SECURE AND RESILIENT
COMPUTING FOR INDUSTRY AND HUMAN DOMAINS

Volume 3
TECHNIQUES, TOOLS AND ASSURANCE CASES
FOR SECURE AND RESILIENT COMPUTING
Edited by Vyacheslav Kharchenko

- Techniques and Tools for Networks and HMI Security Assessment
- Techniques and Tools for Industry FPGA Based Systems Security
- Techniques and Tools of Penetration Testing for Web Applications
- Techniques of Security Assessment for Smart Building Automation Systems
- Security and Resilience Assurance Cases
- Big Data Analysis-Based Assessment of Reliability and Security

БЪЛГАРСКА АКАДЕМИЯ на НАУКИТЕ 1869

CITY UNIVERSITY OF LONDON EST 1894

TUT

KTH

cini

UNIVERSITY OF BANKING OF THE NATIONAL BANK OF UKRAINE

LIMESYSTEMS

radiy

Volume 3. TECHNIQUES, TOOLS AND ASSURANCE CASES
FOR SECURE AND RESILIENT COMPUTING

Tempus 2013 SEREIN 2016

MULTI-
LECTURE
BOOK

SECURE AND RESILIENT COMPUTING FOR INDUSTRY AND HUMAN DOMAINS

VOLUME 3

TECHNIQUES, TOOLS AND ASSURANCE CASES FOR SECURE AND RESILIENT COMPUTING

2017

Co-funded by the
Tempus Programme
of the European Union

Co-funded by the
Tempus Programme
of the European Union

Ministry of Education and Science of Ukraine
National Aerospace University n. a. N. E. Zhukovsky
“Kharkiv Aviation Institute”

O. Gordieiev, V. Kharchenko, V. Sklyar, A. Perepelitsyn, V.Kulanov, M. Poluyanenko, O. Odarushchenko, O. Yasko, E. Babeshko, V. Kulanov, K. Leontiev, I. Zelinko, O.Gordieiev, Ya. Chujkov , O. Illiashenko, V. Duzhyi, O.Rusin, E. Kovalenko, D. Uzun, A. Tetskyi, A. Strielkina , O. Yevsieieva, M. Kolisnyk, I. Piskacheva, S. Yaremchuk, O. Ivanchenko, O. Potii

SECURE AND RESILIENT COMPUTING FOR INDUSTRY AND HUMAN DOMAINS.

Techniques, tools and assurance cases for security and resilient computing

Multi-book, Volume 3

V. S. Kharchenko eds.

Tempus project
SEREIN 543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR
Modernization of Postgraduate Studies on Security and Resilience for Human and Industry Related Domains

2017

O. Gordieiev, V. Kharchenko, V. Sklyar, A. Perepelitsyn, V.Kulanov, M. Poluyanenko, O. Odarushchenko, O. Yasko, E. Babeshko, V. Kulanov, K. Leontiev, I. Zelinko, O.Gordieiev, Ya. Chujkov , O. Illiashenko, V. Duzhyi, O.Rusin, E. Kovalenko, D. Uzun, A. Tetskyi, A. Strielkina , O. Yevsieieva, M. Kolisnyk, I. Piskacheva, S. Yaremchuk, O. Ivanchenko, O. Potii. **Secure and resilient computing for industry and human domains. Techniques, tools and assurance cases for security and resilient computing** / Edited by Kharchenko V. S. – Department of Education and Science of Ukraine, National Aerospace University named after N. E. Zhukovsky "KhAI", 2017.

Reviewers:
Dr. Peter Popov, Centre for Software Reliability, School of Informatics, City University of London
Prof. Stefano Russo, Consorzio Interuniversitario Nazionale per l'Informatica (Naples, Italy)
Prof. Todor Tagarev, Centre for Security and Defence Management, Institute of Information and Communication Technologies of the Bulgarian Academy of Sciences;
Prof. Jüri Vain, School of Information Technologies, Department of Software Tallinn University of Technology

The third volume of the three volume book called "Secure and resilient computing for industry and human domains" contains materials of the lecture parts of the study modules for MSc and PhD level of education as well as lecture part of in-service training modules developed in the framework of the SEREIN project "Modernization of Postgraduate Studies on Security Resilience for Human and Industry Related Domains"[1] (543968-TEMPUS-1-2013-1-EE-TEMPUS-JPCR) funded under the Tempus programme are given. The book describes model and case-based techniques and tools applied to support simulation, assessment and assurance of FPGA and software components and systems.

The descriptions of trainings, which are intended for studying with technologies and means of assessing security guarantees, are given in accordance with international standards and requirements. Courses syllabuses and description of practicums are placed in the correspondent notes on practicums and in-service training modules.

Designed for engineers who are currently or tend to design, develop and implement information security systems, for verification teams and professionals in the field of quality assessment and assurance of cyber security of IT systems, for masters and PhD students from universities that study in the areas of information security, computer science, computer and software engineering, as well as for lecturers of the corresponding courses.

The materials in the book are given in a form as is, desktop publishing of this book is available in hard copy only.

[1] *This project has been funded with support from the European Commission. This publication (communication) reflects the views only of the author, and the Commission cannot be held responsible for any use which may be made of the information contained therein.*

PART 10. TECHNOLOGIES, TECHNIQUES AND TOOLS FOR HMI SECURITY AND USABILITY ASSESSMENT

38.1 ISO 25010 software quality model review

Before describing of basic materials of chapter us need represent short description of new standard ISO/IEC 25010 – «System and software quality model», which include of description of new software quality model (SWQM). It standard include security and usability characteristics also. The standard is part of standards ISO/IEC 25000 series, also known as SQuaRE (System and Software Quality Requirements and Evaluation), has the goal of creating a framework for the evaluation of software product quality.

For represent all materials us need some following terms:

– software quality (SQ) is a degree to which a software product satisfies stated and implied needs when used under specified conditions [1];

– SWQM is [2] usually defined as a set of characteristics and relationships between them which actually provides the basis for specifying the requirements of quality and evaluating quality. A lot of software quality models have been introduced for the last decades [3]. Quality models structure is described by hierarchy whose elements are sets of characteristics (CHs) (subcharacteristics (SubCHs)) and relations of subordination between them. Characteristics (subcharacteristics) included into the models as usual are the basis of software projects requirements.

The product quality model defined in ISO/IEC 25010 comprises the eight quality characteristics:

1. Functional suitability – is degree to which a product or system provides functions that meet stated and implied needs when used under specified conditions. This characteristic is composed of the following subcharacteristics:

– functional completeness – is degree to which the set of functions covers all the specified tasks and user objectives;

– functional correctness – degree to which a product or system provides the correct results with the needed degree of precision;

– functional appropriateness – degree to which the functions facilitate the accomplishment of specified tasks and objectives.

2. Performance efficiency. This characteristic represents the performance relative to the amount of resources used under stated conditions. This characteristic is composed of the following subcharacteristics:

– time behavior – is degree to which the response and processing times and throughput rates of a product or system, when performing its functions, meet requirements;

– resource utilization – is degree to which the amounts and types of resources used by a product or system, when performing its functions, meet requirements;

– capacity – is degree to which the maximum limits of a product or system parameter meet requirements.

3. Compatibility – is degree to which a product, system or component can exchange information with other products, systems or components, and/or perform its required functions, while sharing the same hardware or software environment. This characteristic is composed of the following subcharacteristics:

– co-existence is – degree to which a product can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product;

– interoperability is – degree to which two or more systems, products or components can exchange information and use the information that has been exchanged.

4. Usability – degree to which a product or system can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. This characteristic is composed of the following subcharacteristics:

– appropriateness recognizability – is degree to which users can recognize whether a product or system is appropriate for their needs;

– learnability – is degree to which a product or system can be used by specified users to achieve specified goals of learning to use the product or system with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use;

– operability – is degree to which a product or system has attributes that make it easy to operate and control;

– user error protection – is degree to which a system protects users against making errors;

– user interface aesthetics – is degree to which a user interface enables pleasing and satisfying interaction for the user;

– accessibility – is degree to which a product or system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use.

5. Reliability – is degree to which a system, product or component performs specified functions under specified conditions for a specified period of time. This characteristic is composed of the following subcharacteristics:

– maturity – is degree to which a system, product or component meets needs for reliability under normal operation;

- availability – is degree to which a system, product or component is operational and accessible when required for use;
- fault tolerance – is degree to which a system, product or component operates as intended despite the presence of hardware or software faults;
- recoverability – is degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system.

6. Security – is degree to which a product or system protects information and data so that persons or other products or systems have the degree of data access appropriate to their types and levels of authorization. This characteristic is composed of the following subcharacteristics:

- confidentiality – is degree to which a product or system ensures that data are accessible only to those authorized to have access;
- integrity – is degree to which a system, product or component prevents unauthorized access to, or modification of, computer programs or data;
- non-repudiation – is degree to which actions or events can be proven to have taken place, so that the events or actions cannot be repudiated later;
- accountability – is degree to which the actions of an entity can be traced uniquely to the entity;
- authenticity – is degree to which the identity of a subject or resource can be proved to be the one claimed.

7. Maintainability – is degree of effectiveness and efficiency with which a product or system can be modified to improve it, correct it or adapt it to changes in environment, and in requirements. This characteristic is composed of the following subcharacteristics:

- modularity – is degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components;
- reusability – is degree to which an asset can be used in more than one system, or in building other assets;
- analyzability – is degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified;
- modifiability – is degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality;
- testability – is degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met.

8. Portability – is degree of effectiveness and efficiency with which a system, product or component can be transferred from one hardware, software or other operational or usage environment to another. This characteristic is composed of the following subcharacteristics:

– adaptability – is degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments;

– installability – is degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment;

– replaceability – is degree to which a product can replace another specified software product for the same purpose in the same environment.

38.2 Evolution and correlation of software quality models characteristics security, usability and greenness

Choice of competitive characteristics

Results of applying Structure-Semantic Analysis (SSA) technique [4] to SQM analysis give us the possibility from one side to observe changes in SQMs during more than 40 years of software engineering, and from another side, to determine development trends for each SQM characteristic separately [5]. In several works [4, 5] was proposed metrics for assessing relevance of SQM characteristics, subcharacteristics and models as a whole. Was established, that change of nomenclature and structure of characteristics such as security, usability and characteristics which associated with green and subcharacteristics, which itemize of them. Changes in SQMs are determined by trends in «software engineering» technologies development. Preliminary analysis of derived results in [2-5] allowed us of the paper assume, that changes in competitive characteristics SQMs interconnected among themselves.

General task of the this part of course is analyse evolution and competition for some characteristics of SQMs, such as a security, usability and greenness (i.e. pairs of characteristics «usability-security» and «security-greenness»). Mutual influence and comparative analysis of security, usability and greenness are described using a set of metrics considering relevance of characteristics in SQMs and their changing during last 40 years.

Preliminary analysis of SQMs allowed determined potential most competitive characteristics, which compete with security characteristic. We selected such characteristics - usability and greenness. We review of competition between following pairs of characteristics «usability-security» and «security-greenness»:

– «usability-security». Basis of competition of two characteristics is human-machine interaction. From one side, software user interfaces should be orient on assurance and control of information systems security. Result of this is complexity user interfaces. First of all, complexity of user interfaces is growing of quantity user errors (errors of common cause). From another side, user interfaces should be comfortable, simple and understandable. We can remember of example, when user should make of password (or thinking about new password). From one side, password must have set of requirements to himself, which make his very complicated for memorization. For another side, every user must very easy remember of password. Thus, in sequence of «usability-security» software characteristics must will make special balance;

– «security-greenness». Basic of competition between software characteristics for calculation resources. From one side, security require of additional calculation resources for decision tasks for security. For another side calculation resources need optimize and economize.

Different aspects of attributes relation are described for pairs «usability-security» [6, 7], «security-greenness» [8, 9], «usability-greenness» [10, 11]. Analysis of the works allowed concluding about interesting triangle which is formed by these characteristics and evolution during last decades.

Greenness characteristics are not contained in existing SQMs in an explicit form. Therefore green related or associated with software greenness characteristics and subcharacteristics are analyzed in the paper. Farther, characteristics and characteristics are by SQM structure-semantic analysis (SSA) technique [4].

SSA-technique description

Let us shortly describe the sequence of SSA-technique and to use for analysis of competitive characteristics. The technique describes quality models as a facet-hierarchy structure (graph). Nodes correspond to quality attributes and links take into account hierarchy dependencies. To briefly characterize the proposed technique of analysis let us introduce some initial terms: conceptual model is a model which a model under study is compared with; model under study is a model which is compared with a conceptual model; characteristic under study is a conceptual model characteristic which is compared with model under study characteristics.

SSA-technique is based on comparing a model under study with the conceptual model, i.e. every SQM is compared with the conceptual model. So, the analysis is equivalent to semantically comparing characteristics and subcharacteristics of a model under study and the conceptual model with regard to their structures. Selecting a reference model is usually performed by an expert who has relevant experience and qualifications.

At the following stage comparison of models among themselves should be performed. The simplest and most obvious metrics are offered. Hierarchy of these metrics is presented in Fig. 38.1. The metrics are used to compare models with reference model bottom up, i.e. first at the level of subcharacteristics (subcharacteristics matching metric SMM, cumulative subcharacteristics comparison metric CSCM, characteristics matching metric CMM), then at the level of characteristics (cumulative matching characteristics metric CMCM) and finally at the level of models as a whole (cumulative software quality models comparison metric CSQMCM).

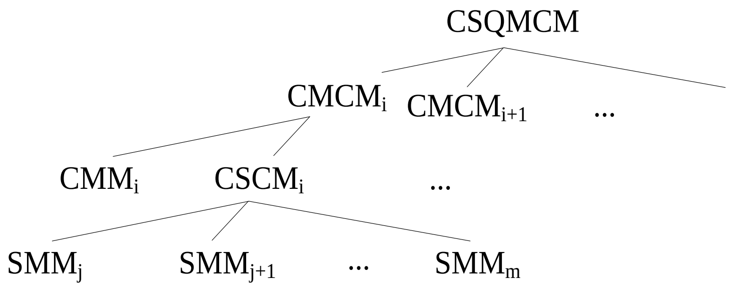


Figure 38.1 – Metrics hierarchy

Features of the metrics are the following:

– subcharacteristic matching metric (SMMj). Every subcharacteristic match value is identified as SMMj = 0,5 / number of reference (conceptual) model elements subcharacteristics of the characteristic under study. Weights of characteristics are not considered when calculating metrics;

– cumulative subcharacteristics comparison metric (CSCM) is evaluated as a sum of SMM:

$$CSCM_i = \sum_{j=1}^{k} SMM_j; \tag{38.1}$$

– characteristics matching metric (CMM) takes the value of 0.5 in case of matching or 0 if the characteristics are different;

– cumulative matching characteristics metric (CMCM) is calculated as a sum of CMM metric and $\sum_{j=1}^{k} CSCM_j$:

$$CMCM_i = CMM_i + \sum_{j=1}^{k} CSCM_j; \tag{38.2}$$

– cumulative software quality models comparison metric (CSQMCM) is calculated according to the formula:

$$CSQMCM_i = \sum_{j=1}^{n} CMCM_j . \tag{38.3}$$

Results of evolution analysis and interference of competitive characteristics

Let us conduct SW QM analysis and first of all, define the reference (conceptual) model. SW Quality Model ISO/IEC 25010 [1] will be considered as uppermost and etalon regarding to all other models. It is the newest introduced model and takes into account main modern software peculiarities from the point of view of quality evaluation. This model is described by international standard of top level.

According with results of analysis CMCM is calculated for the set of characteristics presented in Table 38.1. The results of calculation are shown in Table 38.2 (Chs – characteristics, SChs – subcharacteristics) for Greenness, Usability and Security characteristics.

The histogram of CMCM values for software quality models is presented in Fig. 38.2 (black color for Greenness, gray for Usability and light gray for Security). An abscissa axis corresponds to years of SQM emergence. Initial point (year) is 1970 (as a first year after 1968 which is multiple of ten years).

CMCM values will be further represented and analysed only for so-called basic SWQMs [4]. Basic models were selected considering their support by standards, the international reputation and application. The models of McCall and Boehm are similar, hence first one was selected. Hence, the models of Boehm, Ghezzi, FURPS, Dromey, QMOOD were excluded (Fig. 38.3).

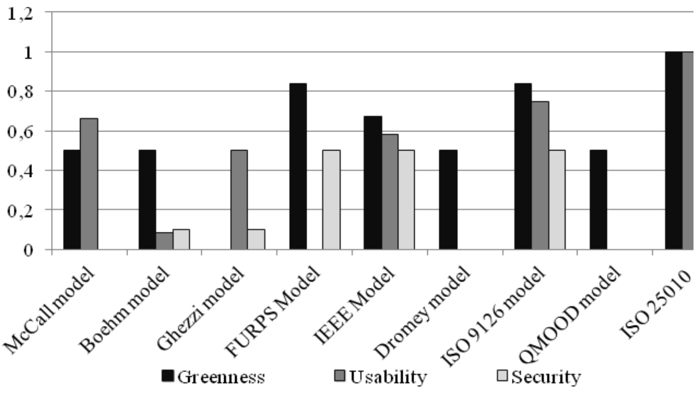


Figure 38.2 – CMCM values for Greenness, Usability, Security characteristics of SQMs

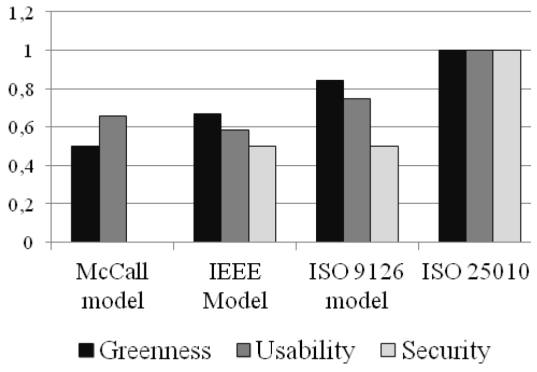


Figure 38.3 – CMCM values for Greenness, Usability, Security characteristics of basic SQMs

Table 38.1 – SWQM characteristics (greenness, security, usability)

№	**SWQMs (years)**	**Greenness characteristics**	**Security characteristics**	**Usability characteristics**
1.	McCall (1977)	4. Efficiency	-	6. Usability
		4.1 Execution efficiency		6.1 Operability
		4.2 Storage efficiency		6.2 Training
2.	Boehm (1978)	2.2 Efficiency	-	3.2 Understability
		2.2.1 Accountability		3.2.1 Legibility
		2.2.2 Accessibility		3.2.2 Conciseness
				3.2.3 Structureness
				3.2.4 Self descriptiveness
3.	Carlo Ghezzi (1991)	-	-	7. Usability
4.	FURPS (1992)	4.Performance	1.Functionality	2.Usability
		4.1 Velocity	1.3 Security	2.1 Human factors
		4.2 Efficiency		2.2 Aesthetic
		4.3 Availability		2.3 Documentation of the user
		4.4 Time of		2.4 Material of training

№	SWQMs (years)	Greenness characteristics	Security characteristics	Usability characteristics
		answer		
		4.5 Time of recovery		
		4.6 Utilization of resources		
		4.6 Capacity		
5.	IEEE (1993)	1.Efficiency	1.Functionality	6.Usability
		1.1 Temporal efficiency	1.3 Security	6.1 Comprehensibility
		1.2 Resource efficiency		6.2 Ease of learning
				6.3 Communicativeness
6.	Dromey (1995)	2.2 Efficiency	-	-
7.	ISO 9126-1 (2001)	4.Efficiency	1.Functionality	3 Usability
		4.1 Time behavior	1.4 Security	3.1 Understandability
		4.2 Resource utilization		3.2 Learnability
				3.3 Operability
				3.4 Attractiveness
8.	QMOOD (2002)	6 Effectiveness	-	3. Understandability
9.	ISO 25010 (2010)	2.Performance efficiency	6.Security	4.Usability
		2.1 Time behavior	6.1 Confidentiality	4.1 Appropriateness recognisability
		2.2 Resource utilization	6.2 Integrity	4.2 Learnability
		2.3 Capacity	6.3 Non-repudiation	4.3 Operability
			6.4 Accountability	4.4 User error protection
			6.5 Authenticity	4.5 User interface aesthetics
				4.6 Accessibility

Table 38.2 – Results of Greenness, Usability, Security characteristics comparison and CMCM calculation

Conceptual model (ISO 25010)		McCall model (1977)				Boehm model (1978)				Ghezzi model (1991)				FURPS Model (1992)				IEEE Model (1993)				Dromey model (1995)				ISO 9126 model (2001)				QMOOD model (2002)			
Chs	SChs	Chs	SChs	CMM	SMM	Chs	SChs	CMM	SMM	Chs	SChs	CMM	SMM	Chs	SChs	CMM	SMM	Chs	SChs	CMM	SMM	Chs	SChs	CMM	SMM	Chs	SChs	CMM	SMM	Chs	SChs	CMM	SMM
Greenness																																	
2		4		-	-	-	-	0	0,5	-	-	0	0	-	4.2	0	0,5	1	-	0,5	0	-	2.2	0	0.5	4	-	0,5	0	2	-	0,5	0
	2.1	-	-	-	-	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	4.1	0	0,17	-	-	0	0
	2.2	-	-	-	-	-	-	0	0	-	-	0	0	-	4.6	0	0,17	-	1.2	0	0,17	-	-	0	0	-	4.2	0	0,17	-	-		0
	2.3	-	-	-	-	-	-	0	0	-	-	0	0	-	1.2	0	0,17	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0
		CMCM=0,5				CMCM=0,5				CMCM=0				CMCM =0,84				CMCM =0,67				CMCM =0,5				CMCM=0,84				CMCM=0,5			
Usability																																	
4		6.	-	0,5	0	-	-	0	0	7	-	0,5	0	-	-	0	0	6	-	0,5	0	-	-	0	0	3	-	0,5	0	-	-	0	0
	4.1	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0
	4.2	-	6.2	0	0,08	-	-	0	0	-	-	0	0	-	-	0	0	-	6.2	0	0,08	-	-	0	0	-	3.2	0	0,08	-	-	0	0
	4.3	-	6.1	0	0,08	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	3.3	0	0,08	-	-	0	0
	4.4	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0

	4.5	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	3.4	0	0,08	-	-	0	0
	4.6	-	-	0	0	-	2.2.2,2.3.1,3.1.4	0	0,08	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0
		CMCM=0,66				CMCM=0,083				CMCM=0,5				CMCM=0				CMCM=0,583				CMCM=0				CMCM=0,749				CMCM=0			
Security																																	
6		-	-	0	0	-	-	0	0	-	-	0	0	-	1,3	0	0,5	-	3.3	0,5	0	-	-	0	0	-	1.4	0	0,5	-	-	0	0
	6.1	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0
	6.2	-	-	0	0	-	2.1.2	0	0,1	1	-	0,1	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0
	6.3	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0
	6.4	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0
	6.5	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0	-	-	0	0
		CMCM=0				CMCM=0,1				CMCM=0,1				CMCM=0,5				CMCM=0,5				CMCM=0				CMCM=0,5				CMCM=0			

Different types of mathematical relations between SWQM appearance year (X axis) and CMCM value (Y axis) for Greenness, Usability and Security characteristics have been determined with the help of graphical analysis of initial data. Let us show such relations and the values of coefficients of determination (R2) for each characteristic:

– to describe the mathematical relation for Greenness characteristic, the most suitable is linear function (Fig. 38.4)

$$y = 0{,}167x + 0{,}335,\ R^2=0{,}999; \qquad (38.4)$$

– to describe the mathematical relation for Usability characteristic, the most suitable is polynomial dependence of second degree (Fig. 38.5)

$$y = 0{,}082x^2 - 0{,}291x + 0{,}861,\ R^2=0{,}987; \qquad (38.5)$$

– to describe the mathematical relation for Security characteristic, the most suitable is polynomial dependence of third degree (Fig. 38.6)

$$y = 0{,}166x^3 - 1{,}25x^2 + 3{,}083x – 2,\ R^2=1. \qquad (38.6)$$

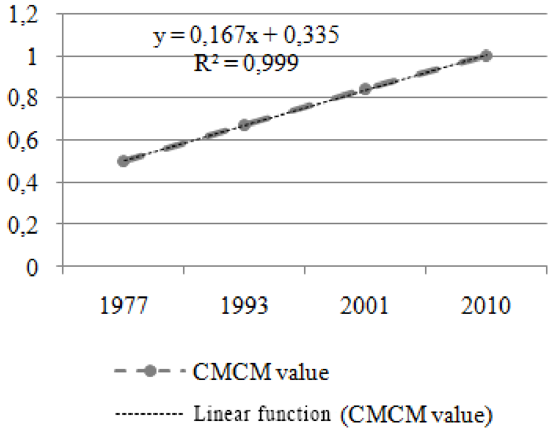


Figure 38.4 – Diagram of change of CMCM values for Greenness

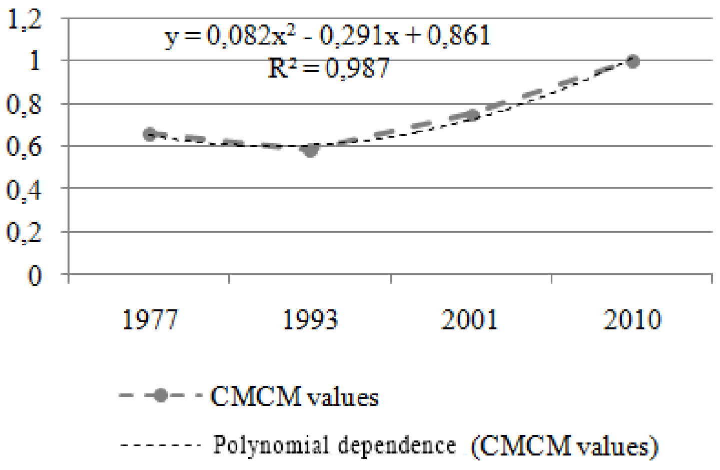


Figure 38.5 – Diagram of change of CMCM values for Usability

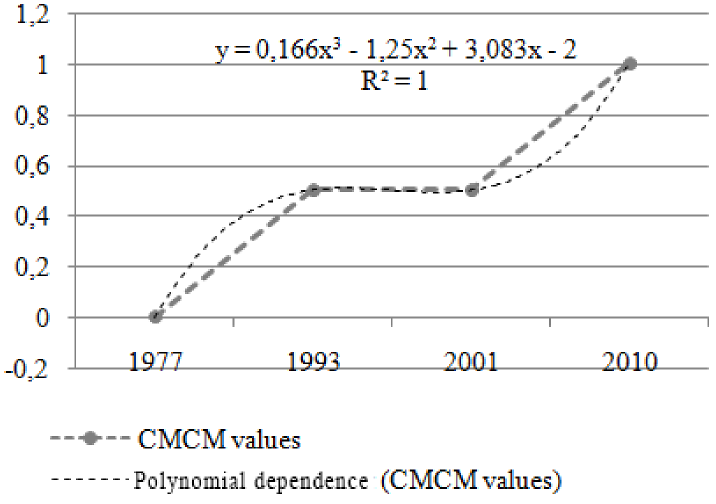


Figure 38.6 – Diagram of change of CMCM values for Security

Formulas 38.4-38.6 and figures 38.4-38.6 illustrate a tendency of SQMs characteristics/ subcharacteristics changes. Analysis of dependencies (Fig. 38.3) allows concluding that weights of Greenness, Usability and Security characteristics became equal in 2010 (the standard ISO/IEC 25010).

Each competitive characteristic was described by different types of mathematical relations. Exact choice of relation type was confirmed high value of coefficient of determination (R2).

Derived results gave us following information:

– difference between minimal and maximal values of metric CMCM (ΔCMCM) determine dynamic of evolution for each characteristic: for Greenness ΔCMCM = 0,5; for Usability ΔCMCM = 0,34; for Security ΔCMCM = 1;

– modifications of CMCM metric values for Security characteristic in SWQM give us the basis to approve that this characteristic has undertaken dynamic development recently. Firstly, as subchracteristic, Security was represented in 1992 year in SWQM FURPS, and as single characteristic only in 2010 year in SWQM ISO 25010. In this perspective, we can confirm that evolution of Security characteristic is developing with delay in SWQM;

– CMCM metric value for Greenness and Security characteristics during their evolution evolution did not decrease, and for Usability slightly decrease by 0,8 in 1993 in IEEE SQM;

– we can confirm, that linear mathematical relation, which was obtained for Greenness characteristic, reveals natural development of Green Software technologies;

– nonlinearity of mathematical relation for Usability, to a lesser degree, and for Security farther give us information, that such type of relation was determined by important external factors. For example, for Security such factor is growth of number intrusion to information systems for insufficient quality of software Security.

Influence of Security on another characteristics (Usability and Greenness) was determined. For this interaction influence of subcharacteristics was determined as shown in Table 38.3.

Conclusion and self-control questions

In result of this work more competitive characteristics of SQMs, such as Greenness, Usability and Security were selected. For each of these characteristics analysis was conducted and interaction was determined. Security characteristic from three competitive characteristics is developing more dynamically. Such tendency is represented by sharp increase of software security requirements recently. Authors consider that in modern conditions

SQMs should change more often than 1 time in 10 years. More often changes in SQMs can be connected with only separated characteristics, for example, with security.

Table 38.3 – Result of subchracteristics interaction

Security	**Usability**						**Greenness**		
	Appropriateness recognisability	Learnability	Operability	User error protection	User interface aesthetics	Accessibility	Time behavior	Resource utilization	Capacity
Confidentiality	~	~	~	↑	↓	↓	↑	↑	↓
Integrity	~	↓	↓	~	↓	~	↑	↑	↓
Non-repudiation	~	~	~	~	↓	~	↑	↑	↓
Accountability	~	~	~	~	↓	~	↑	↑	↓
Authenticity	~	~	~	↑	↓	↓	↑	↑	↓

Table of symbols:

↓ worsening of one subcharacteristic in the presence of improvement of another characteristic (security);

↑ improvement of one subcharacteristic in the presence of worsening of another characteristic;

– worsening of one subcharacteristic in the presence of improvement of another characteristic (security) do not occur ;

~ unknown dependence.

Self-control questions and tasks

1. What is software quality?
2. What is SWQM?
3. What are characteristics (subcharacteristics) SWQM?
4. Represent of characteristics (subcharacteristics) SWQM examples.
5. What is SSA-technique and what is it used for??

6. What is generally between security, usability and greenness?

References

1. International Standard ISO/IEC 25010. Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models, ISO/IEC JTC1/SC7/WG6.
2. Sanjay Kumar Dubey, Soumi Ghosh, Ajay Rana: Comparison of Software Quality Models: An Analytical Approach. In: International Journal of Emerging Technology and Advanced Engineering, Vol. 2 (2), pp. 111-119, February 2012
3. Stefan Wagner: Software Product Quality Control. Springer, Heidelberg (2013)
4. Gordieiev O., Kharchenko V., Fominykh N., Sklyar V. Evolution of software quality models in context of the standard ISO 25010. Proceedings of 9th International Conference on Dependability and Complex Systems (DepCoS-RELCOMEX 2014), 30 Jun-4 July,2014, Advances in Intelligent and Soft Computing, Springer, Brunow, Poland, pp. 223-233,
5. Oleksandr Gordieiev, Vyacheslav Kharchenko and Mario Fusani. Evolution of Software Quality Models: Green and Reliability Issues. Proceedings of the 11th International Conference ICT in Education, Research and Industrial Applications: Integration, Harmonization and Knowledge Transfer (ICTERI 2015), May 14, 2015 Lviv, Ukraine: CEUR-WS.org, pp. 432-445.
6. Bryan D. Payne, W. Keith Edwards. A Brief Introduction to Usable Security, Useful Computer Security, May/June, 2008. pp. 13-21.
7. Dirk Balfanz, Glenn Durfee, D.K. Smetters. In Search of Usable Security: Five Lessons from the Field, IEEE Security & Privacy, September/October, 2004, pp. 19-24.
8. Xun Li, Frederic T. Chong. A Case for Energy-Aware Security Mechanisms. Proceedings of 27th International Conference Networking and Applications Workshops (WAINA 2013), 25-28 March, 2013, Barcelona, Spain, pp. 1541-1546.
9. Carroll M., Merwe A., Kotze P. Secure cloud computing: Benefits, risks and controls. Proceedings of Information Security South Africa (ISSA 2011), 15-17 Aug., 2011, IEEE, Johannesburg, South Africa, pp. 1-9.
10. Effie Lai-Chong Law, Ebba Thora Hvannberg, Gilbert Cockton. Maturing Usability. Part 6 A Green Paper on Usability Maturation. Human-Computer Interaction Series. Springer-Verlag London, 2008, pp. 381-424.
11. Thimbleby H. Interaction Walkthrough: Evaluation of Safety Critical Interactive Systems. The XIII International Workshop on Design,

Specification and Verification of Interactive Systems (DSVIS 2006), Springer Lecture Notes in Computer Science, 2007, pp. 52-66.

Acronyms

– SWQM - software quality model;
– SQuaRE - System and Software Quality Requirements and Evaluation;
– SQ - software quality;
– CHs - characteristics;
– SSA - Structure-Semantic Analysis;
– SMM - subcharacteristics matching metric;
– CSCM - cumulative subcharacteristics comparison metric;
– CMM - characteristics matching metric;
– CMCM - cumulative matching characteristics metric;
– CSQMCM - cumulative software quality models comparison metric.

39 INTRODUCTION IN EYE-TRACKING TECHNOLOGIES

CONTENTS

39.1 Eye-tracking conception

Eye-tracking – the process of tracking human eyes movements by using a special device – eye-tracker. Eye-tracking is also used for measuring the characteristics of eyes movements and physical characteristics of eyes, such as measuring the pupil size. Now eye-tracking is applied to research, aimed at finding weaknesses in the human-machine interaction as well as to improve them.

Eye-tracker is a hardware, which records human's eyes movements, when human looks at a computer screen, at physical object or even at his environment in general. There are several types of eye-trackers, one placed on the respondent's head or worn as glasses, while others can be placed in front of the respondent or mounted under the monitor.

Let's consider the elements of the eye-tracking process. Eye-tracker emits infrared light in eyes of the respondent, and then records the reflected infrared light from the retina of participant (i.e. respondent). This procedure gives the eye-tracker opportunity to find the center of the respondent's pupil, and also to analyze the reflected infrared light from the cornea. Let us make review of the structure of the human eye (Fig. 39.1). Structure includes:

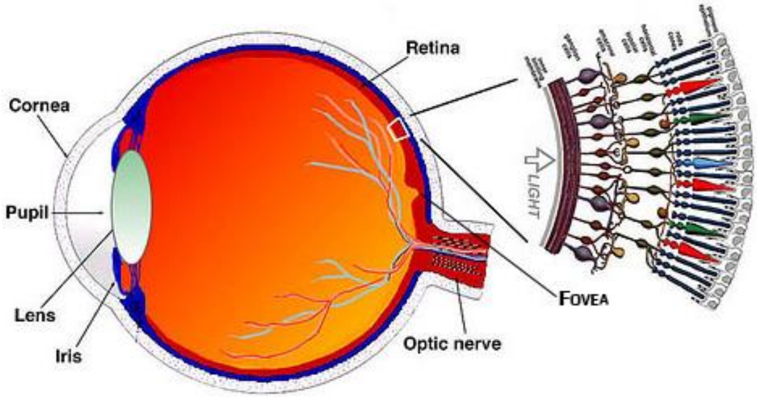


Figure 39.1 – Structure of the human eye

– retina – this light-sensitive layer in the back of the eye, that illuminates the eye-tracker;

– pupil – allows infrared light to penetrate through it to the retina and be reflected from it again through the pupil, to get to the light-sensitive camera of the eye-tracker;

– cornea – the transparent front part of the eye.

If we look at the human's eyes, we can see the reflection of a light from the cornea in each of his eyes. If person keeps his head still and look left, right, up and down, reflection is moving, this is made by the pupil. You can see how changes the relation between the pupil centre and the reflection of a light (Fig. 39.2) [1-4].

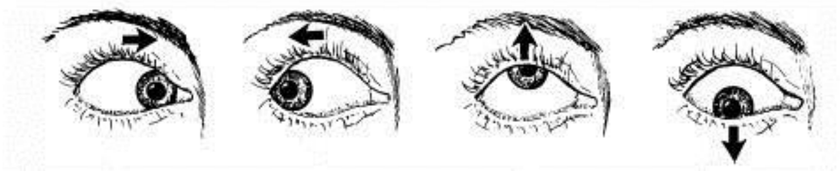


Figure 39.2 – The relative position of the pupil and the reflection of a light changes when eyes are moving, but the head remains stationary

It turns out that the place where person looks can be determined from the position of the pupil centre with respect to the corneal reflection. If person moves his head, looking at the same place, the relation between the pupil centre and the corneal reflection remains unchanged (Fig. 39.3).

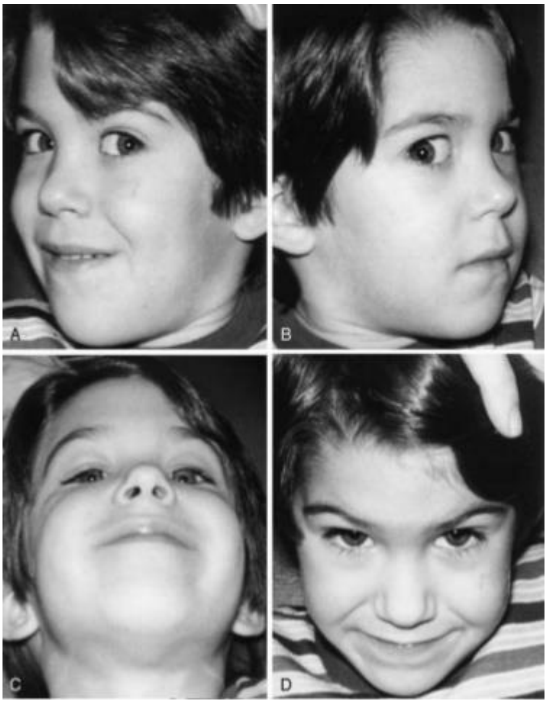


Figure 39.3 – Location of the pupil and corneal reflection does not change

Even if he moves - eye-tracker determines that the person is looking at the same point.

Modern commercial eye-trackers consist of two main components. The first is a light source (approximate to the infrared), it creates a reflection in the human eye. The second component is a video camera, which is sensitive to the infrared light. The camera focuses on eyes of the respondent and records the reflection. Then, by using the software, it calculates and superimposes the view location, for example, on the webpage.

Eye-tracker uses a wavelength that is invisible to humans and therefore does not distract them, but it is reflected by the eye. Infrared light source is not harmful to humans.

Movement of the human's eyes

Human's eyes cover a field of view of about 180 degrees horizontally. This is called the range of visual information (90 degrees to the left and 90 degrees to the right) and 90 degrees vertical (Fig. 39.4).

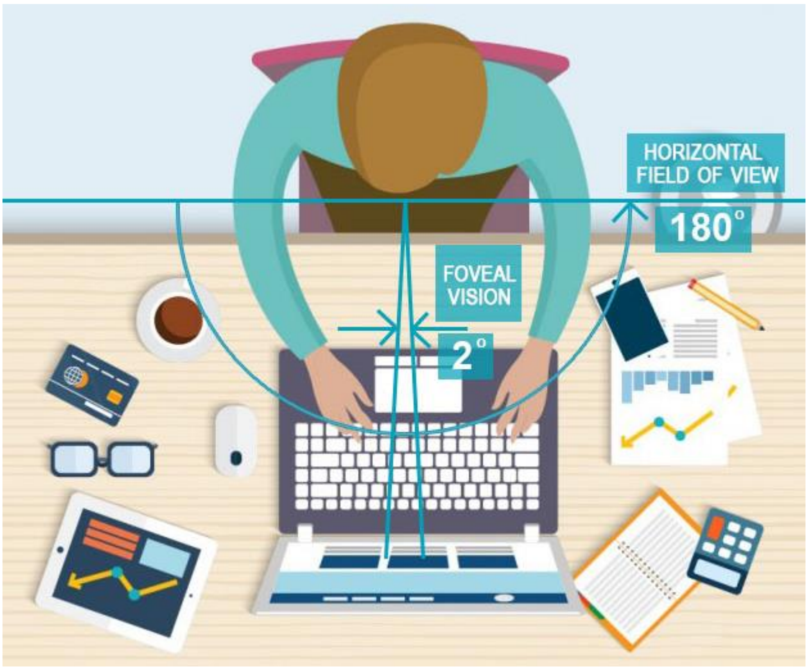


Figure 39.4 – Range of 180 degrees of visual information and the focus (2 degrees)

Every time the respondent's eyes are open, the image of what he sees is projected on the retina. Retinal cells transform this image into signals which then are transmitted to the brain. Cells responsible for visual acuity are concentrated in the centre of the retina, which is called fovea (Fig. 39.1). When a person looks at the object, its image falls on the fovea, and thus, it looks much sharper (clearer) and more colourful than images and objects that are outside the fovea. The area of the fovea is small enough - it covers only 2 degrees of visual information range, which is very often compared to the size of miniature at arm's length. Even if the respondent does not usually aware of this, the image becomes blurry right outside the fovea in the area called the focus of attention (2-5 degrees) and even more blurred on the periphery (Fig. 39.5, NPP systems control interface). Therefore, eyes movements are necessary to keep things in focus. This is an important information-filtering substitution mechanism of human. It helps to unload the respondent's brain. If the focus of attention occupied all 180 degrees, the human brain would be overloaded with information.

From above, on the Fig.39.5, it is shown the centre of attention area, which illustrates the profound vision - the focus of attention. The farther from the focus location, the less object definition.

From below, on the Fig.39.5, it is shown a picture of how eyes movements are helping to create the impression that a person sees everything clearly, allowing you to see just several conditional areas.

Human eyes are moving from place to place several times per second (three or four times, on average). These fast movements, called saccades. They are the fastest movements produced by the external parts of the human body. To prevent turbidity, human vision is almost completely suppressed during saccades. Visual information is seen only when the eyes are relatively immobile and focused on any object (Fig.39.6), i.e. fixation occurs. It lasts from one-tenth to half a second. After that, the eyes move back (through a saccade) to the next part of the field of view. Thus, we can say that human vision is in constant motion, from the current fixation of eyesight through a saccade to a new fixation.

On the Fig. 39.6, it is shown the human's eyes movements, who looks at the flyer of the Cyber forum «DeSSerT», which took place in 2016 in Kiev (Ukraine) [5]. Fixations are presented in the form of points and saccades are presented as lines that connect points. The dot size is proportional to the duration of fixation.

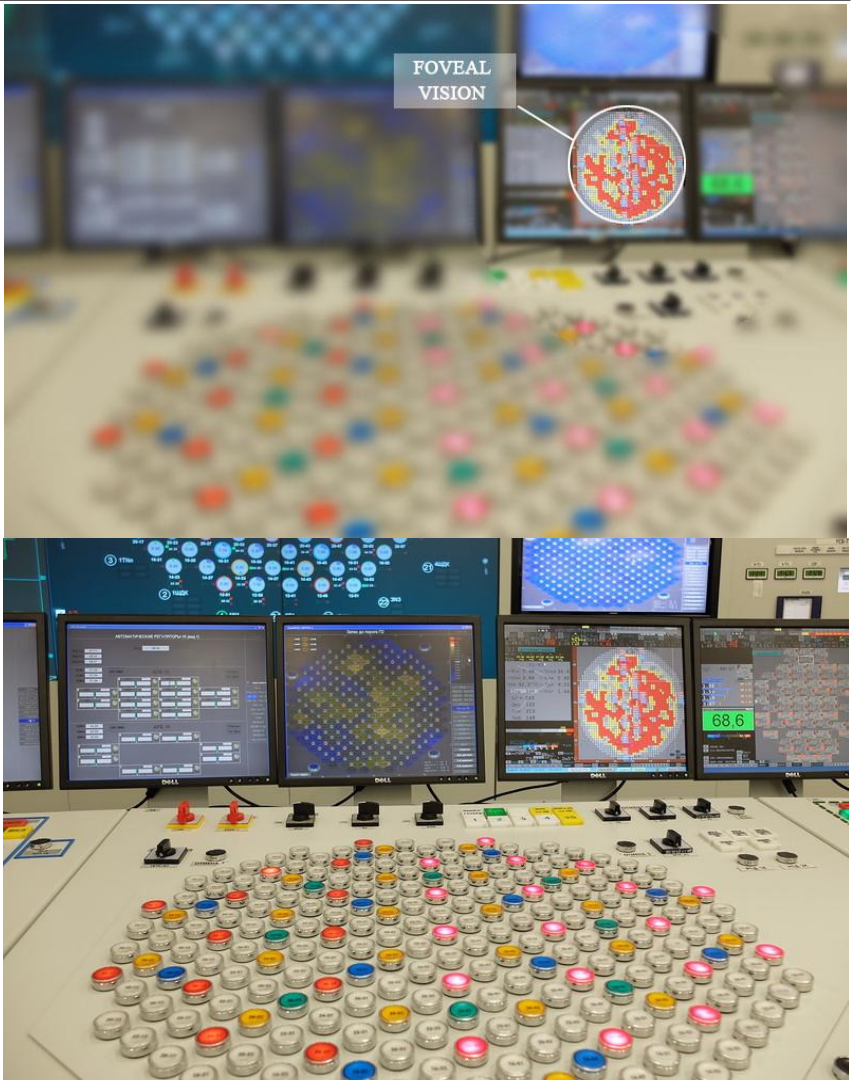


Figure 39.5 – An example of the focus of attention

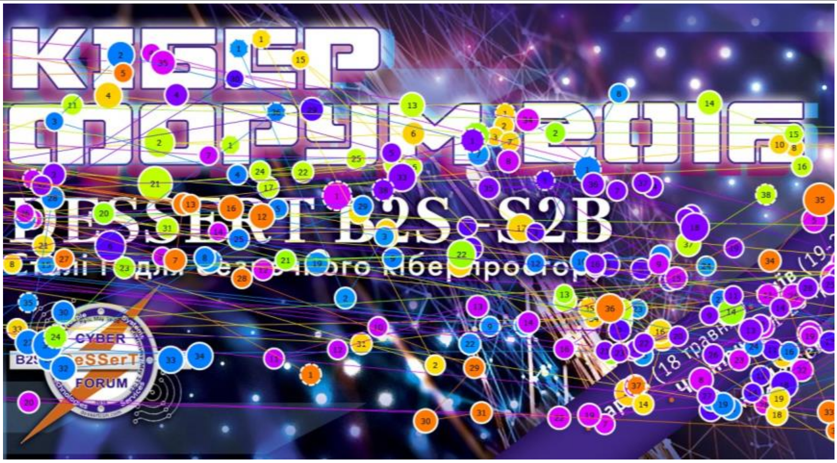


Figure 39.6 – Fixations and saccades

Restrictions of the eye-tracking

Studies have found that mostly people are looking at those objects that attract attention. Especially in focus are those objects which people think about while watching. This is called the hypothesis of the eye of the mind.

However, there are sceptics who do not think that the knowledge of where people are watching may be significant. The argument is usually the next: «I do not have look at the object directly to notice it», - than this argument is followed by the next, - «I look at your face right now, but I still can see the colour of your sweater». Such examples are numerous.

Eye-trackers have several limitations associated with the perception of objects of peripheral vision. Consider such features of eye-tracker:

– for example, if the respondent is looking to someone's face, he could see the colour of his upper garment. But if he wants to consider someone's upper garment, he looks directly at it for two reasons: firstly, the respondent can see things much more clearly, looking directly at them; secondly, if he makes himself deliberately to not look directly at an object, he makes up for this a certain effort. People prefer to change the direction of view moving the visual attention and concentrating on what they are trying to see. However, when people obviously do not look directly at the object, they can not fully credibly affirm that they had not seen it. Eye-tracking catches only those objects that fall into the zone of the focus of attention, not giving any information about what was seen by peripheral vision;

– another argument against the eye-tracking could be the next: «People can look at the object, but do not necessarily see it». Consider the example. If a man closes his eyes after talking face to face to his companion, the other person can not always answer the question about the colour of his eyes, although he looked into the man's eyes during the conversation. Moreover, such a question can not always be answered by people who often communicate and have been long familiar with each other.

Let's make the following conclusions:

– the absence of fixation does not necessarily mean lack of attention, and fixation is not always indicating on the attention, but the fixation and attention are often related;

– attention is usually a little ahead of the eyes, because it plans the next step. After eyes looked elsewhere, attention helps to process and to analyse what has been fixed by the eyesight. With information about the direction of human attention in a given time, the researcher can identify design flaws and human-computer interaction, as well as to improve them.

Visual attention

The visual behaviour of a person is under the influence of all that makes you look (descending attention), as well as your desire to see something, (ascending attention). Descending attention is usually stimulated by a variety of factors (bright colours, speed). Attention shifted to objects unwittingly that are in some way contrasted with its surroundings. For example, bright colours and movements can cause the change of the focus. Things that are new or unexpected in the familiar environment can also attract the attention.

If ascending factors were the only ones that affect people's attention, they would look at the world around us, regardless of external stimuli of the focus of attention (bright colours, speed). Descending attention is based on previous experience and expectations. As a result, people are looking at those objects that correspond to their targets.

Movements of eyes are target-dependent. This means that one person will look at one object differently if he is given different goals. For example, if in the study of packaging of the mobile phone, the person puts two different targets, then his optic routes will be different. On the Fig. 39.7 [6] represented the results of the visual routes which correspond to two different purposes: the definition of brand mobile phone (Fig. 39.7, a.) and does the phone has Internet access function (Fig. 39.7, b.), i.e. does it support Web-browser. Descending attention is responsible for such differences.

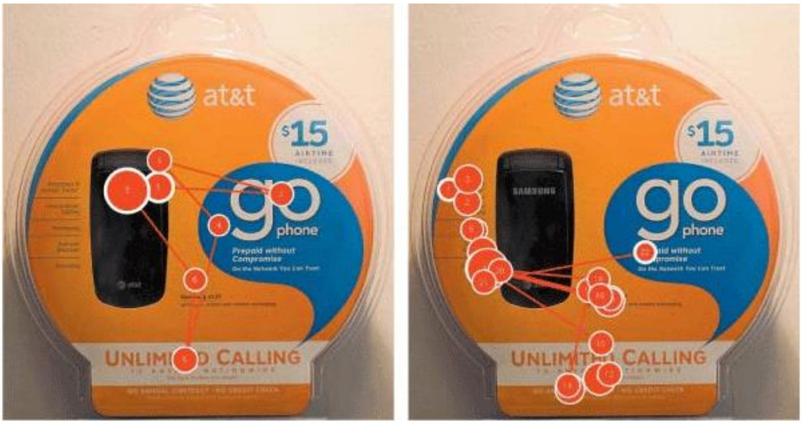
The direction of application for eye-tracking

There are two main applications for eye-tracking:

– usage in methods of research;

– application as a device for input of information and control.

As input devices and control, eye movements are used as the control signals for the computer system, that replace or supplement the computer mouse and keyboard. People with disabilities (patients with ALS or cerebral palsy) used a visual management application for communication. Visual interaction is also used in the field of entertainment (e.g. games) and is now becoming the norm for mobile applications.



а) б)

Figure 39.7 – Features of eye movements at various tasks [6]

Eye-tracking can be used for studies in engineering psychology, design planning, user's research and evaluation design (human-computer interaction) (Table 39.1, text bold) [6].

Table 39.1 – Eye-tracking studies

The subject of the study		
Users		Design
The general idea about the products	The study of engineering psychology	The study of design
Based on the theory or doctrine	Target: to understand the capabilities and limits of human perception, cognition and control of movements. Examples of research issues: how noise affects human performance? what factors influence the switching between different tasks and goals?	Target: to understand how a design or its elements influence on the opinion of people. Examples of research issues: what is the most attractive layout for a web page? How do people search? Do badges and icons stimulate users activity?

Product characteristic	Custom research	User interfaces evaluation
Based on the experience of users, does not go beyond a single product	**Target: to learn more about needs, preferences, motives and user processes, or potential users (consumers) of a particular product. Examples of research issues: why do users of a particular phone model visit the support site and do they get the information they need? Why users are experiencing problems while working with software interfaces?**	**Target: to evaluate a specific product, based on opinions of users. Examples of research issues: how the product can be improved to attract attention? what is the quality of the new design (UI) compared to the old?**

If the researcher has the eye-tracker and he knows how to work with it (e.g. to form a heat map, Fig. 39.8), this does not mean that it needs to be used for all ongoing research. Researcher should know and be able to apply the eye-tracking thus, to obtain reliable and necessary information. Such a process should include planning, training and research. Also, to such a process is usually referred analysis and interpretation of the collected data of the eye-tracking. Note that eye-tracking should be in the form of a complete method.

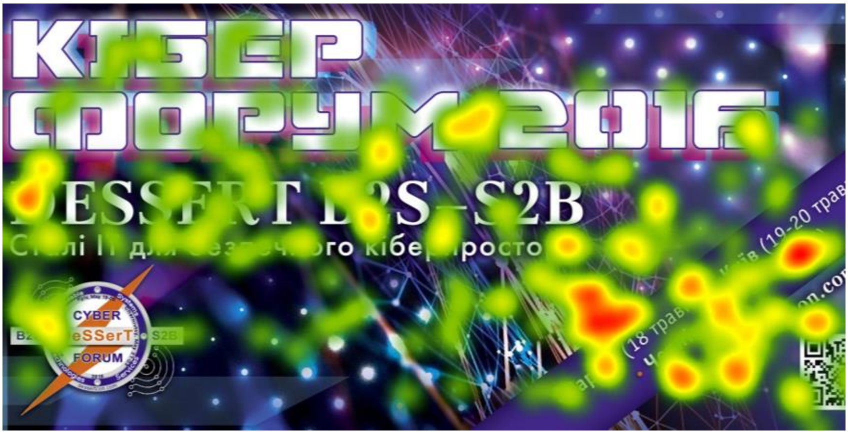


Figure 39.8 – An example of heat map

A heat map (Fig.39.8) is widely used for visualization of eye-tracking. It represents the value of the variable (e.g. the duration of the focus of attention) as the color, where the number of «heat» is proportional to the level provided by the variable.

39.2 Eye-tracking application necessity

Practicing UX-professionals were divided into two opposing camps: on those who advocate for the use of eye-tracking in studies and those who are against it. Supporters want to use the eye-tracking for each study, regardless of its purpose. Opponents, claim that eye-tracking just does not matter much. As usual, the truth lies somewhere in the middle. Eye-tracking can be a valuable addition for particular UX research and inefficient or even wasteful in other cases.

Algorithm of the necessity of applying the eye-tracking

Decision about usage of the eye-tracking in each research comes down to answering three questions, which are presented in the block scheme on Fig. 39.9.

The first and the most important question: «Will eye-tracking provide relevant information that meets the objectives of the research?». To justify usage of eye-tracking in studies, researcher's answer to the first question must be positive. But it's not enough. There are still two questions and we need at least one more positive response.

The second and third questions encourages researcher to think about the economy of chosen approach: «Is eye-tracking simplest or only one technique for obtain the necessary data ?». There are other, more traditional research methods (e.g., observation and user interviews) that can provide answers to the research questions (as, in fact, often the case). In this case, usage of eye-tracking may not be justifiable. The next and last issue is: «Exist necessity for additional investment for future research with eye-tracking ?». A positive answer to the second or third question (or both) gives researcher the opportunity and the right to apply eye-tracking for research.

Conducting the eye-tracking

Supporters of usage of eye-tracking usually accompany their arguments with the following phrases «Here is what eye-tracking can give us». For example, «Eye-tracking can tell us about user's search strategies and about his decision-making processes» or «Eye-tracking can determine what users find interesting». Such statements are not incorrect. However, they are vague and not very convincing for those who are sincerely trying to determine (justify) the need of using the eye-tracking in their research.

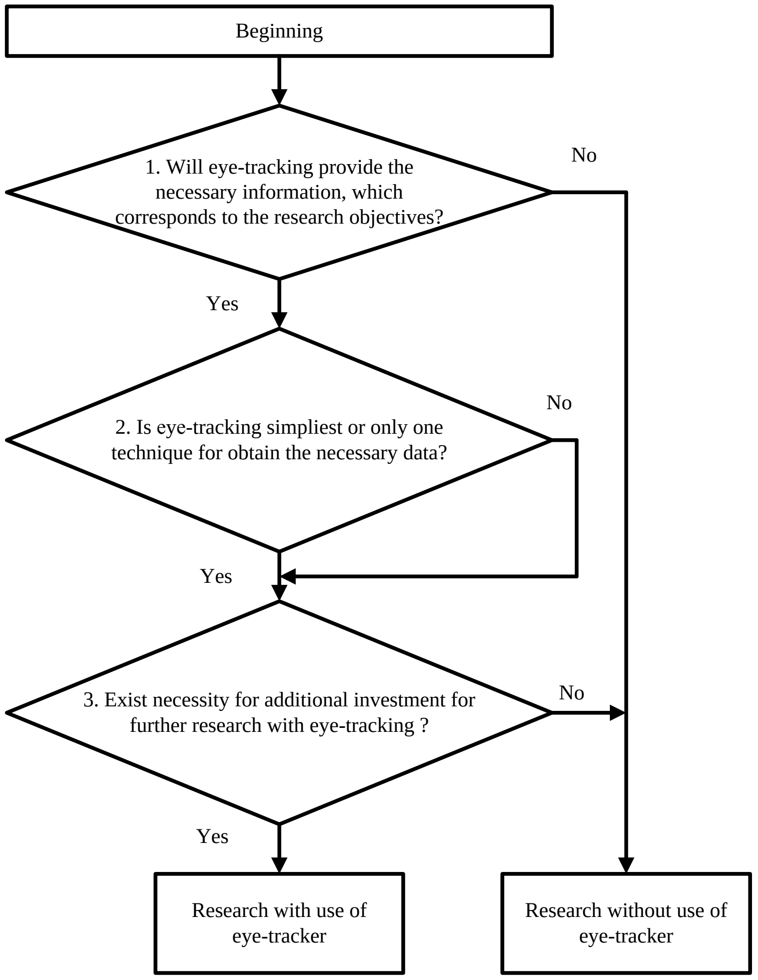

Beginning
1. Will eye-tracking provide the necessary information, which corresponds to the research objectives?
No
Yes
2. Is eye-tracking simpliest or only one technique for obtain the necessary data?
No
Yes
3. Exist necessity for additional investment for further research with eye-tracking ?
No
Yes
Research with use of eye-tracker
Research without use of eye-tracker

Figure 39.9 – Algorithm of the necessity of applying the eye-tracking

UX-researchers usually focus on collecting useful information, usually to have high-level answers. For example, «How can we improve this product?». Eye-tracking can provide researchers with useful information for responses to similar questions, but most of this information does not give direct answers to such practical questions.

Refer to the small example. Let's assume that the instructions for the assembly of the drone become a subject of the research (Fig. 39.10). Eye-tracking data showed that participants spent 10 seconds looking at the front panel before opening the booklet and reading the instructions. If the purpose of the study was to assess the understanding of instructions, such information will be useless. Such information could also be considered as useful information if the aim of the study was the assessment of the effectiveness of new advertisement which is placed on the front panel. Therefore, the core of any study is to clearly articulate the objectives of the study. Note that the eye-tracking is not appropriate to use as long as there are no clearly defined research objectives.

Figure 39.10 – The study of human eyes movements, who reads the user's manual

Also note that the process of eye-tracking is expensive enough (expensive is the equipment and the time of the professionals who work with it). To conduct eye-tracking there should be clearly formulated the purpose and objectives of such a process.

On the Fig. 39.11 displays the types of assessment.

Problems of practical application of eye-tracking

Eye tracking is also used to understand the cognitive processes of users during the study. This understanding sheds light on the user’s experience, which he gains during following events: movements and clicks of the mouse, physical objects manipulations, as well as comments of respondents. Eyes movements help to identify partially unconscious processes that led to the different results. Such information can be used for finding design flaws and to form recommendations for their improvement. A qualitative understanding of all processes can be obtained after collecting the required data.

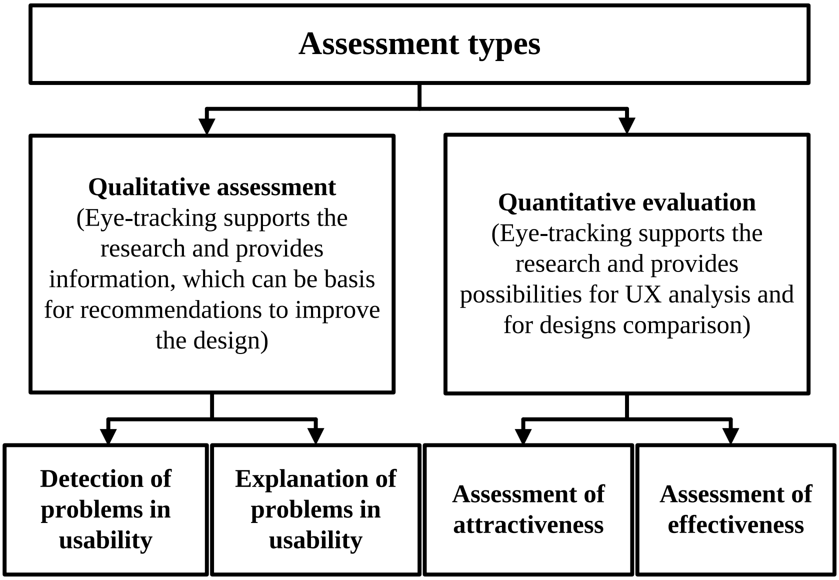

Assessment types
Qualitative assessment (Eye-tracking supports the research and provides information, which can be basis for recommendations to improve the design)
Quantitative evaluation (Eye-tracking supports the research and provides possibilities for UX analysis and for designs comparison)
Detection of problems in usability
Explanation of problems in usability
Assessment of attractiveness
Assessment of effectiveness

Figure 39.11 – The assessment types

There are two fundamentally different approaches for using all advantages of eye-tracking while testing:

– the first approach is called concurrent verbal protocol, thing is that researcher can look at respondents’ eyes movements online and ask them questions during the research process [7, 8];

– the second approach is called retrospective verbal protocol. Unlike the traditional concurrent verbal protocol, respondents are asked to give a verbal report that they make after running the script, which was carried out in silence [7, 8].

Eye-tracking sometimes makes it possible to detect problems when conventional test methods failed to cope with the task (such as the observation of the behaviour of the respondent) and respondents do not indicate that there is a problem. For example, the respondent can almost immediately click on the desired (correct) link, but before clicking on the link can turn his attention to other links or interface elements. Such behaviour of the respondent may indicate about a certain confusion while making the decision. This problem cannot always be diagnosed by standard methods of the study.

The problem of practical application of eye-tracking

Eye-tracking is often used to detect «invisible» problems and also to explain the essence of problems that were discovered by standard methods. For example, the study showed that respondents do not click on necessary buttons or links and it is unclear the reason for such behaviour. To explain this problem, you can apply eye-tracking for a more detailed study of the localized problem. The results of this study may give an answer to the posed questions. Such form of eye-tracking should be used when there is a limitation in time and there is lack of answers to the strange behaviour of respondents, which will be followed-up questions, on which is also difficult to answer.

Example of the questions on which eye-tracking can help to answer:

1. Why respondents are making wrong actions?

If you imagine that a study was conducted on the web site. It was discovered that only a small percentage of respondents chose the link that allowed them to successfully complete a specific task. This could occur because it was hard to find the correct link or because the link was poor and it was difficult to associate it with the task. For a more detailed study of the problem in this case, you should apply eye-tracking.

If respondent never looked at the correct link, the researcher can explain this problem of poor visibility of link. Depending on the nature of the sight and behavioural information, you can narrow down the cause of poor link position , its low importance or other distractions, even to the general confusion of elements on the website. But if the correct link was seen (but not selected), the problem may be in poor labelling or lack of interactivity signs in the link (link, for example, may be similar to plain text).

2. Why do correct actions of the respondent take longer than expected?

While the previous issue was devoted to the lack of design, this question is dedicated to the successful, but not effective implementation of the action. Suppose that respondents found the desired button on the printer, but the search took much longer time than expected. Eye-tracking may help to localize the problem and to explain the behaviour of respondents, offering, for example, a more detailed picture of what happened before the desired action was performed.

3. Why did respondents do not get the necessary information?

Eye-tracking is not limited to the explanation of the physical actions of the respondents, such as clicks of a computer mouse or pressing buttons. This process also includes comments of respondents. Comments can be submitted by answers to questions that are frequently asked during the test and evaluation of educational material, flyers or packaging (i.e. design, etc.). Eye-tracking may indicate reasons for cases when interfaces are unable to transmit full required information to respondents.

39.3 Eye-tracker types and technical characteristics

After it was found that in research it is advisable to use the eye-tracker, it is necessary to determine the type of eye-tracker that will be used in research.

Eye-tracker types

The first task when choosing the eye-tracker is to determine the goals of the proposed research.

There are 2 types of eye-trackers: a head-mounted eye-trackers (Fig. 39.12) and remote eye-trackers (Fig. 39.13). The main difference between them is the location of the equipment. Head-mounted are put on the head of the respondent. Remote eye-trackers are contactless devices and are located at a fixed location in front of the respondent. The Table 39.2 [6] provides a comparison of eye-trackers in part of their application.

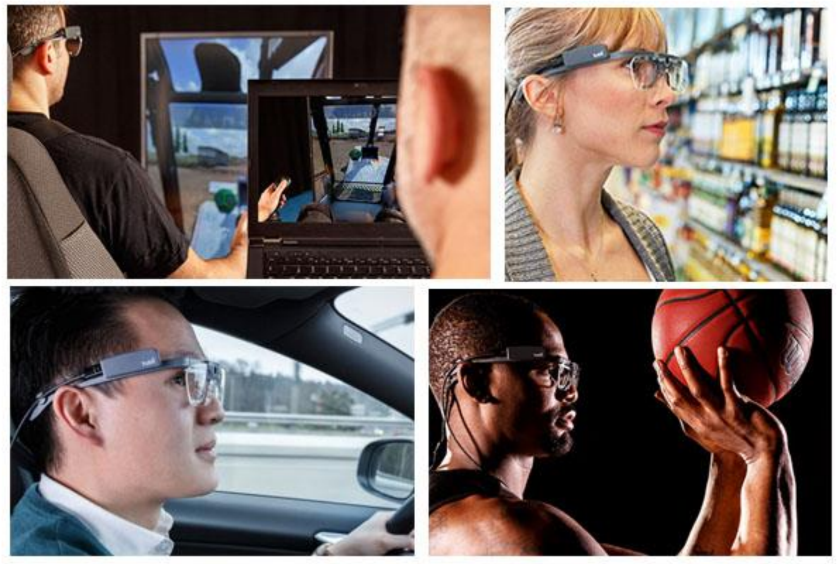


Figure 39.12 – Examples of head-mounted eye-trackers

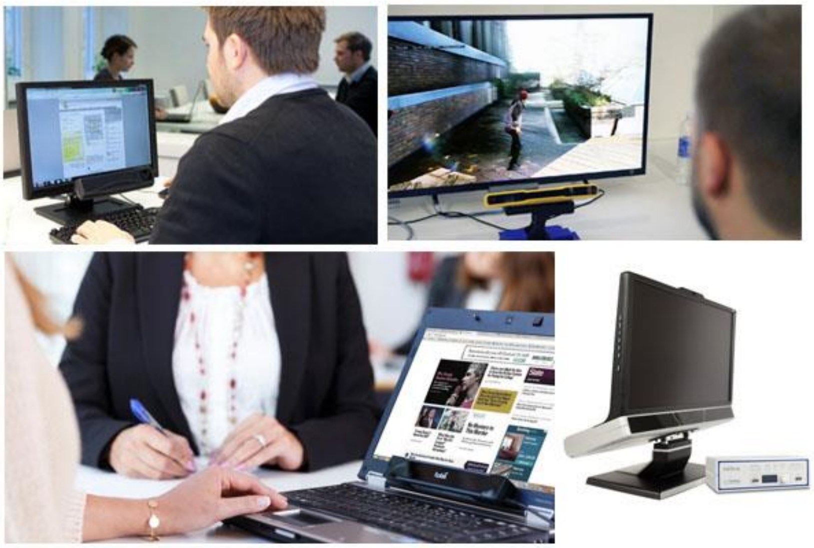


Figure 39.13 – Examples of remote eye-trackers

Table 39.2 - Comparison of the two main types of trackers [6]

	Remote eye-trackers	**Head-mounted eye-trackers**
Placement	Placed in a fixed location in front of the respondent (e.g., on a table or dashboard of a car)	Worn on the head of the respondent (e.g., as a pair of glasses attached to a hat or fasteners on the head)
Application	Applied in studies in which respondents can sit or stand at one place and the object of a study are presented on a fixed surface (e.g., research using on-screen stimuli, such as websites, images and videos)	Applied in studies that require respondents movement and interaction with physical objects or people (e.g., orienteering, shopping or innovative research)

	Remote eye-trackers	**Head-mounted eye-trackers**
Obtrusiveness	Less obtrusive than head-mounted eye-trackers. Respondents can easily forget about the eye-tracker during the research	More obtrusive, because equipment must be worn on the head. Typically, equipment visible to respondents (although respondents can get used to the equipment over time). In addition, head-mounted equipment may be undesirable due to health issues
Freedom of movement	Respondent must be in front of the eye-tracker. Only limited movements of the head are allowed. Also, there should be no obstructions between the eye-tracker and the respondent (for example, respondent can't hold anything in front of face)	Respondent can move freely and manipulate the objects. However, most of these eye-trackers work better for objects that are at the same distance for which they are calibrated and less accurate for objects that are closer or out of the borders of the calibration. This effect is called parallax error (bias) and only some eye-trackers can correct this effect
The simplicity of the analysis	Since the recorded scene does not change when the head is moving, so the view location on the recorded shots indicates that the object of study remains unchanged. Data analysis, usually implemented faster and easier because it can be automated. Note that this is possible only in the case of static stimuli. If visual content is dynamic (e.g., video or web site with large number of overdubs), the lack of a stable coordinate system can make the analysis less automated or manual, and also time-consuming	Recorded scene is constantly changing due to movements of the head. As a result, a fixed position of the glance at the recorded shots and in the surrounding world may be different. Because of this discrepancy, the analysis of the data, generally implemented in a manual form and it is time-consuming. This problem can be at least partially solved by placing infrared cameras in the environment using edge detection of the object, or by adding the main eye tracker.

Types of remote eye-trackers

Remote eye-trackers available in two versions: eye-tracker that is integrated in monitor and eye-tracker (Fig. 39.14), which is autonomous (Fig. 39.15).

Autonomous remote eye-trackers can be used to track physical objects (e. g. magazines, newspapers, products or even entire shelves) because these objects and the respondent are stationary. They also support the analysis of eyes movements on television, when, for example, respondents play games, interact with a menu on the screen or watch videos. Some of these devices they may be installed in vehicles for tracking, focus drivers. For example, to determine how they respond to changes in the instrument panel or that they pay attention to on the road. It should be noted that autonomous eye-trackers can be applied to study the interaction of respondents with mobile devices (Fig. 39.16).

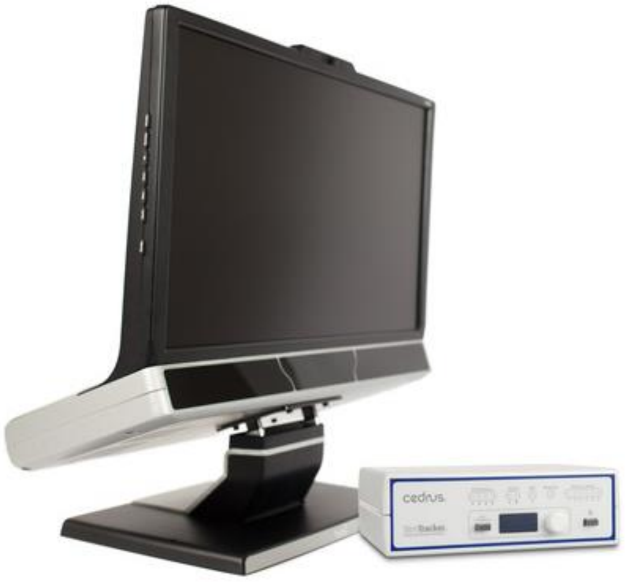


Figure 39.14 – Eye-tracker that is integrated into the monitor

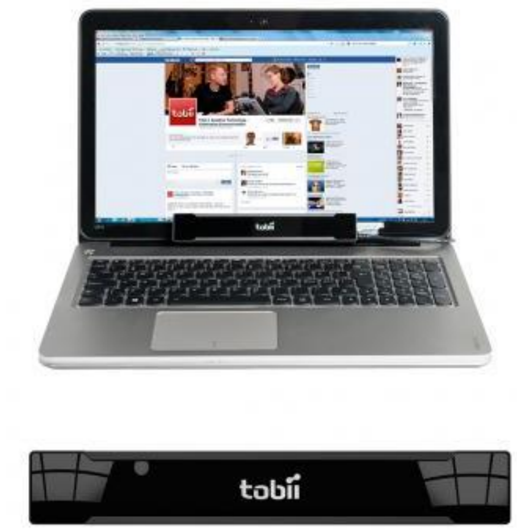


Figure 39.15 – Autonomous eye-tracker from Tobii company [9]

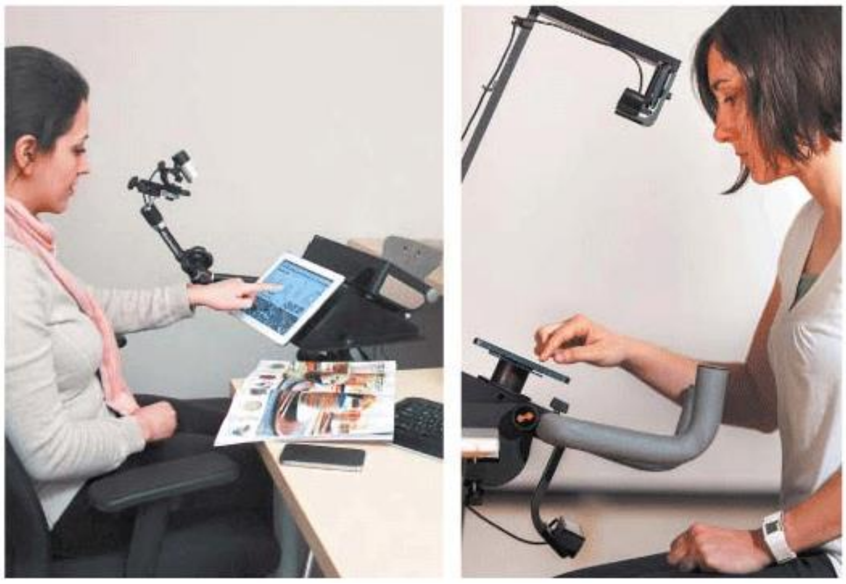


Figure 39.16 – Application for research of eye-trackers mobile interfaces

The advantage of integrated eye-tracker is its less intrusive than autonomous eye-tracker. This feature helps respondents to feel more confident.

The advantages of autonomous eye-tracker include its flexibility of use compared with the integrated eye-tracker. But, despite this advantage, autonomous eye-tracker is more difficult to install, configure and integrate with a PC or a laptop than an integrated eye-tracker.

Despite this drawback, the manufacturers of modern autonomous eye-trackers simplified the setup of their products. And such eye-trackers allow you to save profile settings for later using on other devices. Additional benefits of autonomous eye-trackers are tendency to reduce their size and weight. Among the researchers, these devices are more popular than integrated eye-trackers.

The choice between head-mounted and remote eye-trackers

Most often, researchers selected a remote portable eye-tracker. Such preference is usually due to more effective support and automated analysis of the obtained data.

On the other hand, if the purpose of research is, for example, studying of external signboards at the airport or railway station, head-mounted eye-trackers will be a better option for the research (Fig. 39.17).

Figure 39.17 – Internal airport surroundings

Another example of a better application of head-mounted eye-tracker is to study the interaction of the user and the product. Such research may include following processes: product packaging, opening the box, taking out the product and usage it in accordance with the instructions (Fig. 39.18). Remote eye-tracker can’t be applied in carrying out such research, because respondents have to take and move objects. Herewith, in this case objects can block the space between the eyes of the respondent and remote eye-tracker.

In some situations, the choice of the most suitable type of eye-tracker can not be so obvious. For example, if you want to explore the attention of the operator of nuclear power plants or to compare several variants of control system software interfaces at nuclear power plants can be used both types, head-mounted and remote eye-trackers. Using head-mounted eye-tracker can conduct research directly interfaces with software control systems at nuclear power plants and remote eye-tracker can be used with the monitor screens, which displays examples of the same software management systems interfaces.

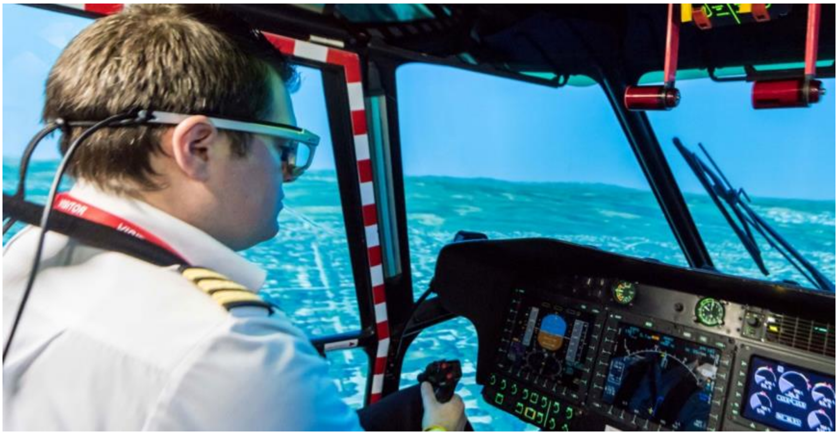


Figure 39.18 – Research using head-mounted eye-tracker

In the case when in research can be applied any eye-tracker, selection of the type of eye-tracker tends to a compromise between the real situation of studies on the one hand and simple (by automation), flexibility, processing of the received information, scientific support (including or excluding the necessary metrics studies) and emulation of environment (option with the picture on the monitor and remote eye-tracker) on the other.

In the example of research with software interfaces of control systems at nuclear power plants with using head-mounted eye-tracker provides real environment (or the actual conditions) for research. The disadvantage of such studies is the complexity of the control of interests of respondents to the interfaces, as close may be third-party stimuli (color interface elements or extraneous movement of workers), which can distract respondents and therefore influence the findings.

In another example, respondents are asked to participate in the research with interfaces of software control systems in nuclear power plants using remote eye-tracker. The reality of the situation is reduced, but it is possible to control the sequence of presentation options for different interfaces.

Research, for example, paper charts or instructions may be implemented using both head-mounted and the remote eye-tracker.

In addition to understanding the compromising associated with the choice of types eye-trackers, it is important to keep in mind that eye-trackers of the same type may be slightly different in their capabilities and limitations. For example, if the research is conducted in the automotive security domain, when we study the behaviour of the driver behind the wheel and how he uses the dashboard of the car, some remote eye-trackers will not work because they are too large and inconvenient to install in the car. However, there are remote eye-trackers that have been specifically designed for research in the car.

The situation is similar with head-mounted eye-trackers. For example, if the head-mounted eye-tracker applied to the study, the interaction of physicians with medical software, which is installed on smart phones in hospitals, it may be the following situation: as a camera that records the survey respondents and fixed to the frame eye-tracker points and can not be tilted down, it may distort the data. For this type of research, it should be used eye-tracker, which has the ability to tilt the camera down or capture a wider vertical view sides.

Technical characteristics

After eye-tracker type selection has been made, must compare their the key characteristics within the same type. Consider some of the most key eye-trackers characteristics.

Sampling frequency

The sampling rate is one of the most important characteristics of eye-trackers. Measured in hertz (Hz), is the number of times that eye-tracker registers the location of human eyes per second. This means that every second 120 Hz eye-tracker collects 120 data points for each of the monitored eye. This is more than 200 thousands of data points recorded during the 30 minutes' session tracking. Next, these points handled by software, i.e. they are combined, analysed and optimized. The researcher has been working with the transformed data.

Sampling frequency range currently available is quite broad – from 25 Hz to 2000 Hz. The higher sampling rate, the more accurately one can measure, when the fixing and, respectively, less likelihood of errors in fixation duration. Duration of fixation is measured by eye-tracker 25-30 Hz may have an error of +/- 20 ms, while the duration of fixation measured by eye-tracker 250 Hz, can have an accuracy of only +/- 2 ms [6]. Therefore, if researchers are interested in measuring not only the direction of sight of people, but also the accuracy of the time during which the respondents are looking at the object of research, they should strive to use eye-trackers with a higher sampling rate.

Given the fact that eye-trackers with a higher sampling rate are more expensive, the general rule is to use for research eye-trackers with a frequency of 50-120 Hz. These will produce a sampling frequency error fixation duration +/- 10 ms or less. Given that a typical fixation lasts from 100 to 500 ms (half a second), the error of this magnitude is generally acceptable for research.

Eye-trackers with a sampling frequency of 250 Hz or more are commonly used in studies measuring the characteristics of saccades (e. g. their speed), as well as micro-movements of eyes, which may be of interest to other areas such as neurology.

Accuracy and precision

Accuracy of eye-tracker – the average value of the difference between what eye-tracker caught and recorded (position view), and what is the view of the position it really is. Researchers should strive to reduce this error, but it will never be equal to zero.

Accuracy is measured in degrees visual angle (Fig. 39.19). Typical values fall in the thoroughness of the tracking range of 0.5 to 1 degree. This means that one degree corresponds to a little less than half an inch (1.3 cm) on the computer monitor, which is located at a distance of 27 inches (68.6 cm).

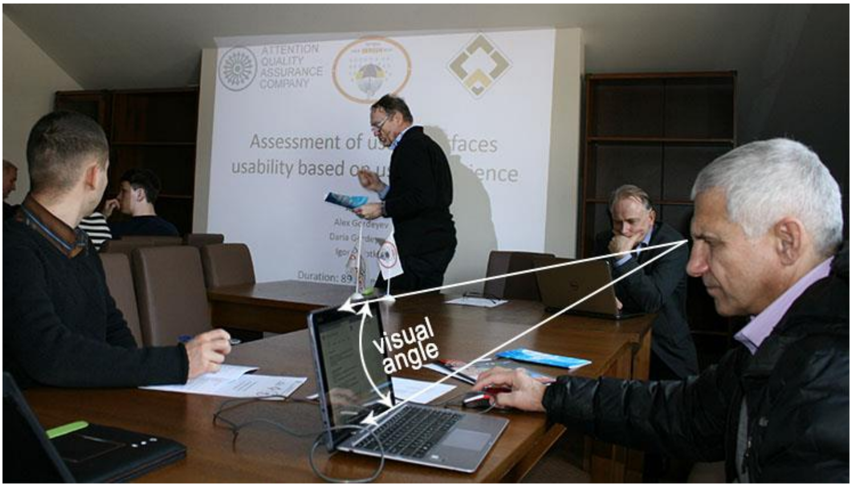
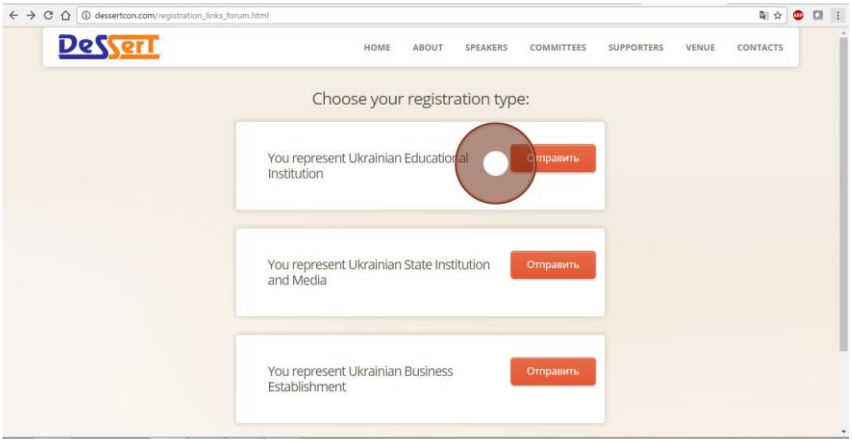

visual angle
ATTENTION QUALITY ASSURANCE COMPANY
Assessment of us... faces usability based o... ...ience

Figure 39.19 – Angle of survey respondents

In other words, the actual location of the respondent's view could be anywhere within a radius of 0.47 from the location recorded by eye-tracker view with one degree of care (Fig. 39.20). The value «Big point» (Fig. 39.20) indicates the location of the view recorded by eye-tracker. Assuming that care eye-tracker 1 degree, with screen resolution of 1680 x 1050 pixels and a viewing distance of 27, the actual location of the respondent's view can be anywhere within the designated range, designated around the value white point.



Choose your registration type:

You represent Ukrainian Educational Institution

You represent Ukrainian State Institution and Media

You represent Ukrainian Business Establishment

Figure 39.20 – Accuracy thoroughness of eye-tracker

The value of accuracy given in textbooks eye-trackers measured under ideal conditions, which usually include testing respondents without corrective glasses, and measured immediately after calibration. During the "real research", the difference between recorded and actual provisions of the view may be greater for respondents who wear glasses or contact lenses, or those who have changed their position before eye-tracker in accordance with the calibration procedure.

Precision is a measure of how well eye-tracker able to measure. Ideally, if the respondent's attention several times is in the same places, eye-tracker has to determine the location of these two as identical. It would be a perfect precision. In fact, precision in the range of 0.01 to 1 degree. These values are calculated as the mean square distance (in degrees viewing angle) between successive samples. Since the values of precision, the manufacturer, is estimated using a fixed artificial eye, tracking the real eye of respondents will show less accurate.

Accuracy and precision is not one and the same index, but they are often confused with each other. Fig. 39.21 [6]. It explains the difference between them. Cross in each cell of the table shows the current location of view, while the point of view of the location are reported by eye-trackers.

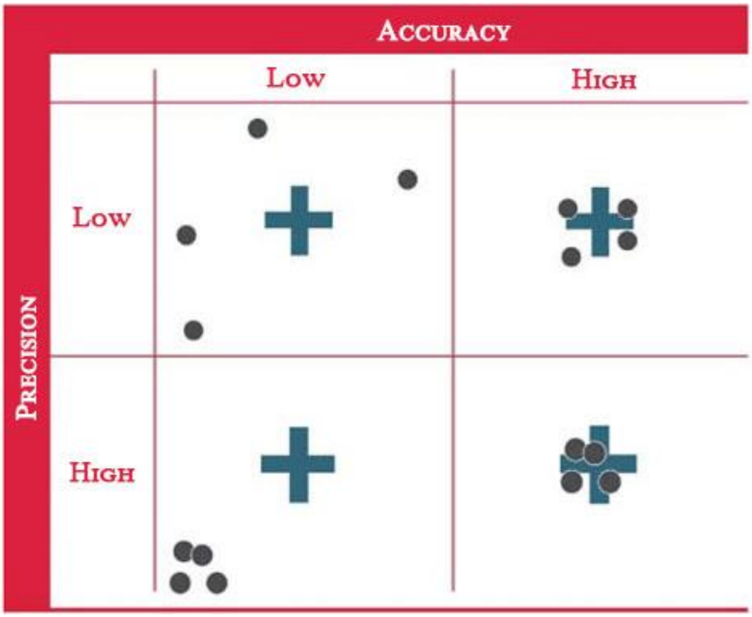


Figure 39.21 – The differences between the accuracy and precision of eye-tracker

The size of the head box

The head box is a notional area that defines boundaries of freedom of movement for the respondent's head of research eye-tracker (Fig. 39.22). While the respondent's head remains in the imaginary field, eye-tracker can track the position of the respondent's eyes.

People can not be completely fixed during eye-tracking sessions. During the research generally the movement of the head of the respondents are not limited by, for example, the chin holder or other similar devices. Therefore, the more the head box, the less data will be lost because of the movement of the head. The size of the head box depends on the eye-tracker, but typical ranges are 12-17 inches (~ 30 to 44 cm) wide, 7 to 9 inches (~ 17 to 23 cm) in height and 8 to 12 inches (~ 20 to 30 cm) in depth.

Head-mounted eye-trackers have no restrictions related to the head box. Since eye-tracker moves with the head of the respondent, restrictions on the movement of the head does not exist.

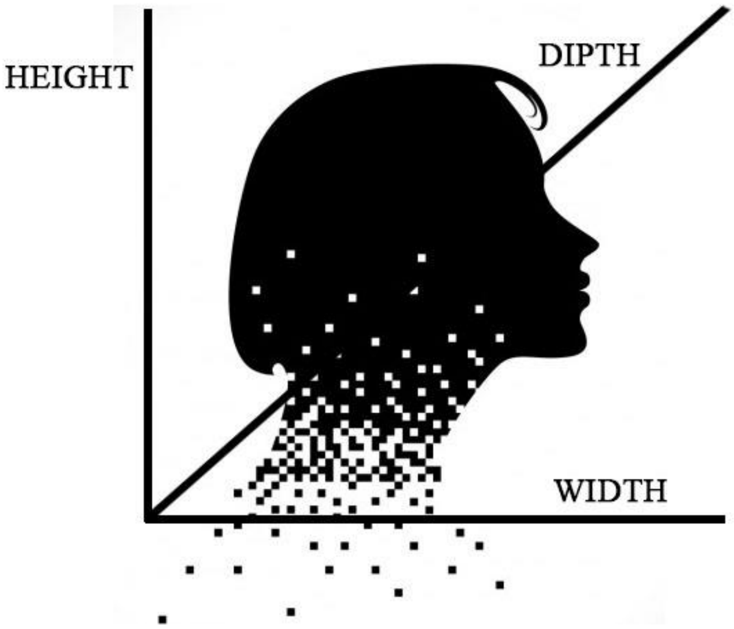


Figure 39.22– Eye-tracker's head box with indication of the allowable range of movements of the respondent's head

Advantages and disadvantages of monocular and binocular eye-tracker

Monocular eye-trackers record movements of only one eye of the respondent, and binocular eye-trackers record movements of both eyes. Theoretically, the monocular eye-tracker should be enough, because usually both respondent's eyes tend to move together, so by analysing the position of one eye, can also determine the position of the other eye. Monocular eye-trackers usually more accessible than binocular. However, use of binocular eye-trackers more preferable for the following reasons:

– firstly, binocular tracking increases accuracy and precision due to the averaging of data from both eyes;

– secondly, if respondent's eye temporarily beyond the head box, no information is lost because there is always the data from the other eye;

– third, knowledge of both positions of eyes helps to correct parallax errors in the head-mounted eye-trackers.

Techniques of lighting pupils

Standard eye-tracker know where the respondent is looking, based on the position of the pupil centre relative to reflection from the cornea. Infrared LEDs illuminate the face of the respondent, and the eye-tracker camera detects and records the characteristics of the two eyes.

Depending on where the diodes are arranged relative to the cameras, two different methods in order to illuminate the eyes of the respondent may be used:

– bright pupil method;
– dark pupil method (Table 39.3).

Table 39.3 - Comparison of bright and dark pupils' methods

	Bright pupil method	**Dark pupil method**
How it works	When the infrared light source is on one path with an optical axis of the camera, the iris pupil becomes brighter, as reflected from the retina is directed towards the camera. (This phenomenon is also called the red-eye effect in photos)	When the infrared light source is directed away from the camera, the pupil appears darker than the iris because the reflection from the retina is directed away from the camera
Effect of lighting conditions	This method works best in dark conditions (when the pupil is large). It is not suitable for tracking outdoors	This method works well in normal room conditions, under normal lighting conditions and even outdoors under the natural light
The impact properties of the eye	Due to the fact that the bright region is determined as an elliptical pupil, there is interference by dark elements, such as shadows or eyelashes. This method works best with bright (blue) eyes than with dark (brown) eyes	Since the dark elliptical region is defined as a pupil, and other dark elements (such as the eyelashes, especially covered with ink) can interfere with the detection of the pupil. This method works best with dark (brown) eyes than the bright (blue) eyes

Both methods have the same goal – to create maximum contrast between the pupil and the iris (coloured part of the eye) to let the pupil be easily distinguishable. However, the efficiency of each method depends on the conditions under which it is used and the physical characteristics of the respondent's eye. Most of eye-trackers support one of the methods, but some eye-trackers support both methods and automatically select the one that provides better care in this situation.

Conclusion and self-control questions

Eye-tracking is a process of measuring the characteristics of the eyes and their movements. Eye-tracking usually supported by the device, which is called eye-tracker. Most commercial eye-trackers work, based on highlighting the human retina with infrared light (or an approximation to it) to determine the location of view on the basis of the relative position of the centre of the pupil and corneal reflection. Human eyes (i.e., pupils) are moved from place to place several times a second. The aim of these movements, also known as saccades, is to bring visual stimulation in the pit (small area, which has the highest visual acuity in the retina). Information extracted from the records, which are short pauses between saccades, that is fixations. Focus of vision covers only two-degree field of view. The farther away from the fovea, the image becomes more blurred and colourless. Even if eye-tracker captures only the focus (e. g., what we're looking directly), it contains useful information about visual attention, because, in most cases, fixing coincides with attention. Saccadic directions are selected based on a combination of ascending and descending cognitive processes. In other words, users are looking at those objects that want to watch in accordance with the goals, experiences and expectations.

When choosing eye-tracker the researcher should determine the type of eye-tracker. Remote eye-tracker is suitable for studies in which respondents will have to sit or stand relatively still and stimulus presented on a stationary surface. Head-mounted device should be used when the respondents move or interact with physical objects.

The technical characteristics eye-trackers include the following:

1. Sampling rate – the rate at which records the location of eye-tracker's view (the higher the better, but 120 Hz is not necessary for the research);
2. Care - the difference between the actual location of the view and the location recorded by eye-tracker (tends to 0,5 ° or less);
3. Accuracy – a measure of how well eye-tracker able to reproduce the measurement (seek to 0,3 ° or less);
4. The size of the head box - freedom of movement of the head which is allowed by remote eye-trackers (the more, the better);

5. Monocular or binocular eye-tracker – record from one eye (monocular) or two respondents' eyes (binocular). Binocular eye-trackers compared with monocular eye-tracker increases accuracy and reduces data loss;

6. Pupil illumination method – a method that identifies the pupil of the human eye. Tracking Method bright pupil is less susceptible to interference from other elements, but dark pupil tracking works in a wider range of lighting conditions. The method of tracking a bright pupil works best with blue eyes and dark pupil tracking method works best with brown eyes;

Self-control questions and tasks

1. To describe of eye-tracking conception.
2. How moves of the human's eyes?
3. What assessment types used for research with eye-trackers?
4. What is visual attention?
5. What kind of technical characteristics connect with eye-tracker?
6. What is remote eye-tracker?
7. What is head-mounted eye-tracker?
8. What is head box for eye-tracker?

References

1. Kootstra, G., de Boer, B. & Schomaker, L.R.B. Predicting Eye Fixations on Complex Visual Stimuli Using Local Symmetry. Volume 3, Issue 1, Cognitive Computation Journal. Springer US. pp. 223–240.
2. Ali Bulent Usakli and Serkan Gurkan, «Design of a Novel Efficient HumanComputer Interface: An Electrooculogram Based Virtual Keyboard» IEEE Transaction on Instrumentation and Measurement, Vol. 59, N0.8, August 2010, pp. 2099-2108.
3. N. H. Cuong, and H.T. Hoang, «Eye Gaze Detection with a Single WebCAM Based on Geometry Features Extraction», 2010 11th International Conference on Control, Automation, Robotics and vision, Singapore, 7-10th December, 2010, pp. 2507-2512.
4. T. Moravcik, «An Approach to Iris and Pupil Detection in Eye Image», 12 International PhD Workshop OWD 2012, 23-26 October 2010, pp. 239-242.
5. Web site of Cyber forum - Dependable systems services and technologies «DeSSerT» (http://dessertcon.com/about_forum.html).
6. Aga Bojko. Eye Tracking the User Experience: A Practical Guide to Research. Rosenfeld Media, 2013, 320 p.

7. Jennifer Romano Bergstrom, Andrew Schall. Eye Tracking in User Experience Design. Morgan Kaufmann, 2014, 400 p.

8. William Albert, Thomas Tullis. Measuring the User Experience, Second Edition: Collecting, Analyzing, and Presenting Usability Metrics (Interactive Technologies). Morgan Kaufmann, 2013, 320 p.

9. Web site of Tobii company (http://tobii.com).

Acronyms

UX – user experience;
UI – user interface.

40 EYE-TRACKING METRICS AND VISUALIZATION TYPES

40.1 Metrics taxonomy

Eye tracking metrics are a method of calculating the necessary quantitative data on the basis of primitives (saccades, fixations and mouse clicks). They enable researchers to calculate the metric value for the study of one object and compare it to the value of the same metrics computed for another research facility. Eye tracking metrics usually help to carry out a more detailed measurement, meaning that they are not only based on the target, but also are problems for the individual components.

Note that one of the important tasks is to create the necessary nomenclature metrics for research using eye tracking and taking into account peculiarities of the research object.

Each researcher before the research makes a decision about what metrics will be used for in the study and to what questions he should receive answers from the respondents to obtain the necessary information from them to calculate metrics.

Types of eye tracking metrics

There are over hundred eye tracking metrics and combinations. Some of them are simple enough to calculate and to interpret the results. Examples of such metrics are: total time that respondent spent considering an object or a number of respondents noted and considered the object of study [1].

Materials of this module are devoted solely to individual subcategories of eye tracking metrics that are most important and most often used in research. Moreover, the original data for the calculation can be easily obtained from the investigation results.

The same metrics may represent various cognitive effects in the context of different stimuluses and purposes. For example, what it could mean for researches when the respondent looks at the subject of the research for a long time. On the one hand, this may mean that the property is interesting for the respondent and thus attracts his attention, on the other hand, it may be a situation where the respondent has difficulty in understanding or solving a problem.

A very important issue is the question of the results interpretation and calculation of metrics. Consider a few examples of interpretation of the metric numbers committed:

- Example 1. When the respondent is trying to find the link «contacts» on the web page, the number of such fixations targeted is a metric search efficiency. The more fixations are required to find the right link, the less effective it is to find as the placing links do not meet the expectations of the respondent;

– Example 2. When viewing online photo albums, the number of fixations in the photo shows the level of interest in it. Photos that are more interesting, cause more fixations;

– Example 3. When registering eye tracker makes greater number of fixations for specific registration form elements than for other ones in the same registration form. As a rule, such a result is indicative for the problem of understanding of the elements on the registration form.

In [1] work there have been four main types of eye tracking metrics identified: (1) the movement of the metrics (2) The provisions of the metrics (3) the number of metrics and (4) the metrics of latency and distance. Since this course is primarily a practical character, we will group eye-tracking metrics according to the types of research facilities for which they are designed (Fig. 40.1). This classification is focused primarily on those metrics that are often used in practical studies. We will consider these metrics group. They are attached to the two main issues of the study of the user experience: (1) issues related to attracting attention and (2) issues relating to the performance.

According to it, metrics for attracting attention can respond, for example, to the following questions:

– what elements of the software interface attract attention;

– what software interface elements would distract operators;

– due to what shortcomings of the software interfaces, the operator makes mistakes.

Performance metrics, for example, answer the following questions:

– whether the updated design simplifies the layout of a Web page search and information comparison;

– how quickly and accurately the software provider for the NPP shall works in critical situations;

– if a new GPS navigator simplifies driver’s work on the road comparing with the existing system.

40.2 Metrics attractiveness

There are three subgroups of metrics attractiveness: (1.1) visibility metrics (1.2) area of interests (AOI) metrics (1.3) emotional arousal metrics. Consider these metrics group in greater detail:

– (1.1) visibility metrics is focused on the identification of objects, how easy it is to notice anything;

– (1.2) AOI metrics are focused on evaluation of the object of interest, as soon as he had seen;

– (1.3) emotional arousal metrics focus on the evaluation of ideas and the desirability of the object. They are often used for evaluating the

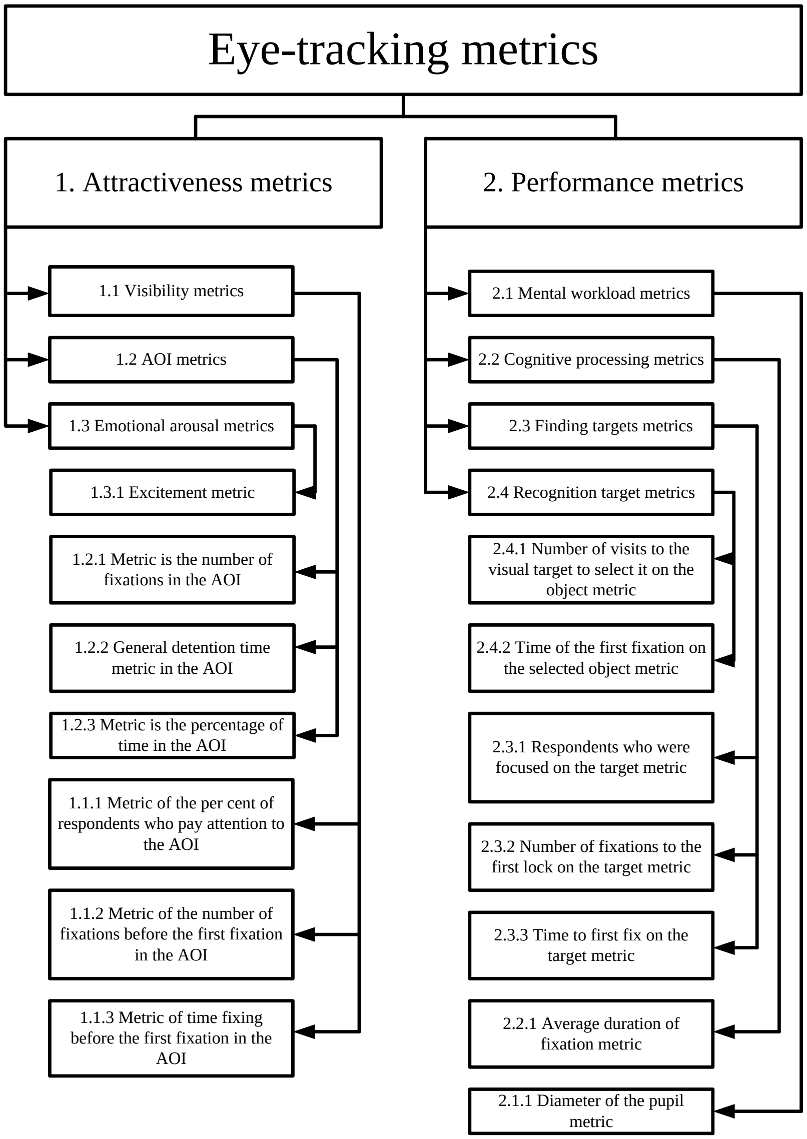

Eye-tracking metrics
1. Attractiveness metrics
2. Performance metrics
1.1 Visibility metrics
1.2 AOI metrics
1.3 Emotional arousal metrics
1.3.1 Excitement metric
1.2.1 Metric is the number of fixations in the AOI
1.2.2 General detention time metric in the AOI
1.2.3 Metric is the percentage of time in the AOI
1.1.1 Metric of the per cent of respondents who pay attention to the AOI
1.1.2 Metric of the number of fixations before the first fixation in the AOI
1.1.3 Metric of time fixing before the first fixation in the AOI
2.1 Mental workload metrics
2.2 Cognitive processing metrics
2.3 Finding targets metrics
2.4 Recognition target metrics
2.4.1 Number of visits to the visual target to select it on the object metric
2.4.2 Time of the first fixation on the selected object metric
2.3.1 Respondents who were focused on the target metric
2.3.2 Number of fixations to the first lock on the target metric
2.3.3 Time to first fix on the target metric
2.2.1 Average duration of fixation metric
2.1.1 Diameter of the pupil metric

Figure 40.1 – Eye-tracking metrics taxonomy

– effectiveness of advertising on the web page, product packaging and on any other object that should attract the attention of not specialized

– for this effort.

Visibility metrics

Visibility metrics are used to estimate the area or object of visibility. The information on the number of respondents who noticed the desired object, and how quickly they notice an object is calculated with the metrics of visibility. These metrics are dependent on the visibility of the visual field, the size of the area, its layout or graphic representation. For example, larger areas are more prominent than the area with smaller space. Moreover, parts of an object, which are very different from other irritants (e.g., within the color and shape) is easier to see than those portions which merge with the general background of the object (Fig. 40.2).

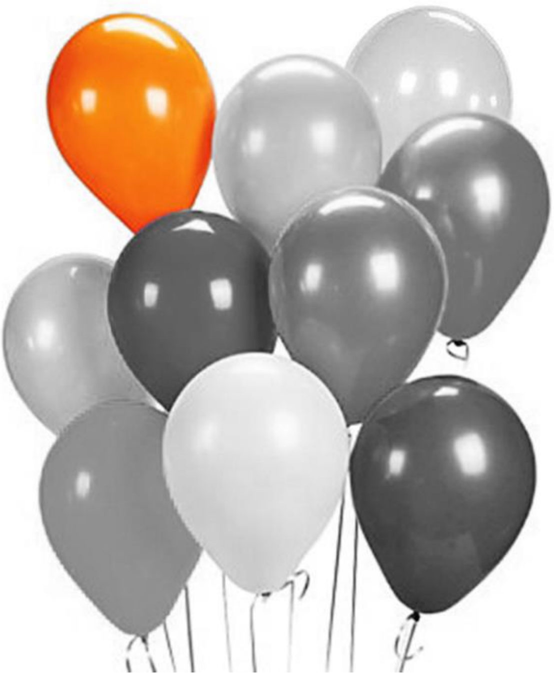


Figure 40.2 – More conspicuous objects that contrast with the background

Another factor that affects the visibility of the field of attention, in addition to visual attention, is the awareness of the importance of the area. Less evolved elements can attract the attention of the experienced user and remind that certain areas of attention contain important information. For example, if the user knows that the navigation system is at the top, it is, first of all there and will look even if it is invisible.

Visibility metrics include the following:

– (1.1.1) metric of the per cent of respondents who pay attention to the AOI. This informative metric that identifies interested respondents. To calculate this metric, it is enough to define the criteria for fixing (the area of interest), and then determine the respondents who at least once drew attention to the area of interest;

– (1.1.2) metric of the number of fixations before the first fixation in the AOI. To assess the visibility the number of commits to the stimulus (web page) to the very first fixed in a AOI (for example, banner) can be measured. For example, the value of this metric will install the following:

– a number of first fixations of respondents in the section of the object of study;

– a number of first fixations in the context of respondents account for third-party sites;

– (1.1.3) metric of time fixing before the first fixation in the AOI. This metric should be used when you want to determine how quickly the area of interest was recognized. It may seem that this metric repeats previous metric of number of commits to the first fixation in the area of interest, but it is not. For example, a stakeholder (for example, customer research) fixing the time value is more understandable metric, in contrast to the number of commits. Interested parties are easier to understand when the researcher, for example, declares that "prior to paying attention to advertising a new product the respondent spent 15 seconds after the opening a web browser the corresponding web page," than, for example, if a researcher, declare that the respondent did 20 commits. When it comes to time, researchers do not need to explain what is fixation and used it in research information.

Note that the last two values of the metrics should be used in conjunction with the percentage of respondents and the fixation recorded in the area of interest. The value of metrics is more important, if the area of interest has been noticed a large number of respondents.

AOI metrics

The aim of the design and human-computer website interaction is to make them simple, understandable and memorable. This combination is not possible in the case where the elements of design attract the attention quickly (as visibility metrics), but can not keep the user’s attention due to lack of information content and relevance. The example on Fig.40.3 illustrates the difference between the concepts of visibility and interest (the area of attention AOI1 only attracts the attention of the respondent - the visibility and attention of AOI2 region is an area of real interest).

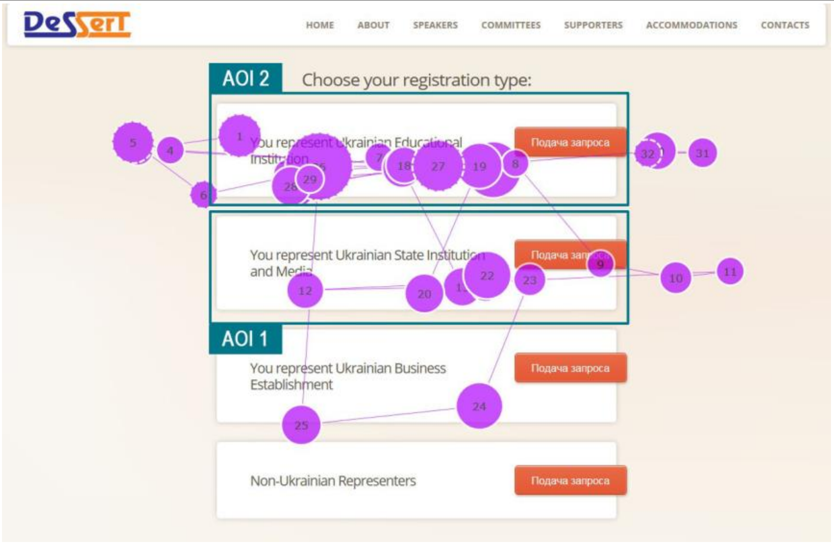

DeSSerT
HOME
ABOUT
SPEAKERS
COMMITTEES
SUPPORTERS
ACCOMMODATIONS
CONTACTS
AOI 2
Choose your registration type:
You represent Ukrainian Educational Institution
Подача запроса
You represent Ukrainian State Institution and Media
AOI 1
You represent Ukrainian Business Establishment
Non-Ukrainian Representers

Figure 40.3 – Illustration of the differences between interest and visibility

The group of interest metrics includes the following:

– (1.2.1) metric is the number of fixations in the AOI. To get the value of such metrics is not difficult, because you need only to count the number of fixations in the area of interest. For example, Fig. 40.3. It illustrates that the respondents showed a greater interest in the field of attention of AOI 2 (12 commits) rather than to the field of attention AOI1 (6 fixation);

– (1.2.2) general detention time metric in the AOI. The general detention time in the area of interest is the time from the beginning of the work with the site until the end with counting of all visitors.

It is important to remember that the general detention time depends not only on the number of fixations, but also on their length. Duration of the fixation may also be related to difficulties in processing. However, the researcher uses the detention time as a metric of interest, it should, first of all, ensure that regions of interest compared between products do not differ in mean duration produced within fixations;

– (1.2.3) metric is the percentage of time in the AOI. To calculate this metric, the researcher needs to know the total time of attention in the area of interest (home page area), and the total time of attention to the stimulus (home page) during the entire operation on a web page.

This metric, for example, may allow the researcher to draw the following conclusions «On the total number of respondents looking at Cyberforum website [2], 15% paid attention to the message that the Cyberforum terms of registration for participants have been changed». Such information makes more sense than the simple phrase «average amount of time spent by respondents on Cyberforum viewing a web page is - 20 seconds». But more complete interpretation of the information obtained may be in the next phrase, «Out of the total average amount of time spent by respondents to view the web page- Cyberforum (20 seconds) they spent 5% (1 second)on viewing the reports about rescheduling Cyberforum terms of registration for the participants».

When calculating the metrics of interest, first of all, we should define the plurality of input data. Metric calculation can be performed for all respondents who participated in the study, regardless of whether they were looking in the area of interest. Or for the calculations can be used data from only those respondents who turned their attention to the region of interest.

To improve these results it is recommended to determine the number or percentage of respondents, who still looked in the area of interest and then determined how big was their interest.

Excitement metric

Pupil is a hole in the colored part of the eye (iris) that controls the amount of light entering the eye (Fig. 40.4).

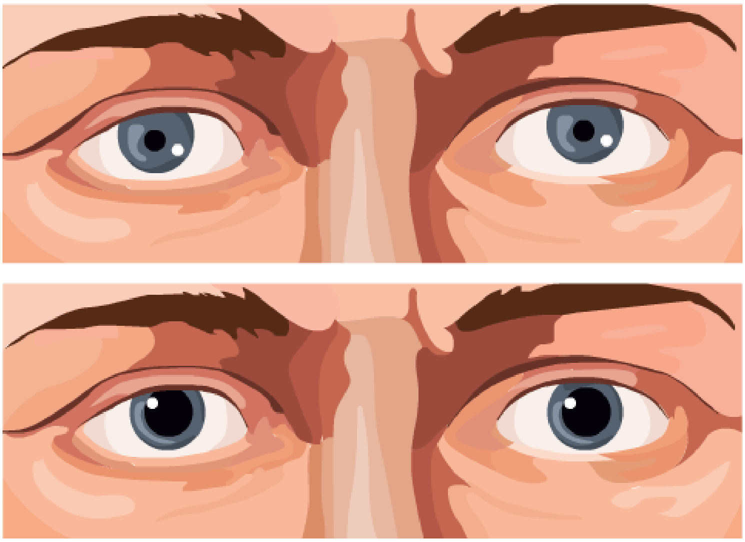


Figure 40.4 – The example of pupil narrowing and dilation depending on lighting conditions

If the light is bright, then the pupil is shrinked to reduce the amount of light that reaches the eye. In the dark the pupil increases, to enable a larger amount of light. It is known that as the pupil expands when a person meets someone who attracts him (Fig. 40.5).

Figure 40.5 – The example of pupils dilation as a result of attracting attention or excitement

Since pupils react to the emotional state the pupil size can be used as an index of excitation caused by a web site or product advertising.

40.3 Performance metrics

At the time when the appeal metrics determine the influence of interfaces and design on awareness, interest and desire of the respondent, helping to identify problems and design interfaces, performance metrics determine how well the respondents reach their own goals. Using the values of performance metrics, a researcher has the opportunity to evaluate the usability of the interface and design appeal, including how much stimulus promotes mental stress, demand for information processing, design, the level of efficiency of search, etc.

Mental workload metrics

Mental load reflects the relationship between the cognitive demands that are made by the respondent and the respondent's limited cognitive resources.

The higher the respondent's mental load is, the higher the probability that the performance will be worse. For example, the situation when the driver is driving in heavy traffic in an unfamiliar area during adverse weather conditions can lead to high mental stress and reduce the quality of driving (Fig. 40.6). Very often, in research it is difficult to establish performance changes as a result of changes in mental workload. Although, it is mental workload metrics are a good way to determine the real-world performance.

Figure 40.6 – Influence of mental load on the performance of the driver of the car

The most commonly used metric is an eye tracking mental load it is the (2.1.1) diameter of the pupil, which is also able to display excitement. As it’s been already mentioned, the mapping of certain metrics in eye tracking purposes depends on the scenario and incentives which are used in the study. Changes in the size of the pupil of the respondent that slowly scans the photo will point to the excitement (if other factors are controlled). But when the same person will be asked to solve a complicated math problem, the diameter of the pupil would be the metric of mental workload.

Mental load, as a rule, is an interest in the study and design of interfaces, which are used in situations with high cognitive demand, and when the low efficiency can have critical effects, for example, transport or medical spheres. Measuring mental workload will be more suitable in the study of the car navigation system, an aircraft cockpit, the control panel at the plant, or even an application in a call center, for example, than in the study of online shopping website.

Cognitive processing metrics

One of the cognitive processing metrics is an average duration of fixation (2.2.1). The duration of fixation usually ranges between 100 ms and half a seconds, an average of 200 -250 ms to read and 280- 330 ms to view the scene. Visually, the longer fixation is usually denoted by circles (Fig. 40.7) and, as a rule, deeper processing means and information retrieval more laborious.

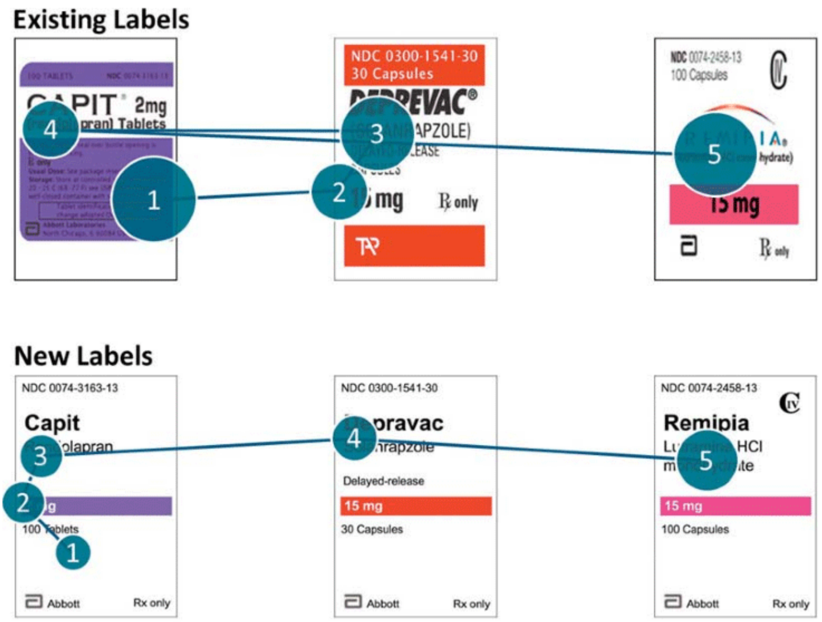


Figure 40.7 – Different lengths of fixations on the labels of medicinal products

The difficulty of such processing may come from various sources, including unusual terminology, unclear images, sophisticated concepts and high density information. For example [3], reducing the average duration of fixation on the new labels presented on Fig. 40.7, is probably due to some kind of uniformity in the formation of names, as well as the fact that some labels improved intelligibility common name, by replacing all capital letters.

The average duration of fixation is often used as the dependent variable in the comparison of stimulus with a lot of information, such as patient records, the instructions of various types of information sites or call center applications. The average duration of a fixing measure is sometimes confused with the exposure time, but the exposure time refers to the whole time spent by the respondent in the study area of interest, which also includes all of the individual duration of fixation. For example, the dwell time in the lower left corner of the label on Fig. 40.7 (new label) from time to time when the opinion was on etiquette (region of interest) until the respondent has left the region of interest (after fixing the number 3). The average duration of the label fixing on the sum of the durations of fixation # 1, # 2 and # 3 divided by three.

Finding targets metrics

To find the right link on a web page, a bottom on a device or a product on the shelf, the respondent must successfully complete two stages:

– firstly, he must determine the location of the target, showing attention to it;

– secondly, he must understand the purpose for which it is looking for the correct object, understand the essence of the object and its relation to its purpose.

Metrics are finding targets for the first phase of this process, and metrics are objective recognition to the second stage. Accumulation of metrics location and recognition goals only make sense to solve this problem, in which a specific element (s) must be selected. For example, in the problem of the online store such items or goals may include the button "add to cart" on the product page or click "Exit". For example, in studying of new variants of identity for one of the largest manufacturers of household appliances in Europe, the Turkish company Beko, the target may be a selection of the best background color and color of the letters for the name brand (Fig. 40.8).

Metrics of finding targets includes the following metrics:

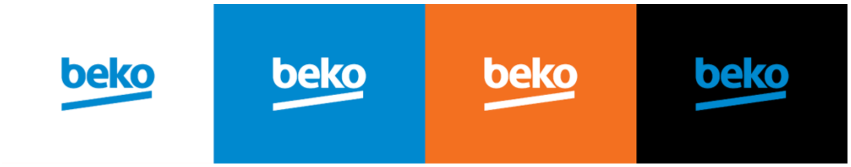


Figure 40.8 – Choice of best background and letters colors

Respondents who were focused on the target metric (2.3.1).

This metric is a metric search efficiency. Sometimes the goals that the respondents have chosen (for example, clicking on the right link) is not determined by the interest in them and the respondents such elements are not at all popular. If the majority of respondents made mistakes and they were unable to select a target (for example, clicked on the wrong link), in this case, the important task is to establish the causes that led to this result. The first step in determining the causes of such a determination of the number of respondents have been focused on the target, at least once.

If it was found that several respondents looked at the target for several times, then the problem is to find the target. The reason for this situation may be bad overall layout of the screen. Or such a cause is the result of non-optimal location of the targets or visual presentation in particular, or other elements (such as other elements can act as a distraction). On the other hand, if the majority of respondents looked at the target and did not choose it then the problem most likely is connected with the sense of purpose.

Number of fixations to the first lock on the target metric (2.3.2).

This metric is the search efficiency - fewer fixations to first fix on the target, the more effective is the search. Since respondents attention was fixed on the target more than once, the names of the metrics "to the first fixation" is an important part of the common sense definition of the metric. This metric should be used together with the percentage of respondents who do notice the goal.

Time to first fix on the target metric (2.3.3).

This metric is related to the previous metric, and is often used instead of the metric number of fixations due to the intuitive nature of the concept of time, what the concept of fixations. As with the previous metric, this metric should be used together with the percentage of respondents who noticed the purpose.

Recognition target metrics (2.4)

After an object has been found and fixed, it should be perceived by the respondent and meet objectives. Successful recognition target depends on how understandable is the object, which is associated with the maintenance target (for example, a link to a web page), and discriminating appropriately represented. Sometimes, problems with the recognition of the objectives are also associated with other links, buttons, or objects that compete with the object of attention of the respondent. For example, the remote control that appears to have a similar control function, which searches for the respondent, or products on the shelf in the store, located near the right for the buyer of goods, can prevent identify the object.

The considered group includes the following metrics:

– **number of visits to the visual target to select it on the object metric (2.4.1).** Usually easy in recognizing the goal selected at the initial recognition of it, but not after a few visits to the visual. After the respondent drew attention to the goal, he should be able to identify the object as a target. For example, on Fig. 40.7 respondents saw the goal twice before he clicked on it. The first visit consisted of a fixation (# 10), and the second visit was already two fixations (# 13 and # 14);

– **time of the first fixation on the selected object metric (2.4.2).** The time required from the start of fixation target to the moment when a respondent selects it (e.g. by clicking the mouse on it) may vary, but a shorter duration indicates a more important objective. Software that supports the analysis of the results of eye tracking allows you to get the value of a metric.

It should be mentioned that the calculation of the metric values to the number of fixations for the first fixations on targets and time to first fix on the target may not be necessary, because both measures provide the same information. But, despite this, more than usual will be the value of the time-to-first-fix for stakeholders on goal.

40.4 Visualization types of eye-tracking research results

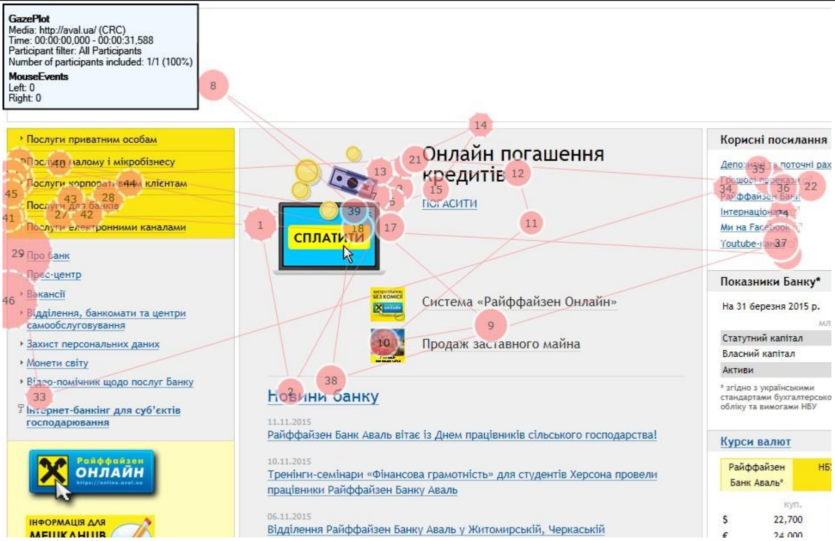
As a rule, eye-tracking research reports includes visual elements, graphics, metrics calculation values and conclusions. In the following materials will be represents of visual data types. Visual information give possibility of researchers very quickly see problems on the object of research and accordingly behave. Consequently, further researcher can calculate data only for problem places of research object. It situation give us economy of time and resources.

In general eye-tracking supports following types of results visualization: gaze plot, gaze video, heat map, opacity map, bee swarm, clusters. We will review such variants of data representation.

Gaze plot

Gaze plot – it is visual interpretation of research results. Gaze plot includes sequence of fixations (circles) and saccades (lines) (Fig. 40.9) [1]. Size of fixations is proportional of fixation duration. Than fixation is longer than size of circle is bigger. Numbering and fixation circles size depend from tuning, which installing before beginning of research. Since fixations have numerations, them transmit not only spatial information, but and time information.

As a rule, eye-tracking supports synchronous representation of all visual trajectories. Each vision motion trajectory will mark separate color. When gaze plot has high density of eye-tracking sequences (fixations) with different colors, such eye-tracking sequences very difficult discernible (Fig. 40.10) [2]. Worth pointing out, that calculation of metrics synchronously for all eye-tracking sequences does not support.


GazePlot
Media: http://aval.ua/ (CRC)
Time: 00:00:00,000 - 00:00:31,588
Participant filter: All Participants
Number of participants included: 1/1 (100%)
MouseEvents
Left: 0
Right: 0

Послуги приватним особам
Послуги малому і мікробізнесу
Послуги корпоративним клієнтам
Послуги для банків
Послуги електронними каналами
Про банк
Прес-центр
Вакансії
Відділення, банкомати та центри самообслуговування
Захист персональних даних
Монети світу
Відео-помічник щодо послуг Банку
Інтернет-банкінг для суб'єктів господарювання
Онлайн погашення кредитів
ПОГАСИТИ
СПЛАТИТИ
Система «Райффайзен Онлайн»
Продаж заставного майна
Новини банку
11.11.2015
Райффайзен Банк Аваль вітає із Днем працівників сільського господарства!
10.11.2015
Тренінги-семінари «Фінансова грамотність» для студентів Херсона провели працівники Райффайзен Банку Аваль
06.11.2015
Відділення Райффайзен Банку Аваль у Житомирській, Черкаській
Корисні посилання
Депозитні та поточні рах
Грошові перекази
Райффайзен Банк
Інтернаціонал
Ми на Facebook
Youtube-канал
Показники Банку*
На 31 березня 2015 р.
Статутний капітал
Власний капітал
Активи
Курси валют
Райффайзен Банк Аваль
НБУ
$ 22,700
Райффайзен ОНЛАЙН
ІНФОРМАЦІЯ ДЛЯ МЕШКАНЦІВ

Figure 40.9. – Gaze plots which put on Raiffeisen Bank Aval web-page [4]

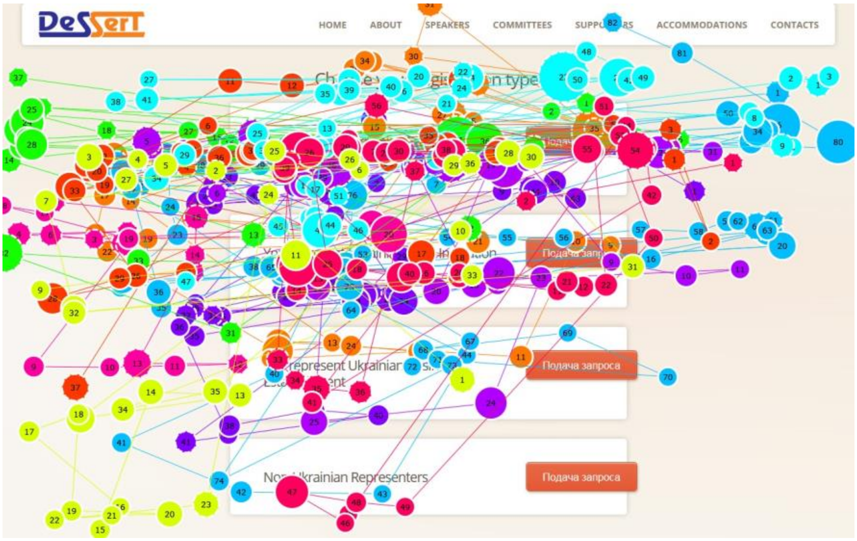
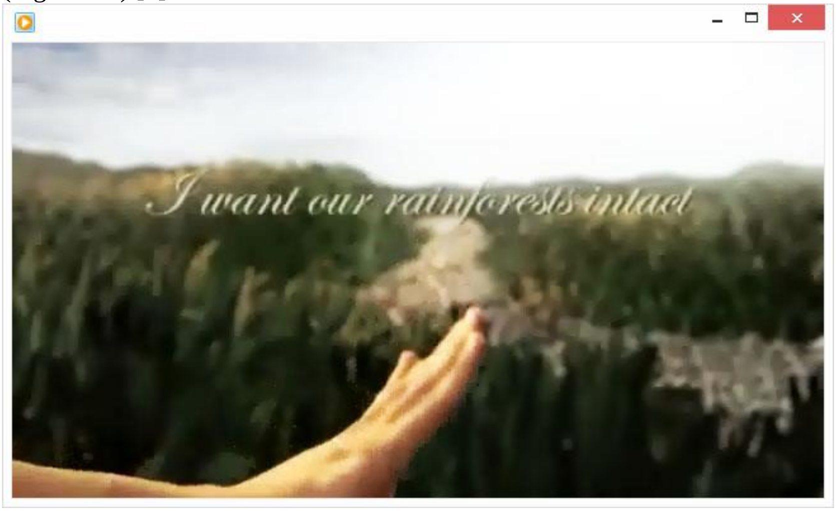


Figure 40.10 – Gaze plots which put on cyber forum registration web-page [2]

Gaze Video

Gaze Video – this is dynamic visualization, which represent as moving circles. The Gaze Video put on stimulus, web-site, pictures or video recording (Fig. 40.11) [5].



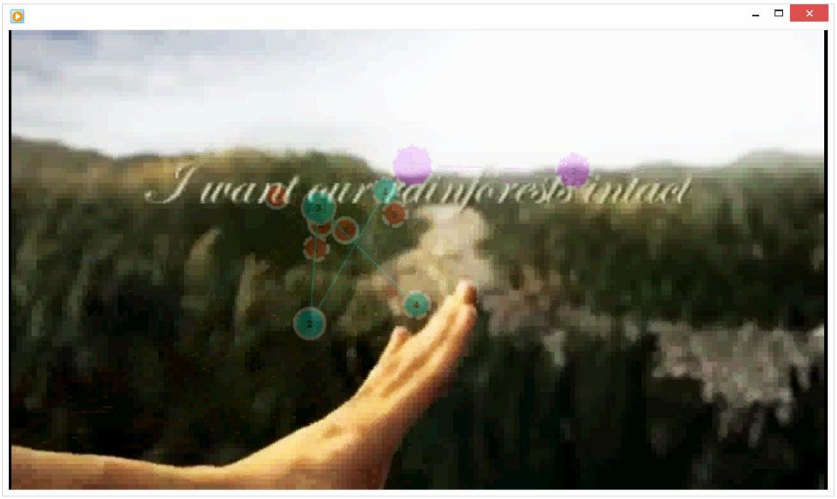


Figure 40.11 – Screenshots of Gaze Video, which put on video «Earth Day: Give Earth a Hand» from «Greenpeace» (on top without Gaze Video, on bottom with Gaze Video)

Repetitions of Gaze Video can look in real time mode. In UX research such possibility will give to researcher come up with questions for respondents in during session not only on basis his actions but and on basis his visions. Besides, Gaze Video in real time mode also does of research more attractive for stackholders, especially when respondent has complexities in verbal description his thoughts or when tasks must be run in full quiet.

Heat Map

Heat Map - this is two-dimensional data representation, in which eye-tracking values demonstrate as colors. Heat Map has intuitive character, because his color gamma correspond to generally accepted temperature: yellow is more warmer, than green or blue, orange is more warmer, than yellow, red is very hotly (Fig. 40.12) [2]. Quantity of «heat» proportional to level of represent value.

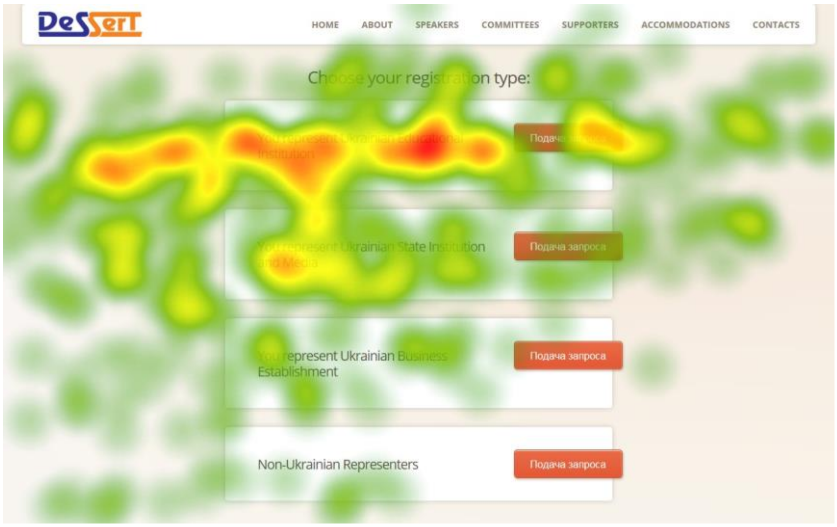

DeSSert
HOME
ABOUT
SPEAKERS
COMMITTEES
SUPPORTERS
ACCOMMODATIONS
CONTACTS
Choose your registration type:
You represent Ukrainian Educational Institution
Подача запроса
You represent Ukrainian State Institution and Media
Подача запроса
You represent Ukrainian Business Establishment
Подача запроса
Non-Ukrainian Representers
Подача запроса

Figure 40.12 – Heat map, which put on cyber forum registration form [2]

As rule, manual for use eye-tracker has very detailed description of algorithms, which use in software for development heat map. Basic idea is representation of fixations on static picture background. Color represent as small zone in around of center of each fixation. Than closer to fixation center, than values of color will be higher. Values of overlapping zones are combining. It is give more smooth of colors of maps. Thus, than higher of fixations density at one small location, than more warmer will be color for heat map.

Size of optimal respondents selection for heat map did not establish. Exist of several opinions about it. Theoretical opinion of minimal selection is only one respondent. But another opinion that more better selection can be 30 respondents.

Information about 30 respondents as optimal selection was represented in book from Jakob Nielsen and Kara Pernice [6]. According to authors of book 30 or 32 respondents can create «stable» heat map. Such number of respondents received by authors as result of following manipulations. They took eye-tracking data from 60 respondents and created 6 heat maps. Each heat map consisted from data of 10 different respondents. Heat maps were very different. All respondents were divided in 3 groups with 20 persons in each. When authors received only 2 heat maps with 30 respondents in each, they said that such 2 heat maps very similar. Therefore Jakob Nielsen and Kara Pernice made of conclusion, that 30 persons this is optimal selection for respondents.

Aga Bojko [3] has another opinion. Opinion of Aga Bojko that 30 not optimal (minimal) number of respondents for eye-tracking research. For confirmation such opinion, Aga represented following conclusions:

– firstly, similarity of heat maps was estimated only by sight, with not armed gaze. Static or another scientific methods did not use;

– secondly, particularities of data research did not use for create of heat map. Changing of top gamma color values, can make very good correction of heat maps identity for visual identification;

– third, subgroups with 10, 20, 30 respondents, were formed only once from big group from 60 respondents. More better approach will be form such groups by random pattern.

Thus, theoretically heat maps can be receive on basis data from once respondent.

Opacity map

A gaze opacity map (Fig. 40.13) is a type of outcome that depicts the areas in which the visual fixations of the respondent took place. It is often called a «reverse heat map», because although the outcome is similar to a heat map, the areas the respondent focuses on arc not covered by a temperature map, but the areas that the respondents missed are darker. The results displayed are more precise because in a gaze opacity map. the ignored areas are darker and therefore only the parts of the advertisements that attracted the eye of the respondent appear brighter [1].

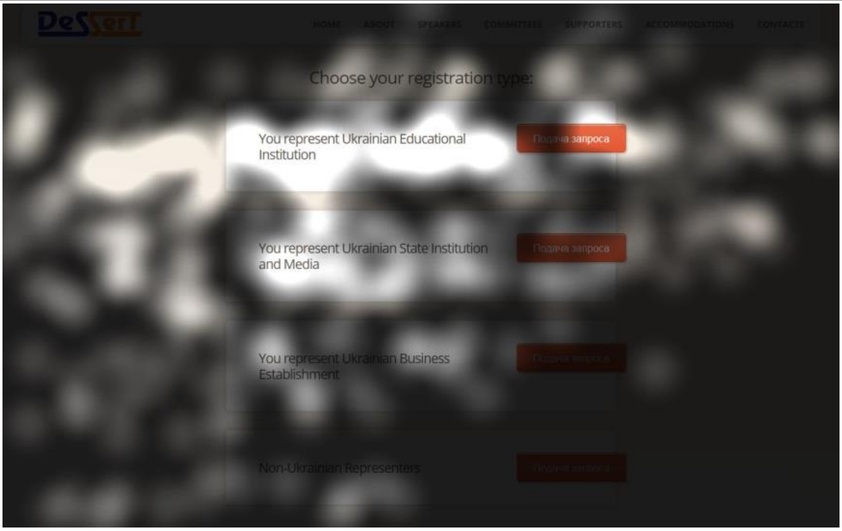

DeSSerT
HOME
ABOUT
SPEAKERS
COMMITTEES
SUPPORTERS
ACCOMMODATIONS
CONTACTS
Choose your registration type:
You represent Ukrainian Educational Institution
Подача запроса
You represent Ukrainian State Institution and Media
Подача запроса
You represent Ukrainian Business Establishment
Подача запроса
Non-Ukrainian Representers
Подача запроса

Figure 40.13 – Opacity map, which put on cyber forum registration form [2]

Bee Swarm

In order to verify visual patterns which was aggregated for all users, can use the bee swarm analysis tool of the eye tracking software. This tool shows the gaze points over time for all participants as little dots. By moving along the timeline, these dots resemble a bee swarm where, despite of apparently chaotic movements of the individual members of the swarm, the overall behavior seems to follow some higher-level patterns [7]. As a rule, respondents look in first seconds on most interesting element of object. For illustration of Bee Swarm possibilities we will view of cyberforum web-site research, as example. [2]. We asked of respondents watch on one cyberforum web page [2]. After first second of research, gazes of respondents concentrate on only one element of cyberforum web page (Fig. 40.14). Similar situation and after 2 seconds (Fig. 40.15). After 5 seconds of research, gazes of respondents concentrate on another elements of object already (Fig. 40.16).

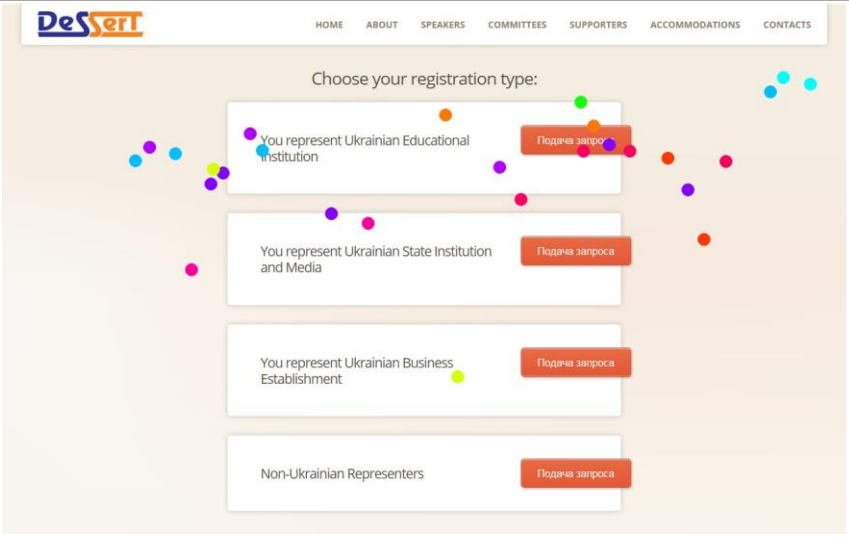

DeSSerT
HOME
ABOUT
SPEAKERS
COMMITTEES
SUPPORTERS
ACCOMMODATIONS
CONTACTS
Choose your registration type:
You represent Ukrainian Educational Institution
Подача запроса
You represent Ukrainian State Institution and Media
Подача запроса
You represent Ukrainian Business Establishment
Подача запроса
Non-Ukrainian Representers
Подача запроса

Figure 40.14 – Bee Swarm of web page [2] after first second of review

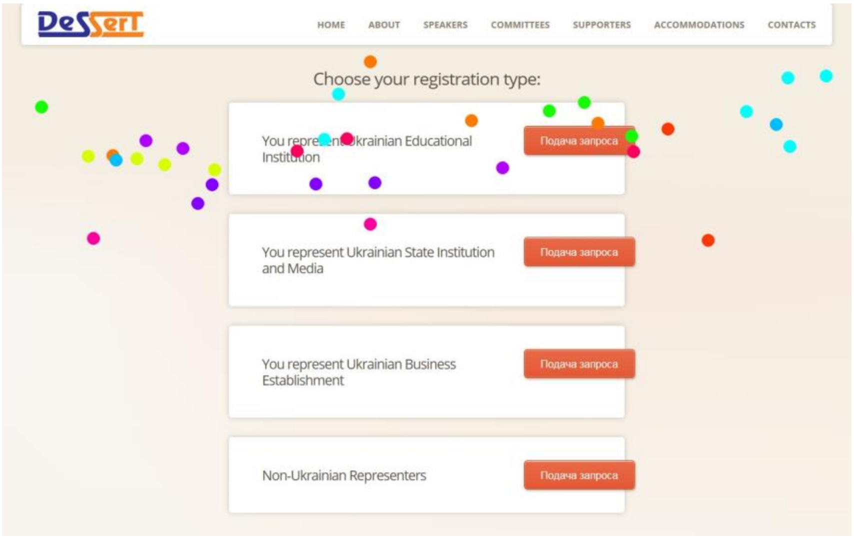

DeSSerT
HOME
ABOUT
SPEAKERS
COMMITTEES
SUPPORTERS
ACCOMMODATIONS
CONTACTS
Choose your registration type:
You represent Ukrainian Educational Institution
Подача запроса
You represent Ukrainian State Institution and Media
Подача запроса
You represent Ukrainian Business Establishment
Подача запроса
Non-Ukrainian Representers
Подача запроса

Figure 40.15 – Bee Swarm of web page [2] after 2 second of review

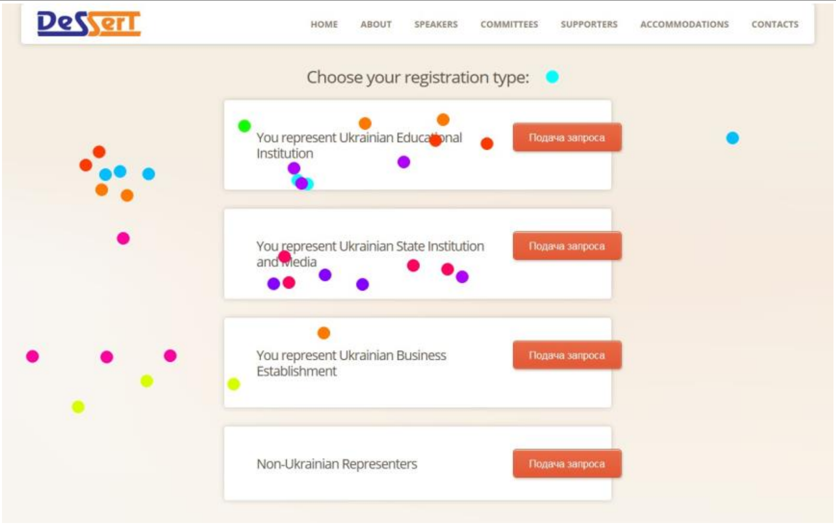

DeSSerT
HOME
ABOUT
SPEAKERS
COMMITTEES
SUPPORTERS
ACCOMMODATIONS
CONTACTS
Choose your registration type:
You represent Ukrainian Educational Institution
Подача запроса
You represent Ukrainian State Institution and Media
Подача запроса
You represent Ukrainian Business Establishment
Подача запроса
Non-Ukrainian Representers
Подача запроса

Figure 40.16 – Bee Swarm of web page [2] after five second of review

Clusters

Cluster tool visualizes areas of visual attention in a slightly different way than the heat map tool. To define the cluster areas, a spatial threshold is set, which limits how far away two fixations can be from each other before being separated into different clusters (Fig. 40.17). For each cluster, the information is given of how large percentage of the recordings or participants who have fixations included in the cluster. With this tool researcher can also zoom in on time segments. Researcher can replay the clusters for numerous participants over time.

Area of Interest

Area of Interest (AOI) this is part of object area research, which forming from tasks and goals of research. (Fig. 40.18). In during research can selecting several AOIs. Each AOI denote by different colors. Only AOIs are places for researches after selection of AOI borders. Accordingly, Eye-tracking metrics calculate for each AOI only.

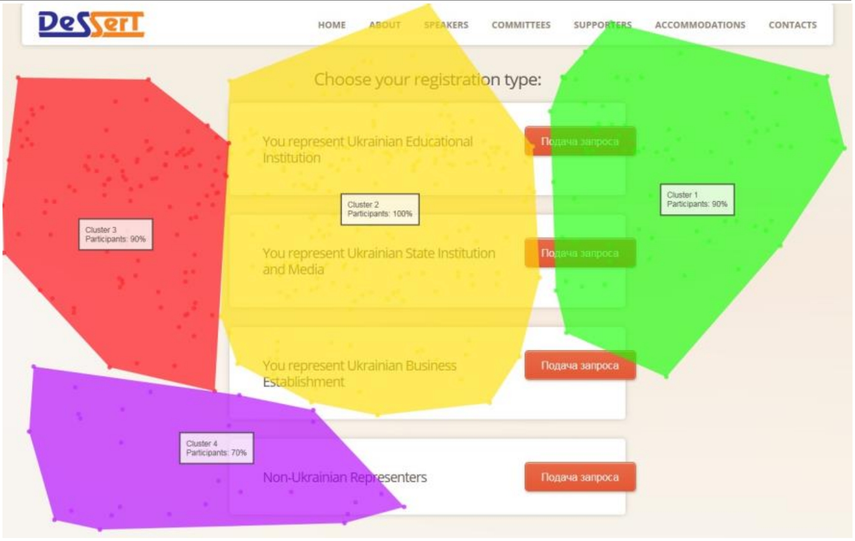

DeSSerT
HOME
ABOUT
SPEAKERS
COMMITTEES
SUPPORTERS
ACCOMMODATIONS
CONTACTS
Choose your registration type:
You represent Ukrainian Educational Institution
You represent Ukrainian State Institution and Media
You represent Ukrainian Business Establishment
Non-Ukrainian Representers
Подача запроса
Cluster 1 Participants: 90%
Cluster 2 Participants: 100%
Cluster 3 Participants: 90%
Cluster 4 Participants: 70%

Figure 40.17 – Clusters of cyberforum web page [2]

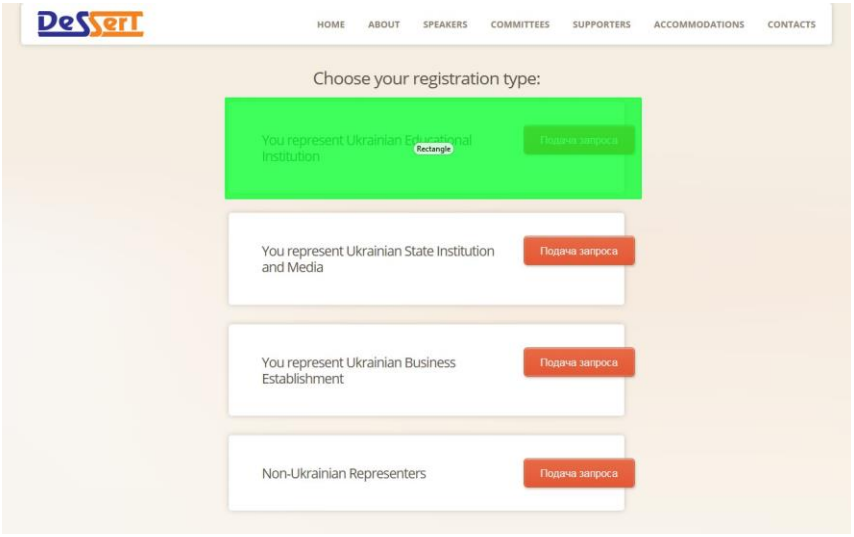

DeSSerT
HOME
ABOUT
SPEAKERS
COMMITTEES
SUPPORTERS
ACCOMMODATIONS
CONTACTS
Choose your registration type:
You represent Ukrainian Educational Institution
Rectangle
You represent Ukrainian State Institution and Media
You represent Ukrainian Business Establishment
Non-Ukrainian Representers
Подача запроса

Figure 40.18 – AOIs of cyberforum web page [2]

Conclusion and self-control questions

Metrics support of process of assessment human-machine interaction. Eye-tracking metrics taxonomy has represented, which include two main groups: attractiveness metrics and performance metrics.

Additional instrument for human-machine interaction research is possibility of information visualisation. Tobii Studio supports following types of visualisation: gaze plot, gaze video, heat map, opacity map, bee swarm and clusters.

Tobii Studio supports of Area of Interest also. It is very important mechanism, which is part of research sequence.

Self-control questions and tasks

1. What is metrics?
2. What kind of metrics you know?
3. What is gaze plot?
4. What is gaze video?
5. What is heat map?
6. What is opacity map?
7. What different between gaze plot, gaze video and opacity map?
8. Let represent of examples Area of Interests.

References

1. Kenneth Holmqvist, Marcus Nyström, Richard Andersson, Richard Dewhurst, Jarodzka Halszka, Joost van de Weijer. Eye tracking: A comprehensive guide to methods and measures. Oxford University Press. 2011, 560 p.
2. Web site of Cyber forum - Dependable systems services and technologies «DeSSerT» (http://dessertcon.com/about_forum.html).
3. Aga Bojko. Eye Tracking the User Experience: A Practical Guide to Research. Rosenfeld Media, 2013, 320 p.
4. Raiffeisen Bank Aval web site (https://www.aval.ua).
5. Green peace organization. Video clip «Earth Day: Give Earth a Hand» (https://www.youtube.com/watch?v=Ep9MFiWXR8M&t=30s).
6. Jakob Nielsen and Kara Pernice. Eyetracking Web Usability, 2009, 456 p.
7. M. Horsley, N. Toon, B. Knight, R. Reilly, Current Trends in Eye Tracking Research. Springer International Publishing Switzerland, 2014, 336 p.

Acronyms

GPS – Global Positioning System;
AOI – area of interests;
UX – user experience.

41 EYE-TRACKING TECHNIQUE ASSESSMENT AND EYE-TRACKING APPLICATION IN SECURITY DOMAIN

41.1 Eye-tracking assessment technique

Consider the method of estimating the usability of human-computer interaction [1]. It includes the following sequential steps (Fig. 41.1):

- preparation stage (1) aimed at carrying out the preparation work before the test. It includes the following works: a preliminary analysis of the object of research, selection the type of eye-tracker, development research scenario, the formation of groups of respondents, connection and adjustment of the equipment;
– the research stage (2) is designed specifically for the necessary research. It includes the following works: a pilot test to familiarize respondents with the script, writing the research results, the visualization of the results, the selection of areas of interest, the selection and calculation of metrics;
– analytical stage (3) is used directly for the analysis and presentation of results. It includes the following works: interpretation of the results, preparation of the report, the presentation of the results, recommendations for optimizing the human-machine interfaces.

Consider more detailed each work.

1. Preparation stage

1.1 A preliminary analysis of the object of research. The aim is to analyse properties and determine the characteristics of the object of research. In carrying out such work the whole set of properties and characteristics of the object should be taken into account. Inaccuracies may affect the results of studies in subsequent phases.

1.2 Selecting the type of eye-tracker. The aim is to select the type of reasoned eye-tracker. Type eye-tracker should be chosen considering characteristics of research. Firstly, decision between the remote and the head-mounted eye-trackers.

1.3 Development of the research scenario. The aim of this work is to develop a sequence of actions for being performed by each respondent during the investigation. In this paper we consider the following: The script must be described verbally in advance (before the study). Before the research scenario should pass verification by one, and preferably several independent experts. If at the result of verification, it was found that the script is difficult to perform and (or) takes a long time, it is better to divide it into several parts.

1.4 Formation of groups of respondents. The aim of this work is the selection of participants for the studies that are directly involved in the process of investigation of human-machine interface software. In this paper, it should be noted that the respondent's selection criteria should be correlated with the overall goal of the research and potential users of the investigated human-machine interfaces. For example, in studies of human-machine interfaces for software information security monitoring systems, it is necessary to form a group of respondents, professionals who have experience with similar systems.

1.5 Connection and adjustment of the equipment. The purpose of this is to prepare the equipment (laptop computer, eye-tracker and webcams), run it and set the settings for the study. Note that in the course of research data should to be written from the eye-tracker and webcam. On a portable computer equipped with specialized software, for example, Tobii Studio [2], which supports eye-tracker and eye-tracker should pass calibration and calibration verification.

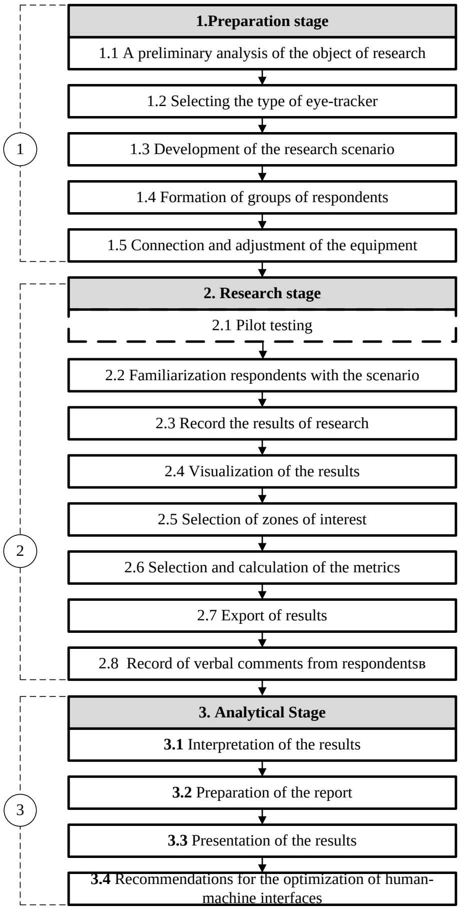

1.Preparation stage
1.1 A preliminary analysis of the object of research
1.2 Selecting the type of eye-tracker
1.3 Development of the research scenario
1.4 Formation of groups of respondents
1.5 Connection and adjustment of the equipment
1
2. Research stage
2.1 Pilot testing
2.2 Familiarization respondents with the scenario
2.3 Record the results of research
2.4 Visualization of the results
2.5 Selection of zones of interest
2.6 Selection and calculation of the metrics
2.7 Export of results
2.8 Record of verbal comments from respondentsв
2
3. Analytical Stage
3.1 Interpretation of the results
3.2 Preparation of the report
3.3 Presentation of the results
3.4 Recommendations for the optimization of human-machine interfaces
3

Figure 41.1 – The sequence of the usability assessment of human-computer interaction technique

Research stage

1.6 Pilot testing. Before the actual research it is strongly recommended to conduct a pilot eye-tracking study. Even if all necessary requirements are complied for the study and researcher completely sure about the quality of training in research, pilot testing often helps identify unexpected problems.

1.7 During the pilot testing, usually follows 4 goals:

1.8 the first goal is to eliminate all technical problems. Researcher must be sure that functions of hardware and software of eye-tracker performed reliably and fully. Also all the output files and data should be successfully generated. If the software and (or) hardware problems are found, then they should seek solutions asking technical support of company-manufacturer;

1.9 second goal is to conduct the scenario in order to make sure that the script is not difficult to implement it and all parts of instructions understood by the respondents;

1.10 third goal of the pilot test is that the researcher received the experience of preparing a specific study using eye-tracker. Achieving this goal is also very important, as an experienced moderator makes fewer mistakes, gets more qualitative data and knows how to control the duration of the studies. Even the most experienced moderators recommended pilot testing, especially for more complex studies apparently;

1.11 the fourth and the final goal of the pilot test is to confirm the accuracy of organization studies by other researchers. Even if leadership for research is described in detail, there are features that are associated with the uniqueness and precision of perception instructions. The only way to identify possible inconsistencies or shortcomings in the organization of research is to attract other researchers for consultation and observation.

1.12 This work is not mandatory and its necessity is determined by researcher.

1.13 Familiarization respondents with the scenario. The aim of this work is to read respondent verbal (sometimes schematic) description of the sequence actions, which he has to perform in the framework of research. It should be noted that in the process of familiarization with the scenario, respondent can ask researcher clarifying questions. If the issues concern, research and (or) the script, the respondent should receive full answers that will satisfy him. If the script is not fully understood by the respondent or will he remain some unanswered questions, it can negatively affect the results of research.

1.14 Record of results of research. The work is to maintain the real-time data about the visual routes from eye-tracker to respondents, as well as facial expression data via a web-camera at the time of research.

1.15 Visualization of the results. The aim of this work is preview and analysis of the results – visualization of the optic routes and, if necessary, a mimicry of respondents.

1.16 Selection of zones of interest. The work is to identify areas of research on the subject that are most interesting to the researcher. In fact, in this work are localized areas of the research object.

1.17 Selection and calculation of the metrics. The rresearcher should define a profile of metrics whose values will be calculated for each region of interest.

1.18 Export of results. Results of research can be to export. In future the results of research will use for prepare of presentation about results of research.

1.19 Record of verbal comments from respondents. In general after completion of eye-tracking research respondent has possibility share own impressions about performance of scenario. All words of respondents must be write on paper or record with microphone. Such data can help more exactly interpret of received results.

2. Analytical Stage

2.1 Interpretation of the results. The aim is accurate and complete interpretation of the results of research. Such work requires a sufficient concentration of researchers in analysing the results of research, attention to detail and experience in conducting such studies. It is better to perform in a group of 2 or more researchers.

2.2 Preparation of the report. The aim is filing and decoration the document on the results obtained by research, their interpretation and preliminary conclusions of identified deficiencies in the human-machine interfaces.

2.3 Presentation of the results. This work is to present the results to the customer or customer's representatives for them in understandable terms.

2.4 Recommendations for the optimization of human-machine interfaces. The aim is a set of constructive suggestions to improve the human-machine interfaces.

41.2 Possible additional applications eye-tracking for security, usability and control

Consider the possible additional tasks applying eye-tracking technology in the fields of security, safety, usability and control (Fig. 41.2).

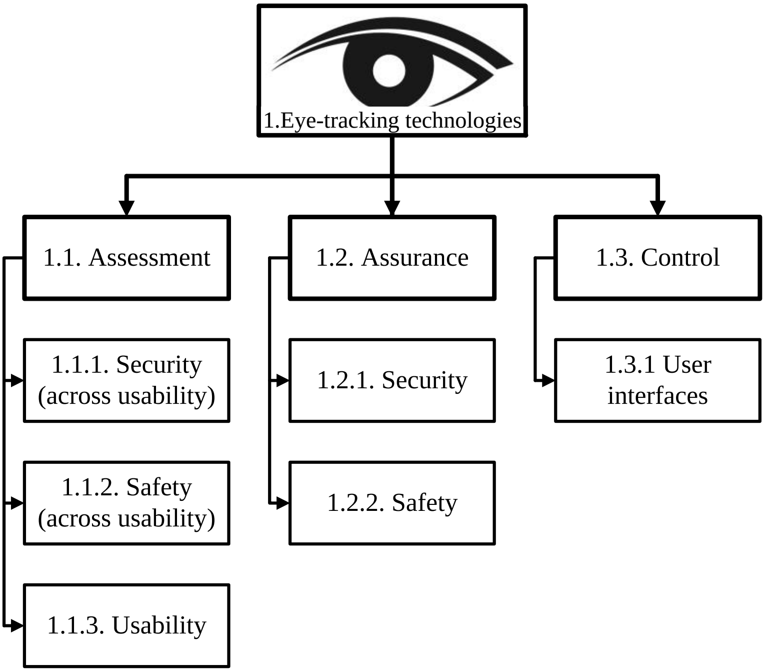


Figure 41.2 – Taxonomy of additional application tasks eye-tracking technology

All tasks can be divided into areas of assessment, assurance and control. Let’s look more detailed and give examples for each direction:

1.1 Assurance

1.1.1 Security (across usability). Eye-tracking technology can be use for human-machine interfaces usability assessment of security control software

1.1.2 Safety (across usability). Eye-tracking technology can be use for human-machine interfaces usability assessment of, for example, driver and car

1.1.3 Usability. Eye-tracking technology can be use for software usability assessment in general

1.2 Assurance

1.2.1 Security. Eye-tracking technology can be used for analysis of retina to control access to the systems. Also, such technologies can be used for the graphical key, for example, for mobile phones.

1.2.2 Safety. Here eye-tracking technology can be used for tracking driver's eyes and in case if driver falls asleep can wake him up by special signal.

1.2.3 Usability. Eye-tracking can be used for evaluation the usability of interfaces of software in general.

1.3 Control

1.3.1 User interfaces. The eye-tracking technology assists people with disabilities to manage software interfaces and enter the information using the view, i. e., with the help of movement and concentration of their gaze.

A separate area for research is a relatively new area that has been crystallized and is located at the junction of the two areas (properties) usability and cybersecurity. This area is called the usable cybersecurity or usability of security [3]. It should also be noted that in the National Institute of Standards and Technologies includes special special unit Usability of Cybersecurity Team [4]. Here are some examples and describe in more detail usable cybersecurity. Usability and cybersecurity complement one another. A well-designed system needs to make it easy for the user to do the right thing, hard to do the wrong thing, and easy to recover when the wrong things happen anyway. Another term for this is «user-centred security». However, accomplishing this is easier said than done. Passwords provide an example of the challenges involved in creating usable cybersecurity measures. From a usability perspective, passwords should be easy to remember, changed infrequently, and reusable across multiple systems. From a security perspective, passwords should be long and random, changed on a regular basis, and not reused across multiple systems.

In general, the best way to ensure usability in a system is to test it with real users prior to full implementation. In other respects, however, testing for and implementing usability in any kind of system is not always straightforward. There are three factors that make it challenging to design systems for usability:

Cybersecurity usability is a multidisciplinary field that involves experts from domains such as computer science, cognitive psychology, HCI, and (of course) cybersecurity. Since experts from these domains use different terminology and practices, it can be difficult for them to coordinate with each other

As a relatively young field, cybersecurity usability has little empirical data from which to develop usability best practices (although this is changing).

Usability is very context-specific and influenced by a number of factors, such as the nature of the user population, organizational culture, and the specifications of the organization’s systems and cybersecurity measures themselves.

41.3 Standard ANSI/ISA-101.01 2015 «Human machine interfaces for process automation systems» particularities

Usability assessment of human-computer interaction technique can be use for assurance of requirements of standard ANSI/ISA-101.01-2015 for assessment HMI of automation systems [5].

General information

Safe and efficient facility operations are directly related to the Control Room Operator's access to reliable and accurate, real time operational information. A well-designed Human Machine Interface (HMI) should give the Operator the ability to monitor and interact with the processes, by displaying process data as organized and usable information. HMIs have often been designed to be simplistic graphical representations of the process data, through individual displays depicting states (colors), values, and trended data on charts. ANSI/ISA-101.01-2015, the new ISA-101 Human Machine Interfaces for Process Automation Systems standard is a consensus created document that addresses the design, implementation, and maintenance of HMIs from a holistic standpoint and identifies documentation and design practices that will lead to more effective and maintainable HMI implementations.

We represent short characteristic of ANSI/ISA-101.01-2015 standard.

Use of this standard should:

– provide guidance to design, build, operate, and maintain effective HMIs which result in safer, more effective, and efficient control of the process, in both normal and abnormal situations;

– improve the user's abilities to detect, diagnose, and properly respond to abnormal situations.

Scopes of the standard are:

– HMIs for equipment and automated processes;

– application of practices and methodology for safety, security and reliability of process automation systems;

– the practices in this standard are applicable to discrete processes and any process using an HMI for interfacing to a controlled system.

Potential audience of standard can include of following participants:

– users, which responsible for safe and productive operation of equipment and facility;

– integrators, designers, engineers, which build of HMI design and develop of applications;

– vendor, which develop the software tools needed to build the HMI applications; and which develop the interfaces / drivers needed for an HMI to transfer data and information to and from multiple sources.

Benefits use of standards can be are following:

– consistency in execution. I.e. easier to implement new applications and easier to hand off to third party developers;

– decrease of time to train users;

- decrease of operator error as a result of consistent effective design;
- more easier to move between platforms or versions of HMI.

HMI life cycle concept

HMI life cycle concept has represented on Fig. 41.3. HMI life cycle include of following logical blocks: system standards, design, implement, operate, continuous work processes. We view of each logical block more detail.

System standards stage. Basic of goal such stage it is develop documents that set the foundation for all HMI design decisions. System standards stage include following substages: philosophy, style guide, toolkit.

The HMI philosophy is a strategic document addressing the guiding principles that govern the design structure of the HMI. Philosophy as a rule, include of human factors; user, task and functional requirements for all modes of operation that require HMI support; design standards and work practices for the development and management of the HMI. Philosophy provide a foundation of concepts such that new developers and users can grasp the underlying principles and technical rationales.

A style guide will include general design principles for the displays and implementation standards. Typically, it includes:

- work practices recommended to manage the HMI;
- guidance on display types and their preferred use;
- guidance on expected HMI performance targets.

As a rule, HMI toolkit is developing for decision of concrete tasks and includes templates and examples of all necessary graphic symbols and elements to implement an HMI application that meets the style guide requirements. Additional description of substages represent in table. 41.41 as directions Activity, Objectives, Inputs, Outputs.

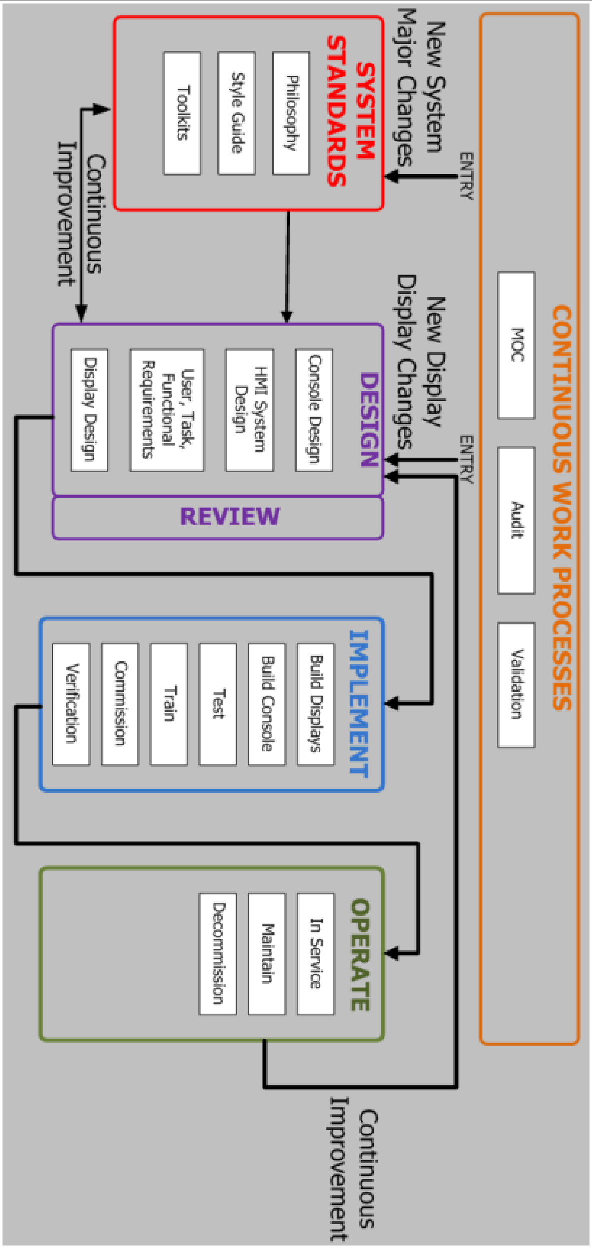


Figure 41.3 – HMI life cycle concept

Table 41.1 – System standards substages

Activity	Objectives	Inputs	Outputs
HMI Philosophy	Provides guiding principles and conceptual foundation for HMI design. This includes details on how the HMI is used and designed. (Independent of platform)	User Experience, Conceptual User Functional Requirements, Best Practices, Standards, Guidelines and Human Factors Engineering Considerations	HMI Philosophy Document
Style Guide	Turns the guiding principles and concepts of the HMI Philosophy and turns them into implementation examples and guidance. (This does not include all technical details, though the style guide needs to be feasible in all target platforms)	HMI Philosophy, Platform experience and expertise (to confirm feasibility; develop early proof of concept designs)	HMI Style Guide Document
Toolkit	Generate all requirement graphical symbols and other supporting elements as required to implement the Style Guide	HMI Style Guide Document, Platform experience and expertise, Conceptual User, Task, Functional Requirements	HMI Toolkit (Platform-specific)

Descriptions of Design, Implement, Operate and Continuous work process represents in Table 41.2 respectively as directions Activity, Objectives, Inputs, Outputs.

Table 41.2 – Design, Implement, Operate and Continuous work substages descriptions

Activity	Objectives	Inputs	Outputs
Design substages			
Console Design	To provide hardware and software design for the Console. This includes furniture and supporting systems (phone, radio, LAN PC, cameras, etc.)	User, Task, Functional Requirements; Vendor Specifications, Human Factors Engineering Design Standards	Console design documents
HMI System Design	Identify design basis for the HMI system	User, Task, Functional Requirements. Control System Design Standards, Network Design Standards, Preliminary Network Design	HMI system design documents
User, Task and Functional Requirements	Identify primary and secondary requirements that be supported in the HMI	HMI Philosophy, HMI Style Guide, Console Design, User, Task and Functional Requirements Analysis	Requirements document(s)
Display Design	Identifies conceptual design for displays and the navigation hierarchy. (This may include some prototype displays on complex applications	HMI Philosophy, HMI Style Guide, User, Task, Functional Requirements document(s), User Input in Review(s)	Display design document(s)

Activity	Objectives	Inputs	Outputs
	or processes)		
Implement substages			
Build Displays	Complete construction of displays and supporting items	Display design documents	Displays
Build Console	Complete construction of console hardware and software	Console design documents	Console
Test	Integrated Test of HMI and Console	User, Task, Functional Requirements documents, Usability and performance standards. Console and Displays	HMI Ready to Commission, Testing documents
Train	Train Users	Console, Displays, User Manuals and Online Help (as required)	HMI Ready to Qualify (as required), Commissionin g documents
Commission	Final testing of HMI in production environment	Console, Displays, User manuals and Online help (as required)	HMI Ready to Quality (as required) Commissionin g documents
Verification	Verify HMI Ready to Operate	Qualification Plan, Commissioning documents	Qualification documents, HMI Ready to Operate
Operate substages			
In Service	HMI In Service	Commissioning/Qualificati on Approval, User Manuals and Online Help	HMI in Service
Maintain	Ensure HMI	Approved Change	Approved

Activity	Objectives	Inputs	Outputs
	is Valid and Reflects Current Process Conditions	Management requests to fix errors or to add enhancements or updates to reflect changes in the process	Change Management requests to fix errors or to add enhancements or updates to reflect changes in the process
Decommission	HMI Removed from Service (End of Life)	Change Management Change Requests	HMI (or part of HMI) removed from use, archived for approved records period
Continuous Work Processes substages			
Management of Change (MOC)	Manage change, ensuring consideration of allimpacts	Changes in process or User, Task and Functional Requirements	Change completed following the approved work practices
Audit	Verify that the HMI is being managed under the approved work practices	HMI Philosophy, HMI Style Guide, Related Documents	Audit Records, Change requests to Correct any Deviations, Updates (as needed) to HMI Philosophy, HMI Style Guide, HMI Toolkits and Related Documents
Validation	Verify HMI meets User, Task and Functional	Validation Plan	Validation System, Validation Records

Activity	Objectives	Inputs	Outputs
	Requirements		

Conclusion and self-control questions

Studies using eye-tracking technology commonly used in the form of a single technology, in which the sequence of work may not be violated, while some work can be optional (for example, pilot testing).

Eye-tracking technologies can be applied in the fields of security, safety and usability solutions for assessment tasks, assurance and control.

One from new standards in HMI development and assessment field is standard ANSI/ISA-101.01 2015 «Human machine interfaces for process automation systems». It standard include of following logical blocks: system standards, design, implement, operate, continuous work processes. According with standards requirements eye-tracking technologies can be use for testing, verification and validation of HMI.

Self-control questions and tasks

1. What three main parts included in sequence of eye-tracking assessment?
2. For what applications we can use eye-tracking technology?
3. What difference has applications eye-tracking technologies for safety, security and usability assessment?
4. What is ANSI/ISA-101.01-2015?
5. Let represent examples of cases use eye-tracking technologies for security, safety and usability.

References

1. Kootstra, G., de Boer, B. & Schomaker, L.R.B. Predicting Eye Fixations on Complex Visual Stimuli Using Local Symmetry. Cognitive Computation, 2011, pp. 223-240.
2. Web site of Tobii company (http://tobii.com).
3. Nurse, J., Crccsc, S.. Goldsmith, M.. Lamberts, K.: Guidelines for usable cybersecurity: past and present. In: Third International Workshop on Cyberspace Safety and Security (CSS), 2011, pp. 21-26.
4. NIST Usability of Cybersecurity Team (http://csrc.nist.gov/security-usability/HTML/about.html).
5. ANSI/ISA-101.01 2015 «Human machine interfaces for process automation systems», 2015, 60 p.

Acronyms

HMI – human machine interaction

42 TECHNIQUES AND TOOLS FOR INDUSTRY FPGA-BASED SYSTEMS SECURITY

42.1 Special FPGA capabilities related with security assurance

FPGAs combine the features of processors with the performance of custom hardware. As they are widely used in safety and security critical systems, special techniques are needed to manage security in FPGA designs. FPGAs can provide a useful balance between performance, rapid time to market, and flexibility [1].

Also there are following FPGA features, which can be beneficial from the point of view of security assurance [2]:

– There are no known viruses and malware designed to attack HDL coded configurations; FPGA-based platforms have a simple and structured design, therefore the corresponding V&V processes performed at each stage of design are more likely detect the presence of potential threats and malicious design;

– FPGAs do not use operating systems which could be the target of potential cyber-attacks;

– Physical access to the FPGA chips is also strictly controlled by design. For example, the HDL code is located in a flash memory (on a separate chip) without offering any physical access for modification while in on-line operating mode;

– FPGA programming and reprogramming can be done only through a special interface. It is impossible to connect common storage media or communication devices that could infect the control logic code, as was the case in the Stuxnet attack [3].

Material of the actual section are mainly based on article “Managing Security in FPGA-Based Embedded Systems” [4], written by researcher from Naval Postgraduate School (Monterey, California), University of California, and Special Technologies Laboratory (Santa Barbara, California). This article, from the one hand, provides pioneer results in the area of FPGAs security, from the other hand, this article is still state-of-the-art research through the years.

Because major FPGA manufacturers outsource most of their operations, as named Intellectual Property cores (IP-cores) theft from a foundry may be a serious concern. FPGAs provide a viable solution to this problem because the sensitive IP-core is not loaded onto the device until after it has been manufactured and delivered, making it harder for adversaries to target a specific application or user. Furthermore, FPGAs use bitstream encryption and other methods to protect IP-cores once it is loaded onto the FPGA or external memory.

However, techniques beyond bitstream encryption are necessary to ensure FPGA design security. To save time and money, FPGA systems are typically cobbled together from a collection of existing computational cores, often obtained from third parties. These cores can be subverted during the design phase, by tampering with the tools used to translate the design to the cores or by tampering with the cores themselves. Building every core and tool from scratch is not economically feasible in most cases, and subversion can affect both third-party cores and cores developed in-house. Therefore, embedded designers need methods for securely composing systems comprising both trusted and untrusted components.

FPGAs use programmability and an array of uniform logic blocks to create a flexible computing fabric that can lower design costs, reduce system complexity, and decrease time to market, using parallelism and hardware acceleration to achieve performance gains. The growing popularity of FPGAs has forced practitioners to begin integrating security as a first-order design consideration, but the resource-constrained nature of embedded systems makes it challenging to provide a high level of security.

An FPGA is a collection of programmable gates embedded in a flexible interconnect network that can contain several hard or soft microprocessors. FPGAs use truth tables or lookup tables (LUTs) to implement logic gates, flip-flops for timing and registers, switchable interconnects to route logic signals between different units, and I/O blocks for transferring data into and out of the device. A circuit can be mapped to an FPGA by loading the LUTs and switch boxes with a configuration, a method that is analogous to the way a traditional circuit might be mapped to a set of AND and OR gates.

An FPGA is programmed using a bitstream. This binary data, loaded into the FPGA through specific I/O ports on the device, defines how the internal resources are used for performing logic operations.

Fig. 42.1 shows some of the many different design flows used to compose a single embedded system. Such cores with different pedigrees and various trust requirements occupy the same silicon. The presented system-on-the-chip includes reconfigurable logic, hardwired soft-processor cores, Static Random Access Memory (SRAM) blocks and other soft IP-cores all share the FPGA and the same off-chip memory, block RAM (BRAM), digital signal processors (DSP), and microprocessor (μP).

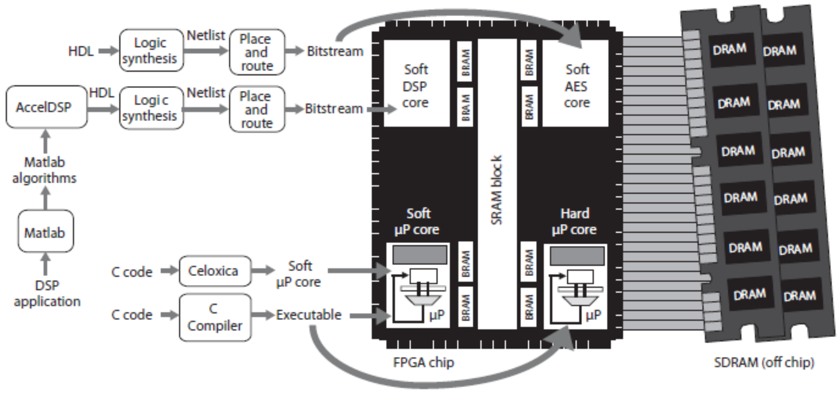


Fig. 42.1 – An example of multicomponent FPGA-based system (source: [4])

The FPGA implementation relies on several sophisticated software tools created by many different people and organizations. Special-purpose processing cores, such as an Advanced Encryption Standard (AES) core, can be distributed in the form of the hardware description language (HDL), netlists (which are a list of logical gates and their interconnections), or a bitstream. These cores can be designed by hand, or they can be automatically generated by design tools. For example, the Xilinx Embedded Development Kit (EDK) generates a soft microprocessor on which C code can be executed.

A synthesis tool uses a place and route algorithm to convert this netlist into a bitstream, with the final result being an implementation of a specialized signal-processing core. Security vulnerabilities can be introduced into the life cycle inadvertently because designers sometimes leave “hooks” (features included to simplify later changes or additions) in the finished design. In addition, the life cycle can be subverted when engineers inject unintended functionality, some of which might be malicious, into both tools and cores.

The subversion of design tools could easily result in malicious design being loaded onto a device. For example, a malicious design could physically destroy the FPGA by causing the device to short circuit. In fact, major design-tool developers have few or no checks in place to ensure that attacks on specific functionality are not included. An approach to eliminate such vulnerability may lay in safely combination of trusted cores, developed with trusted tools (perhaps using in-house tools that might not be fully optimized) with untrusted cores. FPGA manufacturers provide signed cores that embedded designers can trust. Freely available cores obtained from open sources might have vulnerabilities introduced after distribution from the original source. However, a digital signature does not prevent vulnerability either.

Given that different design tools produce a set of interoperating cores, and in the absence of overarching security architecture, you can only trust your final system as much as you trust your least-trusted design path. If there is security-critical functionality (such as a unit that protects and operates on secret keys), there is no way to verify that other cores cannot snoop on it or tamper with it.

One major problem is that it’s now possible to copy hardware, not just software, from existing products, and industry has invested in mechanisms to protect IP-cores.

Multiple scales of design present different potential attack vectors. Attacks at the device level can involve malicious software as well as sophisticated sand-and-scan techniques. Attacks at the board level can involve passive snooping on the wires that connect chips and the networks that connect boards as well as active modification of data traffic. There are security advantages to using a separate chip for each core, because doing so eliminates the threat of cores on the same device interfering with one another. This advantage must be weighed against the increased power and area cost of having more chips and the increased risk of snooping on the communication lines between chips.

Solutions to reconfigurable security problems fall into two categories: life cycle management and a secure architecture.

Life cycle management is an approach to ensure the trustworthiness of all the tools involved in the complex FPGA design flow. FPGA-based systems designers already deal with problems like software configuration management, which covers operating systems, security kernels, applications, and compilers.

Configuration management stores software in a repository and assigns it a version number. The reputation of a tool’s specific version is based on how extensively is has been evaluated and tested, the extent of its adoption by practitioners, and whether it has a history of generating output with a security flaw. The rationale behind taking a snapshot in time of a particular version of a tool is that later versions of the tool might be flawed. For example, because automatic updates can introduce system flaws, it is often more secured to delay upgrades until the new version has been thoroughly tested.

A similar strategy is needed for life cycle protection of hardware to provide accountability in the development process, including control of the development environment and tools, as well as trusted supply chain of the chips from the factory. Both cores and tools should be placed under a configuration management system. Ideally, it should be possible to verify that the output of each stage of the design flow faithfully implements the input to that stage through the use of formal methods such as model checking. However, such static analysis suffers from the problem of false positives, and a complete security analysis of a complex tool chain is not possible with current technology, owing to the exponential explosion in the number of states that must be checked.

An alternative is to build a custom set of trusted tools for security-critical hardware. This tool chain would implement a subset of the commercial tool chain’s optimization functions, and the resulting designs would likely sacrifice some measure of performance for additional security. Existing research on trusted compilers could be exploited to minimize the development effort. A critical function of lifecycle protection is to ensure that the output (and transitively the input) does not contain malicious artifacts. Testing can also help ensure fidelity to requirements and common failure modes. For example, it should consider the location of the system’s entry points, its dependencies, and its behavior during failure.

Life-cycle management also includes delivery and maintenance. Trusted delivery ensures that the FPGA has not been tampered with from manufacturing to customer delivery. For an FPGA, maintenance includes updates to the configuration, which can occur remotely on some FPGAs. For example, a vendor might release an improved version of the bitstream that fixes bugs in the earlier version.

Programmability of FPGAs is a major advantage for providing on-chip security, but this malleability introduces unique vulnerabilities. Industry is reluctant to add security features to Application Specific Integral Circuits (ASIC), because the edit compile run cycle cost can be prohibitive. FPGAs, on the other hand, provide the opportunity to incorporate self-protective security mechanisms at a far lower cost.

In this section we consider the following secure architecture features: memory protection, isolation, identification tags attachment, and secure communications.

One example of a runtime security mechanism we can build into reconfigurable hardware is memory protection. On some embedded devices, memory is flat and unprotected. A reference monitor, a well-understood concept from computer security, can enforce a policy that specifies the legal sharing of memory (and other computing resources) among cores on a chip. A reference monitor is an access control mechanism that possesses three properties: it is self-protecting, its enforcement mechanisms cannot be bypassed, and it can be subjected to analysis that ensures its correctness and completeness. Reference monitors are useful in composing systems because they are small and do not require any knowledge of a core's inner workings.

Although synthesis tools can generate designs in which the cores are intertwined, increasing the possibility of interference, FPGAs provide a powerful means of isolation. Because applications are mapped spatially to the device, we can isolate computation resources such as cores in space by controlling the layout function. A side benefit of the use of physical isolation of components is that it more cleanly modularizes the system. Checks for the design's correctness are easier because all parts of the chip that are not relevant to the component under test can be masked. Also some regions of the chip can be isolated isolate each other by placing a buffer between them.

As opposed to explicitly monitoring attempts to access memory, the ability to track information and its transformation as it flows through a system is a useful primitive for composing secure systems. A tag is metadata that can be attached to individual pieces of system data. Tags can be used as security labels, and, thanks to their flexibility, they can tag data in an FPGA at different granularities. For example, a tag can be associated with an individual bit, byte, word, or larger data chunk. Once this data is tagged, static analysis can be used to test that tags are tightly bound to data and are immutable within a given program. Although techniques currently exist to enhance general-purpose processor with tags such that only the most privileged software level can add or change a tag, automatic methods of adding tags to other types of cores are needed for tags to be useful as a runtime protection mechanism.

To run designated features, cores must communicate with one another and therefore cannot be completely isolated. Cores can communicate via shared memory, direct connections, and a shared bus. For communication via shared memory, the reference monitor can enforce the security policy as a function of its ability to control access to memory in general. For communication via direct connections, static analysis can verify that only specified direct connections are permitted, as we discussed earlier. Such interconnect-tracing techniques can be applied at both the device and board levels. Communication via a shared bus must address several threats. If traffic sent over the bus is encrypted, snooping is not a problem. To address covert channels resulting from bus contention, every core can be given a fixed slice of time to use the bus. Although various optimizations have been proposed, this type of multiplexing is clearly inefficient, because cores needs may change over time.

Secure system composition on an FPGA should employ many different techniques, both static and runtime, including life-cycle management, reconfigurable mechanisms, spatial isolation, and a coherent security architecture.

Successful security architecture must help designers manage system complexity without requiring all system developers to have complete knowledge of the inner workings of all hardware and software components, which are far too complex for complete analysis.

An architecture that enables the use of both trusted and commercial (partly trusted) components would let us build systems without having to reassess all the elements for every new composition.

16.2 Industrial Case: Security assurance for FPGA-based RadICS PLC

The RadICS Platform is a state-of-the-art digital control system platform specifically designed for safety-related control and protection systems in NPP applications [5]. The RadICS Platform features a modular and distributed FPGA-based architecture. This is a set of general purpose building blocks (modules) that can be configured and used to implement application specific functions and systems (see Fig. 42.2).

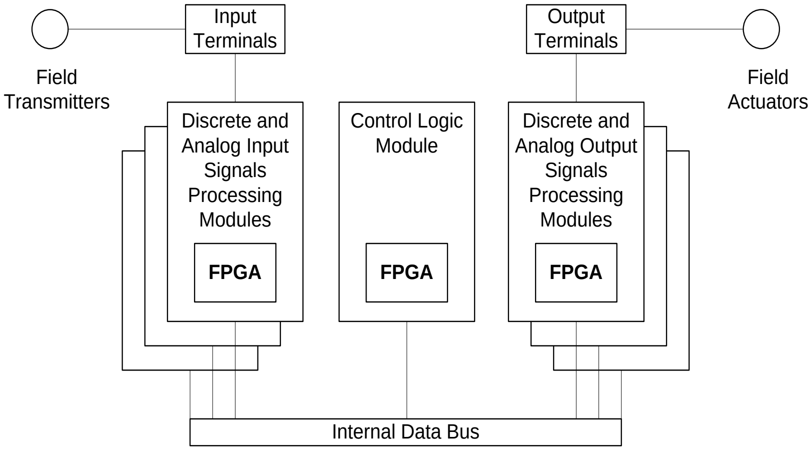


Fig. 42.2. Typical structure of Instrumentation and Control system based on Field Programmable Gates Array (FPGA)

The basic architecture of the RadICS Platform consists of an instrument chassis containing a logic module, as well as up to 14 other Input/Output (I/O) and fiber optic communication modules. Logic modules gather input data from input modules, execute user specific logic, and update the value driving the output modules. They are also responsible for gathering diagnostic and general health information from all I/O modules. The I/O modules provide interfaces with field devices (e.g., sensors, transmitters, and actuators). The functionality of each module is defined by the logic implemented in the FPGA that are part of the above modules. The backplane of the RadICS Platform provides interfaces to power supplies, process I/Os, communication links, and indicators. The internal backplane provides interfaces to the various modules installed within each chassis by means of a dedicated, isolated, point-to-point low voltage differential signaling (LVDS) interface.

For application development, RPC Radiy provides a tool called Radiy Product Configuration Toolset (RPCT). This tool can be used to configure logic for various applications using the Application Functional Block Library (AFBL).

In addition, the RadICS Platform includes extensive on-line self-surveillance and diagnostics at various levels, including control of FPGA power, watchdog timer, cyclical redundancy check (CRC) calculations, and monitoring of the performance of FPGA support circuits, I/O modules, communications units, and power supplies.

Safety application may be designed on the base of the RadICS Platform using rigorous life cycle implementation. The main market for the RadICS Platform is Instrumentation and Control (I&C) systems for Nuclear Power Plants (NPP). So when NPP I&C systems on the base of RadICS Platform are licensed and certified, specific security requirements for a nuclear domain have to be taken into account.

The RadICS Platform has the following security benefits which are caused by FPGA features:

– FPGA reconfiguration is performed through interfaces protected physically and by passwords. Software used in configuration and verification processes is password protected;

– User Radiy’s I&C systems are intended for use in a wide range of nuclear safety and process control applications. In most cases, systems are composed of multiple modules and chassis, each including one or more FPGA chips as their computational engine. Internal interfaces are implemented via dedicated and isolated connections between chassis modules, while interfaces external to our systems chassis provide secured and reliable connections to prevent unauthorized access;

– Radiy’s I&C systems use one-direction User Datagram Protocol (UDP) and point-to-point communication to transfer plant and diagnostic data from logic control level of HMI;

– Manuals are issued with supplied I&C systems to encourage operation and maintenance personnel to implement protective measures;

– HDL code (VHDL or Verilog) without an operating system is used for FPGA programming. At the present there are no known viruses and malware for HDL.

Safety and security life cycle of the RadICS Platform [6] is presented on Fig. 42.3. The RadICS Platform life cycle implements specific stages of FPGA design development and verification. Specific technique of fault insertion testing has been performed both for hardware and software parts.

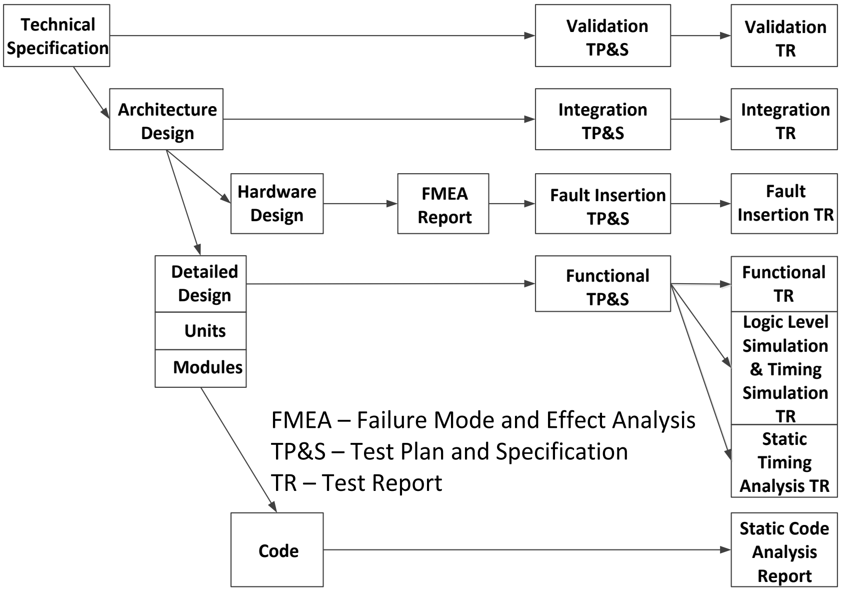


Fig. 42.3. Safety and security life cycle of the FPGA-based RadICS Platform

This life cycle complies with requirement traceability concept which requested the following:

– Every requirement has a child (either a lower level requirement or a solution);

– Every lower level requirement or solution to a higher level requirement: an orphan represents unjustified functionality;

– Every requirement has been tested.

An example for an application of the RadICS Platform is given on on Fig. 42.4. The same life cycle as per Fig. 42.3 has been applied for this application.

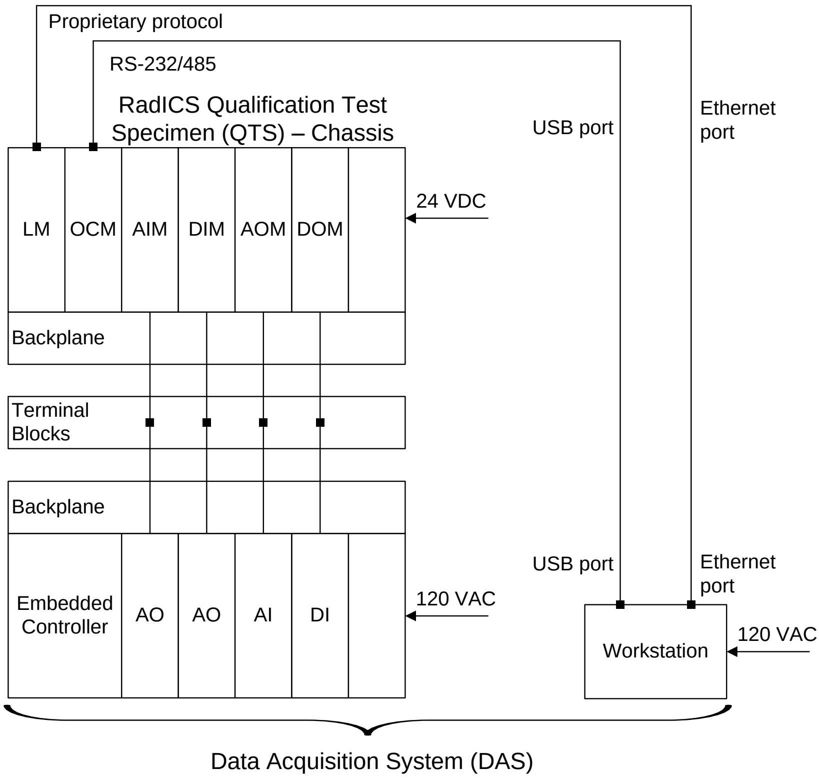


Fig. 42.4. The RadICS Qualification Test Specimen with the connected Data Acquisition system

The given application represents the RadICS Qualification Test Specimen (QTS), which has been designed and manufactured for the project of the RadICS Platform certification against the United States Nuclear Regulatory Commission (U.S. NRC) requirements [7].

The QTS is assembled from the safety-related components of the RadICS Digital Safety Instrumentation and Control Platform. After manufacturing, the QTS shall be tested to confirm compliance with the U.S. NRC requirements to Equipment Qualification (EQ) testing.

The Data Acquisition System (DAS) is needed to simulate the QTS Input / Outputs as well to exchange data via communications and to monitor data.

The QTS is the RadICS chassis powered with 24 VDC. Chassis in-houses a representative set of the RadICS platform modules: Logic Module (LM), Optical Communication Module (OCM), Analog Input Module (AIM), Discrete Input Module (DIM), Analog Output Module (AOM), Discrete Output Module (DOM). These modules are connected with Input / Output simulator via terminal blocks. LM and OCM transmit data to a workstation which also is a part of the DAS.

Four standard types of interfaces are used in the RadICS Platform:

– External Interfaces are used to communicate with another Chassis or specific device

– Internal Interfaces are used for on-board communications within a Module or for Module to Module communications within a Chassis

– Online Interfaces are interfaces that can directly influence a safety-critical system during normal operation

– Offline Interfaces are interfaces used for a safety-critical system configuration when it is out of operation

External interfaces are the most critical from the security point of view. The RadICS Platform (chassis) external interfaces are:

LM-to-OMS (Online Monitoring Station) interface using the UDP-based interface protocol for communications via fiber optical medium. This communication interface is online and safety related;

– Tuning interface with Tuning Computer using the proprietary Radiy Tuning Interface protocol via fiber optical medium. This communication interface is offline and not safety critical;

– Tuning Access Interface provides a safety related discrete input to de-energize Tuning interface. This communication interface is online and safety related.

RadICS uses several types of data communication:

– Client-server (safety-related point-to-point as in the I/O Module response to interrogation by the LM);

– Broadcast (non-safety related as in the on-line reporting of plant data and RadICS Platform status to OMS);

– Transformational (as in a transceiver used for communication between LMs in the same division, safety-related, or between LM and Tuning Computer, non-safety related)

In all cases, two things are specified for the communications link to meet safety requirements:

– Sequence of operations;

– Data packet construction, including validation data and demonstration that it is of adequate capability to detect errors for the.

The RadICS Platform data transmission protocols are divided into two levels:

– Transport Level Protocols – intended to transmit (transport) fixed amounts of any data and to provide Channel Level Protocol organization; data integrity checks are part of protocol service data.

– Channel Level Protocols – intended to organize a data exchange channel with defined data structure; consist of multiple frames of Transport Level Protocols; data integrity check means can be present as part of data.

RadICS Platform security assurance measures include secure development environment, the RadICS Platform vulnerability assessment, and implementation of the secure development and operational environment controls.

The RadICS Platform secure development environment is designed to provide measures and controls taken to establish a secure environment for development of the digital safety system against undocumented, unneeded, and unwanted modifications and protective actions taken against a predictable set of undesirable acts (e.g., inadvertent operator actions or the undesirable behavior of connected systems) that could challenge the integrity, reliability, or functionality of a digital safety system during operations.

The QTS design process shall comply with requirements to safety and security. Results of safety assurance and assessment of the RadICS Platform and the RadICS based I&C systems can be found in [5-7].

Security assessment of safety critical I&C system shall be done in accordance with the U.S. Regulatory Guide (RG) 5.71, “Cyber security programs for nuclear facilities” [8].

RG 5.71 is a direct guidance for activities implementation under the Code of Federal Regulations 10 CFR 73.54 “Protection of digital computer and communication systems and networks” and describes the methods and security activities for the operation and maintenance of a NPP, including appropriate I&Cs. Accordingly, cyber security features should be designed and implemented during the development phase of the I&C, before its installation. The guide embodies the findings by standards organizations and agencies, such as the International Society of Automation, IEEE, and NIST, as well as guidance from the U.S. Department of Homeland Security; however, it does not provide any specific processes related to life cycle of the I&C [9].

In this way, security controls, which are in compliance with the RG 5.71, additionally should be planned, designed, and implemented during development phase of the I&C.

One of the possible ways in such establishment is in development and maintenance of a cyber security program, regulated by appropriate Cyber Security Program Plan. Such document should contain description for specific security assurance processes for the I&C, which should be followed in development phase: namely the set of activities, including measures and controls taken to establish a secure environment for development of the digital I&C against undocumented, unneeded and unwanted modifications, as well as protective actions taken against a predictable set of undesirable acts that could challenge the integrity, reliability, or functionality of a I&C during operations.

Secure development and operational environment establishment process requires that the development process should identify and mitigate potential vulnerabilities in each phase of the I&C lifecycle that may degrade the secure development or operational environment or degrade the reliability of the I&C. Vulnerabilities assessment process can be performed in different ways, and sections below describe one of the possible approaches.

From the variety of I&C’s attributes one of the most important is dependability, which is the ability to deliver required services (perform functions) that can justifiably be trusted.

Dependability is a complex attribute that can be decomposed into a set of first-order attributes, including safety and security.

Safety of an I&C ensures absence of catastrophic consequences for the user(s) and the environment.

Cyber security, in terms of international normative documents, is defined as a protection mechanism that ensures:

– Confidentiality (the property that information is not made available or disclosed to unauthorized individuals, entities, or processes);

– Integrity (protection of the accuracy and completeness of the information and methods of processing);

– Authenticity (the confidence that the information comes from the correct source and / or the system trust the source code);

– Availability (access to information and associated assets of authorized users as needed);

– Reliability (entities involved in the processing, or communication, should not be able to refuse to exchange data).

In order to associate I&C life cycle of exact attribute with each of the I&C's components within the component-oriented V-model (see Fig. 42.3) we consider one more plane, which is the attributes plane formed by a set of components' attributes. Hence, there are already two planes: components plane and attributes plane; further, in a bundle with the life cycle, it is possible to address the aspect under interest in three-dimension space defined by the three coordinates, related with the I&C component, I&C attribute, and the life cycle stage. Attribute-oriented extension of a component-oriented V-model of I&C's life cycle allows us to independently assess each of I&C components and attributes of the component at the component-specific life cycle stage. In terms of modern FPGA-based I&C s, the main components within life cycle are represented by hardware (including printed circuit boards with FPGA chips), FPGA electronic designs and software tools intended for the I&C (re)configuration.

The proposed extension allows separation of specific life cycle stages for each of components' attributes (for example, confidentiality) to reveal discrepancies of appropriate development processes that can potentially result in anomalies (for example, vulnerabilities) of the final product (complex I&C).

Appropriate criticality matrix is depicted in Fig. 45.5. Each of the numbers inside the matrix represents an appropriate row number of Intrusion Modes and Effect Criticality Analysis (IMECA) table. From cyber security assurance point of view, the possible way of risk reduction is in decreasing of attacks' occurrence probability, since related damage is constant. A worst-case criticality diagonal for the matrix is marked out by bold line; acceptable values of risks are below the diagonal. Cases of probability decreasing for numbered rows are denoted by dotted lines with arrows: the problem is in decreasing of the probability by the degree sufficient to move a row of IMECA table below the criticality diagonal.

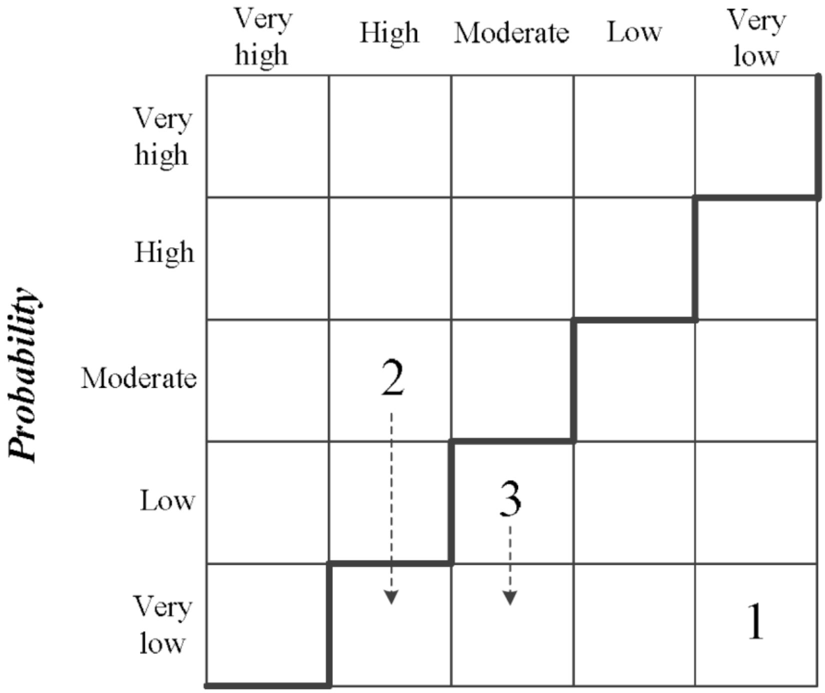


Fig. 42.5. The Criticality Matrix based on IMECA

In terms of FPGA-based NPP I&C systems, such decreasing of the probability can be achieved, for example, by implementation of certain process countermeasures during implementation of development processes or specific countermeasures during operation and maintenance stage on the basis of results of proposed approach application.

The RadICS Platform designer (Company Radiy) conducted vulnerability assessments for the RadICS Platform development environment to identify security vulnerabilities and identify potential security measures to mitigate identified vulnerabilities. RadICS also used RG 5.71 to understand potential U.S. customer requirements.

The RadICS Platform development environment vulnerability assessment included:

– Analysis of potential hardware vulnerabilities for the development environment;

– Analysis of potential software vulnerabilities for the development environment;

– Analysis of potential configuration vulnerabilities for the development environment;

– Analysis of potential network vulnerabilities for the development environment.

A list of possible threats to the development environment was developed. Each potential development environment vulnerability was assessed against the possible threats. Appropriate security measures for each vulnerability and effective means to implement them were identified. The results of the RadICS Platform development environment vulnerability assessment were documented in a vulnerability assessment report.

RPC Radiy has implemented a secure development procedure that specifies the security controls for the RadICS Platform development and operational environment. The procedure provides guidance for designing digital systems (both hardware and software) to ensure that they are free from vulnerabilities that could affect the reliability of the system.

The RadICS Secure Development and Operational Environment Procedure specifies the security controls for the RadICS Platform development and operational environment. It describes the methodology for complying with regulatory requirements for promoting high functional reliability, design quality, and a secure development and operational environment for the use of safety-related digital systems used in nuclear power plants. The procedure specifies the measures and controls taken to establish a secure environment for development of the digital system against undocumented, unneeded, and unwanted modifications and protective actions taken against a predictable set of undesirable acts, for digital assets, that could challenge the integrity, reliability, or functionality of a digital system during operations.

The RadICS Platform secure development environment is supported by corporate Information Security Management System policies and procedures. Company instructions define the process for identification of information that is commercial secret or confidential information and define controls for circulation, processing, storage, and disposal, define the control of physical access to development areas.

Company instructions also define formal security training and awareness program that was developed to keep staff up to date on prescribed organizational security policies and procedures, define the RadICS Platform development process that implement security controls identified in the development environment vulnerability analysis

Periodic audits of security-related activities are conducted to verify that controls are effectively implemented.

The RadICS Platform design features identified in the development environment vulnerability analysis are related with communications, platform diagnostics, and access control features.

The RadICS Secure Development and Operational Environment Procedure specifies that a project-specific vulnerability assessment be prepared that contain results of vulnerabilities assessment and appropriate phase-specific protective actions and controls. This document is intended for implementation at development environment to cover all the revealed vulnerabilities for the development stages.

16.3 Platform for PLC/FPGA security assessment

FPGA-based Security Controller (Platform for PLC/FPGA security assessment) has been developed to provide an opportunity to test security features of PLCs [10]. Security Controller architecture is a shield based so investigated FPGA-based board is a shield for a control motherboard. Therefore many types of shield boards with many types of programmable chips (MCUs and FPGAs) cam be designed and investigated in the future.

Security Controller implementation has been started from a concept with specification development. This specification describes requirements to the product as for a “black” box. After the product implementation these requirements have been used as an input for testing. Architecture of a tool set includes the following components:

– Arduino MEGA serves as a motherboard with performing functions of control and measurement as well as data transmission via USB-interface to a Personal Computer (PC);

– Arduino MEGA software supports functions of the motherboard Arduino MEGA;

– Specially designed Security Controller board with a target device what is Altera Cyclone IV FPGA; this device is used as a connected shield for Arduino MEGA; FPGA is programmed with a target code and security features can be investigated with support of Arduino MEGA;

– A target Security Controller board code (FPGA electronic design) which is installed in Cyclone IV to investigate security features;

– A software monitor for PC desktop using named Security Controller GUI (Graphical User Interface); this software provide service functions such as data visualization and archiving.

The following ideas have been considered as inputs for Security Controller concept:

– Security Controller may be designed with using both commercially available kits and self-developed Printed Circuit Boards (PCB);

– Security Controller shall include power supply unit to provide different voltage ranges for the boar component; this feature also could be used to investigated operation in different failure modes (for example, with high or low power supply voltage);

– Security Controller shall include current measurement unit for monitoring of current and energy consumed by the investigated FPGA;

– Security Controller shall support different clock frequencies for FPGA to investigate operability in different modes;

– Security Controller shall include temperature measurement unit;

– Security Controller shall support standard interfaces (USB, RS-232, Ethernet, IrDA) to be easy connected with PC and other devices.

The developed concept has been used as an input for design of Security Controller. A functional diagram is presented on Figure 6.

A design core of Security Controller includes Arduino MEGA board connected with a shield on the base of Altera Cyclone IV (Security Controller board). This board supports interfaces with Security Controller board and interfaces with PC. Built-in USB interface supports data transmission to PC as well Arduino board power supply and Arduino MCU programming. Two the last functions can be performed with alternative power supply and programming interfaces as it is presented on Fig. 42.6.

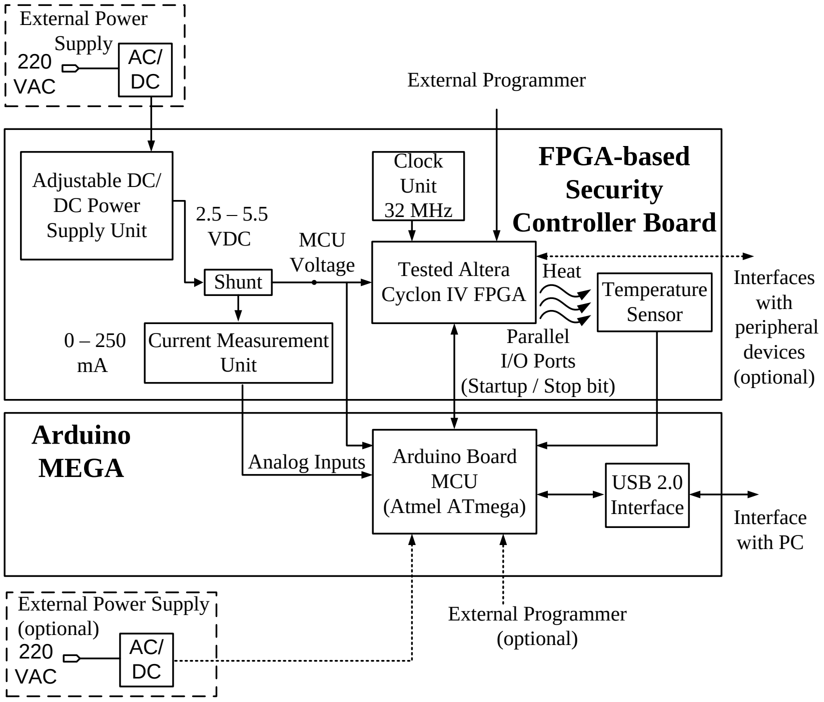

External Power Supply
220 VAC
AC/DC
External Programmer
FPGA-based Security Controller Board
Adjustable DC/DC Power Supply Unit
2.5 – 5.5 VDC
Clock Unit 32 MHz
MCU Voltage
Tested Altera Cyclon IV FPGA
Heat
Temperature Sensor
Shunt
Interfaces with peripheral devices (optional)
0 – 250 mA
Current Measurement Unit
Parallel I/O Ports (Startup / Stop bit)
Arduino MEGA
Analog Inputs
Arduino Board MCU (Atmel ATmega)
USB 2.0 Interface
Interface with PC
External Power Supply (optional)
220 VAC
AC/DC
External Programmer (optional)

Fig. 42.6. Functional diagram of Security Controller

An adjustable DC/DC convertor is used to provide stable voltage for the FPGA in the range 1.0-5.5 VDC. At the DC/DC convertor output an additional precise resistor (shunt) is installed. This shunt provides both value of voltage and value current to be measured and converted to electrical energy consumption. Security Controller board contains also temperature sensor and current measurement unit. It provides an opportunity to research power consumption depending on installed security features.

Clock unit provides up to 50 MHz frequency for the FPGA chip. The above features provide opportunities to investigate different frequency modes of the FPGA.

3-D model of Security Controller board is presented on Fig. 42.7.

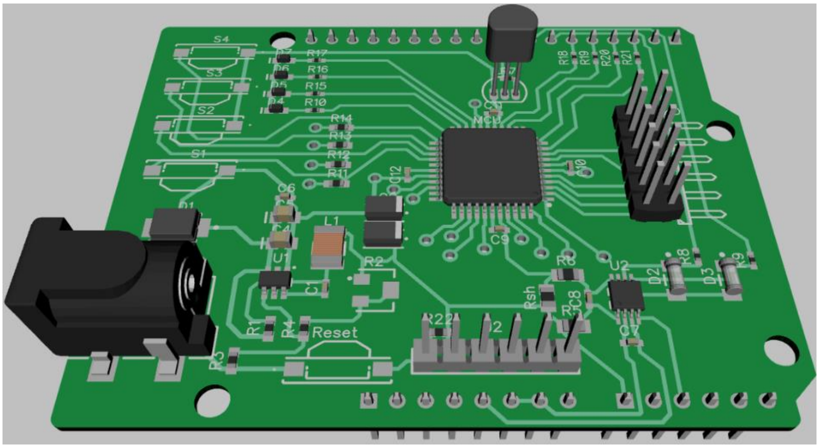


Fig. 42.7. 3-D model of Security Controller board

Conclusions

FPGAs combine the features of processors with the performance of custom hardware. As they are widely used in safety and security critical systems, special techniques are needed to manage security in FPGA designs. FPGAs can provide a useful balance between performance, rapid time to market, and flexibility. This section discuss the FPGA features, which can be beneficial from the point of view of security assurance.

The FPGA implementation relies on several sophisticated software tools created by many different people and organizations. Special-purpose processing cores, such as an Advanced Encryption Standard

(AES) core, can be distributed in the form of the hardware description language (HDL), netlists (which are a list of logical gates and their interconnections), or a bitstream.

One major problem is that it’s now possible to copy hardware, not just software, from existing products, and industry has invested in mechanisms to protect IP-cores.

Multiple scales of design present different potential attack vectors. Attacks at the device level can involve malicious software as well as sophisticated sand-and-scan techniques. Attacks at the board level can involve passive snooping on the wires that connect chips and the networks that connect boards as well as active modification of data traffic. There are security advantages to using a separate chip for each core, because doing so eliminates the threat of cores on the same device interfering with one another. This advantage must be weighed against the increased power and area cost of having more chips and the increased risk of snooping on the communication lines between chips.

Solutions to reconfigurable security problems fall into two categories: life cycle management and a secure architecture.

The RadICS Platform is a state-of-the-art digital control system platform specifically designed for safety-related control and protection systems in NPP applications. The RadICS Platform features a modular and distributed FPGA-based architecture.

The RadICS Platform design features identified in the development environment vulnerability analysis are related with communications, platform diagnostics, and access control features.

RadICS Platform security assurance measures include secure development environment, the RadICS Platform vulnerability assessment, and implementation of the secure development and operational environment controls.

The RadICS Secure Development and Operational Environment Procedure specifies that a project-specific vulnerability assessment be prepared that contain results of vulnerabilities assessment and appropriate phase-specific protective actions and controls.

FPGA-based Security Controller (Platform for PLC/FPGA security assessment) has been developed to provide an opportunity to test security features of PLCs. Security Controller architecture is a shield based so investigated FPGA-based board is a shield for a control motherboard.

Questions to self-checking

1. Which FPGA features can support system security?
2. Which are the main challenges in security assurance of FPGA-based systems?
3. Which FPGA components can affect security?
4. What is a purpose of FPGA security life cycle management?
5. Which components does FPGA secure architecture consist?
6. Describe the main features and parts of the RadICS platform.
7. Describe safety and security life cycle of the FPGA-based RadICS Platform.
8. Describe interfaces of the RadICS platform.
9. How is the RadICS Platform secure development environment implemented?
10. How is the RadICS Platform vulnerability assessment implemented?
11. How is the RadICS Platform operational environment control implemented?
12. Describe the main features and parts of Security Controller.

References

1. T. Huffmire, C. Irvine, T. Nguyen, T. Levin, R. Kastner, T. Sherwood. Handbook of FPGA Design Security. – Springer, 2010. – 177 p.
2. Nuclear Power Plant Instrumentation and Control Systems for Safety and Security / Yastrebenetsky M., Kharchenko V. (Edits). – IGI Global. – 2014. – 470 p.
3. K. Zetter. Countdown to Zero Day: Stuxnet and the Launch of the World's First Digital Weapon. Crown Publishers, 2014. – 433 p.
4. T. Huffmire, B. Brotherton, T. Sherwood, R. Kastner T. Levin, T. Nguyen C. Irvine. Managing Security in FPGA-Based Embedded Systems // IEEE Design and Test of Computers. – 2008. Vol. 25, Issue 6. – P. 590-598.
5. V. Sklyar, V. Kharchenko, E. Bakhmach, A. Andrashov. FPGA-Based I&C Systems: A Technological Trick or a Way to

Improve NPPs Safety and Security? // Proceeding of 20th International Conference on Nuclear Engineering ‘ICONE20-POWER 2012’.

6. V. Kharchenko, V. Sklyar. Assurance Case Driven Design for software and hardware description language based systems // Radioelectronic and Computer Systems. – 2016. – No. 5(79). – P. 98-103.

7. V. Sklyar. Safety-critical Certification of FPGA-based Platform against Requirements of U.S. Nuclear Regulatory Commission (NRC): Industrial Case Study // Proceedings of the 12th International Conference on ICT in Education, Research and Industrial Application ‘ICTERI 2016’. – P. 129-136.

8. Regulatory Guide 5.71, “Cyber security programs for nuclear facilities”, U.S. Nuclear Regulatory Commission, 2010.

9. V. Kharchenko, A. Kovalenko, V. Sklyar, O. Siora. Security Assessment of FPGA-based Safety-Critical Systems: US NRC Requirements Context // Proceedings of the International Conference on Information and Digital Technologies ‘IDT 2015’. – P. 117-123.

10. V. Kharchenko, V. Sklyar, E. Brezhnev, A. Boyarchuk, O. Starov, C. Phillips. University-Industry Cooperation in Cyber Security Domain: Multi-Model Approach, Tools and Cases // Practitioners Proceedings 2016 University-Industry Interaction Conference (UIIC). – P. 265-283.

АННОТАЦИЯ

Раздел содержит обзор специфических особенностей ПЛИС, относящихся к обеспечению информационной безопасности, поскольку ПЛИС комбинируют свойства процессоров и аппаратных средств.

В разделе рассмотрено обеспечение информационной безопасности для платформы RadICS на базе ПЛИС (в качестве индустриального кейса). Платформа безопасности RadICS была создана для разработки систем управления и защиты АЭС.

Также раздел содержит описание устройства, которое было разработано для тестирования информационной безопасности программируемых контроллеров.

Розділ містить огляд специфічних особливостей ПЛІС щодо забезпечення інформаційної безпеки, оскільки ПЛІС комбінують якості процесорів и апаратних засобів.

У розділі розглянуто забезпечення інформаційної безпеки для платформи RadICS на базі ПЛІС (у якості індустріального кейсу). Платформу безпеки RadICS було розроблено для реалізації систем управління та захисту АЕС.

Також розділ містить опис пристрою, який було розроблено для тестування функцій інформаційної безпеки програмованих логічних контролерів.

The section contains survey of special FPGA capabilities related with security assurance since FPGAs combine the features of processors with the performance of custom hardware.

Security assurance for FPGA-based RadICS safety Platform is considered as an example of industrial case. The RadICS Platform is a state-of-the-art digital control system platform specifically designed for safety-related control and protection systems in NPP applications.

Finally section contains a general description of Security Controller which has been developed to provide an opportunity to test security features of Programmable Logic Controllers.

Part 11.
Techniques and Tools for Industry FPGA-Based Systems Security

ABBREVIATIONS

CPU	Central Processing Unit
FaaS	FPGA as a Service
FfaS	FPGA for a Service
GPU	Graphics Processing Unit
JTAG	Hardware interface based on IEEE 1149.1
LFSR	Linear Feedback Shift Register
NLFSR	Nonlinear Feedback Shift Register
PRS	Pseudo-Random Sequence
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus

43. IMPLEMENTATION OF SECURITY FUNCTIONS USING FPGA AS A SERVICE

The growing demand for the services provided by cloud technologies is due to their advantages over traditional computing. Accessibility anywhere, relatively low requirements for computing power of the client machine, and as a result, lower power consumption for end user, saved hardware and time resources – all greatly accelerate this trend.

Applying the FPGA technology in a cloud computing is of great interest nowadays [1-3]. It allows to propose more energy-efficient solutions for data centers [4, 5], especially in cases where some part of a cloud infrastructure can be deployed on FPGA platform.

It also can significantly speed up many performance-intensive tasks (or services): from digital signal processing (digital video processing, audio signal processing, spectral estimation, speech recognition, imaging processing, biomedicine, radar, sonar, etc.) to specific mathematical calculations for science, such as finding of generator polynomials for nonlinear feedback shift registers (NLFSRs) [6]. This task is exactly suitable to be solved using FPGA and may drive the progress of remote FPGA programming for science purposes.

This chapter provides the analysis of existing FaaS solutions, proposes FaaS deployment techniques and the case-study of applying FaaS for finding of polynomials for stream encryption systems. This example is about how to use FPGA as a part of secure device and as a tool for creation of the new crypto primitives. The provided information will allow security engineers to create their own efficient task-dedicated cloud service for solving problems that can not be done using classical architectures.

43.1 FPGA as a Service

43.1.1 FPGA and Cloud technology Interaction

The FPGA application analysis shows that FPGA can interact with a cloud infrastructure on three different levels [7]. They are the following:

– FPGA as a Service (FaaS) – providing end users with "raw" FPGA resources and giving them ability to define their own projects.

– FPGA for a Service (FfaS) – providing end users with already defined services (e.g. audio/video processing, specific DSP algorithms etc.). In this case customer may not even know about the hardware configuration/characteristics.

– FPGA for Cloud Infrastructure – the case when FPGA is used to support cloud infrastructure itself (networking, virtualization platform etc.).

Among the variety of existing levels, FaaS still requires more attention and research because there are many methods, tools and techniques that can be applied.

43.1.2 FPFA as a Service Task Classification

One of the main advantages of the FPGA technology is the ability to implement non-standard hardware based solutions, especially when microprocessors show low efficiency/resource capability, and production of ASIC is still unreasonable due to the price per product unit. Also, there are special types of computational tasks, which show better performance and energy-efficiency for FPGA compared with CPU-based solutions and the increase is immense for GPU-based solutions [8].

Exactly for such type of tasks FPGA resources may be provided as a service. Typically that is data processing for science, imaging, cryptography, medicine and industry. The data flow of such tasks is shown in Figure 43.1.

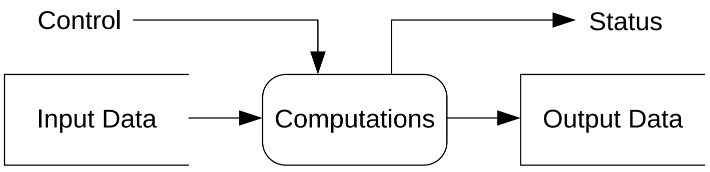


Figure 43.1 – The data flow of typical task for FPGA as a service

According to symmetry of data path throughputs the FaaS tasks is classified into:

– Symmetric Tasks – symmetrical throughputs (for symmetric algorithms);

– Input-oriented Tasks (IOT) – dominance of input dataflow (computationally intensive tasks such as filtering, inverse problem solving, convenient number search, or brute force search);

– Output-oriented Tasks (OOT) – dominance of output dataflow (all types of data generators).

– Symmetrical or output oriented throughput tasks require a communication channel with a high bandwidth. In this case, the PCI Express or Gigabit Ethernet interfaces are strongly recommended. A large amount of tasks with non-intensive data-exchange operations may be implemented in

FPGA even without fast communication channels, simply via USB, UART and other widely used interfaces.

43.1.3 FPGA as a Service Deployment Approach

In spite of all the advantages of the FPGA PCIe-based platforms, they are still too expensive, and when dealing with input-oriented tasks more cost-effective solutions can be applied.

The proposed coarse-grained FaaS infrastructure consists of several components (see Figure 43.2):

– FPGA Development Kit Boards, connected to a server via the USB interface;

– A server machine with running JTAG Server, FaaS Tasks WEB Server and Server-side application for distributing and collecting data;

– User PC with CAD Tool and a WEB browser.

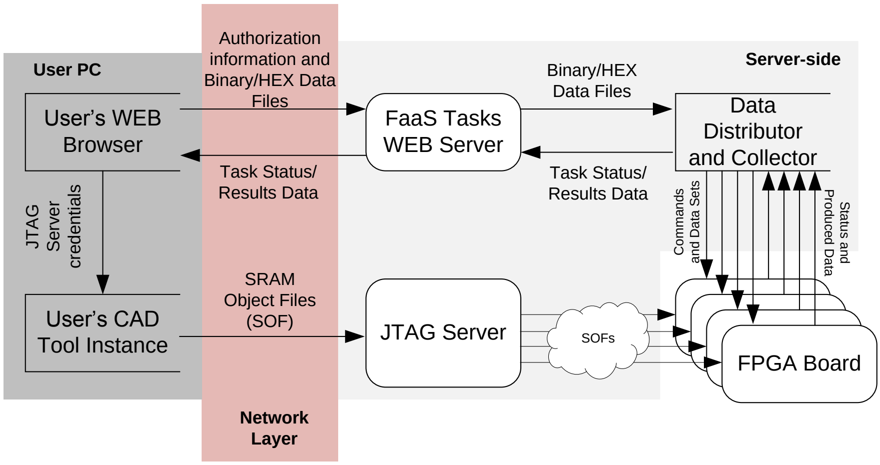


Figure 43.2 – Coarse-grained FaaS infrastructure

The efficiency of task implementation in FaaS depends on its parallelization feature. If the data dependency can be reduced and the algorithm allows the parallelization, it's possible to organize the scaling of the FPGA system in a wide range of available resources. The implementation of tasks using multiparametrization allows creating a universal project, which can be used in a wide range of FPGA chips without redesigning.

43.2 Finding of polynomials for NLFSR using FPGA

43.2.1 Implementation of Pseudo-Random Generators in a Hardware

The basic element of the cryptographic stream encryption system is a generator of pseudo-random sequences (PRSs). The main requirements to PRS are the indiscernible of the sequence, complexity, speed and repetition period. Cryptographic primitives that meet these requirements can be based on linear feedback shift registers (LFSRs).

Common cryptographic algorithms, which are built using LFSRs, are: stream cipher A5/1, used to ensure the privacy of telephone mobile communication of GSM standard; stream cipher E0, used in the Bluetooth protocol [9], etc.

The main disadvantage of LFSR is the linearity, which leads to a relatively simple cryptanalysis.

43.2.2 Applying of Nonlinear Feedback Shift Registers

Nonlinear Feedback Shift Registers can be used as an alternative to LFSRs in PRSs generation for the stream cipher. The NLFSRs are included to different stream ciphers: Achterbahn, Dragon, Grain [10], Trivium and VEST.

NLFSRs are more resistant to cryptanalytic attacks than LFSRs. Using L cells NLFSRs, a cryptanalyst can take up to $O(2^L)$ [11]. A sequence of $L(L+1)/2 +L$ bits is necessary to determine the structure of L-bit NLFSR that is used to generate this sequence.

At the same time, finding of polynomials for large size NLFSR with a guaranteed period is unsolved problem.

It is known that LFSRs generate maximum length sequence (M-sequence) equal to 2^L-1 if and only if the characteristic polynomial is primitive. For NLFSRs such property is not found to this day. Small NLFSRs with a maximum period can be built with the help of simulation. Nevertheless, modern computing capacities allow to simulate NLFSRs with a size only $L<35$ [12]. This is insufficient for cryptographic applications that require long periods, for example, 2^{128}.

Non-standard FPGA-based implementation of cryptographic algorithms allows to accelerate the search of polynomials for NLFSR.

One of the important advantages of the implementation of cryptographic algorithms in FPGAs is the ability to construct a parallel and asynchronous architectures. The performance of such approach is higher than for GPU and CPU-based platforms.

43.2.3 Common structure of NLFSR of second order

If NLFSR register use multiplication of only two cells such registers are called NLFSRs of the second order. General NLFSR architecture for the register of L=4 cells is shown in Figure 43.3.

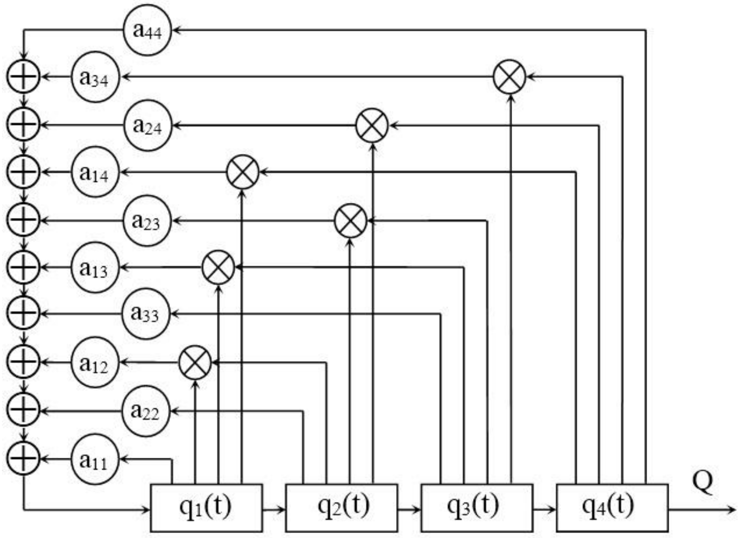


Figure 43.3. Universal NLFSR architecture

Where a_{ij} defines the presence or absence of the feedback in register, $q_i(t)$ is value of the i-th register cell at the time t, Q is the generated bit of sequence. Preliminary feedback coefficients a_{ij} are limited that equate some of the coefficients to zero. The total number of variables (initially not equal to zero) of feedback coefficient a_{ij} denoted as n_1. The number of feedback coefficients a_{ij} for NLFSR with size of L is n_L=L(L+1)/2, therefore, n_1 can take values in the range between 0 and n_L. [13]

43.2.4 NLFSR Polynomials Search Method

NLFSR polynomials search method consists of two stages that can be carried out simultaneously or sequentially.

At the first stage some NLFSR polynomials are excluded. Only generator polynomials for generation of M-sequence shoud be presented in the collection. The total amount of rejected polynomials is selected in the range of 90-99% of the possible amount. The volume of rejected polynomials is limited by available memory and time resources.

Rejected set is defined by an analytical method. This step implemented as software tool. The analytical method consists of checking the feedback coefficients to meet certain requirements. The essence of these requirements is to analyze the place, type and arrangement of non-zero coefficients of feedback a_{ij}. Mismatch of a_{ij} coefficients to specified requirements means the inability of NLFSR to generate M-sequence.

After the first stage, we obtain the set of polynomials, acceptable (for time-consuming) to check in the second stage.

At the second stage all polynomials prepared at first step should be verified. During direct verification it is necessary to find the polynomials that produces the M-sequence.

To increase the performance of M-NLFSR search it is possible move this process from CPU to a FPGA implementation.

43.3 Case-Study: Brute Force Search of NLFSR Polynomials

To verify the proposed approaches, FaaS was deployed. A set of Altera DE2 boards was connected to the host computer (server) via USB interface. In order to give the remote access and ability to program FPGA boards, the JTAG server was configured.

The described service was used for solving a scientific problem dealing with the brute force search of polynomials for nonlinear feedback shift registers. This task is input-oriented, which means that it has the dominance of pre-generated on the user side input data. The data set consists of millions of coefficients for non-linear polynomials. The coefficients of each polynomial can be processed separately. Therefore, this task is completely suitable for parallelization by means of multiparametrization. Only a small part of these coefficients can generate maximum length sequence.

To achieve the best performance, the implementation of search block was separated from the other parts of the project using dual port RAMs. The dataflow in one channel (FPGA) is shown in Figure 43.4.

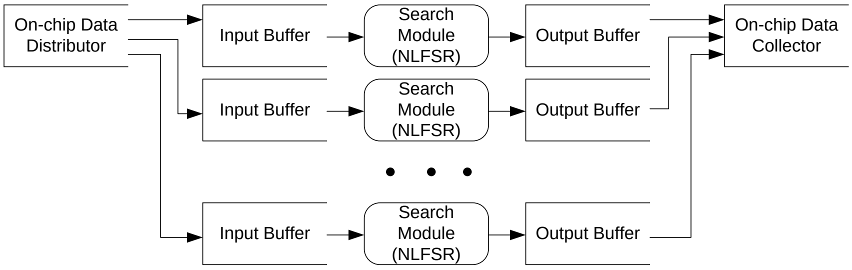


Figure 43.4. Scalable implementation of brute force search task for FaaS

Full amount of input data was divided between four FPGA boards programmed with the same project (the same SOF). The distribution and collection of data was carried by a custom communication software.

This example of FaaS shows that such type of tasks can be parallelized not only inside of an integrated circuit (chip) but also between independent boards. If multiparametrization was used during development of task implementation, the project may be ported to another chip family without redesigning. FaaS in this case can include various types of boards.

43.4 Conclusion

Over the past years the FPGA technology has taken a big leap towards conquesting the cloud service market. The common trend is that the growth in this area is still in progress due to the advantages that programmable logic can offer.

In this chapter FPGA as a Service was considered. The analysis of existing techniques showed that one should take into account a great variety of different approaches of deploying FaaS. In most cases the choice depends on many factors, from the final cost of FaaS infrastructure to type of task that is going to be executed. It is also showed that according to the symmetry of data path throughputs the FaaS tasks can be classified to symmetric, input-oriented, and output-oriented tasks.

A cost-effective solution for FPGA as a Service was proposed. This architecture can be recommended for so called input-oriented tasks, where the requirements for the input and output dataflow are not so strict. To verify the proposed approach, a high performance computational task was carried out, used for solving scientific problem based on brute force search of polynomials for nonlinear feedback shift registers of second degree. It allowed saving the time and resources to get final results.

Also the method for M-NLFSRs search using hardware-software system is given. It is shown that the use of FPGA-based solutions can significantly improve the performance of complex for search of M-NLFSR. The time for M-NLFSR search is increased by more than 150 times per chup compared to using only the computing power of PC.

FPGA-based NLFSR performance capabilities and recommendations on optimizing the performance of the complex to search of M-NLFSR are given.

On the basis of the results it can be developed stream ciphers with enhanced cryptographic strength. The application of obtained non-linear polynomial allows to obtain stream ciphers, which are analogues of modern developments in this field, and the use of the described search method allows to find higher degree M-NLFSR.

Questions and tasks for self-control

1. Which ways of interaction between FPGA and Cloud exist?
2. What kind of tasks is applicable for FPGA platform?
3. What is FaaS?
4. Which parts includes typical FaaS infrastructure?
5. What is NLFSR?
6. What kind of algorithms and standards use NLFSR?
7. How to determine the order of NLFSR?
8. What are the advantages of applying NLFSR?
9. What is M-polynomials?
10. What is the main challenge in finding of M-polynomials?

References

1. Chen, F., Shan, Y., Zhang, Y., Wang, Y.: Enabling FPGAs in the Cloud. In: Proceedings of the 11th ACM Conference on Computing Frontiers Article No. 3, pp. 1–10. Cagliari, Italy (2014). doi: 10.1145/2597917.2597929

2. Gupta, P.: Xeon+FPGA platform for the data center, https://www.ece.cmu.edu/

3. Fahmy, S. A., Vipin, K., Shreejith, S.: Virtualized FPGA accelerators for efficient cloud computing. In: IEEE International Conference on Cloud Computing Technology and Science, pp. 430–435. Vancouver, Canada (2015). doi: 10.1109/CloudCom.2015.60

4. Yanovskaya, O., Yanovsky, M., Kharchenko, V.: The concept of green Cloud infrastructure based on distributed computing and hardware accelerator within FPGA as a Service. In: Proceedings of the IEEE East-West Design & Test Symposium (EWDTS), pp. 45–48. Kiev, Ukraine (2014). doi: 10.1109/EWDTS.2014.7027089

5. Neshatpour, K., Malik, M., Ghodrat, M. A., Sasan, A., Homayoun, H.: Energy-efficient acceleration of Big Data analytics applications using FPGAs. In: BIG DATA '15 Proceedings of the 2015 IEEE International Conference on Big Data (Big Data), pp. 115–123. Washington, DC, USA (2015). doi: 10.1109/BigData.2015.7363748

6. Poluyanenko, N.: Development of the search method for non-linear shift registers using hardware, implemented on field programmable gate arrays. In: EUREKA: Physics and En-gineering, pp. 53-60 (2017). doi: 10.21303/2461-4262.2017.00271

7. Kolesnyk, I. N., Kulanov, V. O., Perepelitsyn, A. E.: Analysis of FPGA Technologies Ap-plication as a Part of Cloud Infrastructure. In: Radioelectronic and computer systems, 6 (80), pp. 130–135 (2016).

8. Fowers, J., Brown, G., Cooke, P., Stitt, G.: A performance and energy comparison of FPGAs, GPUs, and multicores for sliding-window applications. In: FPGA '12 Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays, pp. 47–56. Monterey, California, USA (2012). doi: 10.1145/2145694.2145704

9. Shaked, O. (2006). Cryptanalysis of the Bluetooth E0 cipher. citeseer.ist.psu.edu/744254.html.

10. Hell, M., Johansson, T., Meier, W. (2005). Grain - a stream cipher for constrained environments. citeseer.ist.psu.edu/732342.html.

11. Dubrova, E., Teslenko, M., Tenhunen, H. (2008). On analysis and synthesis of (n,k) – non – linear feedback shift registers. in Design and Test in Europe, pp. 133–137.

12. Dubrova, E. (2014). A Method for Generating Full Cycles by a Composition of NLFSRs. Designs, Codes and Cryptography, ISSN 0925 – 1022, E – ISSN 1573 – 7586, November, Vol. 73, № 2, 469 – 486 p.

13. Potii, A., Poluyanenko, N. (2008). Analyz svojstv reghystrov sdvygha s nelynejnoj obratnoj svjazjju vtorogho porjadka, gheneryrujushhykh posledovateljnostj s maksymaljnim peryodom. Prykladnaja radyoelektronyka, № 3, S. 282 – 290.

44 TECHNIQUES AND TOOLS FOR FPGA-BASED SYSTEMS SAFETY AND SECURITY ASSESSMENT USING FAULT INJECTION

44.1 Design Fault Injection-Based Technique and Tool for FPGA Projects Verification

44.1.1 Design Fault Injection Approach and General Technique

General

General design fault injection life cycle is described in the [1]. It includes the following main phases:

– fault profiling;

– fault injection;

– results processing.

Thus, using this standard structure the design fault injection life cycle will be specified for FPGA projects below.

The first phase is design fault profiling, which means that the FPGA design fault set (profile) should be formed, i.e. those types of design faults, which are going to be injected, should be selected from the general fault profile and presented as one collection.

At the second phase the selected type's faults are being inserted in the description (VHDL code).

Some known techniques [2] insert faults during the simulation process but it restricts the space of code which is accessible for injection. Besides, in the case of design faults in the real practice faults are being made before simulation and implementation. So there is no need to make design able to rebuild itself. The faulty VHDL description generation is more convenient and suitable.

When the faulty code is obtained it is possible to process in the appropriate way.

Design Faults Profiling

As was above mentioned there is a necessity of specifeing general FPGA design fault profile.

The following classification of design faults may be chosen [3]:

- faults in signal/variable names;
- faults in constants;
- operator faults;
- faults in assignments;
- conditional statement faults;
- faults in component instantiation.

Taking into account this classification it is proposed to classify faults accordingly to the HDL-statements. It will extend the possibilities of fault impact analysis. Consequently, the following design fault profile was formed:

- wrong signal assignment (simple signal assignment, conditional signal assignment, selected signal assignment);
- wrong variable assignment;
- wrong condition (if, elsif, if generate, while, wait until, conditional signal assignment);
- wrong declaration (signal declaration, variable declaration, port declaration, constant declaration, type (subtype) declaration);
- wrong port instantiation.

Every fault can be caused by the designer's errors. The set of possible errors is listed below:

- wrong object name (signal, variable, attribute, constant, component);
- wrong value;
- wrong statement location;
- wrong arithmetic operator;
- wrong logical operator;
- wrong relative operator;
- wrong type;
- wrong brackets location.

But it is considered to leave the term "design fault" at the VHDL statement level.

Therefore, the following table can be constructed (see Table 44.1).

Table 44.1 Design faults classification

Design Fault / **Error (design fault source)**	Wrong signal assignment	Wrong variable assignment	Wrong condition	Wrong declaration	Wrong port instantiation
Wrong value	▨	▨	▨	▨	▨
Wrong statement location	▨	▨		▨	
Wrong arithmetic operator	▨	▨			
Wrong logical operator	▨	▨	▨		
Wrong relative operator			▨		
Wrong type	▨	▨		▨	
Wrong brackets location	▨	▨	▨		
Wrong name	▨	▨	▨	▨	▨

Design Fault Injection

Fault injection procedure is very responsible phase of fault injection life cycle.

As the modern FPGA designs is complex (for some safety related NPP I&C systems number of the LOCs equal thousand) it would be reasonable to use an appropriate tool for the fault injection. According to this, the fault injection procedure can be performed as:

- automatic (user only can choose a file and fault type);
- semiautomatic (user can chose a file, a fault type, code area to inject fault and a fault from offered fault set);
- manual (user can choose a file, a fault type, code area to inject fault, fault is inputted manually by user).

We suggest to use manual fault injection. The result of this procedure will be the faulty description (faulty file will be generated).

Such way of injection aims to free the initial design description from possible residual faults, which could be present in the case of original file modification. The drawback lies in the required additional free disc space for the faulty descriptions storing.

Faulty VHDL code processing

Fault injection procedure has no sense without any analysis of faulty code. Therefore the modeling of faulty VHDL design is proposed. The simulation results will show:

- whether applied test-cases reveal all injected faults;
- whether any latent faults present;
- how different faults affect the design.

44.1.2. Tool for FPGA Fault Injection

To make the proposed approach feasible the design fault injection tool was developed.

Tool functions are the following:

- design fault profiling for selected VHDL description;
- fault injection in accordance to obtained fault profile.

Besides, the crucial requirement consists in the following: fault injection procedure has to provide syntactically and semantically correct VHDL files.

Tool Architecture

Before the tool architecture development the fault injection environment was analyzed [2]. Proposed tool consists of the next modules (Figure 44.1):

- graphical user interface (GUI);
- syntax analyzer;
- design fault profile generator;
- fault injection manager;
- faulty files generator.

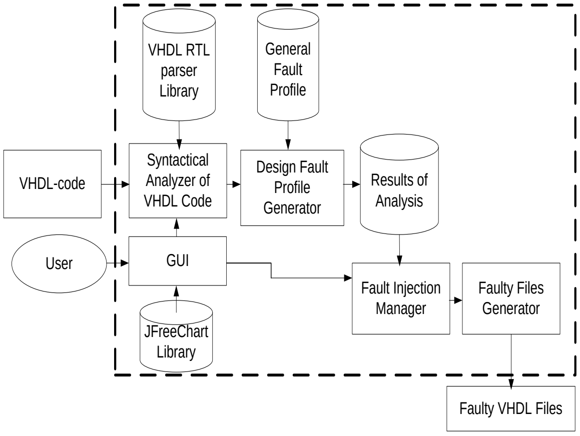


Figure 44.1 Architecture of fault injection tool

Key features

Developed design fault injection performs design fault profiling and design fault injection. It has easy to understand Graphic User Interface (GUI). Concerning the former task developed tool possesses the following features:

- one or more file(s) selection for analysis;
- automatic file correctness check;
- file viewer;
- design fault selection;
- generation of the design fault profile diagram for selected files;
- generation of text information about obtained design fault profile.

In order to inject faults the tool affords a fault injection manager, with which help user can:

- select target file;
- select type of fault;
- select the target line of code;
- chose type the fault into selected code;
- inject one or more faults in one file;
- view the faulty file with marked faulty lines of code.

44.1.3. Application of Design Fault Injection in Invariant-based Verification

The fault injection approach may be used in order to assess the quality of invariants, which are the base of model-checking invariant-based verification process for FPGA designs [4,5]. There is a invariant quality model including the following attributes:

- conciseness and formality of description;
- clarity and physically meaning;
- measurability;
- criticality;
- traceability;
- universality;
- ability to control;
- ability to diagnose;
- flexibility;
- non-redundancy.

IIf the quality assessment of HDL-model invariants is going to be performed the design fault injection procedure will be required. Such attributes as criticality, ability to control and consequently ability to diagnose can be determined in this way.

44.2 Multi-Fault Injection Testing: Cases for FPGA-Based NPP I&C Systems

44.2.1 Fault-injection for FPGA-based NPP I&Cs safety

Last ten years FPGA technology is intensively developed and applied in safety important instrumentation and control systems (I&Cs) of nuclear power plants (NPP) and other critical domains due to flexibility, dependability, security and maintainability of system decisions. Application of FPGA-based I&Cs is attended by evolving of the verification methodology and techniques.

The fault injection/insertion testing (FIT) is one of the effective techniques in an arsenal of independent verification and validation (V&V) procedure for software and FPGA-based safety critical systems including NPP I&Cs [6]. Besides, FIT is a procedure to certify platforms and I&Cs against requirements of IEC 61508 according with safety integrity level (SIL) [7,8]. FIT is based on design fault injection into the software code (VHDL code for FPGA), physical faults into programmable chips and hardware modules to assess test coverage and quality of testing as a whole, on-line testing and fault-tolerance [6-9].

Initially FIT technique was developed and implemented for software and software-based I&Cs. Taking into consideration the features of FPGAs and FPGA-based I&Cs methodology and technique of fault injection should be modified according with the requirements of the standard IEC 61508 [10] and regulatory document NUREG/CR-7151 "Development of a Fault Injection-Based Dependability. Assessment Methodology for Digital I&C Systems" [6]. This document, in particular, contains the requirements to multi-fault injection testing (MFIT). MFIT implies injection of a few faults on the one or several levels of system hierarchy.

The key idea of using MFIT is the stress-testing of the modernized or newly created I&Cs or its components by two, three or more faults injecting at the same time. This technique speeds up the occurrence and the propagation of faults in a system for the purpose of observing the effects of faults on the system performance and behavior. The final goal of such testing is the increase of the functional and informational safety of NPP I&Cs.

44.3 t-Wise-Based multi-fault injection technique for the verification of safety critical I&C systems

44.3.1 t-Wise-Based MFIT for FPGA-based module

In this section, we present the industrial realization of a few elements from the t-wise-based, multi-fault injection technique. For this purpose, we observed the fault injection procedure over the Logic Module (LM), which was performed during the SIL-3 certification, according to the requirements of the standard IEC 61508 [11]. LM is a part of the FPGA-based digital RadICS platform [12].

The FMEDA approach is applied at the first stage of the t-wise-based, multi-fault injection technique. The list of injected fault types and subtypes presented in the form of FIT-table is a result of

FMEDA[13]. This table contains the complete list of injected faults with their detailed description. Such a table can comprise the following information:

- Definite names of the detectable symptoms.
- Modules implicated of FPGA-based platform; name of units with injected faults.
- The list of modules that are presented in the chassis for the test.
- Designers' recommendation how to make this symptom appear.
- Module's mode for the test.
- Importance of module or software modification for testing.

The obtained FIT-table is analyzed to determine the number of injected fault types and subtypes. Table 44.2 shows fragments of the analysis. LM is analyzed to find the points for injection of all faults subtypes. A few limitations are taken into account: element parameters are not changed under temperature/mechanical influences; fault injection for elements does not cause a failure or an unacceptable parameter changing of other ones; any element must be acceptable for fault injection.

Table 44.2 Description of types and subtypes of injected faults for Logic Module

Types of injected faults		**Subtypes of injected faults**		
Type	Description	Subtype	Subtype ID	Description
Lost V (LV)	Loss of voltage	Lost V PS	LV-P	Loss of voltage in power supply
		Lost V DIU	LV-D	Loss of voltage in discrete input unit
V change (VC)	Change of voltage	V low PS	VC-L	Decrease of voltage in power supply
		V high PS	VC-H	Overvoltage in power

				supply
Stuck on (SO)	Stuck on in high or low level	Stuck on DIU links	SO-D	Bad data caused by stuck on in DIU links
		Stuck on DIU switch	SO-S	Bad data caused by stuck on in DIU switch
Communication and clock faults (CC)	Loss or short of circuit	Loss of communication	CC-L	Loss of communication between modules
		Clock faults	CC-C	Short of circuit clock
Configuration and memory faults (CM)	Configuration error of FPGA unit	Package failure	CM-P	Bad data of package
		Configuration memory faults	CM-C	Change configuration data

Combinatorial coverage of different types of injected faults

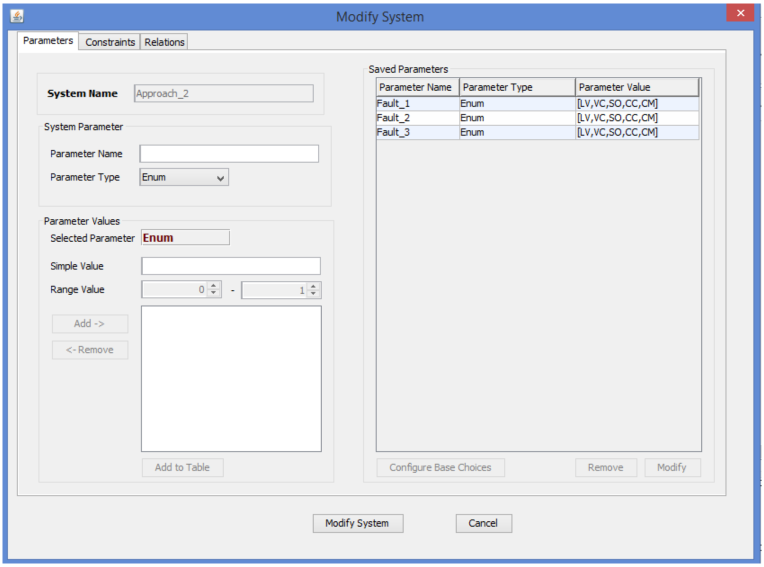
At the next step, we apply the t-wise approach to reduce the number of fault injections, but we still have a proper coverage of different combinations of types of faults. Let k be the number of possible types/subtypes of faults and n be the number of faults that are simultaneously injected. The value of k does not depend on n and could be much bigger than n. One n-tuple of values of fault types represents one simultaneous injection of n faults. Our approach is to generate a set of n-tuples in such a way that all combinations of fault types of any t faults (t<n) are covered.

The t-wise coverage is usually used in practice for t from 1 to 6. However, the number of requested combinations can be too large for fault injection when t>2. That is why the most practical approach is the 2-wise (pair-wise) coverage with three injected faults. The aim is a selection of the minimum number of injections of three faults which cover all possible pairs of types for all pairs of faults. In this case study, values n=3, t=2, k=5, and k=10 are considered. To generate a set of fault injections, tools can be used.

ACTS tool

We use the Advanced Combinatorial Testing System (ACTS) tool [14, 15] to generate different variants of fault injections according to the t-wise (pair-wise) approach. ACTS has been developed by the U.S. National Institute of Standards and Technology (NIST) and the University of Texas at Arlington for combinatorial t-wise software test generation for $1 \le t \le 6$. The tool is based on the mathematical covering array technique and uses several different algorithms, including various versions of In-Parameter-Order-General (IPOG) algorithm [16].Though ACTS is a software testing tool, it can be directly applied without any changes for the generation of the different variants of multi-fault injection.

Input data of ACTS are a list of testing parameters (in our case, faults for injection) and their values (in our case, types of the faults). ACTS allows using a big number of parameters and their values and, in the case of fault injection, this numberis more than sufficient for any practical application. The output value of the tool is a list of tests (in our case, variants of multi-fault injection) that guarantees the t-wise coverage. Users can establish some logical constrains (restrictions) that must be satisfied by output values. It is very important in our case because, as mentioned above, some combinations of faults may not be valid for injection. The tool has a simple and convenient GUI interface as well as a command line interface and generates output results very fast.



System Name: Approach_2

Parameter Type: Enum

Selected Parameter: Enum

Range Value: 0 - 1

Saved Parameters

Parameter Name	Parameter Type	Parameter Value
Fault_1	Enum	[LV,VC,SO,CC,CM]
Fault_2	Enum	[LV,VC,SO,CC,CM]
Fault_3	Enum	[LV,VC,SO,CC,CM]

Figure 44.2. Description of possible faults in ACTS

44.4 Cyber Security Assessment of Component Off-the-Shelf Based NPP I&C System Using IMECA Technique

44.4.1 Cots vulnerability analysis of NPP I&CS

COTS software can be used in safety systems and other systems important to safety in nuclear power plants. Examples of such products are compilers, operating systems, software supplied in Programmable Logic Controllers (PLCs), and software in commercial industrial digital control systems [17].

Buying commercial products seems to be an easy decision. COTS application allows reducing development time and decreasing production and maintenance costs. Vendors of commercial products have track records that prove qualification of both the assignable product and the developers.

Examples of commercial components in modern NPP I&C systems are:

– Emerson Ovation Distributed Control System;

– Triconex Tricon;
– ABB AdvantPower;
– Siemens/Areva Teleperm XS;
– Siemens SPPA-R3000;
– Alstom Alspa P320.

COTS components are required to meet standards for software quality assurance, verification and validation for the use in nuclear power plants. These standards are presented in Table 44.3.

NP-T-3.12 regulatory guide [18] describe procedure of commercial off-the-shelf products application. Functional qualification of COTS is difficult. It is named “commercial dedication” and consists of several stages: identifying the critical characteristics required by its safety functions, special tests, inspections, supplier evaluations, verification during fabrication, operating history to demonstrate that both the functions and the quality of the product are appropriate for the specific system. The main complication is that functional qualification of COTS is often impossible without interaction between vendor and user. Many vendors are unwilling to give proprietary information or sources codes.

Table 44.3 The standards required for COTS

Standard	**Name**
IEEE 730 (Now 730.1)	IEEE Standard for Software Quality Assurance Plans
IEEE 828	IEEE Standard for Software Configuration Management Plans
IEEE 1042	IEEE Guide to Software Configuration Management
IEEE 7-4.3.2	Standard Criteria for Digital Computers in Safety Systems of Nuclear Power Generating Stations
ISO 9000-3	Guidelines for the Application of ISO 9000-1 to the Development, Supply, and Maintenance of Software
ANSI/ANS-10.4	Guidelines for the Verification and Validation of Scientific and Engineering Computer Programs for the Nuclear Industry"
ANSI/IEEE 1012	IEEE Standard for Software Verification and Validation Plans

IEC 880,	Software for Computers in the Safety Systems of Nuclear Power Stations
IEC 987	Programmed Digital Computers Important to Safety for Nuclear Power Stations

COTS products may be of higher quality and reliability than components special developed for NPP systems. Although safety assurance of commercial facilities may be very expensive, purchase of commercial solutions is an easy decision.

Commercial and public databases

In nuclear power industry, many databases provide information of vulnerabilities and possible failures. In those sources information is available in the form of standards, book, and electronic database. Some of such sources are listed below:

– International Atomic Energy Agency (IAEA);

– EIREDA European Industry Reliability Data Handbook;

– T-Book Reliability Data of Components in Nordic Nuclear Plants;

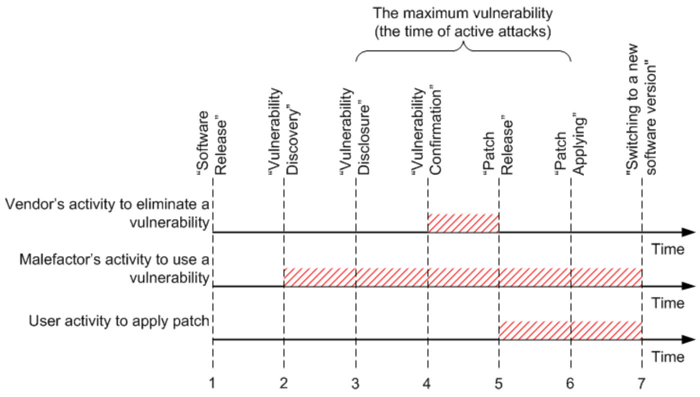
– Nuclear Plant Reliability Data Systems (NPRDS);

– U.S. NRC, Data Summaries of Licensee Event Reports of Selected Instrumentation and Control Components at US Commercial Nuclear Power Plants [17].

Public vulnerability databases also contain information about NPP vulnerabilities. Examples of such sources are Common Vulnerabilities and Exposures (CVE) and National Vulnerability Database (NVD).

Tool for collection of vulnerability date

Vulnerability information is stored in databases. Some of them are public and open for common Internet users. Every day, about a dozen new vulnerabilities are published there. Various sources publish information about the vulnerability at certain stages of its lifecycle. The vulnerability lifecycle is depicted on Figure 44.3.



The maximum vulnerability (the time of active attacks)
"Software Release"
"Vulnerability Discovery"
"Vulnerability Disclosure"
"Vulnerability Confirmation"
"Patch Release"
"Patch Applying"
"Switching to a new software version"
Vendor's activity to eliminate a vulnerability
Malefactor's activity to use a vulnerability
User activity to apply patch
Time
Time
Time
1
2
3
4
5
6
7

Figure 44.3 Vulnerability lifecycle

Information in the CVE (Common Vulnerabilities and Exposures) database appears in the time of vulnerability discovery. Usually you can see the vulnerability name, description and status. More detailed information of vulnerable products is made available in NVD (National Vulnerability Database) after the patch release. The most dangerous are zero-day attacks that target publicly known but still unpatched vulnerabilities.

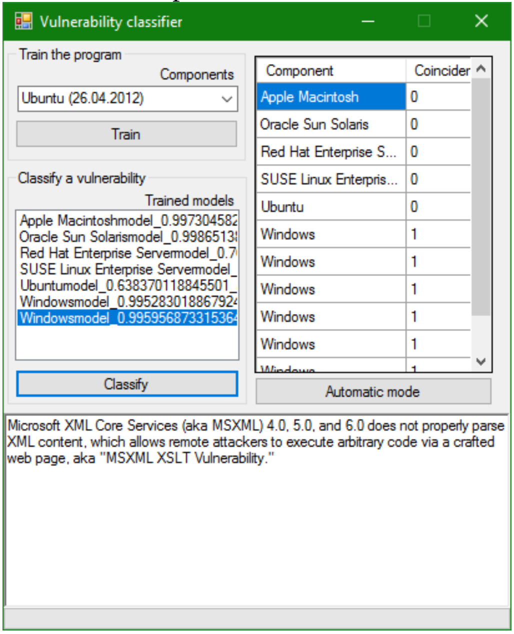
We developed a tool for collecting data from public vulnerability databases (CVE and NVD) and detection of zero-day vulnerabilities. The main idea was to use description from CVE for making prediction about components, which can be vulnerable. The "Vulnerability classifier" tool interface is shown in Figure 44.4.

For text classification, SVM (Support Vector Machine) algorithm was applied. The machine learning based on the training model. Each training example marked as concerned one or other categories. SVM algorithm is non-probabilistic binary linear classifier because SVM algorithm assigns new items to one or other category based on training model. In this case, unique training model was created for each operation system. Vulnerability was ranged in two categories: related to that operation system or not. SVM algorithm allows determining which component is vulnerable before publication official information in NVD.

Checking the validity of the results has been proved based on data from NVD. Table 44.4 shows how the obtained results correspond to reality.

For example, the prediction of vulnerabilities related to Apple Macintosh were correct in 98% of cases. For Linux distributions, this figure is much lower, so that the prediction is less accurate. Other algorithms can be applied to improve the accuracy. It will be the next step of our research.

Therefore, suggested tool helps to determine vulnerable products before they become public. In this case, components can be changed or updated until vendor creates patch.



Apple Macintoshmodel_0.997304582
Oracle Sun Solarismodel_0.9986513
Red Hat Enterprise Servermodel_0.7
SUSE Linux Enterprise Servermodel_
Ubuntumodel_0.638370118845501_
Windowsmodel_0.99528301886792
Windowsmodel_0.99595687331536

Component	Coincider
Apple Macintosh	0
Oracle Sun Solaris	0
Red Hat Enterprise S...	0
SUSE Linux Enterpris...	0
Ubuntu	0
Windows	1
Windows	1
Windows	1
Windows	1
Windows	1

Microsoft XML Core Services (aka MSXML) 4.0, 5.0, and 6.0 does not properly parse XML content, which allows remote attackers to execute arbitrary code via a crafted web page, aka "MSXML XSLT Vulnerability."

Figure 44.4 Vulnerability classifier user interface

Table 44.4 Correspondence table (%)

Apple Macintosh	Windows	Oracle Sun Solaris	Red Hat Enterprise Server	Ubuntu	SUSE Linux Enterprise Server
98.26	99.27	99.46	85.64	52.25	56.63

44.4.2 IMECA Technique

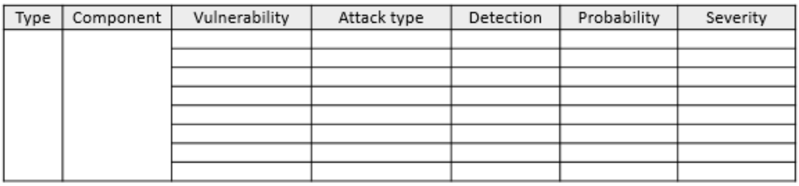
IMECA analysis

Various methods are used to analyze cyber security of I&C systems important to safety. In this paper, the Intrusion Modes and Effects Criticality Analysis (IMECA) technique was applied for cyber security assessment. The “Standards and Standardization: Concepts, Methodologies, Tools, and Applications” [20] contains the following definition of IMECA: IMECA is a modification of FMEA that takes into account possible intrusions to the system.

FMEA as rule is used during the design period to prevent failures in future. IMECA technique can be used when an intrusion or failure occurs.

The process of analyzing components based on IMECA technique is the following:

Step 1: Each component represents a special IMECA table. Example of IMECA table is depicted on a figure 44.5.



Type	Component	Vulnerability	Attack type	Detection	Probability	Severity

Figure 44.5 IMECA table

One table may contain many vulnerabilities related to component. Each vulnerability has a description that is made of columns of the table (attack type, detection, probability and severity).

Step 2: Each of these vulnerabilities is assessed in five grades in the following characteristics: detection, probability and severity. Grades that were used are the following: “Very high”, “High”, “Medium”, “Low”, “Very low”. Usually table cells are filled manually. We suggest a tool that allows filling the table automatically.

Step 3: Creating of criticality matrix based on grades described above.

Criticality matrix

Criticality matrix is the way for formalizing results of IMECA analyze. Each number in matrix represents a special row in IMECA table. Example of criticality matrix is shown in figure 44.6.

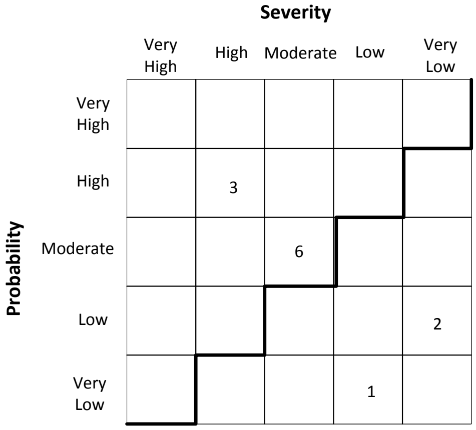


Figure 44.6 Criticality matrix

The bold line is criticality diagonal. Acceptable values are below critical diagonal, values that are above are the worst case. The key challenge is to reduce probability, which is depicted on Figure 44.7. Such decreasing of probability can be solved by implementing appropriate countermeasures.

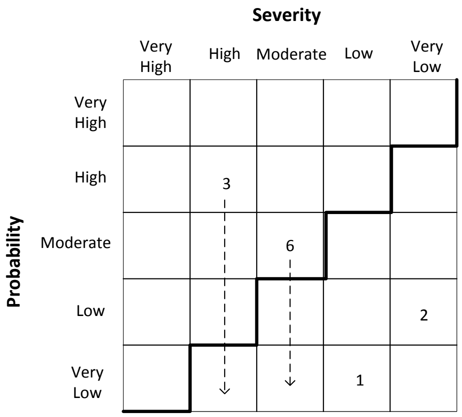


Figure 44.7 Decreasing of probability

44.4.3 Technique of cyber security analyses

The procedures

The suggested technique of the NPP I&C systems security assessment consists of the following procedures:

Step 1: Analysis of I&C architecture to assess influence of OTS components failures on dependability (reliability and safety) of the system.

Step 2: The IMECA based assessment of OTS components and their configuration. Each component is presented as an individual IMECA table, each vulnerability is a single row in that table. Creation of criticality matrixes is based on IMECA analysis.

Step 3: Determination of the cyber security of the system depending on the criticality of the components present in the system.

Step 4: Developing of Security Assurance Case and selecting of countermeasures according to safety (cyber security) COTS criteria.

Analysis of I&C architecture

On the first stage, we determine components, which require analysis of I&C configuration. At this stage, we suggest an integrated approach to the evaluation system presented on Figure 44.8. The automation tool loads the configuration of the system and scan information from public databases (NVD and CVE). Expert selects the components for further analysis.

As a result, we get list of components for next evaluation.

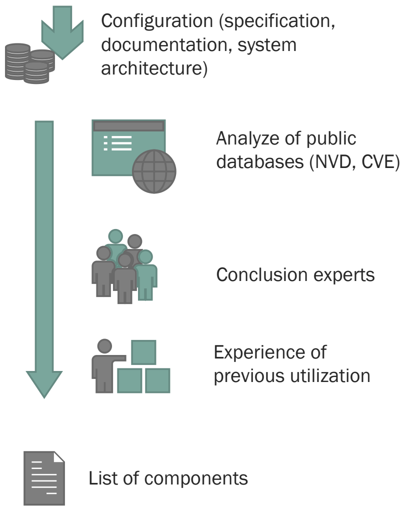

Configuration (specification, documentation, system architecture)
Analyze of public databases (NVD, CVE)
Conclusion experts
Experience of previous utilization
List of components

Figure 44.8 Analysis of I&C architecture

The IMECA-based assessment of OTS components

Each component represents a local IMECA table, which contains the following columns: vulnerability, attack type, detection, probability and severity. Each vulnerability is special row in that table. The suggested tool provides the ability to automatically download vulnerabilities related to this component from public NVD database and/or to add them manually.

Expert evaluates each vulnerability by determining its level of severity, probability and detection in range: “Very low”, “Low”, “Medium”, “High”, “Very high”.

Processing of experts’ evaluation of appropriate columns in IMECA table allows creating IMECA criticality matrix. By this means, each row in IMECA table forms one cell in critical matrix. Suggested technique is show on Figure 44.9.

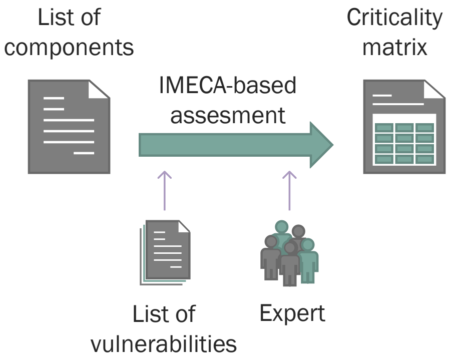


Figure 44.9 IMECA-based assessment

Joining of criticality matrixes

Joining of criticality matrixes allows getting an overall picture of the criticality of the system. The tool combines all criticality matrixes that have been received in the previous stage. The total criticality matrixes Software IMECA Criticality Matrix (SwICM) and Hardware IMECA Criticality Matrix (HwICM) are obtained by merging of the components of criticality matrixes. Principle of criticality matrixes joining is shown on Figure 44.10.

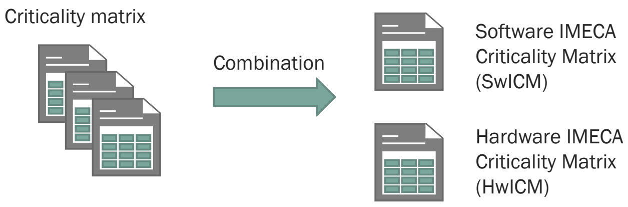

Criticality matrix
Combination
Software IMECA Criticality Matrix (SwICM)
Hardware IMECA Criticality Matrix (HwICM)

Figure 44.10 Joining of criticality matrixes

For this purpose, the following rule was used:

$$e_{yz}^{G} = \bigcup_{k=1}^{n} e_{yz}^{L_k}$$

were:

– e^{G} is an element of the total criticality matrix,

– e^{Lk} is the corresponding element of the k-th local criticality matrix,

– n is the total number of local criticality matrixes [19].

Determination of the cyber security of the system

Before defining the countermeasures to eliminate vulnerabilities and improve safety it is necessary to calculate the level of criticality of the system.

Calculation of the total risk of R was calculated as follows:

$$R = \sum_{i=1}^{n}\sum_{j=1}^{m} p_{ij}D_{ij}$$

where:

– n is the total number of components,

– m is the total number of rows in the IMECA table,

– p is the occurrence probability,

– D is the corresponding damage [19].

In this paper criticality matrix is 3-dimensional (K=3).

This step aimed to identify the most critical places in I&C system. It is possible to manage the security risks by prioritization of appropriate rows in the total matrix (the smallest row number in the matrix corresponds to the smallest risk of occurrence) [19].

After prioritization of vulnerabilities it is possible to achieve security of the I&C system by implementing special countermeasures.

44.5 Verification of FPGA based NPP I&C systems considering multiple faults: technique and automation tool

44.5.1 FPGA based NPP I&C systems verification considering MFA

There are several techniques for verification of NPP I&C systems safety and reliability, that include FMEA (as well as its modifications FMECA, FMEDA etc.), RBD, FTA, MM and others. One of the most commonly used is FMEDA technique.

FMEDA technique

FMEDA is systematic way to identify and evaluate effects of system components' failure modes, to determine what could eliminate or reduce the possibility of system failure. FMEDA is an extension of FMEA technique, that takes into consideration online diagnostic capabilities of a system and the failure modes relevant to safety instrumented system design[25].

Usually experts manually apply FMEDA technique. Due to complexity of NPP I&C systems, application process is very time and resource consuming, including a lot of routine work with different kinds of documents (datasheets, regulatory documents, guides, etc.).

Hence, manual technique application indirectly leads to errors caused by human factor (i.e. loss of attention) which will probably cause system failures.

Techniques automation is one of the ways of minimization of experts' involvement in assessment process and decreasing of experts-related errors number. Examples of tools that are used for FMEDA automation include Exida FMEDA tool [21], AXMEA tool [20] developed by Department of Computer Systems and Networks, National Aerospace University "KhAI" in cooperation with RPC Radiy, PLATO-SCIO-FMEDA system [22] and others.

Although, usage of automation tools brings higher level of results quality, it does not allow complete experts exclusion from assessment process. In addition, there are stages that cannot be formalized and fully automated (i.e. identification of components failure modes and their effects on the system).

In consideration of the foregoing, we need to identify new approaches for improvement of trustworthiness of FMEDA-based NPP I&C systems safety and reliability assessment.

FIT-based verification of FMEDA

Generally, FIT technique is used to verify FMEDA results. The goals of FIT are described below:

– to assess the quality of tests considering test coverage or trustworthiness issues;

– to assess efficiency of online testing capabilities;

– to analyze fault-tolerance;

– to analyze security and intrusion-tolerance.

Considering FPGA-based NPP I&C systems, faults can be injected on design, chip, module, interaction levels of system (Fig. 44.11).

Moreover, during multiple FIT it is possible not only to inject few faults in one layer, but inject faults in different layers of system hierarchy. In our research we consider fault injection applied only on module level.

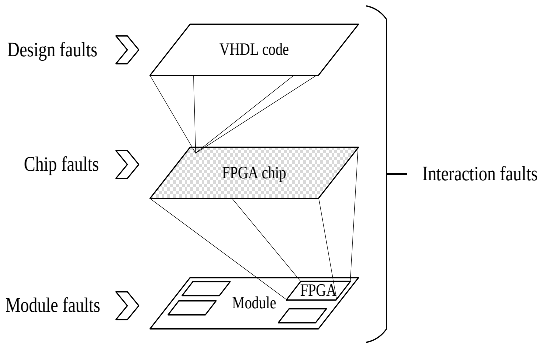


Figure 44.11 FIT targets

Design faults are injected in VHDL code, i.e. stuck-at-one, stuck-at-zero bits, data words bit flips. Chip fault injection implies modification of software flashed into the chip. Module faults include physical intrusion in hardware components (resistors or capacitors unsoldering, overvoltage, under voltage, short circuits, etc.). Interaction faults imply components communication problems, attacks on vulnerabilities.

Typical FIT environment is shown on Fig. 44.12.

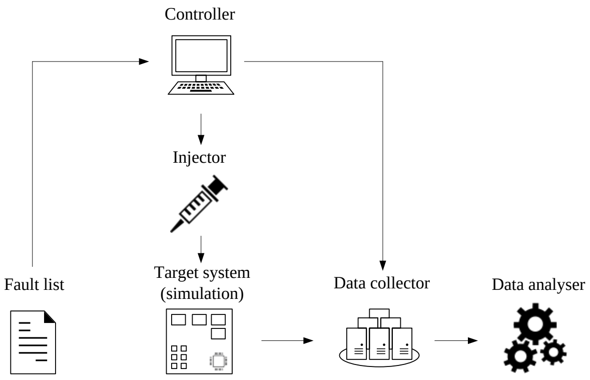


Figure 44.12 Typical FIT environment

Input for FIT technique is fault list obtained after FMEDA application. FIT controller provides fault list for injector that injects faults into the real system. Data collector collects data from system under test and provides it to the data analyzer.

Then basing on FIT results the design of system can be reviewed or previous FMEDA-based assessment results can be proven.

FMEDA and FIT bottlenecks

During FMEDA technique experts analyze system components for failure modes and possible failure effects. Due to the large number of components, this task becomes very monotonous and routine, especially, in case of manual FMEDA application.

Dimensionality problem is inherent for FIT as well as for FMEDA. Application of FIT is very time consuming, because the number of faults is large and each one has to be injected into the real system (i.e. components should be unsoldered). After injection the behavior of system should be analyzed for correspondence to FMEDA results, especially to defined failure effects.

Then system should be returned to initial state in order to prepare for next injection. In addition, physical FIT is expensive, because it requires real system instances to be under test. Thus, after testing they should not be used in production due to introduced damages.

In case of FIT application in computer simulated system, results depend too much on the accuracy of model.

Additionally, FMEDA and FIT technique processes are based on assumption that only one fault at a time is considered. In real systems, faults can chain exposure of others leading to system failures that were not considered at verification and validation stages. This level of uncertainty is inacceptable in safety critical systems.

44.5.3 The technique and tool

Joint MFA-based FMEDA and FIT verification technique (MF&F) is a combination one that is to be applied on safety-critical FPGA-based NPP I&C systems. It is focused on multiple fault nature of real systems. The input data for MFIT are obtained after FMEDA application, such as:

– the list of fault combinations for injection;

– their effects on the system;

– online diagnostic capabilities of the system.

The technique consists of several stages. The first is MFMEDA (Multiple FMEDA) process (Fig. 44.13). MFMEDA is combination of the standard FMEDA and multi-fault approach.

Decomposition of FMEDA technique as well as its application process are described in details in [23]. In order to improve the trustworthiness of safety assessment, the research in minimization of exerts' involvement in assessment process was carried out in [20].

Automation of FMEDA technique application was proposed and then implemented in AXMEA software tool. This tool is a platform for our current research activities in multiple faults nature and related problems.

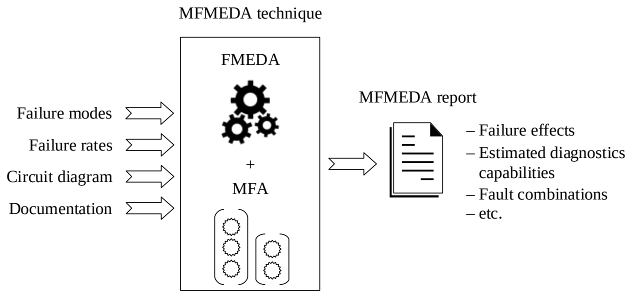


Figure 44.13 MFMEDA technique

Multi-fault approach for FMEDA allows improving of assessment coverage, because one of the most significant restrictions of FMEDA is the single fault at a time assumption, which is accepted in order to reduce dimensionality problem. The decomposition of MFA is performed and its structure is shown on Fig. 44.14.

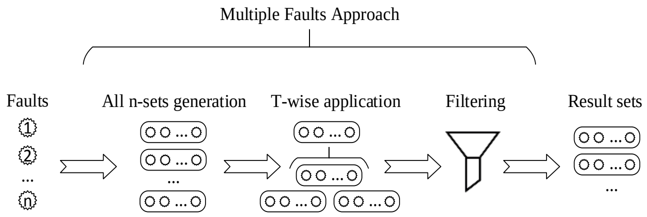


Figure 44.14 MFA structure

The inputs for multiple faults approach are the list of single faults obtained during FMEDA application. Next stage is to generate all combinations of input faults, which fault number in single group should be limited by n. Usually the number of faults in single group is three.

The number is small because it is not practically usable to inject more faults during FIT, moreover, even for n equal to three the number of fault combinations is too large. If we have 10 single faults as input and n equal to three, the total number of fault combinations is equal to 175. This number is already not injectable. In real systems the number of components is very large and each component has tens of possible faults. For MFA dimensionality problem is very significant, hence, it is necessary to decrease combinations number by application of special techniques.

The next stage is usage of t-wise combinatorial technique that allows combinations of multiple faults to be found and provides full coverage of all combinations of any t-types of faults with a minimum number of checks. The first case of t-wise application for fault injection is described in details in [24].

After reduction of fault combinations number, it is important to analyze each set for blocking, masking and prohibitive fault combinations. Such filtering procedure is proposed and its process stages are shown on Fig. 44.15.

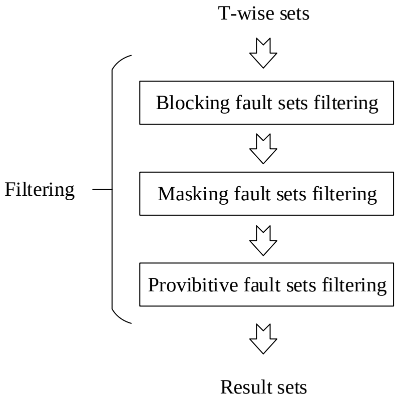

T-wise sets
Blocking fault sets filtering
Masking fault sets filtering
Provibitive fault sets filtering
Filtering
Result sets

Figure 44.15 Fault combinations filtering

Due to the three types of challenges, i.e. blocking, masking, prohibitive faults, the process consists of three stages. During each stage experts should analyze fault combinations for corresponding challenges. In case of inappropriate fault combination being found, it should be excluded from the result sets. Is some cases, after filtering is performed, the result sets cannot fully cover the initially generated non-limited fault combinations, because of sets exclusion during filtering. This cases require adding of new combinations that cover appeared gaps.

The extended procedure of new fault combinations addition as well as coverage verification is shown on Fig. 44.16. We propose coverage verification and new sets generation performing after each challenge type analysis.

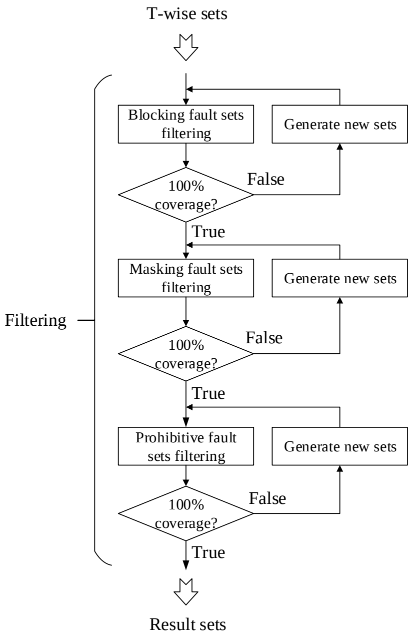


Figure 44.16. Extended fault combinations filtering

The last stage of proposed MFA-based FMEDA&FIT verification technique is application of automated multiple fault injection testing. The process of this stage is shown on Fig. 44.17.

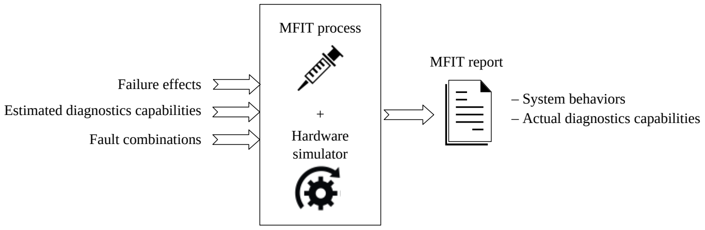


Figure 44.17 Fault combinations filtering

In order to minimize number of fault injections into the hardware of the real system, we propose usage of 3rd party simulation tools like National Instruments Multisim. The AMXEA software tool uses API of simulator for automatic faults injections. During simulation, experts observe the behavior of the model of the system under test and then fill the MFIT report that includes behaviors, estimated and actual online diagnostics capabilities of the system.

Research in verification methods for assessment techniques is carried out (shown in Table 44.5) in order to define indicators, which can be used for classification of MF&F results as correct or incorrect.

Table 44.5 Verification techniques for different quality models

Technique	Quality model	Verification technique		
		FIT	*Review*	*FAT, SAT, Operation*
FMEDA	Components coverage	±	+	–
	Failure modes coverage	±	+	–
	Qualitative indicators assignment	+	+	±

FIT	Accuracy of injection by component	n/a	+	+
	Accuracy of injection by fault	n/a	+	+
	Incorrect behavior interpretation	n/a	+	+

Basing on MFIT results the following activities should be performed (Table 44.6).

Table 44.6 Activities to perform after MF&F technique application

Results		Activities to perform
MFMEDA	MFIT	
Incorrect	Incorrect	Reapply MFMEDA
Incorrect	Correct	
Correct	Incorrect	Reapply MFIT
Correct	Correct	No actions to perform

Usage of proposed MF&F technique improves completeness of safety and reliability assessment by extension of fault coverage that is achieved by application of multiple faults approach. Moreover, trustworthiness of assessment is also improved by automation of technique and therefore minimization of experts' involvement.

Conclusion

Developed design fault injection procedure and tool can be applied not only for independent verification processes. They can be used to assess fault-tolerance and multi-version FPGA-based systems. In other words the comparison of different versions (developed according to one specification) using fault injection procedure is very interesting and may be direction of future research and engineering activities.

The MFIT is applied in process of certification to meet SIL requirements of IEC 61508. Formal concepts of MFIT procedure, technique and examples of MFIT development for Levels1&2 have been described.

The multi-fault injection technique is becoming one of the important and modern variants of the well-known FIT procedure. This technique allows researching the behavior and verifying a lot additional properties of safety critical I&C systems such as a tolerance to multiple faults of different components and different types (design, physical, interaction.

In the process of analyzing data from publicly available databases of vulnerabilities it was to established that the modules that make up the commercial components contain many vulnerabilities and pose a potential threat to security of I&C systems.

IMECA method was used to determine the criticality degree of system. To automate the process of cyber security assessment we used CyberSAs tool. This allowed minimizing the human impact on the use of IMECA technique.

The technique and tool are applied to assess cyber security at the preliminary stage of safety critical I&C development. It concerns software-based NPP I&C as well. Next steps will be dedicated to embedding of technique and tool into processes of cyber security assessment assurance for FPGA-based platform RadICS and NPP I&C systems as a whole.

The quality of MFIT simulation relies on the accuracy of the model. In order to decrease the number of physical fault injections it is necessary to perform research in different simulation tools and develop a procedure for selection of the most suitable one.

Questions to self-checking

1. Phases of general design fault injection life cycle?
2. What kind of procedures can be performed for fault injection?
3. What kind of attributes can be included into invariant quality model?
4. What is most effective technique which is in an arsenal of independent verification and validation (V&V) procedure for software and FPGA-based safety critical systems including NPP I&Cs?
5. What is the key idea of using MFIT?
6. What components are required in COTS?
7. The process of analyzing components based on IMECA technique?
8. Kind for formalizing results of IMECA analyze?
9. From which assessments consists technique of the NPP I&C systems security.
10. What is the main idea of FMEDA technique?

References

1. A. Gordeyev, V. Kharchenko , A. Andrashov et al. Case-based Software Reliability Assessment by Fault Injection Unified Procedures. // Proc. Software Engineering in East and South Europe(SEESE'08), May 13, 2008. Leipzig (Germany). – ACM, 2008.– P. 1–8.

2. H. Ziade, R. Ayoubi, R. Velazco. A Survey on Fault Injection Techniques. The International Arab Journal of Information Technology, Vol. 1, No. 2, July 2004.

3. M.R. Movahedin., P. Kindsmüller, V.Stechele. Modeling of VHDL Design Errors and Methods for their Correctability, 1996.

4. B.M. Konorev, V.S. Kharchenko (edits). Invariant-oriented quality assessment of space systems software, National Aerospace Agency of Ukraine, State Centre for Regulation of Deliveries and Services Quality, National Aerospace University "KhAI", Kharkiv, 2009.

5. A. Andrashov, V. Kharchenko, L. Reva et al. Verification of FPGA Electronic Designs for Nuclear Reactor Trip Systems: Test- and

Invariant-Based Methods // IEEE East-West Design & Test Symposium. – Sankt-Petersburg, Russia. – 2010. – P.92-97.

6. Fault Injection-Based Dependability. Assessment Methodology for Digital I&C Systems", Vol. 3, United States Nuclear Regulatory Commission

7. Kharchenko, V., Sklyar, V., Odarushchenko, O., Ivasyuk, O., 2014, "Fault Insertion Testing of FPGA-based NPP I&C Systems: SIL Certification Issues", Proceedings of the 22nd International Conference on Nuclear Engineering, Prague, Czech Republic, July 7-11

8. Kharchenko, V., Sklyar, V., Odarushchenko O., Ivasuyk A., 2013, "Fault-Injection Testing: FIT-Ability, Optimal Procedure and Tool for FPGA-Based Systems SIL certification", Proceeding of 11th East-West Design and Test Symposium, September, Rostov-on-Don, Russia, September 25-27

9. Cotroneo, D. (editor), 2013, "Innovative Technologies for Dependable OTS-Based Critical Systems: Challenges and Achievements of the CRITICAL STEP Project", Springer, Milan, Italy

10. Medoff, M., Faller, R., "Functional Safety - An IEC 61508 SIL 3 Compliant Development Process", www.exida.com

11. M. Medoff, R. Faller, "Functional Safety - An IEC 61508 SIL 3 Compliant Development Process," Exida (2010).

12 ."RadICS Platform," Radiy, http://radiy.com/en/nuclear/products/radics-platform.html (2014).

13 Failure Modes, Effects and Diagnostic Analysis – Rosemount 07-10-08 FMEDA Report 2051 R001 V1R1.doc, Page 2 of 20.

14 L. Yu; Y. Lei; R. Kacker, R. Kuhn, "ACTS: A Combinatorial Test Generation Tool," Proceedings of the Sixth International Conference on Software Testing, Verification and Validation (ICST 2013), March 18-22, Luxembourg, pp.370-375 (2013).

15 "ACTS tool," NIST, http://csrc.nist.gov/groups/SNS/acts/ (2014).

16 Y. Lei, R. Kacker, R. Kuhn, V. Okun, J. Lawrence, "IPOG: A General Strategy for T-Way Software Testing," Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems (ECBS '07), Tucson, Arizona, USA, pp. 549-556 (2007).

17 IEC 61226:2009 “Nuclear power plants - Instrumentation and control important for safety - Classification of instrumentation and control functions”, International Electrotechnical Commission, 2009.

18 IAEA Nuclear Energy Series Core No. NP-T-3.12, “Knowledge on Instrumentation and Control Systems in Nuclear Power Plants”, International Atomic Energy Agency Vienna ISBN 978-92-0-113710-4 ISSN 1995-7807.

19 Information Resources Management Association (USA), Standards and Standardization: Concepts, Methodologies, Tools, and Applications, 2015.

20 Kharchenko, V., Gordieiev, A., Vilkomir, S., Odarushchenko, O., 2015, “T-Wise-Based Multi-Fault Injection Technique for the Verification of Safety Critical I&Cs”, Proceedings of the 9th International Conference on Nuclear Plant Instrumentation, Control & Human-Machine Interface Technologies, Charlotte, North Carolina, pp. 13-22.

21 Babeshko, E., Kharchenko, V., Odarushchenko, O., Sklyar, V., 2015, “Toward automated FMEDA for complex electronic products”, 15402695, Proceedings of 2015 International Conference on Information and Digital Technologies (IDT), Zilina, pp. 22-27.

22 Yasko, A., Babeshko, E., Kharchenko, V., 2016 “FMEDA-based NPP I&C systems safety assessment: toward to minimization of experts’ decisions uncertainty”, ISBN: 978-0-7918-5005-3, Proceedings of 2016 24th International Conference on Nuclear Engineering, Charlotte, North Carolina, pp. V005T15A022.

23 SILcal tool, Exida, http://www.exida.com/Software

24 PLATO-SCIO-FMEDA tool, PLATO, http://www.plato.de/fmeda-495.html

25 FMEDA and Proven-in-use Assesment – gm international 04-10-27 r003 v2r0.doc, July 10 2007, Page 2 of 4

45 TECHNIQUES AND TOOLS FOR FPGA BASED SYSTEM DIVERSITY ASSESSMENT AND IMPLEMENTATION

45.1 Analysis of regulatory documents

45.1.1 Diversity related standards

As for today, there were developed basic regulatory documents that cover various aspects in the areas of FPGA, NPP I&C systems, and diversity issues. To assess diversity for NPP I&C systems the following safety standards are to be applicable:

a) documents of International Electrotechnical Commission (IEC):

- IEC 60880:2006. Nuclear power plants - Instrumentation and control systems important to safety - Software aspects for computer-based systems performing category A functions;
- IEC 62340:2007. Nuclear power plants – Instrumentation and control systems important to safety – Requirements for coping with common cause failure (CCF);
- IEC 61508:2010 Ed.2 Functional safety of electrical/electronic/programmable electronic safety-related systems;

b) documents of International Atomic Energy Agency (IAEA):

- IAEA. Safety standards series No. NS-G-1.1:2000. Software for computer based systems important to safety in nuclear power plants. Safety guide;
- IAEA. Safety standards series No. NS-G-1.3:2002. Instrumentation and control systems important to safety in nuclear power plants. Safety guide;

c) documents of the American National Standards Institute of IEEE and NUREG documents:

- ANSI/IEEE-ANS-7-4.3.2-2003. Criteria for Programmable Digital Computer System Software in Safety-Related Systems of Nuclear Power Plants;
- Nuclear Regulatory Commission. Diversity strategies for nuclear power plant instrumentation and control systems. NUREG/CR–7007: 2009.

All terminologies are agreed with [1]. Diversity is the presence of two or more redundant systems or components to perform an identified function, where the different systems or components have different attributes so as to reduce the possibility of common cause failure, including common mode failure.

45.1.2 Profiling of requirements to diversity

The analysis of standards allows determining the following groups of I&C NPP diversity requirements to: application of principle of diversity (types of systems); version product-process redundancy (diversity types); necessity of risk analysis of diversity application (additional risks which can be caused); diversity assessment (techniques of actual value diversity assessment).

The Table 45.1 presents a normative profile of qualitative requirements to diversity application in NPP I&C systems.

Table 45.1. Requirements to application of principle of diversity

Requirement	*Requirement wording*
R1. Requirement to application of principle of diversity	In order to achieve the safety and reliability requirements for I&C NPP that performs the reactor protection function the diversified means of carrying out the safety function should be incorporated (diversity principle).
R2. Requirements to types of diversity	The following types of diversity are applicable: (1) Design, (2) Equipment Manufacturer, (3) Logic Processing Equipment, (4) Function, (5) Life-cycle, (6) Signal, (7) Logic. As a rule, the combinations of these types are used.
R3. Requirements to necessity of risk analysis of diversity application	Degree of differences between I&C versions should be evaluated and the preferences shall be given for such combination that provides the biggest diversity. Besides, while carrying out the selection of versions some additional factors (complexity, cost, maintenance expenses) shall be considered as well.

Requirement	*Requirement wording*
R4. Requirements to diversity assessment	It shall be noted that diverse I&C shall also meet the reliability requirements. To prove that all possible I&C vulnerabilities the CCFs shall be considered. For all I&C important to safety, such features as diversity, redundancy, its feasibility for validations, stability, etc. shall be justified in the context of its adequacy for required reliability of safety function accomplishment. Such confirmation shall be based upon combination of deterministic criteria and quantitative reliability analysis.

45.2 Diversity assessment of NPP I&C systems

45.2.1 Diversity assessment technique NUREG7007-A

This method is an implementation of metric approach for diversity assessment. It uses the diversity classification described in NUREG 7007. The following attributes of diversity are considered: design diversity as application of different approaches, including both software and hardware, to solve the same, or a similar, problem; equipment manufacturer diversity as difference in vendors, manufactures, etc.; logic processing equipment diversity as differences in logic processing architecture, logic processing version in the same architecture, component integration architecture, data-flow architecture; functional diversity as characteristic of two systems as being functionally diverse if they perform different physical functions; life-cycle diversity as involvement different human beings in the design, development, installation, operation, and maintenance of safety systems; logic diversity as differences between both systems in terms of algorithms, logic, and program architecture, timing and/or order of execution, runtime environment, etc.; signal diversity as differences in sensed parameters to initiate protective action.

This method uses the double level classification of diversity, including the types and subtypes (attribute and criteria, correspondently). The method described in [2] is developed for two-version I&C. The diversity assessment is based on application of the two levels check-list filled by the experts. The check-list that contains the evidences (column details) supporting variant diversity assessment. The expert determines and marks diversity types presented in I&C systems. It is marked in check-list using value Yes or No. If diversity types or subtypes is applicable in the system and marked Yes, value INT = intentional (X) is marked as well.

Filling of particular diversity types/subtypes into check-list can automatically cause the appearance of corresponding ones, which either expert marks by INH = inherent (i).The weight of attribute depends on rate of application of the diversity type/subtype in I&C system. After the filling of check-list the diversity metrics are calculated as sum of weighted values of diversity types/subtypes (attributes and criteria). The diversity metric obtained after calculation is not normalized and can take any values in the range [0 – 1.76]. In this method, the diversity metric equaled 1.0 is considered as acceptable for two-version I&C system.

45.2.2 Diversity assessment of digital NPP I&C systems: extreme values

Let us calculate extreme diversity metric values for digital (CPU- or FPGA-based) NPP I&C systems for two variants: 1) primary (main) analog subsystem and secondary (diverse) digital subsystems (A-D); 2) primary (main) and secondary (diverse) subsystems are digital, but one of them is based on CPU, other one is based on FPGA (DC-DF). The results of calculations for the variant DC-DF is presented by the Tables 45.2. Analysis of calculation results allows concluding that a set of different diversity types and subtypes is enough to make a lot of decisions concerning selecting ones and to meet requirements. Extreme values of diversity metrics equal for two-version systems 1.44 and 1.38 correspondingly. These tables are an initial to analyse the possible variants of decreasing number of applied diversity types and subtypes.

Table 45.2. Diversity metrics for digital-digital I&C system

	Attribute criteria (diversity types/subtypes)			**Metrics**		
		Rank	**DCE WT**	**INT**	**INH**	**Score**
DESIGN	Design					
	Different technologies	1	0,500			0,000
	Different approaches within a technology	2	0,333	x		0,333
	Different architectures	3	0,167		i	0,167
	DAE weight and subtotals		1,000		0,500	0,500
EQUIP.MANUF.	Equipment Manufacturer					
	Different manufacturers of fundamentally different equipment designs	1	0,400	x		0,400
	Same manufacturer of fundamentally different equipment designs	2	0,300			0,000
	Different manufacturers of same equipment design	3	0,200			0,000
	Same manufacturer of different versions of the same equipment design	4	0,100			0,000
	DAE weight and subtotals		0,250		0,100	0,400
LOGIC PROC.EQUIP.	Logic Processing Equipment					
	Different logic processing architectures	1	0,400	x		0,400
	Different logic processing versions in same architecture	2	0,300			0,000
	Different component integration architectures	3	0,200	x		0,200
	Different data flow architectures	4	0,100	x		0,100
	DAE weight and subtotals		0,644		0,451	0,700

	Attribute criteria (diversity types/subtypes)			**Metrics**		
		Rank	**DCE WT**	**INT**	**INH**	**Score**
FUNCTION	Function					
	Different underlying mechanisms to accomplish safety function	1	0,500	x		0,500
	Different purpose, function, control logic, or actuation means of same underlying mechanism	2	0,333	x		0,333
	Different response time scale	3	0,167	x		0,167
	DAE weight and subtotals		0,600		0,600	1,000
LIFE-CYCLE	Life-Cycle					
	Different design companies	1	0,400	x		0,400
	Different management teams within the same company	2	0,300			0,000
	Different designers, engineers, and/or programmers	3	0,200	x		0,200
	Different implementation/validation teams	4	0,100	x		0,100
	DAE weight and subtotals		0,683		0,478	0,700
SIGNAL	Signal					
	Different reactor or process parameters sensed by different physical effect	1	0,500	x		0,500
	Different reactor or process parameters sensed by the same physical effect	2	0,333	x		0,333
	The same process parameter sensed by a different redundant set of similar sensors	3	0,167	x		0,167
	DAE weight and subtotals		0,867		0,867	1,000

Attribute criteria (diversity types/subtypes)				**Metrics**		
		Rank	**DCE WT**	**INT**	**INH**	**Score**
LOGIC	Logic					
	Different algorithms, logic, and program architecture	1	0,400	x		0,400
	Different timing or order of execution	2	0,300	x		0,300
	Different runtime environments	3	0,200	x		0,200
	Different functional representations	4	0,100	x		0,100
	DAE weight and subtotals		0,733		0,733	1,000
Score(*100)				373		
Normalized score				1,38		
Basis for normalizing		271				

The Table 45.3 contains values of diversity metric for extreme and other variants when some diversity subtypes are not applied, in particular F1(Different underlying mechanisms to accomplish safety function), LC1 (Different design companies), S1 (Different reactor or process parameters sensed by different physical effect), S2 (Different reactor or process parameters sensed by the same physical effect). In addition to two variants A-D and AC-AF described above we analyse third one when main and diverse subsystems are developed using CPU (DC-DC) or FPGA (DF-DF) only.

Table 45.3. Values of diversity metrics for two-version I&C systems

Variants	Values of diversity metrics				
	Maximal	Without F1	Without F1,LC1	Without F1,LC1,S1	Without F1,LC1,S1,S2
A-D	1.44	1.33	1.23	1.07	0.96
DC-DF	1.38	1.27	1.16	1.00	0.90
DC(DF)-DC(DF)	1.25	1.14	1.04	0.88	0.78

45.3 Diversity types selection model

Let a set of diversity (or version redundancy) types

$$MD = \{d_1, d_2, \ldots, d_D\}, \tag{45.1}$$

can be used to develop a multi-version system. It may be diversity of chip technologies (d_1), manufacturers of chips (d_2), families of chips (d_3), and others [3].

Let a set of diverse elements

$$MDE_j = \{e_{j1}, e_{j2}, \ldots, e_{jmj}\}, \tag{45.2}$$

corresponds to diversity type d_j. For example, the set MDE_1 consists of technologies FPGA (e_{11}), program logic controllers (e_{12}), microprocessors (e_{13}) and their subtechnologies.

A set

$$MDD = MDE_1 \, X \, MDE_2 \, X \ldots X \, MDE_D \tag{45.3}$$

forms all variants of project decisions (versions) for multi-version system (X is operation of Cartesian product).

In general we have

$$m = m_1 \cdot m_2 \ldots m_D \tag{45.4}$$

versions to develop multi-version system. But taking into account existing of dependencies between diversity types the number of versions will be less [8].

Version L_t is described as a vector of elements

$$L_t = (e_{(t)1}, e_{(t)2}, \ldots, e_{(t)D}) \tag{45.5}$$

where $e_{(t)j} \in MDE_j$. This vector corresponds to one way in a direct acyclic graph G consisted of m + 2 nodes including nodes "Enter" and "Exit". This graph may be represented in compressed form GC when all nodes (elements $e_{j1}, e_{j2}, \ldots, e_{jmj}$ of j-th diversity level are joined in one node. Graph GC step by step is transformed in a special form of the graph GS in case of dependencies between diversity types [8]. The graph GS allows to search sequentially all ways (and vectors L) taking into consideration such dependencies.

If two-version system is developed according to project requirements a pair $PL(L_t, L_k)$ of vectors L_t and L_k, $t \neq k$, should be selected. In general vectors L_t and L_k may differed one, two or D elements.

A set of pairs

$$MPL = \{PL(L_1, L_2), PL(L_1, L_3), \ldots, PL(L_{m-1}, L_m)\} \tag{45.6}$$

contains $r = C_m^2$ elements.

The version L_t and pair $PL(L_t, L_k)$ can be described by one and two ways correspondingly in special graph which is called a graph of multi-version technologies [19].

Each pair PL_i is characterized by a metric of diversity DPL_i and cost CPL_i. The values of DPL_i and CPL_i depend on pairs of elements for selected vectors L_t, and L_k:

$$PL(L_t, L_k) = (e_{(t)1}, e_{(k)1}), (e_{(t)2}, e_{(k)2}), \ldots, (e_{(t)D}, e_{(k)D}). \tag{45.7}$$

Diversity metric for the pair PL_i is calculated at the following way

$$DPL_i = \sum_{j=1}^{D} \omega_{i,j} \cdot DPL_{i,j}, \tag{45.8}$$

where $\omega_{i,j}$ is weighting coefficient, $0 \leq \omega_{i,j} \leq 1$, a sum of weighting coefficients equals 1.

The model assumes D diversity levels, and the sum of the weighting coefficients for them should be 1. To add a new level of diversity, in order to preserve the normalization condition we need to recalculate the value of weighting coefficients and the sum also should be 1. If part of specified diversity types is not used, i.e. the pair of elements $(e_{(t)h}, e_{(k)h})$ consists of identical elements corresponding weighting coefficient $\omega_{t,k}$ will equal zero and the sum of ones will be less 1.

It is not always possible to evaluate the project by simply summing the cost value of each element. In reality, model should consider some variants of elements assessment. Each element on j-th level of diversity can include element by default from other levels and we need to consider this property for the cost. For example, technologies of chips (SRAM, Flash, Antifuse for FPGAs, etc) or manufacturers of chips (companies Altera, Xilinx, Microsemi, etc) can include families of chips (Cyclone, Aria, Stratix, Virtex) and there is no need to consider the cost. In this case, we should set the cost to zero for each element of diversity level families of chips.

Let j_i – factor characterizing the need to consider the cost of elements at the j-th level, where $j_i = \{0,1\}$. The value is zero $j_i = 0$ if the cost already included to the element at other level of diversity and the value is one if we need to consider the cost:

$$CPL_i = \sum_{j=1}^{D} j_i \cdot CPL_{i,j} \qquad (45.9)$$

,

where $CPL_{i,j}$ – a cost of pair L_i and L_j.

To design one subsystem (version) of a multi-version system, it is necessary to choose a specific value from each set. If there are no dependencies among diversity types, any combination of values is possible.

An optimal design decision should contain a pair of elements according to the model (45.7), we need to find one pair of decision according to selected criteria (45.1 or 45.2).

Task 1. Find an optimal design decision with various types of diversity, which provides minimum cost $CPL \to \min$ with required degree of diversity DPL_{req}:

$$f = \begin{cases} DPL \geq DPL_{req} \\ CPL \to \min \end{cases} . \qquad (45.10)$$

The solution of this problem includes following sequence of steps.

Step 1. Determine a set of versions. This task may be solved in case of existing diversity type dependencies by development a direct acyclic graph GS according to the model [8].

Step 2. Determine diversity value $DPL_{i,j}$ and a cost $CPL_{i,j}$ for each pair of design decisions.

Step 3. Determine the weighting coefficient for each $j-th$ level of diversity.

Step 4. Calculate diversity value and a cost for each pair of decisions $PL_i = \langle DPL_i, CPL_i \rangle$.

Step 5. Determine the pairs of decisions PL_i, which provide required degree of diversity DPL_{req}.

Step 6. Determine one pair of decisions PL_i, which provides minimum cost $CPL \to \min$.

Task 2. Find an optimal design decision with various types of diversity, which provides maximum diversity $DPL \to \max$ with assumed cost CPL_{assum}:

$$f = \begin{cases} CPL \leq CPL_{assum} \\ DPL \to \max \end{cases} . \qquad (45.11)$$

The solution of this problem includes following sequence of steps.

Step 1-4. Repeat steps 1-4 (task 1).

Step 5. Determine the pairs of decisions PL_i, which provides assumed cost CPL_{assum}.

Step 6. Determine one pair of decisions PL_i, which provides minimum cost $DPL \to \max$.

45.3.1 Example

According to the proposed model it’s needed to choose two pair of decisions for optimal configuration of two-version system based on (45.6) or (45.7) criteria. A basic example was taken from the article [8]. Based on diversity types presented at the Fig. 45.1 an example of the diversity model is developed using abstract sets of diversity values. This makes the example more general and applicable for various types of computer systems. There are diverse technologies of chips (TC) (SRAM, Flash and Antifuse for FPGAs; program logic controller-, microprocessor- and microcontroller-based technologies); manufacturers of chips (MC) (companies Altera, Xilinx, Microsemi, Intel, Motorola, etc); families of chips (FC) (e.g., Cyclone, Aria, Stratix, Virtex, etc); technologies of printed circuit board production (TP) based on different materials, dielectrics, technological processes, etc.; manufacturers of printed circuit boards (MP) (companies in different countries); languages (L) (VHDL, JHDl, C, C++, etc.); technologies of development and verification (TO), etc.

We consider these seven diversity types and dependencies among the values (Table 45.4), which are typical for many safety-critical systems. For example, application of chips Altera company (MC) stipulates use of SRAM-FPGA technology-producing languages (L) and technologies and case tools of development and verification (TO). Dependencies between diversity types are shown in the Table 45.4 by arrows for corresponding types and subtypes.

Table 45.4. Diversity types, elements and dependencies among diverse elements

	Diversity type	Diverse elements	Dependencies among diverse elements	
1	TC	TC1, TC2, TC3, TC4, TC5, TC6	TC → MC	TC1, TC2, TC3 → MC1, MC2, MC3
				TC4, TC5, TC6 → MC4, MC5

	Diversity type	Diverse elements	Dependencies among diverse elements	
2	MC	MC1, MC2, MC3, MC4, MC	MC → FC	MC1, MC2 → FC1, FC2
				MC3, MC4, MC5 → FC3, FC4, FC5, FC6
3	FC	FC1, FC2, FC3, FC4, FC5, FC6	FC → TP	FC1, FC2, FC4 → TP1, TP2
				FC3, FC5, FC6 → TP3, TP4, TP
4	TP	TP1, TP2, TP3, TP4, TP5	TP → MP	TP1, TP3, TP5 → MP1, MP2
				TP2, TP4 → MP3, MP4
5	MP	MP1, MP2, MP3, MP4	TC → L	TC1, TC3 → L1, L2, L3
				TC2, TC4, TC5, TC6 → L4, L5
6	L	L1, L2, L3, L4, L5	L → TO	L1 → TO1
				L2, L3, L5 → TO2
				L4 → TO3
7	TO	TO1, TO2, TO3	TC → TO	TC1, TC3, TC5, TC6 → TO1, TO2
				TC2, TC4 → TO3

According to the model [8], input data for the model were prepared and next operations were performed.

1. First step of the algorithm starts from the graph GC (Fig. 45.1), which describes possible diversity types, but does not reflect any dependencies between these elements according to Table 45.4.

2. Next step includes four steps: splitting a subgraph, labeling ingoing and outgoing edges of split subgraphs, eliminating dead nodes and edges, and merging nodes. In result, we get graph shown in Fig. 45.1.The final model of the complete example according to [8] is

presented in Fig. 45.2. The model contains 26 different decisions with 374 feasible diversity combinations, as shown in Table 45.5.

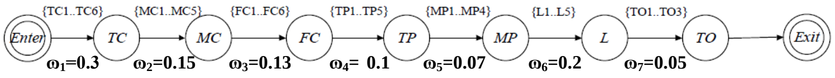


Figure 45.1. An example of the graph GC with weight coefficients [8]

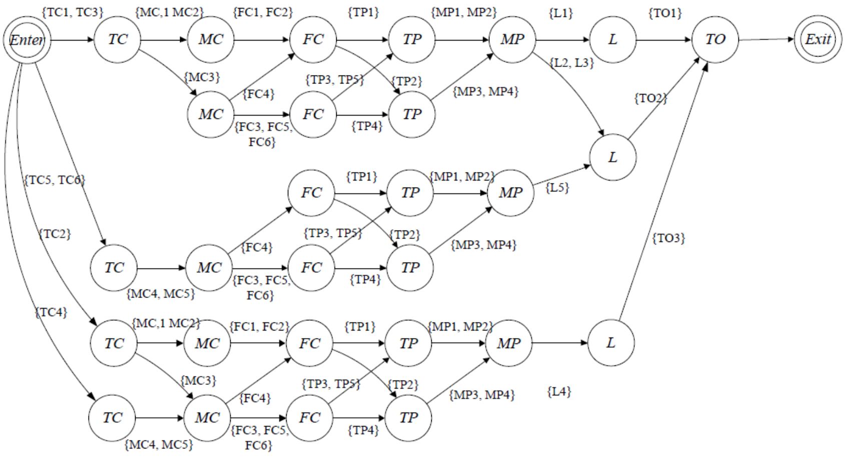


Figure 45.2. The graph GS (model of dependencies 1 – 7) [8]

Table 45.5. Feasible Combinations of Diversity Types

Path	TC	MC	FC	TP	MP	L	TO	Number of combi-nations
1	TC1, TC3	MC1, MC2	FC1, FC2	TP1	MP1, MP2	L1	TO1	16
2	TC1, TC3	MC1, MC2	FC1, FC2	TP1	MP1, MP2	L2, L3	TO2	32
3	TC1, TC3	MC1, MC2	FC1, FC2	TP2	MP1, MP2	L1	TO1	16

Path	TC	MC	FC	TP	MP	L	TO	Number of combi-nations
4	TC1, TC3	MC1, MC2	FC1, FC2	TP2	MP3, MP4	L2, L3	TO2	32
5	TC1, TC3	MC3	FC4	TP1	MP1, MP2	L1	TO1	4
6	TC1, TC3	MC3	FC4	TP1	MP1, MP2	L2, L3	TO2	8
7	TC1, TC3	MC3	FC4	TP2	MP1, MP2	L1	TO1	4
8	TC1, TC3	MC3	FC4	TP2	MP3, MP4	L2, L3	TO2	8
9	TC1, TC3	MC3	FC3, FC5, FC6	TP3, TP5	MP1, MP2	L1	TO1	24
10	TC1, TC3	MC3	FC3, FC5, FC6	TP3, TP5	MP1, MP2	L2, L3	TO2	48
11	TC1, TC3	MC3	FC3, FC5, FC6	TP4	MP3, MP4	L1	TO1	12
12	TC1, TC3	MC3	FC3, FC5, FC6	TP4	MP3, MP4	L2, L3	TO2	24
13	TC5, TC6	MC4, MC5	FC4	TP1	MP1, MP2	L5	TO2	8
14	TC5, TC6	MC4, MC5	FC4	TP2	MP3, MP4	L5	TO2	8
15	TC5, TC6	MC4, MC5	FC3, FC5, FC6	TP3, TP5	MP1, MP2	L5	TO2	24
16	TC5, TC6	MC4, MC5	FC3, FC5, FC6	TP4	MP3, MP4	L5	TO2	24
17	TC2	MC1, MC2	FC1, FC2	TP1	MP1, MP2	L4	TO3	8
18	TC2	MC1, MC2	FC1, FC2	TP2	MP3, MP4	L4	TO3	8
19	TC2	MC3	FC4	TP1	MP1, MP2	L4	TO3	2

Path	TC	MC	FC	TP	MP	L	TO	Number of combi-nations
20	TC2	MC3	FC4	TP2	MP3, MP4	L4	TO3	2
21	TC2	MC3	FC3, FC5, FC6	TP3, TP5	MP1, MP2	L4	TO3	12
22	TC2	MC3	FC3, FC5, FC6	TP4	MP3, MP4	L4	TO3	6
23	TC4	MC4, MC5	FC4	TP1	MP1, MP2	L4	TO3	4
24	TC4	MC4, MC5	FC4	TP2	MP3, MP4	L4	TO3	4
25	TC4	MC4, MC5	FC3, FC5, FC6	TP3, TP5	MP1, MP2	L4	TO3	24
26	TC4	MC4, MC5	FC3, FC5, FC6	TP4	MP3, MP4	L4	TO3	12
Total								374

According to the proposed model at step 2 we should assign diversity and cost value for each pair of elements on all levels of diversity (TC, MC, FC, etc.). Table 45.6 shows the input data for pair of elements from 1 and 16 paths (Table 45.5). Each level of diversity (TC, MC, FC, etc.) must be characterized by weighting coefficients (step 3) and Fig. 45.1 presents weighting coefficients for each level of diversity. Diversity values and weighting coefficients are normalized between 0 and 1. Weighting coefficients determines the importance of diversity type and the sum of them is 1. The cost value is given in absolute units.

Table 45.6. Diversity value for some pair of elements

Pair of elements	**TC1**	**TC5**	**MC2**	**MC4**	**FC2**	**FC3**	**TP1**	**TP4**	**MP1**	**MP3**	**L1**	**L5**	**TO1**	**TO2**
Cost value	44	20	68	55	62	66	70	13	30	79	23	68	12	13
$j_i = \{0,1\}$	1	1	1	1	0	1	1	1	1	1	1		1	1
Diversity value	0.29		0.5		0.15		0.73		0.13		0.86		0.8	
DPL_1	DPL_1=0.29·0.3+0.5·0.15+0.15·0.13+0.73·0.1+0.13·0.07+0.86·0.2+0.8·0.05=0.46													
CPL_1	CPL_1=44·1+20·1+68·1+55·1+62·0+66·1+70·1+13·1+30·1+79·1+23·1+68·1+12·1+13·1=561													
Pair of elements	**TC1**	**TC3**	**MC2**	**MC3**	**FC2**	**FC3**	**TP1**	**TP2**	**MP1**	**MP4**	**L1**	**L3**	**TO1**	**TO2**
Cost value	44	80	68	54	62	66	70	55	30	76	23	59	12	13
$j_i = \{0,1\}$	1	1	1	1	0	1	1	1	1	1	1		1	1
Diversity value	0.11		0.79		0.15		0.61		0.7		0.49		0.8	
DPL_2	DPL_2=0.11·0.3+0.79·0.15+0.15·0.13+0.61·0.1+0.7·0.07+0.49·0.2+0.8·0.05=1.01													
CPL_2	CPL_2=44·1+80·1+68·1+54·1+62·0+66·1+70·1+55·1+30·1+76·1+23·1+59·1+12·1+13·1=650													

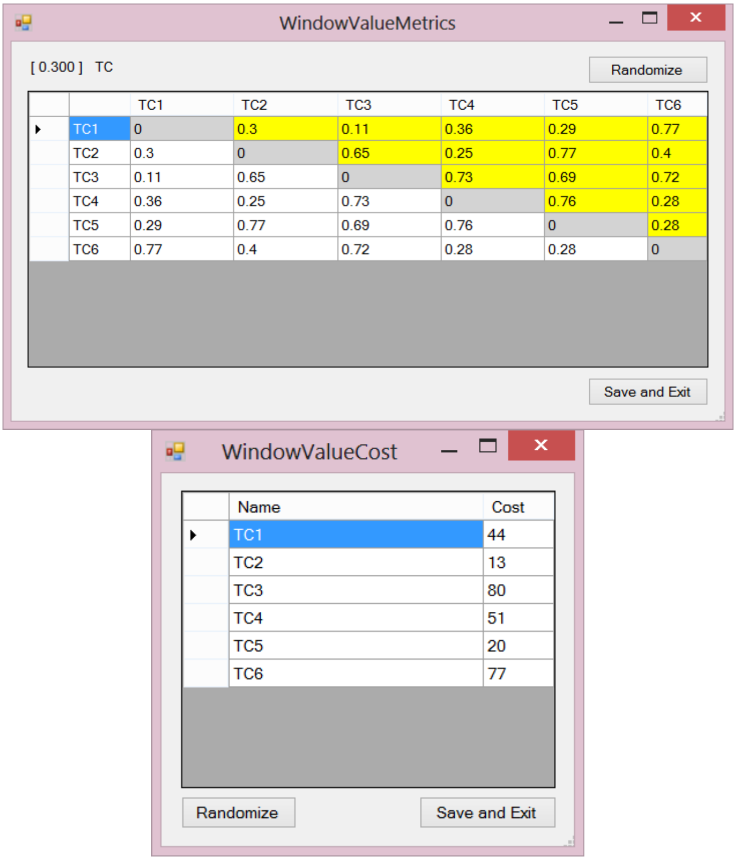
So, we have all feasible combinations of diversity types (Table 45.5) which can be presented as a set of decisions. We need to make simple calculations for each pair of decisions to get an optimal design decision according to our model for two-version system.

For example, let show calculations for some pair of decisions from Table 45.6. First two paths contain such pairs of elements (TC1, TC5), (MC1, MC4), (FC1, FC3), (TP1, TP4), (MP1, MP3), (L1, L5), (TO1, TO2). Diversity value for this decision should be calculated according to the model (45.8) and the result is DPL_1. Cost value for this decision should be calculated according to the model (45.9) and the result is CPL_1. The cost of FC1 already included into TC level of diversity and because of it we set the cost for FC_1 to zero. The same calculations DPL_2, CPL_2 presented for second pair of decisions and to get optimal design decision we need to make calculations for each pair of decisions according to Table 45.5. Thus, based on this proposed model, an optimal design decision for two-version system can be presented in the form of two decisions. We have to make all calculations for each pair of decisions from feasible combinations of diversity types (Table 45.5) to find an optimal design decision according (45.10 or 45.11) criteria. To make the calculations we used developed software Diversity Analyzer based on the proposed model. To get the optimal design decision we need to do following steps.

– Initial data of model can be defined by expert (Fig. 45.3).
– Create full group of elements for each level of diversity (Table 45.4).
– Set level of diversity for each pair of elements.
– Set the cost for each element and price relationship between elements (Fig. 45.3).
– Set weighting coefficient for each level of diversity (Fig. 45.3).
– After all initial data entered, the user can start working with the system (Fig. 45.4).

The result can be presented in the text form and in the form of a graph (Fig. 45.4). This software takes into consideration the dependencies among diversity types, diversity metrics and the cost and presents a decision in the form of two decisions for two-version system.

According to the model it's needed to choose two pair of decisions for optimal configuration of two-version system based on (45.10) or (45.11) criteria. According to (45.10) criteria we need to assign value for required degree of diversity DPL_{req} and according to (45.11) criteria we need to assign value for with assumed cost CPL_{assum} (Fig. 45.4). The optimal design decision of two-version system according to (45.10) is (TC3,TC5)(MC2,MC5)(FC1,FC5)(TP2,TP5)(MP4,MP1)(L1,L5)(TO1, TO2) and presented in the top of the tool (Fig. 45.4).

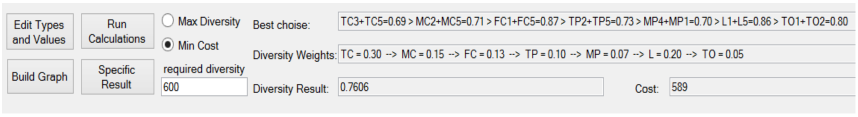


[0.300] TC

	TC1	TC2	TC3	TC4	TC5	TC6
TC1	0	0.3	0.11	0.36	0.29	0.77
TC2	0.3	0	0.65	0.25	0.77	0.4
TC3	0.11	0.65	0	0.73	0.69	0.72
TC4	0.36	0.25	0.73	0	0.76	0.28
TC5	0.29	0.77	0.69	0.76	0	0.28
TC6	0.77	0.4	0.72	0.28	0.28	0

Name	Cost
TC1	44
TC2	13
TC3	80
TC4	51
TC5	20
TC6	77

Figure 45.3. Diversity Analyzer tool. Values of diversity metrics and cost



Max Diversity [✓ Min Cost]

required diversity: 600

Best choise: TC3+TC5=0.69 > MC2+MC5=0.71 > FC1+FC5=0.87 > TP2+TP5=0.73 > MP4+MP1=0.70 > L1+L5=0.86 > TO1+TO2=0.80

Diversity Weights: TC = 0.30 --> MC = 0.15 --> FC = 0.13 --> TP = 0.10 --> MP = 0.07 --> L = 0.20 --> TO = 0.05

Diversity Result: 0.7606

Cost: 589

Figure 45.4. Diversity Analyzer tool. Criteria and results

Conclusions

FPGA technology application increases number of project decisions for multi-version NPP I&C systems and decreases CCF risks in comparison with CPU-based systems. However, there are a few challenges regarding multi-version FPGA-based NPP I&C systems development and implementation caused by the uniqueness of such systems.

NUREG7007- and CLB-based assessment techniques allow calculating diversity metrics and comparing the different variants of CPU- and FPGA-based NPP I&C systems. Extreme values of diversity metrics were calculated for such systems. Quantitative assessment adds qualitative analysis according with profile of the requirements to diversity.

Application of the diversity allows decreasing the probability of CCF. Complexity of diversity type choice is caused by very large number of version pairs and dependencies between different diversity types (e.g., between different manufacturers of chips and technologies of chips, between technologies and families of chips, etc.). Suggested model can be used during safety-critical systems development or modernization for making an optimal design decision based on a criterion of safety-cost.

The proposed model takes into consideration the dependencies among diversity types, diversity metrics and the cost and presents a decision in the form of two decisions for two-version system. Future steps may be related to development of general procedure for n-version systems (n more than 2) and taking into account reliability and other indicators and limitations.

Questions to self-checking

1. Which standards are applicable to assess diversity for NPP I&C systems?
2. Which requirements are applicable to principle of diversity?
3. Which attributes of diversity are described in NUREG 7007?
4. What does the weight of diversity attribute depend on?
5. How is the diversity metric calculated based on filled check-list?
6. How is the diversity metric calculated when some diversity subtypes are not applied?
7. How to build graph of design decisions with only feasible combinations of diversity types?
8. What advantage has excluding non-feasible combinations of diversity types?
9. How to find an optimal design decision which provides minimum cost with required degree of diversity?
10. How to find an optimal design decision which provides maximum degree of diversity with assumed cost limit?

References

1. IAEA Safety Glossary, “Terminology used in nuclear safety and radiation protection. Ed.2.0” (2007)
2. NUREG/CR-7007, “Review Guidelines for Field-Programmable Gate Arrays in Nuclear Power Plant Safety Systems,” U.S. Nuclear Regulatory Commission (2009)
3. NUREG/CR-7007, Diversity Strategies for Nuclear Power Plant Instrumentation and Control Systems, ONL, Oak Ridge, USA, 2009.
4. NUREG/CR-7006, “Review Guidelines for Field-Programmable Gate Arrays in Nuclear Power Plant Safety Systems,” U.S. Nuclear Regulatory Commission (2010)
5. M. Yastrebenetsky, V. Kharchenko (eds), Nuclear Power Plant Instrumentation and Control Systems for Safety and Security, IGI-Global, USA (2014).
6. W. Postma, J. Brinkman, “The Contribution to Safety of a Diverse Backup System for Digital Safety I&C Systems in Nuclear Power Plants, a Probabilistic Approach”, 12th Conference Probabilistic Safety Assessment and Management (2014)
7. V. Kharchenko, A. Siora, E. Bakhmach. Diversity-scalable decisions for FPGA-based safety-critical I&Cs: from Theory to Implementation, Proceedings of the 6th American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Controls, and Human Machine Interface Technology, NPIC-HMIT, Knoxville, TN, USA, 2009, pp. 78-84.
8. S. Vilkomir, V. Kharchenko, A Diversity Model for Multi-Version Safety-Critical I&C Systems, 11th International Probabilistic Safety Assessment and Management Conference and the Annual European Safety and Reliability Conference PSAM-ESREL, 2012, pp. 1791 – 1799.
9. V. Kharchenko, V. Sklyar (Edits), FPGA-based NPP Instrumentation and Control Systems: Development and Safety Assessment, Research and Production Corporation “Radiy”, National Aerospace University named after N.E. Zhukovsky “KhAI”, State Scientific Technical Center on Nuclear and Radiation Safety (2008)

10. EPRI TR1019181, “Guidelines on the Use of Field Programmable Gate Arrays (FPGAs) in Nuclear Power Plant I&C Systems,” Electric Power Research Institute (2009)

11. EPRI TR1022983, “Recommended Approaches and Design Criteria for Application of Field Programmable Gate Arrays in Nuclear Power Plant I&C Systems,” Electric Power Research Institute (2011)

12. Report of RPC Radiy, “FPGA-Based Road to Business Success”, Release 3 (2014)

13. J. Knight. Diversity. In Dependable and Historic Computing / C. Jones, J. Lloyd (editors) Lecture Notes in Computer Science, Volume 6875, Springer, 2011, pp. 298-312

14. K. Salako, L. Strigini. When does "diversity" in development reduce common failures? Insights from probabilistic modelling. IEEE Transactions on Dependable and Secure Computing, 99, 2013. doi: 10.1109/TDSC.2013.32

15. V. Kharchenko, O. Siora, V. Duzhyi, D. Rusin, “Standard Analysis and Tool-Based Assessment Technique of NPP I&C Systems Diversity”, 22nd International Conference on Nuclear Engineering (2014)

16. V. Kharchenko, O. Siora, V. Sklyar, A. Volkoviy, V. Bezsaliy, “Multi-Diversity Versus Common Cause Failures: FPGA-Based Multi-Version NPP I&C Systems”, 7th American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies (2010)

17. V. Kharchenko, T. Nikitina, S. Vilkomir, “Optimal Selection of Diversity Types for Safety-Critical Computer Systems”, 12th Conference Probabilistic Safety Assessment and Management (2014)

18. V. Kharchenko, A. Siora, V. Sklyar, A. Volkoviy. Multi-Diversity Versus Common Cause Failures: FPGA-Based Multi-Version NPP I&C Systems, Proceedings of the 7th American Nuclear Society International Topical Meeting on Nuclear Plant Instrumentation, Controls, and Human Machine Interface Technology NPIC-HMIT, Las-Vegas, Nevada, USA, 2010, pp. 85 – 96.

19. Standard IEC 60880 Ed. 2.0 b: 2006, Nuclear power plants - Instrumentation and control systems important to safety - Software aspects for computer-based systems performing category A functions.

20. L. Pullum , Software Fault Tolerance Techniques and Implementation, Artech House Computing Library, 2001.

АННОТАЦИЯ

Раздел содержит обзор видов диверсности, применяемых к многоверсионным информационно-управляющим системам критического применения.

В разделе рассмотрен метод расстановки весовых коэффициентов для видов и подвидов диверсности и вычисления результирующего показателя диверности системы.

Также раздел содержит описание методики и информационной утилиты для выбора оптимального проектного решения по критерию диверсность-стоимость.

Розділ містить огляд видів диверсності, що застосовуються у багатоверсійних інформаційно-управляючих система критичного призначення.

У розділі розглянуто метод розстановки вагових коефіцієнтів для видів і підвидів диверсності та обчислення результуючого показника диверсності системи.

Також розділ містить опис методики та інформаційної утиліти для вибору оптимального проектного рішення за критерієм диверсність-вартість.

The section contains survey of diversity types which are applied to multiversion I&C systems of critical application.

Approach for assessment of weight coefficients for diversity types and subtypes is considered, as well as calculation of diversity metric of the system.

Finally section contains approach and tool designed to find optimal design decision by diversity-cost criteria.

PART 12
Techniques and tools of penetration testing for web systems and networks

ABBREVIATIONS

AJAX	Asynchronous Javascript and XML
OSI	The Open Systems Interconnection model
CMM	Capability Maturity Model
CMS	Content Management System
CI/CD	Continuous integration and continuous deployment
CSRF/XSRF	Cross-Site Request Forgery
DBMS	The Database Management System
DDD	Data Display Debugger
DDoS	Distributed Denial-of-Service
DLP	Data Loss Prevention
DNS	Domain Name System
DoS	Denial-of-Service
DVL	Damn Vulnerable Linux
ET, ETA	Event Tree, Event Tree Analysis
ELF	An interactive, modular, and scriptable ELF (Executable & Linking Format) machine for static binary instrumentation of executable files, shared libraries and relocatable ELF objects
FT, FTA	Fault Tree, Fault Tree Analysis
FTP	The File Transfer Protocol
GCC	The GNU (*GNU's Not Unix*) Compiler Collection
GDB	The GNU Project Debugger
ICMP	The Internet Control Message Protocol
IDS	Intrusion Detection System
IPS	Intrusion Prevention System
IPv6	Internet Protocol version 6
ISP	Internet Service Provider
ITD	Iterative and Incremental Development
LIDa	Linux Interactive Disassembler
MITM attack	Man-In-The-Middle attack
NASM	The Netwide Assembler
OSINT	Open-Source Intelligence

OWASP	Open Web Application Security Project
OWASP ZAP	OWASP Zed Attack Proxy
PHP	Hypertext Preprocessor
PKI	Public Key Infrastructure
PTES	Penetration Testing Execution Standard
RFID	Radio-Frequency Identification
SAML	Security Assertion Markup Language
SDLC	Software Development Life Cycle
SOA	Service-Oriented Architecture
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SQLi	Structured Query Language injection
SSH	Secure Shell
SSL / TLS	Secure Sockets Layer / Transport Layer Security
TCP	The Transmission Control Protocol
UDP	The User Datagram Protocol
VOIP	Voice over Internet Protocol
VPN	Virtual Private Network
W3C	World Wide Web Consortium
WAF	Web Application Firewall
WSP	Web Systems Pentesting
WSS	Web Services Security
WVS	Acunetix Web Vulnerability Scanner
XACML	eXtensible Access Control Markup Language
XML	eXtensible Markup Language
XSS	Cross Site Scripting

46. Introduction to penetration testing of web systems and networks

The urgency of ensuring the security of web-systems is growing rapidly along with the number of such systems in the world information community. Objective is also the fact that the issue of the value of the content of various web resources is different and it is necessary to apply various, most effective in each case, decisions regarding the security of the investigated web-systems.

The actualization of information security problems also requires an increase in the number of highly qualified personnel. To train and acquire skills in penetration testing, platforms with previously known vulnerabilities are created. At these sites, novice testers can test their skills, try out new tools. Anyone can view the code and figure out what exactly led to the vulnerability in open source systems. This also gives additional experience, because the tester knows not only how to attack the application, but also how to get rid of the found vulnerabilities.

Nowadays penetration testing occupies an important niche in the process of ensuring the quality of web services and networks. The procedure for penetration testing is included in the security audit of such systems. In the era of the total dependence of mankind on the level of preparedness and availability of critical infrastructures, issues are increasingly emerging that are related to the security and safety of such systems. In particular, in [1] the authors of the report on the impact of cyber-attacks on the resilience of complex industrial systems and describe the approach to building a hybrid model to ensure the cyber security of the of the computer and communication system used for monitoring, protection and control. Dependability is an important characteristic that a trustworthy computer system should have. It is a measure of availability, reliability, maintainability, safety and security. And so, the modern tendencies of buiding the web services define security requirements for them. The security-aware selection of Web Services for reliable composition is described in [2].

Penetration testing is one of the most effective ways to identify systemic flaws and defects in software. Trying to bypass security assurance technics, the penetration tester is able to determine the ways and tools in which a hacker can compromise an organization's security and damage the organizations as a whole. The testers’ task is to show in a safe and controlled manner how an attacker can cause a serious harm to the organization and affect its ability to provide services, maintain its reputation, protect the customers. The difficulty of pentesting is that the quality of its implementation depends on the tester experience and knowledge of the target system.

The subjective opinion of the authors grounds on the fact that there is some launch pad with which it is possible to start efficiently ensuring of the security of web-based systems easily enough in the event of the need. The foundation of such a launch pad is a set of data and knowledge about the web system being researched, a set of tools for conducting penetration testing, and the appropriate skills of using such a toolkit. Manually check a large amount of code to identify all possible values of all data items is the task that is complex and time-consuming, requiring great knowledge in various areas. For this reason, it is increasingly necessary to use specialized security scanners operating at the application level.

The chapter provides an overview of cutting-edge technologies and tools for testing of web systems and networks including walkthroughs that consider conducting certain types of attacks aimed at compromising one of the basic properties of information security – confidentiality

The tools discussed in the section 48 will help novice testers to master penetration testing of Web applications and networks, give an understanding of the causes of the vulnerabilities and the processes that they implement, eliminating the tester from repeatedly repeating monotonous tests. . Such knowledge will allow security engineers to develop holistic mechanisms of web systems and networks cyber security ensuring.

46.1. The importance and specificity of Web systems and networks security

46.1.1 Meeting the security requirements for Web services

Several well-known organizations such as OASIS, W3C, Liberty Alliance, and various industry representatives have compared numerous security standards and methods to ensure the security of Web services. Basically, all of the following technologies and standards complement each other, but there is some contradiction or even competition between some. Let’s consider the existing standards and their relationship within the framework of threat protection and security of web systems.

To form a holistic image of the research object, it is necessary to consider the stack of standards for securing web systems. Modern IT communities that promote open standards for creating web-based systems have developed a large number of standards for securing web systems. As a recommendation for ensuring the security of such systems, the notation on their interaction and correspondence, presented in Figure 46.1, was proposed [3-6]. The figure also presents the relationship between the organizations' security standards for Web systems developed with the Open Systems Standard (OSI).

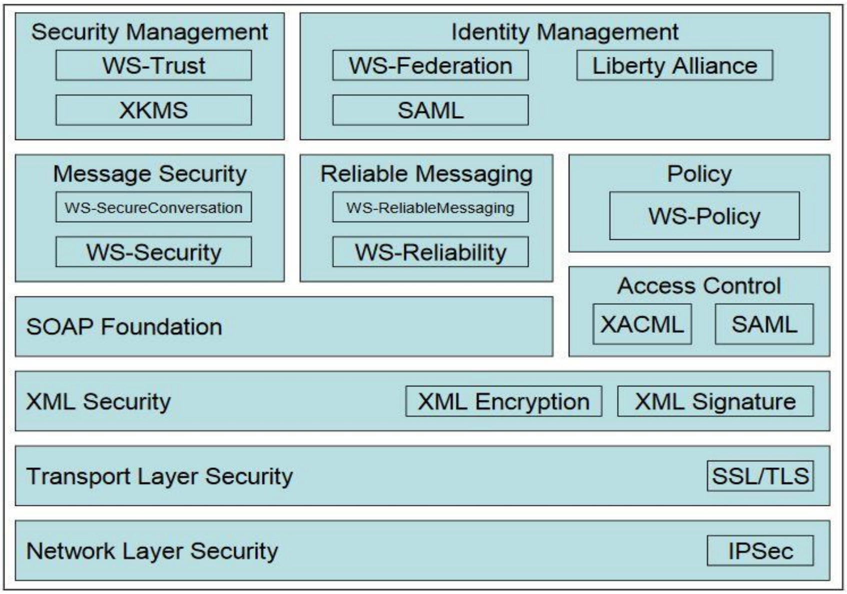


Fig. 46.1. – The interrelation of security standards for web-based systems

The security standards for the network, transport and XML layers are used to ensure the security of messages during the transmission over the network. And such security standards as IPsec, SSL / TLS, XML Encryption and XML Signature work with SOAP messages on a different level. In addition, over the level of XML Security, there are two more types of standards: standards based on SOAP and stand-alone standards.

The WS-Security and WS-SecureConversation security standards define how to use XML signatures, XML Encryption and credentials to secure SOAP at the message level, while standards that ensure the reliability of message exchange determine the protocols necessary to ensure the receipt of the message.

Access control standards are not unique to web systems; XACML can define an access policy to any system and SAML can be used in any environment. The WS-Policy policy defines a grammar for linking the requirements of a Web system policy.

Security management features define other Web services to manage credentials, such as PKI certificates within the SOA framework. Identity management standards take precedence over access control standards, policy standards and SOAP standards when providing services for the dissemination and management of user identification and credentials within the SOA.

46.1.2. Threats to the security of web systems.

Security solutions should always be based on an analysis of threats from which the system is planned to protect.

Although there are a huge number of security standards and technologies available to ensure the security of web services, they can not always be adequate or necessary for a separate organization or individual service.

For this reason, it is important to analyze the threats that web systems face in order for an organization to determine what threats their Web services must be protected from.

In accordance with WS-I, the main threats for Web systems are:

– Message alteration. An attacker inserts, deletes, or modifies information in a message to mislead the recipient;

– Loss of confidentiality. The information in the message may become available to an unauthorized person (third party);

– Falsified messages. An attacker misleads the recipient that the message is sent from an authorized sender;

– Man in the middle. The third party is between the sender and the recipient and forwards the messages in such a way that the two parties are not aware of its existence, which allows an attacker to view and change all messages;

– Principal spoofing. An attacker generates and sends a message with credentials in such a way that it is from an authorized source;

– Forged claims. An attacker creates a message with fake credentials, which, however, is accepted by the recipient as verified;

– Replay of message. The attacker resends the previously sent message;

– Replay of message parts. An attacker includes parts of one or more previously sent messages in a new message;

– Denial of service. An attacker causes the system to expend resources disproportionately in such a way that real queries can not be executed. The danger of these threats can vary depending on the needs and goals of the organization. In some cases, messages should not be confidential, so losing confidentiality is not an issue. In addition, organizations can offer a web service to the public. For example, a web service that provides information about the current weather forecast does not need to worry if the request is from a falsified sender. However, it is important to understand these threats and what technologies are available to eliminate them.

To protect against these threats, the following Web services and HTTP standards exist:

– W3C XML Encryption. WS-Security uses W3C XML Encryption to encrypt messages and to ensure the confidentiality of a portion (or total) of a SOAP message;

– W3C XML Signature. It uses WS-Security to digitally sign a message and ensure the integrity of the message and the authentication of the sender;

– WS-Security Tokens. Allows messages to include credentials to assist the recipient in determining whether or not the sender of the message is allowed to perform the requested action.

Supported marker types include:

– Username / password. The most common credentials in web applications;

– OASIS SAML Assertion. Asserts that the sender has been identified and / or the attributes belong to the sender;

– IETF X.509 certificate. In conjunction with the XML Signature, the recipient can verify that the CA issued the certificate used to sign the SOAP message;

– ISO Rights Expression Language. Used to provide public key information, attributes of these keys, as well as information about the sender's license;

– IETF Kerberos token. Allows web services to function in a domain serviced by Kerberos;

– W3C WS-Addressing IDs. Allows the message sender to provide a unique identifier for the message;

– IETF SSL / TLS. Secure the HTTP protocol for sending / receiving SOAP messages;

– SSL / TLS with client authentication. Requires pre-authentication of the sender and the recipient before the security of the HTTP protocol;

– IETF HTTP authentication. Allows usernames, passwords (via HTTP Basic) or password digests (via HTTP Digest) to be passed as part of the HTTP header.

In addition to all of the above, web systems are also threatened by potential vulnerabilities associated with the presence of defects in the software. Also, web systems are available for remote access, so attackers can use this access to investigate potential exploits.

Thus, it should be noted that for web systems, as for any remote access system, it is necessary that web services be implemented in accordance with security requirements with the use of network technologies to restrict access to web systems based on the developed recommendations.

46.2. Pentesting as stage of lifecycle of web-system

From the point of view of software development, formally any web-based system is software or a part of it. Therefore, the development process is based on currently used technologies (software development methodologies) that define the requirements for quality, reliability, functionality, and, of course, security, etc.

The genesis of methodologies and technologies for creating software can be represented as follows [7]:

– waterfall model;

– iterative model;

– spiral model.

Waterfall model of software development assumes consistent execution of all stages of the project in a predetermined order. The transition to the next stage is possible only with the complete completion of the previous stage. The requirements formulated at the stage of requirements formation are approved in the terms of reference and are unchanged throughout the project implementation time. The result of the completion of each stage is the complete set of documents necessary to continue the development of another team of specialists.

In accordance with the cascade ("waterfall") model, the following stages are distinguished:

1. Requirements;
2. Design;
3. Implementation;
4. Verification;
5. Maintenance.

As advantages of software development in accordance with the cascade ("waterfall") model, the following can be noted:

- completeness and consistency of the project documentation at each stage of development;

- certainty in calculating the timing and costs for the implementation of the project.

The disadvantages of the cascade ("waterfall") model are:

- 100% correctness of the output of the previous stage;

- incorrect rollback is equivalent to the beginning of the previous stage, resulting in a loss of time, money and the "spirit" of the development team;

- can lead to an unplanned completion of the project.

As a resultant conclusion on the use of the cascade model, it can be noted that it is inefficient to implement large projects with its help, whereas for the development of small projects the waterfall model can be objectively claimed.

The main alternative to the cascading model is the iterative and incremental development (ITD) or iterative model (IID).

In the iterative model, the software development life cycle is divided into iterations. Each iteration is the implementation of a part of the project's functional and contains all the necessary steps. The iteration goal is to get a functioning version (build) as part of the software, which includes all the previous iterations and also the current one. The result of the last (final) iteration will be software that fully implements the requirements for the functionality of the product.

The disadvantages of the iterative model include the following:

- an integral definition of the "environment" of the project appears in the process of project implementation;

- partial loss of results obtained in previous stages;

- iterative effect on quality through psychological assumption about the possibility of making corrections at the next stage.

Practical implementation of the iterative model is presented in the form of modern development methodologies (Rational Unified Process, Microsoft Solutions Framework, Extreme Programming).

The spiral model of software development is based on the classical Deming cycle *"plan -> do -> check -> act"*. With this approach it is understood that the software is developed for several iterations (spiral turns) through the prototyping method.

Each "turn" (iteration) involves the creation of a "build" (version) of the software, the requirements, goals and characteristics of the project being implemented are clarified, the qualitative and quantitative indicators of the project are assessed, work plans for the next iteration ("turn") are recorded.

At each "turn" an assessment is made:

- the risk of exceeding the time and cost of the project;

- the feasibility of another "turn";

- the results of the requirements management subsystem;

- the need for staff / freelance completion of the project.

It should be noted that the spiral model is a branch (a particular case) of the iterative (evolutionary model or IID) and not an alternative. The most common mistakes include the use of a spiral model as a synonym for the iterative model, or they speak of the independence and independence of the model from the IID.

For the implementation of pentesting, there are various techniques in which you detail how to perform penetration testing, and what tools to use. Currently, the most relevant are the following [8]:

– OWASP Testing Guide [9];

– NIST Special Publication 800-115: Technical Guide to Information Security Testing and Assessment (NIST SP 800-115) [10];

– Penetration Testing Execution Standard (PTES) [11];

– Open Source Security Testing Methodology Manual (OSSTMM) [12].

To conduct pentesting on the basis of current information, it is necessary to use constantly updated special resources about software vulnerabilities, for example:

– Database of vulnerabilities in the software SecurityFocus [13];

– Bagtrack SecLists.Org [14];

– Vulnerability database CVE [15];
– Open Source Vulnerabilities Data Base (OSVDB) [16];
– Vulnerability database Secunia [17];
– Database Bugs Collector [18]
– Database of XSS-vulnerabilities [19];
– Microsoft Security Bulletins [20].

The dramatic increase in the number of crimes directed directly at the web system motivates developers to take a closer look at cybersecurity issues and include technologies that ensure the security of web applications in the software development process. This situation arises because traditional techniques such as manual testing or a firewall for web applications become ineffective to protect web systems from cyber attacks by highly skilled attackers.

A system solution to the above-mentioned task of ensuring the security of the developed web-based systems can be a set of activities to regulate the stages of software development (alignment with one of the existing software development methodologies) and additional penetration tests (which can be part of automated testing and empirical influences of pentester).

As an example, you can consider the Secure Software Development Life Cycle (SDLC), which is shown in Figure 46.2 [21]. At each stage there should be a security audit, in which case it is necessary to take into account the results of the previously conducted pentesting. Ideally, incorporating penetration testing into the automated testing suite will identify and eliminate potential vulnerabilities in earlier stages of software development [22].

In general, the software development life cycle (in our case, a web system) includes six stages:

– **Analysis** (work with requirements) involves defining and following the overall strategy of the development process, conducting system analysis, supplementing new functions and making changes in the development of the process itself. The information obtained as a result of the security audit (including the activities of pentesting) allows us to evaluate the results obtained from the point of view of ensuring the cybersecurity of the software being developed;

– **Design** is the stage in the development of the software architecture based on the requirements obtained from the results of the previous stage (usually the result of the work of the business analyst);

– **Development** is the writing of the software code. This stage is the longest and most time consuming in the whole cycle;

– **Verification** determines the scope and consistency of the implementation of the procedures responsible for quality control. This phase also includes software security testing;

– **Implementation** implies the functioning of the created software in real conditions (for example, the customer);

– **Maintenance** Maintenance provides for the elimination of vulnerabilities (including on the basis of the results of pentesting), problems identified during the operation of the web system, as well as customization and expansion of functionality in accordance with software specification. An additional help is the translation of all the stages to the CI/CD and the application of the corresponding methodologies / technologies (Scrum / DevOps).

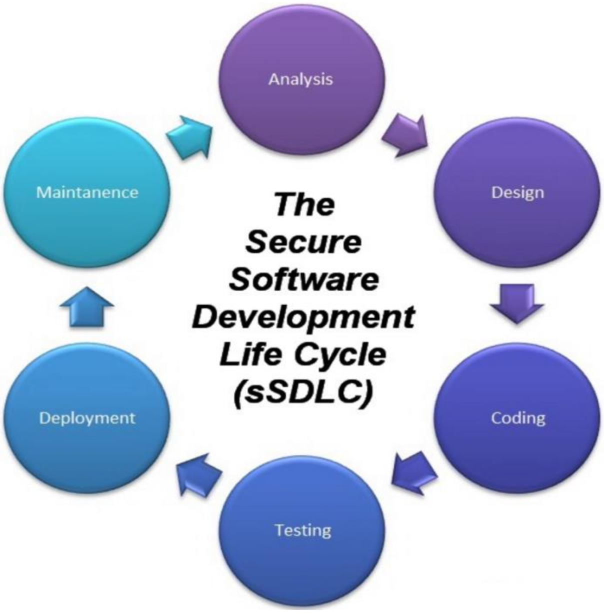


Fig. 46.2 – The secure software development lifecycle (web-system)

It should also be noted that penetration testing can not objectively guarantee 100% invulnerability of the developed web systems and networks, but it will facilitate the creation of countermeasures against potential intruders. Additional time and money spent at this stage will help to identify some vulnerabilities that could potentially cause material and moral damage to both the customer and the developer.

Thus, the introduction of pentesting into the stages of the life cycle of web systems is an urgent need. In some cases, you can use specialized scanners, code analyzers and automatic test methods at all stages of the life cycle of web systems.

46.3. Methods for analyzing the security of Web systems

In most cases, a combination of methods and tools such as OWASP (the appropriate tool in Kali Linux), PTES (mechanisms implemented in Metasploit), and some others are used to analyze the security of web systems. However, the question of formalizing the approach of penetration testing with the aim of its deeper (systematic) quantitative and qualitative research through modeling the main stages is still open. The following methods for analyzing web systems can be applied both independently (only the modeling process), and in conjunction with existing tools for conducting penetration testing of Web systems.

46.3.1 Attack trees

Event tree (ET) [23] is an algorithm for considering events originating from the main event. ET is used to identify and analyze options for achieving the main event. This is a powerful and flexible tool for analyzing possible problems and finding ways to resolve these problems. In this regard, the analysis of event trees has been widely applied [24-26] in the field of risk assessment, since it is possible to assess the probability of a major event (emergency, failure, successful attack).

Fault tree [27] (attack, accident, incident, unwanted event, etc.) is the basis of logical and probabilistic model of the cause-and-effect relationships of system failures with failures of its elements and other events (impacts); analyzing the occurrence of a failure, it consists of sequences and combinations of violations and malfunctions, and thus it is a multi-level graphological structure of causal relationships obtained as a result of tracing dangerous situations in reverse order to find possible causes of their occurrence.

Trees of failures can use quantitative and qualitative values, as well as linguistic variables, as input parameters. Using the variables of fuzzy logic allows the modeling in cases where no statistical data.

Events in the tree can be combined with AND or OR elements. Formulas 46.1 and 46.2 are used to calculate the probability of independent events:

$$P(A\ and\ B) = P(A \cap B) = P(A)P(B), \qquad (46.1)$$

$$P(A\ or\ B) = P(A \cup B) = P(A) + P(B) - P(A \cap B), \qquad (46.2)$$

where $P(A)$ is a probability of an A event;

$P(B)$ is a probability of an B event.

An example of an abstract tree is shown in Figure 46.3.

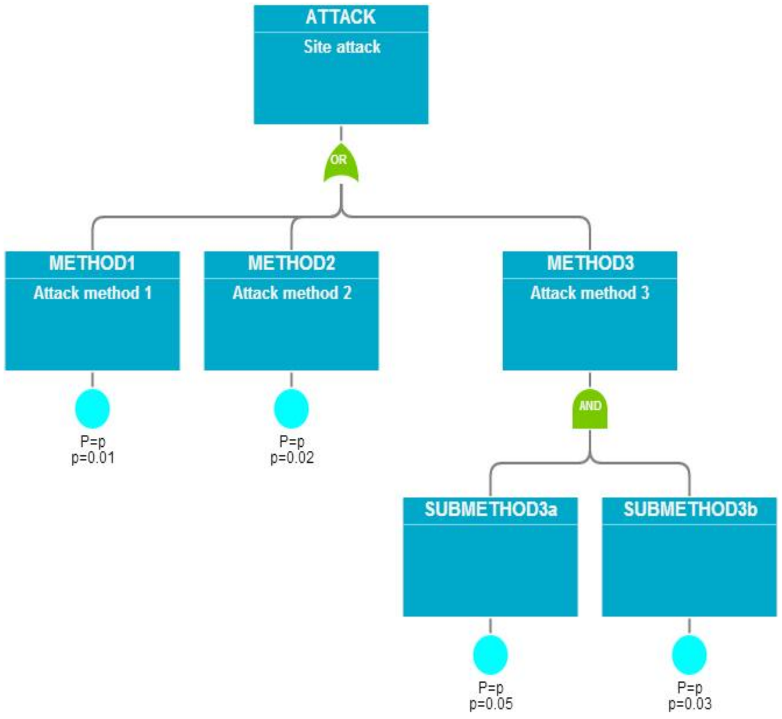


Fig. 46.3. Example of a tree with AND and OR elements

In this case, the site can be successfully attacked by three methods: METHOD1, METHOD2 and METHOD3. These events are combined by the logical condition "OR". In turn, the METHOD3 event will occur only in the event of SUBMETHOD3a and METHOD3b events, thus, these events must be combined by the logical condition "AND".

Figure 46.4 shows a structural tree model containing real scenarios for a successful attack of content management systems. For simplicity, there was no binding to specific control systems and no specific vulnerability identifiers. Vulnerabilities are defined as vulnerabilities that can lead to a successful outcome of a higher-level event. For example, in the attack associated with the addition of a new administrator to the database using a direct connection, the vulnerability should realize the ability to retrieve the contents of the configuration file, that ususally stores the datavase name user, password and database name that is used by the CMS.

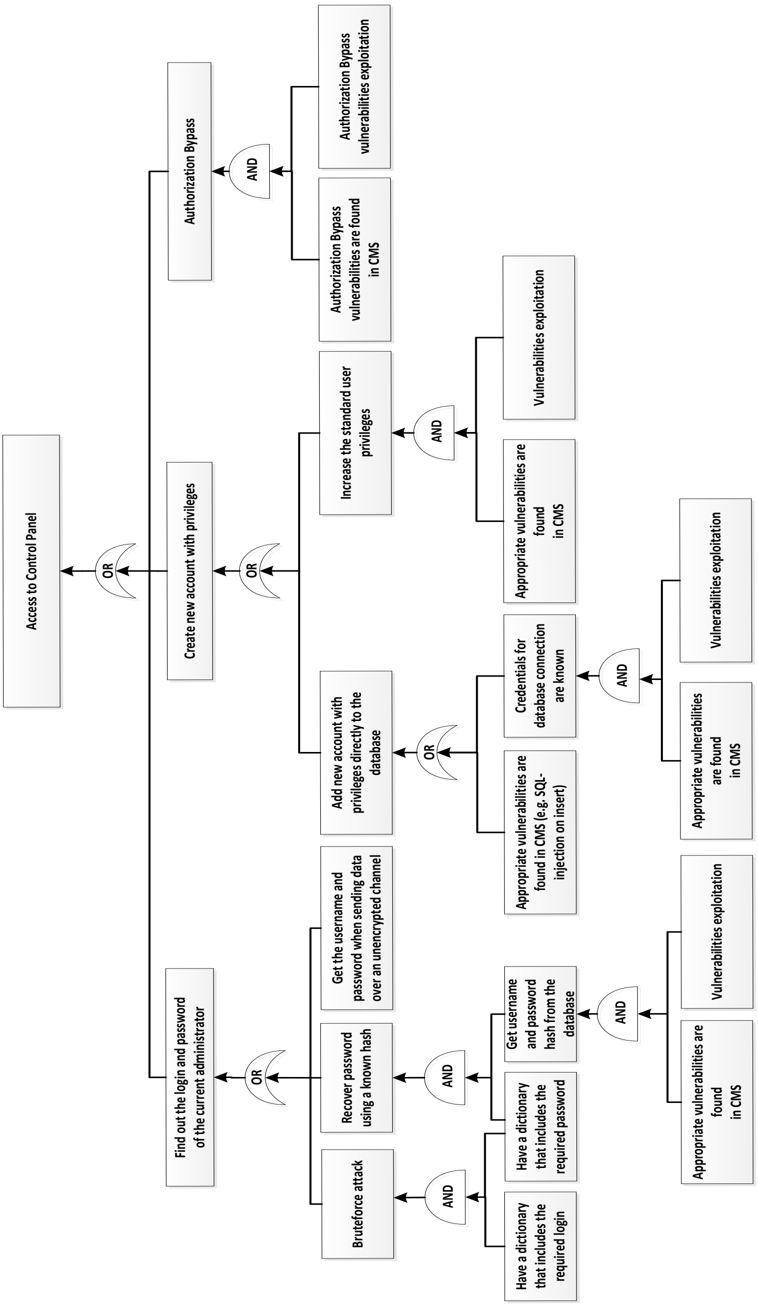

Access to Control Panel
OR
Find out the login and password of the current administrator
Create new account with privileges
Authorization Bypass
OR
Bruteforce attack
Recover password using a known hash
Get the username and password when sending data over an unencrypted channel
OR
Add new account with privileges directly to the database
Increase the standard user privileges
AND
Authorization Bypass vulnerabilities are found in CMS
Authorization Bypass vulnerabilities exploitation
AND
Have a dictionary that includes the required login
Have a dictionary that includes the required password
AND
Have a dictionary that includes the required password
Get username and password hash from the database
OR
Appropriate vulnerabilities are found in CMS (e.g. SQL-injection on insert)
Credentials for database connection are known
AND
Appropriate vulnerabilities are found in CMS
Vulnerabilities exploitation
AND
Appropriate vulnerabilities are found in CMS
Vulnerabilities exploitation
AND
Appropriate vulnerabilities are found in CMS
Vulnerabilities exploitation

Fig. 46.4. Attack scenarios made in the form of a tree

The main difficulty in assessing the security of Web applications is the inability to fully cover the security problem since security is a complex task. To minimize the probability of a successful attack, it is not enough to have a Web application without vulnerabilities, it is also necessary to:

– ensure the security of the application software used on the server side (OS, DBMS, etc.);

– ensure the security of network protocols (encryption of traffic using an SSL certificate);

– provide security on the user side (malware can pass passwords and other information to third parties);

– provide protection from the methods of social engineering (the key factor is the level of user's knowledge in the field of information security).

If you take into account all the above aspects when building a tree, then the tree will turn out to be too cumbersome and difficult to understand. There are difficulties in choosing the parameters of the model because often these data are based on statistics that are difficult or impossible to find. There are also problems with the detailing of each variant of the attack, as this leads to the growth of the event tree.

46.3.2 Markov's analysis

There are several tools for assessing information security, modeling security tools and possible attacks, the main ones of which are given in [28]. These include: probability theory, fuzzy sets, game theory, graphs, automata, Petri nets and stochastic processes.

The tools of information security assessment, modeling and protection of possible attacks based on practical considerations usually include such requirements:

– possibility to calculate the probability of threat realization;

– possibility to calculate the time of threat realization;

– possibility to calculate attack detection time;

– possibility to calculate information security risks;

– the simplicity of definition of input parameters of the model.

All necessary requirements should depend on the means of protection used, the vulnerabilities in them, the level of training, and the equipment of the attacker.

Models using the theory of stochastic processes are presented in the works [29-33]. Here are allocated models using:

– Markov processes [30];

– semi-Markov processes [31];

– Hidden Markov processes [33];

– branching processes [29].

These models have the following advantages and disadvantages. On the one hand, they make it possible to calculate the probability of implementing a particular threat, and, in the case of using the semi-Markov process, to estimate the time parameters of the threat realization. On the other hand, the approach to determining the states in these models leads to difficulties in practical application, because it is difficult to determine the flux density parameters for Markov models due to the fact that they are affected by many parameters that are not taken into account in the model. Such as, the vulnerability of the means of protection, the source of the threat and its characteristics, etc. or the semi-Markovian model, there is no explanation of the state of the safe functioning of the system, which also makes its practical application difficult.

Markov stochastic processes are widely used in theory and practice. According to [30] these processes can be used to assess the influence of attacks on the security of information in that case if the attack is a rare and independent event.

Proceeding from this, for the investigation of the impact of attacks on the information system, it is justified to use Markov stochastic processes.

Let the information system is affected by *n* independent attacks for a finite time *t*. At the same time, another attack affects the system only after the implementation of the previous. The transition of the system from the state to the state occurs until it is in an inoperative state, corresponding to the successful implementation of the attack on the system by an attacker. From this state, the system can exit after patching with or without fixing the vulnerability. In addition, the system periodically carries out preventive measures, during which existing vulnerabilities are eliminated and updates can be developed. There is always a *"zero day" n+1* vulnerability in the system.

In this paper, an attack is a potentially dangerous event that occurs during the operation of the system, which creates a particularly dangerous situation for the system. Such attack is characterized by the following features:

– it can be intentional or unintentional;

– it is a deterministic event;

– impacts on the information system typically are stochastic processes, which makes it possible to apply stochastic processes.

With regard to the threat of vulnerability and attack (when the vulnerability is removed, the threat of attack also disappears), the information system can be viewed as a system with failures and restores of the information security characteristic.

On the basis of this problem, take the following notation:

P_j and $\overline{P_j} = 1 - P_j$ – probabilities of successful and unsuccessful parrying of the *j*-th attack that arises, respectively;

$\lambda_j, j = 1,n$ – intensity of attacks on the information system;

μ_j – intensity of reflection of consequences of *j*-th attack;

$\mu_j \cdot P_j$ – intensity of reflection of attacks on the information system;

$\mu_j \cdot \overline{P_j}$ –intensity of the successful implementation of the attack on the information system;

λ_{prof} – intensity of conducting preventive security audit activities;

μ_{prof} – intensity of system recovery after preventive security audit activities;

P_{sv} – probability of eliminating one vulnerability;

μ_{reb} – intensity of restarting the system after the attack;

$\lambda_{patchup}$ – intensity of development of updates that address vulnerabilities;

λ_{patch} – intensity of patch development after a vulnerability attack;

μ_{patch} – intensity of system recovery after installing the patch (or updates);

λ_{fix} – intensity of vulnerability elimination;

P_{fix} – probability of restart with removal of vulnerability;

$\Delta\lambda_a$ – changing of the intensity of defects manifestation in the system, caused by the appearance or elimination of vulnerabilities in the system.

In addition, the following assumptions are adopted:

– the parrying and non-parrying flow of attacks on the system is the simplest;

– the possibilities for parrying the consequences of the impact on the information system of the j-th attack are not limited, that is $\mu_j \geq \lambda_j$.

For any *j*-th impact with the intensity $\lambda_j^{i'}$ the system can be with probability $P_j^{i'}$ and intensity $\mu_j^{i'}$ in the initial state $S_0^{i'}$ (Fig. 46.5) that corresponds to a successful parry of the *j*-th attack by available methods and means.

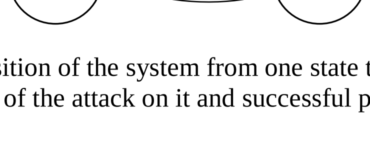


Fig. 46.5. The transition of the system from one state to another under the impact of the attack on it and successful parrying

With some periodicity, the system carries out preventive measures (states $S_{aud}^{i'}$), as a result of which it can be detected and eliminated from *0* to *n* vulnerabilities (Fig. 46.6).

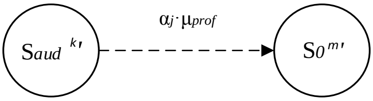


Fig. 46.6. The transition of the system from one state to another if vulnerabilities are eliminated after preventive measures

Detection and removal of several vulnerabilities from the set $[1,...,n]$ is possible in the prevention. For this it is necessary to introduce a paremeter α_j of the probability of detection and elimination of the *j*-th vulnerability $\left(j \in [1,...,n]\right)$. It is obvious that $\sum_{i=1}^{n} \alpha_i = 1$, and $\alpha_1, \alpha_2, ..., \alpha_j, ..., \alpha_n$ values have a discrete distribution law. To ensure equality $\sum_{i=1}^{n} \alpha_i = 1$ values of the α_j coefficients should be calculated ftom the firmulas in Table 46.1. In the transition to a new state after prevention, the intensity of attacks on the information system is redistributed to $\Delta\,\lambda_a$ (since the sum must be 1).

Table 46.1. Determination of the probability of *j*-th vulnerability detection

j	1	2	3	…	$n-1$	n
α_j	p	$q \cdot p$	$q^2 \cdot p$	…	$q^{n-2} \cdot p$	$1-\sum_{i=1}^{n-1} \alpha_i$

After the successful attack (transition to a state $S_c^{i'}$ with the intensity $\mu_j^{i'} \cdot \overline{P_j^{i'}}$ that is depicted in Figure 46.7), the system loses working capacity.

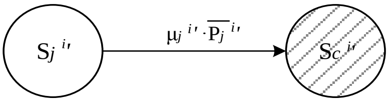


Fig. 46.6. The transition of the system from one state to another after a successful attack

After the manifestation of vulnerability there are two possible case scenarios: system service restarts if the vulnerability is not detected in a timely manner, then transition from the state $S_c^{i'}$ to the state $S_o^{i'}$ performs with the intensity $\lambda_{fix} \cdot \overline{P_{fix}}$ that is shown in Figure 46.8); vulnerability is detected and the system remains in the state $S_c^{i'}$ and the development of patch begins, and then the system goes into a state of patching $S_{pat}^{i'}$ with the intensity λ_{patch} (Figure 46.9).

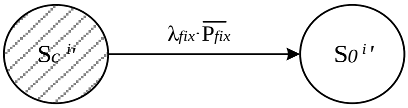


Fig. 46.8. The transition of the system from one state to another if the vulnerability is not detected in a timely manner

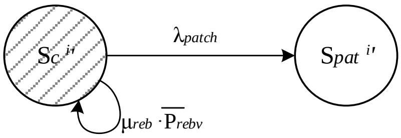


Fig. 46.9. The transition of the system from one state to another if the vulnerability is detected

After installing the patch, there are also several possible situations: one vulnerability is eliminated (Figure 46.10), group of vulnerabilities (Figure 46.11), the vulnerability is not eliminated (Fig ure 46.12), previously removed vulnerabilities are introduced (Figure 46.13).

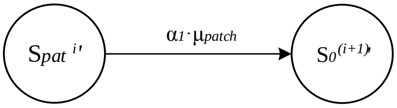


Fig. 46.10. The transition of the system from one state to another after patching if one vulnerability was eliminated

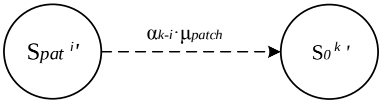


Fig. 46.11. The transition of the system from one state to another after patching if group of vulnerabilitis were eliminated

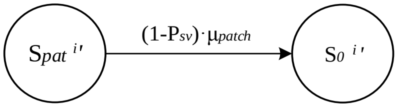


Fig. 46.12. The transition of the system from one state to another after patching if the vulnerability was not eliminated

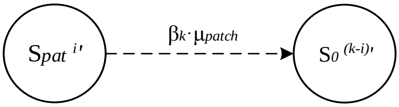


Fig. 46.13 The transition of the system from one state to another after patching if previously removed vulnerabilities were introduced

If the j-th vulnerability is eliminated $\left(j \in [1,...,n]\right)$, it is also necessary to use the paramemter α_j of the probability detection and elimination of vulnerabilities, as in the case of prevention. It is also calculated by the formulas from Table 46.1.

Since it is possible to introduce several vulnerabilities from the set $[1,...,n]$ after patching, it is also necessary to introduce a parameter β_j of the probability of making vulnerability, which has the same properties as α_j.

It is also possible to develop updates, as a result of which the system from the initial state goes into the state of patching with the intesity $\lambda_{patchup}$, that is depicted in Figure 46.14.

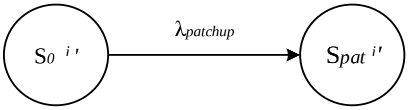
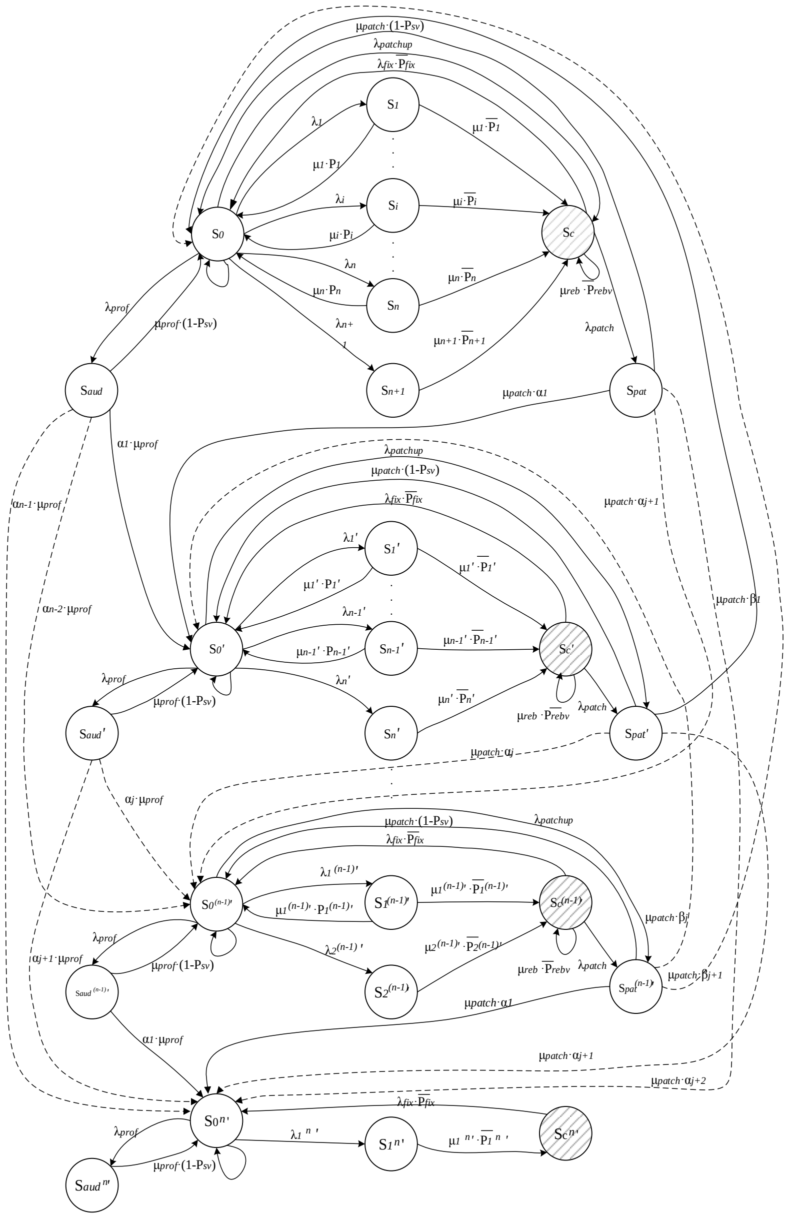


Fig. 46.14. The transition of the system from one state to another due to the development of updates

After detecting and eliminating all vulnerabilities (except the vulnerability of the "zero day") the system continues to function in normal mode.

For the case under consideration, a complete graph of states and transitions is depicted in Figure 46.15.

Thus, Markov processes can be used to identify the most dangerous threats, as well as the development of organizational and preventive measures to prevent the impact of attacks on the information system.



S_0
S_1
S_i
S_n
S_{n+1}
S_c
S_{aud}
S_{pat}
$\mu_{patch}\cdot(1-P_{sv})$
$\lambda_{patchup}$
$\lambda_{fix}\cdot\overline{P_{fix}}$
λ_1
$\mu_1\cdot\overline{P_1}$
$\mu_1\cdot P_1$
λ_i
$\mu_i\cdot\overline{P_i}$
$\mu_i\cdot P_i$
λ_n
$\mu_n\cdot\overline{P_n}$
$\mu_n\cdot P_n$
$\mu_{reb}\cdot\overline{P_{rebv}}$
λ_{prof}
$\mu_{prof}\cdot(1-P_{sv})$
λ_{n+1}
$\mu_{n+1}\cdot\overline{P_{n+1}}$
λ_{patch}
$\mu_{patch}\cdot\alpha_1$
$\alpha_1\cdot\mu_{prof}$
$\alpha_{n-1}\cdot\mu_{prof}$
$\alpha_{n-2}\cdot\mu_{prof}$
$\mu_{patch}\cdot\alpha_{j+1}$
$\mu_{patch}\cdot\beta_1$
S_0'
S_1'
S_{n-1}'
S_n'
S_c'
S_{aud}'
S_{pat}'
λ_1'
$\mu_1'\cdot\overline{P_1'}$
$\mu_1'\cdot P_1'$
λ_{n-1}'
$\mu_{n-1}'\cdot\overline{P_{n-1}'}$
$\mu_{n-1}'\cdot P_{n-1}'$
λ_n'
$\mu_n'\cdot\overline{P_n'}$
$\mu_{patch}\cdot\alpha_j$
$\alpha_j\cdot\mu_{prof}$
$S_0^{(n-1)'}$
$S_1^{(n-1)'}$
$S_2^{(n-1)'}$
$S_c^{(n-1)'}$
$S_{aud}^{(n-1)'}$
$S_{pat}^{(n-1)'}$
$\lambda_1^{(n-1)'}$
$\mu_1^{(n-1)'}\cdot\overline{P_1^{(n-1)'}}$
$\mu_1^{(n-1)'}\cdot P_1^{(n-1)'}$
$\lambda_2^{(n-1)'}$
$\mu_2^{(n-1)'}\cdot\overline{P_2^{(n-1)'}}$
$\mu_{patch}\cdot\beta_j$
$\mu_{patch}\cdot\beta_{j+1}$
$\alpha_{j+1}\cdot\mu_{prof}$
$\mu_{patch}\cdot\alpha_{j+2}$
$S_0^{n'}$
$S_1^{n'}$
$S_c^{n'}$
$S_{aud}^{n'}$
$\lambda_1^{n'}$
$\mu_1^{n'}\cdot\overline{P_1^{n'}}$

Fig. 46.15. Graph of states and transitions of the system under the impact of *n* independent threats and the removal of vulnerabilities

Conclusions

The increasing number of successful cyber crimes and cyber attacks performed by highly skilled attackers targeted at the web system motivates developers to make a shift from the traditional testing techniques for ensuring the cyber security of web applications (such as manual, automation testing, firewall, etc.) because of their ineffectiveness to the new ones.

A system solution to the task of ensuring the cyber security of the process of Web-based systems under development (as well as after development) can be a set of activities to regulate the stages of software development and penetration tests which can be part of the security audit.

As an example of web-system development process which includes penetration testing the Secure Software Development Life Cycle is reviewed. Incorporating penetration testing into the automated testing suite will identify and eliminate potential vulnerabilities in earlier stages of software development.

During the penetration testing in most cases, combinations of methods and tools are used to analyze the security of web systems. For the implementation of pentesting, there are various techniques in which one details how to perform penetration testing, and what tools to use.

The task of formalization of the penetration testing approach related to its quantitative and qualitative research through modeling of the main stages is challenged in this chapter. Markov models and attack tree analysis as methods for web systems analysis which can be applied independently or in combination with some existing tools for conducting penetration testing of Web systems are described.

Questions and tasks for self-control

1. What is NIST?
2. The main purpose of the OSI model.
3. The relationship between Web security standards and the OSI model.
4. List the threats to the security of web systems.
5. Technologies, services, protocols used to ensure the security of web systems.
6. Features of using SSL / TLS
7. Advantages and disadvantages of IPSec.
8. Genesis of software development methodologies and technologies.
9. Disadvantages of the cascade model.
10. Disadvantages of the iterative model.
11. Basic methods of implementing pentesting.
12. Sources of up-to-date information about software vulnerabilities.
13. Life cycle of the development of secure software (web-system).

14. What is an event tree?
15. Give a formula for calculating the probability of events combined by AND.
16. Give a formula for calculating the probability of events combined by OR.
17. What can act as input data for the model?
18. What are the disadvantages of the method of assessing the security of Web applications using attack trees?
19. What are the advantages and disadvantages of using the Markov's analysis to assess the security of Web applications?
20. Under what conditions is it justified to use the apparatus of Markov stochastic processes?
21. What are the input and output parameters of the Markov model?

47. Techniques for penetration testing of web systems and networks

47.1 Positioning of penetration testing for Web systems in PTES

Currently, there are various standards in the field of cybersecurity in general, and in pentesting in particular. One of the most well-known de facto standards for penetration testing is PTES or Penetration Testing Execution Standard. First information about this standard appeared in public access in 2011 [34], and taking into account the rapidly changing realities of the IT sphere, the fact that the PTES project exists for more than 6 years, objectively indicates its relevance and consistency.

In addition, the possibility of formal description of the pentesting process in accordance with known recommendations (it means the unification of the reporting documentation about the conducted penetration testing) allows us to move from *"handicraft"* in the study of cybersecurity to the initial (basic) properties of industrial IT systems. The next step is to audit compliance with one of the CMM (Capability Maturity Model) levels [35].

For the proper positioning of the task (process) of Web Systems Pentesting (WSP), it's necessary a detailed studying of the de facto standard in the field of penetration testing. Obviously, there is a separate item in the standard devoted to web systems testing, However, the whole pentesting ecosystem generally can contain elements that are in interaction with the task under study. Therefore, to obtain an objective view of the WSP positioning without detailed consideration of PTES is not possible. That is why a short but highly structured description of the standard is proposed for studying, based on information from an official source [36].

The following are the main stages defined by the standard as the basic ones in performing the pentesting:

1. Pre-engagement Interactions (agreement)
2. Information (intelligence) gathering
3. Threat modeling
4. Vulnerability Analysis
5. Exploitation
6. Post Exploitation
7. Forming reports

Structurally and graphically the above stages can be represented in the form of mind maps [37]. Hierarchical models in this case are applicable only when decomposing each of the pentesting stages. An example of a fragment of such a mind map of the PTES standard is shown in Figure 47.1 [38].

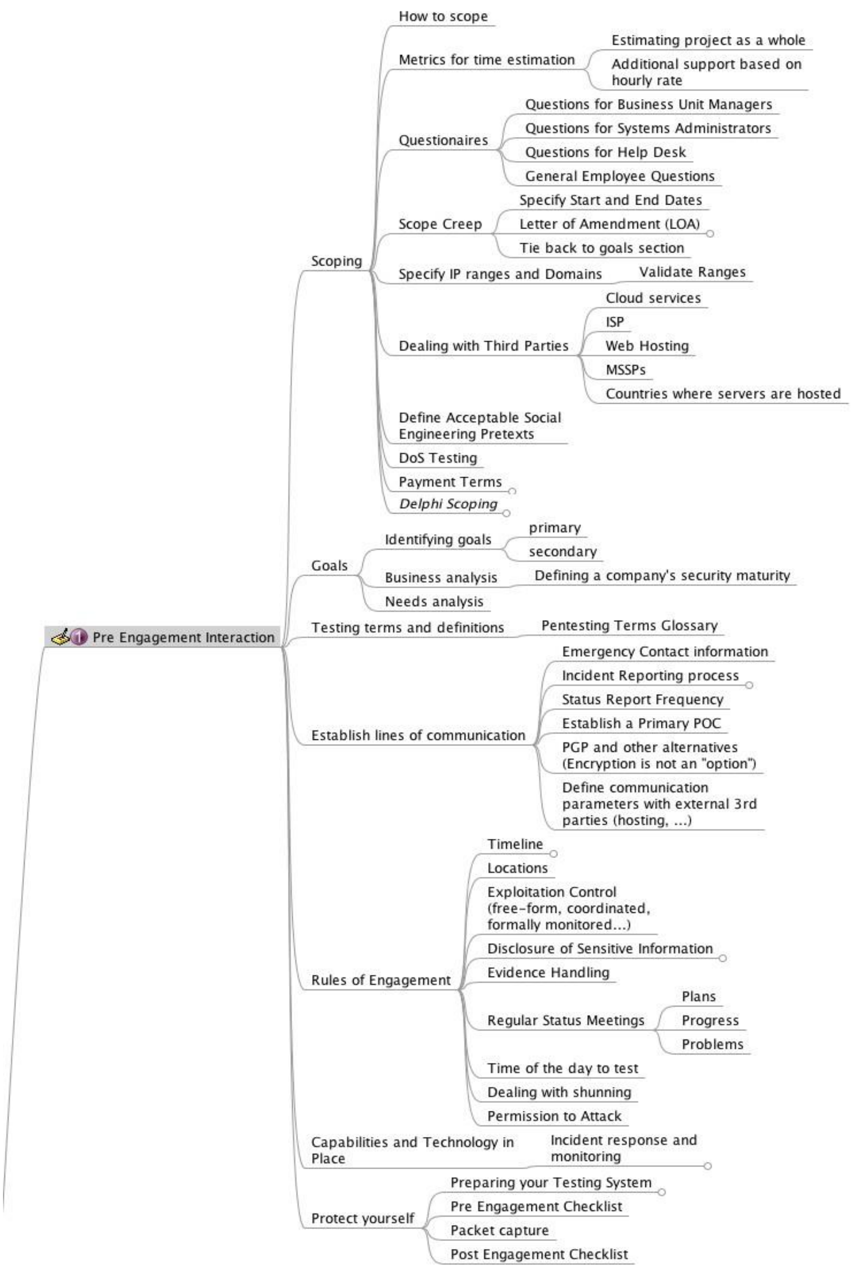

Pre Engagement Interaction
Scoping
How to scope
Metrics for time estimation
Estimating project as a whole
Additional support based on hourly rate
Questionaires
Questions for Business Unit Managers
Questions for Systems Administrators
Questions for Help Desk
General Employee Questions
Scope Creep
Specify Start and End Dates
Letter of Amendment (LOA)
Tie back to goals section
Specify IP ranges and Domains
Validate Ranges
Dealing with Third Parties
Cloud services
ISP
Web Hosting
MSSPs
Countries where servers are hosted
Define Acceptable Social Engineering Pretexts
DoS Testing
Payment Terms
Delphi Scoping
Goals
Identifying goals
primary
secondary
Business analysis
Defining a company's security maturity
Needs analysis
Testing terms and definitions
Pentesting Terms Glossary
Establish lines of communication
Emergency Contact information
Incident Reporting process
Status Report Frequency
Establish a Primary POC
PGP and other alternatives (Encryption is not an "option")
Define communication parameters with external 3rd parties (hosting, ...)
Rules of Engagement
Timeline
Locations
Exploitation Control (free-form, coordinated, formally monitored...)
Disclosure of Sensitive Information
Evidence Handling
Regular Status Meetings
Plans
Progress
Problems
Time of the day to test
Dealing with shunning
Permission to Attack
Capabilities and Technology in Place
Incident response and monitoring
Protect yourself
Preparing your Testing System
Pre Engagement Checklist
Packet capture
Post Engagement Checklist

Fig. 47.1. Fragment of the mind map *"Pre-Engagement"* from PTES

Accordingly, each of the pentesting stages corresponds to an individual mind map, and the interconnection of all maps is a decomposition of the PTES standard.

From the point of view of WSP positioning, there will be different stages in the standard related to web-systems testing by the 3 ways. From the statistical point of view, this will be a strong, medium and weak correlation [39]. However, there is an objective difficulty in identifying such a correlation (especially for medium and weak coupling) before taking the necessary number of system behavior measurements (So-called representative sample), moreover, a change in the pentesting object will entail a change in the interrelation between the stages of PTES and the performance of the WSP.

However, the initial establishment of the interrelation between the WSP and the PTES stages can be done in expert way, based on the experts knowledge and skills as well as on the logic of the functioning of the software and hardware and, of course, knowing the basic principles of human psychology. And then, after obtaining a representative sample, based on the obtained results to verify the reliability of the established interrelations and, if necessary, to change it.

For example, for WSP in the context of PTES such steps are important:

1. Pre-engagement Interactions (IP range definition, domain, ISP, "cloudiness" of services, the possibility of DoS, etc.).
2. Intelligence Gathering (target selection, OSINT, hidden intelligence, footprinting, identification of security systems, etc.).
3. Threat modeling (business assets analysis, business processes analysis, threat agents analysis, potential damage analysis , etc.).
4. Vulnerability analysis (research, validation, testing).
5. Exploitation (targeted impact, circumvention of protective measures, exploit testing, etc.).
6. Post-exploitation (infrastructure research, business influence, "cleaning up tracks", "looting", etc.).
7. Reports formation (report on implementation, technical report, strategic development (implementation) plan).

Thus, the WSP positioning in PTES in the context of general direction of research methods and techniques of web systems penetration testing will allow to identify the potential ways of formalizing empirically-intuitive actions and results of pentester, which will in the future create appropriate projects that can be described by SaaS models [40], Which will allow to automate and increase the trust level and commercial attractiveness of such projects.

47.2 Stages of pentesting

Penetration Testing Execution Standard [36] consists of seven stages that cover all the processes that occur during testing, beginning with preliminary arrangements with the object owner and ending with the report creation which is the main document for the customer of services and is of the greatest interest to him. The interrelation of these stages to specific use scenarios is shown in the source [41].

The main stages of conducting the penetration testing below.

47.2.1 Pre-engagement interactions

This stage implies carrying out all the necessary arrangements between the customer of the service (the owner or representative of the attack object) and executor of the service (group of testers who will conduct pentest). There are a number of points that must be discussed. Below are some of them.

- *How long will the testing take?*

Like most tasks, the testing process is time-limited. Based on this value, the cost of the service is determined.

- *At what time should testing take place?*

The customer may wish that the testing conducts at night, when the system is not busy with customer service, and, in case of unforeseen disruptions, there will be time before the next working day to restore the system's working state.

- *What should do if the tester breaks the system?*

Perhaps, it is one of the most important issues to be solved at this stage. After all pentesting can do unforeseen consequences. Since pentesting is carried out at the exploration stage of system, a malfunction of the system can lead to losses. It is necessary to notify the customer about this and think over a set of measures to restore the operability as soon as possible (rebooting the equipment, recovering data from backups, etc.).

- *What kind of testing will be used?*

There are three types of testing: the "black box" testing, the "gray box" testing and the "white box" testing. There is a need to decide which species is best suited for the particular case:

- *The "black box" testing* means that the testers do not receive any additional information from the owner of the testing object, they extract all the information themselves in the next stage of testing, which is devoted to data collection. This approach is most similar to the malefactor actions who are trying to find weaknesses in the attacked object.
- *The "white box" testing* implies that the testers receive all information about the object: documentation, knowledge of the architecture, the technologies used, the source code, etc. That is, all the information that will give testers the most complete idea about the tested object.
- *The "gray box" testing* represents something between the approaches described above. Testers can be given some information that will expand their

knowledge about the testing object. This is done in order to increase the coverage of tests (for example, testers can not find all entry points to the application), and to reduce the time required for data collection.
Moreover, the following questions are need to be addressed by the pentester:

- Will social engineering be used?
- Will the personnel (employees, security service) know about the fact that pentesting is being conducted?
- Which services / domains / servers do not need to be tested?

This is only part of the general issues that need to be solved before moving on to the next stage. More complete list can be found on the PTES website [42].

47.2.2 Intelligence Gathering

As the name implies, the main purpose of this stage is an intelligence gathering. Any data can be useful.

First, there is a need to define test objects (if there is no information about them). Domains, IP addresses, used services, etc. should be defined. Such actions usually do not cause suspicion, since they do not require an active execution phase. After this, it is necessary to proceed to the active phase - gathering information about the software used, its versions, port scanning, determining the entry points to the Web services.

Using the common content management systems that has a modular architecture, there is a need to set the name and CMS version, the modules used and their versions. Having such knowledge, may be checked in the vulnerability databases [43], whether this software has known vulnerabilities.
The decomposition of the intelligence gathering phase was carried out using the technology of mind maps, the results of the developers (Chris Nickerson, Ian Amit, Wim Remes, Stefan Friedli) are presented in Figure 47.2 [38].

Using popular CMS and its components with open source code (WordPress, OpenCart, Joomla, DLE) allows testers to get the source code of the software they use and to search for vulnerabilities by code analysis.
In the case of large-scale testing, the gathering of personal information about the personnel can also make sense for creating thematic dictionaries with passwords and their further use in bruteforce selection.

At this stage, various tools come to help testers. For example, to determine working machines on a subnet the nmap utility used. It can also scan open ports and determine which services are running on that machine and which versions of the software are being used.

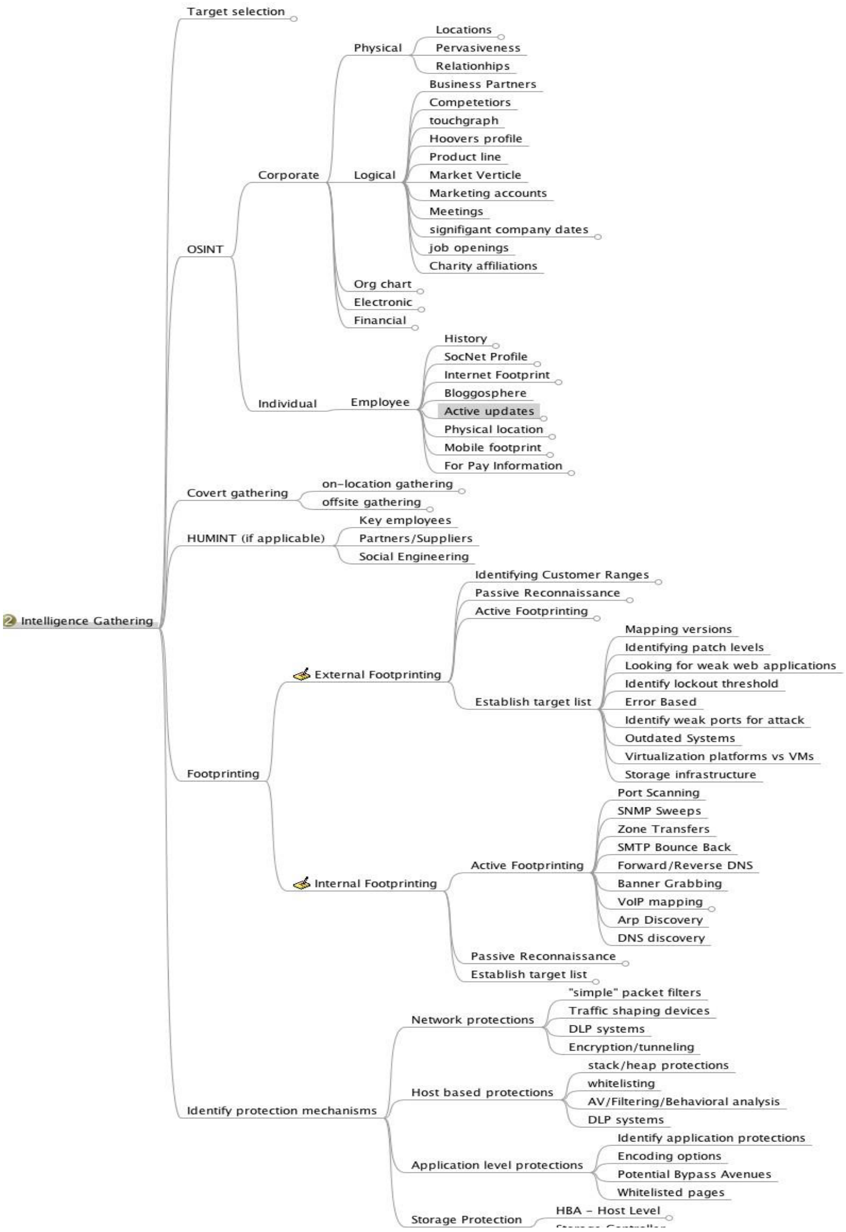


Fig. 47.2. Fragment of the mind map *“Intelligence Gathering”* from PTES

47.2.3 Threat Modeling

At this stage, the testers must understand which cyber-attacks can be of interest to third parties and which may serve as a reason for hacking.
As an example, consider an online store. What information might be of interest to malefactors:

- Bank card details – Malefactors can steal this information and use it for further fraud with the customers accounts of this store.
- Customers contact information – Malefactors can sell a customer's database to the another store owner.
- Product data – Similarly, you can sell this information, as human efforts were spent to add these goods to the store, descriptions and characteristics of the goods were compiled. The buyer of such information will not need to re- write descriptions of these goods.

After all, the site in a denial of service condition can also cause a loss, because while it does not work, customers are forced to buy products on competitors' sites.

Malefactors can also be attracted by a large number of site visitors (for example, any thematic portal). There are many different ways to monetize traffic, one of them is redirecting visitors of this site to paid advertising pages.
Even if a hacked site does not contain any cyberactivities, it can still be of use to malefactors. Computing power can be used to send spam, participate in network attacks, as well as cybercriminals can store malicious software on a hacked site and distribute links to it.

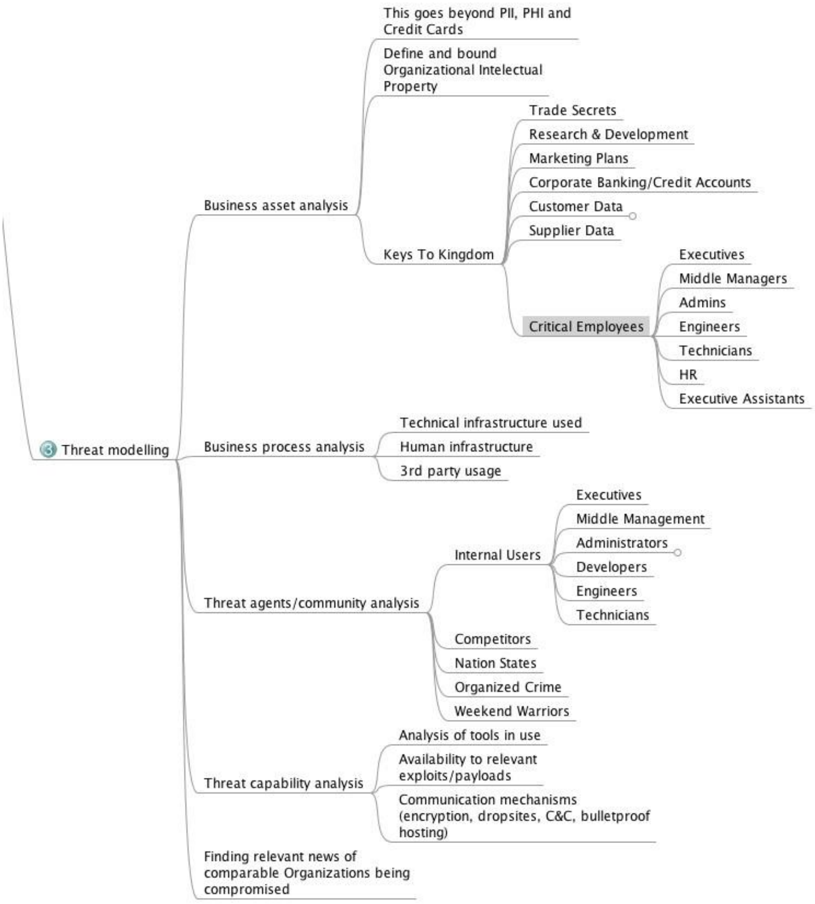
It is necessary to show the importance of these data and to let the customer know what he can lose in case of a successful attack.

In the process of threat modeling it is important to have a generalized holistic view of their number, structure, typification and other specific features. To this end, the standard developers created a mind map, a fragment of which is presented in the Figure 47.3 [38].

47.2.4 Vulnerability Analysis

Vulnerability Analysis – the process of defects identifying in the systems and applications that can be used by an attacker. Often these are design defects that have been committed at various stages of the software development life-cycle. To find vulnerabilities, there are a large number of scanners and other utilities that avoid repetition of monotonous actions manually.

The vulnerability search on the website usually begins with the compilation of a list of pages on the site. The scanner opens the main page of the site, finds links to the pages of the same site, remembers them and goes to one of the found pages. This happens until the scanner goes around all the site pages, like a search engine bot. If the content management system generates many similar pages, it is possible to configure the scanner to ignore similar pages. This setting will allow to crawl the site faster.



Threat modelling
Business asset analysis
This goes beyond PII, PHI and Credit Cards
Define and bound Organizational Intelectual Property
Keys To Kingdom
Trade Secrets
Research & Development
Marketing Plans
Corporate Banking/Credit Accounts
Customer Data
Supplier Data
Critical Employees
Executives
Middle Managers
Admins
Engineers
Technicians
HR
Executive Assistants
Business process analysis
Technical infrastructure used
Human infrastructure
3rd party usage
Threat agents/community analysis
Internal Users
Executives
Middle Management
Administrators
Developers
Engineers
Technicians
Competitors
Nation States
Organized Crime
Weekend Warriors
Threat capability analysis
Analysis of tools in use
Availability to relevant exploits/payloads
Communication mechanisms (encryption, dropsites, C&C, bulletproof hosting)
Finding relevant news of comparable Organizations being compromised

Fig. 47.3. Fragment of the mind map *“Threat Modeling”* from PTES

After the site map is compiled, the utility can test the possibility of SQL-injection or XSS-attacks. The utility will automatically find the GET-parameters of the page and the form for data entry, after that the utility will substitute specially generated values into these forms, which must be correctly processed on the server side. This method is called fuzzing. If there is a page on which the user data is not processed properly, the utility will issue a corresponding message.

While site scanning using the above method, in sight are only those pages that are referenced from the main or from other pages of the site. However, the web server may contain files that are not referenced from the site pages, and these files may be of interest to attackers. The search method is used to find such files and folders. There are dictionaries with common names of folders and files, with the help of special utilities it is possible to use these dictionaries and check the presence of some files on the server. This approach can also be used when determining the content management system used, since each CMS is characterized by its unique set of system folders and files.

Also at this stage, it is possible to check the ready exploits for the vulnerabilities that are present in the specific version of the software. Using information about software versions, it is possible to find vulnerabilities specific to these versions, and for them to find exploits on thematic sites, for example, exploit-db.com. There are no guarantees that the exploits will be working, but it's still worth trying.

Exploits can be represented in various ways. For example, as separate scripts in any programming languages or as modules for Metasploit – a convenient platform for creating and using exploits. This framework includes an archive with exploits for exploiting vulnerabilities in a variety of software. It is possible to create your own exploit using the Ruby programming language.

The decomposition of the vulnerability analysis phase allowed the developers of the PTES standard to identify the structural elements and present them in the form of a mind map, shown in the Figure 47.4 [38].

47.2.5 Exploitation

The previous stage was aimed at identifying vulnerabilities and the formation of attack vectors. The purpose of the current phase is to attack and gain access to certain previously cyberactivities.

The attack vector may include one or more elements of different levels of the exploitation phase structure. Detailed information on the structure and elements is shown in the Figure 47.5 4 [38].

At this stage, there is an active use of a variety of utilities that will help testers to solve the tasks. For example, to create a site map and test SQL injections and XSS vulnerabilities, the owasp-zap utility can be used, it has an intuitive interface that allows novice testers to easily master it. However, this utility claims universalization and represents a wider functionality than the sqlmap utility, which is designed to search and operate SQL injections. A special feature of this utility is the console interface, which can cause some inconvenience, since before starting to work there is a need to read the manual to this utility. To find directories and files by brute force the nikto utility is used.

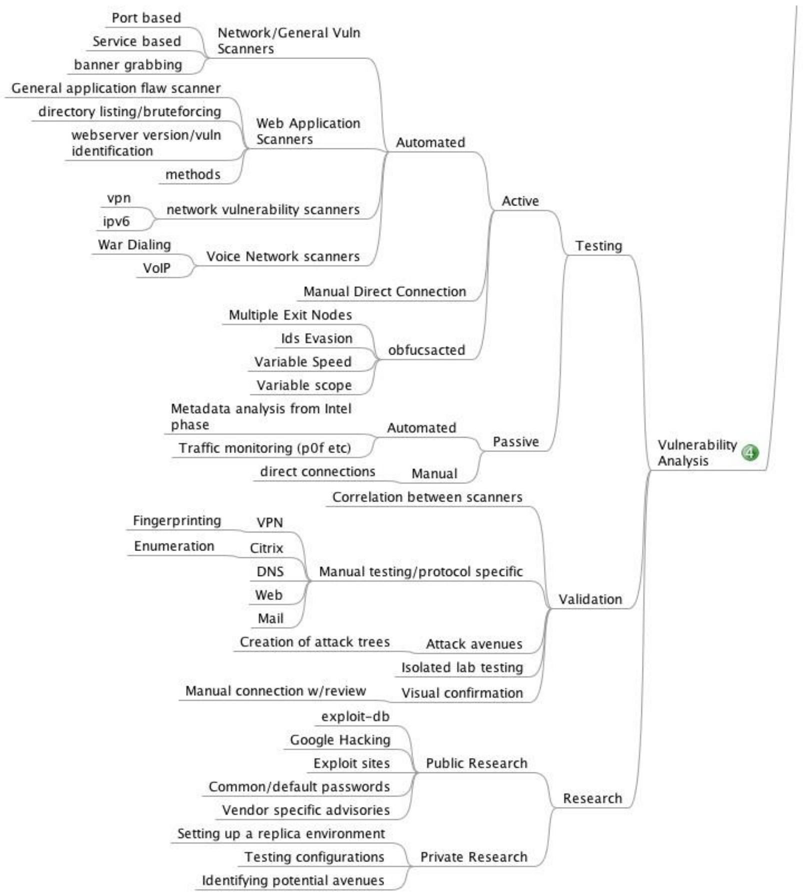


Fig. 47.4. Fragment of the mind map *“Vulnerability Analysis”* from PTES

To protect against attacks on the server side, various additional software can be used, for example, WAF. Web Application Firewall [44] – An application-level traffic filtering tool designed to detect and block attacks on Web applications, including using zero-day vulnerabilities. The use of WAF has traditionally been considered the most effective approach to protecting Web resources, since protection assurance is already occur during the exploitation of web applications.

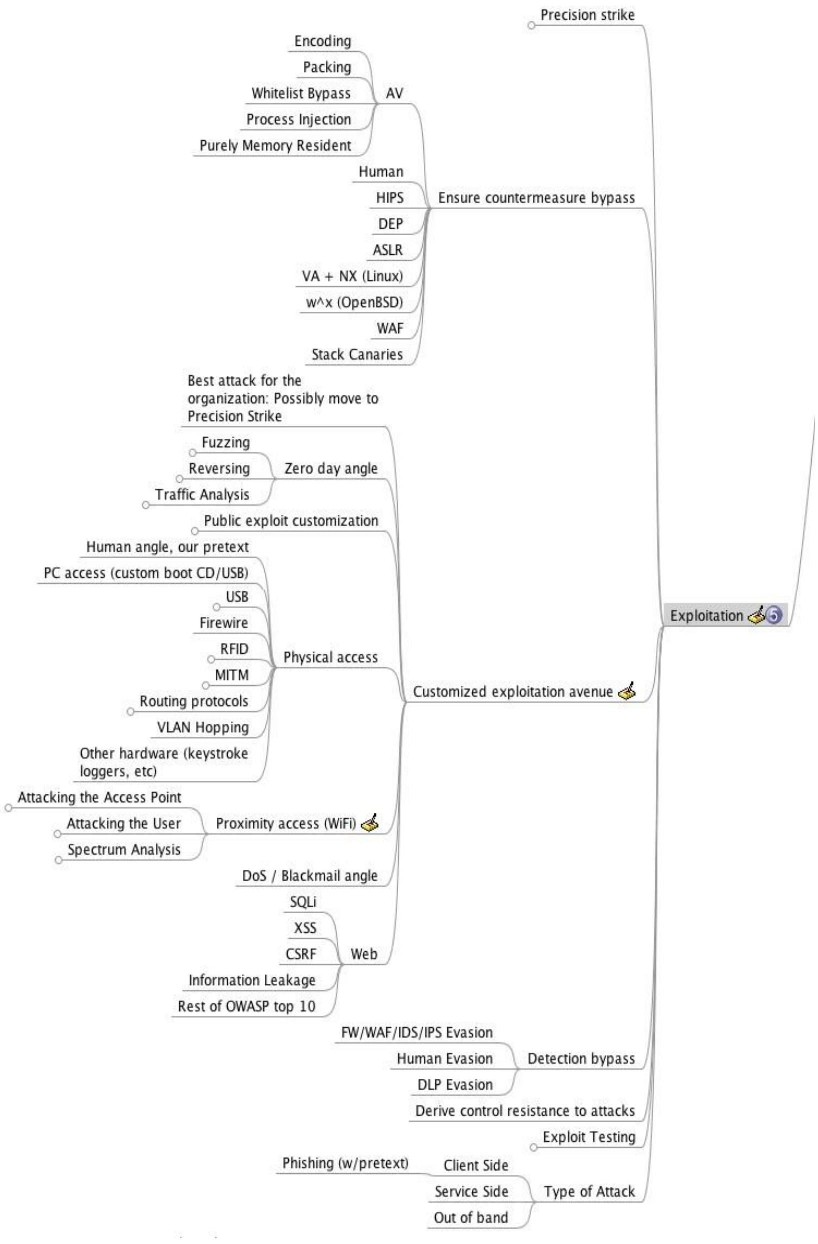


Fig. 47.5. Fragment of the mind map *“Exploitation”* from PTES

Intrusion Detection System (IDS) – a software or hardware tool designed to detect the facts of unauthorized access (intrusion or network attack) into a computer system or network. IDS is increasingly becoming a necessary complement to the network security infrastructure. In addition to firewalls, which work on the basis of security policy, IDS serve as mechanisms for monitoring suspicious activity. They can detect attackers who have bypassed the firewall and issue a report to the administrator, who in turn will take further steps to prevent the attack. The penetration detection technology does not make the system absolutely safe. Nevertheless, the practical benefits of IDS exist and is not small.

Intrusion Prevention System (IPS) – A software or hardware tool that monitors a network or computer system in real time to detect, prevent or block malicious activity. In general, IPS of its classification and its functions is similar to IDS. Their main difference is that they function in real time and can automatically block network attacks. Each IPS includes an IDS module.

47.2.5 Post-Exploitation

At this stage, it is necessary to determine the value and importance of those cyber-attacks that have been accessed. It is necessary to fix the methods and configuration settings with which was exploited the vulnerability.

Decomposition of the post exploitation stage in accordance with the identified criteria allows structuring the potential threats and risks that arise from the successful invasion. Graphically information is shown in the Figure 47.6 [38].

Since testers already possess of sensitive information, it is necessary to protect themselves and the client from possible threats associated with this. For example, any changes in the configuration of the system under test should be documented and there was a possibility to return the original configuration after testing. During the testing, additional files (for example, shells) could be downloaded to the web server, changes to database tables could be made - all this should be returned to the original state. A carelessly forgotten shell or an administrator account with a simple password added by the testers can lead to significant damage if access to it obtain a real malefactor. All confidential data received by the testers during the exploitation should be securely protected (encrypted) when stored on testers' computers. The received information should not be transferred to the third parties without the coordination with the customer.

Due to its own specifics, pentesting can be regarded as a violation of the law, because during the testing there is interference in the operation of the system. Data stored on servers may be protected by copyright. All these moments must be fixed in the contract.

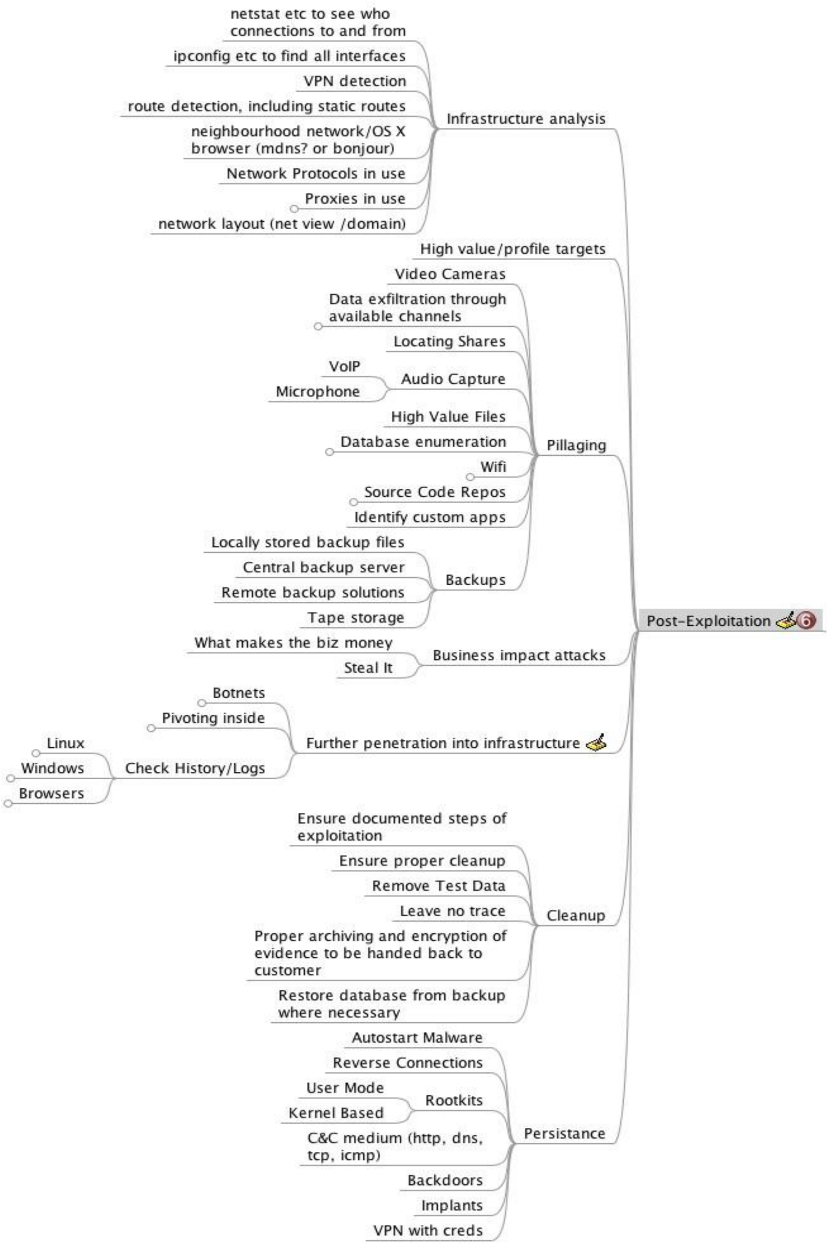


Fig. 47.6. Fragment of the mind map *“Post-Exploitation”* from PTES

47.2.6 Reporting

The report is the main way of transferring test results from performers to the customer. Therefore, the report makers should try and describe the results more fully.

The report includes goals, methods, results obtained and recommendations for the removal of identified gaps. It is necessary to indicate the information received for each of the above-described pentesting stages.

The targets are the cyberassets detected by the testers. The customer may not be able to imagine what damage might be done to him if these data fall into the attackers hands.

Also, the report consider as a document confirming the lack of actions on the pentester side, which could cause damage to the customer.

In addition, the structuring and detailing of this stage allows to significantly increase the customer's understanding of the actual amount of performed work, which may be grouding of services cost. The PTES developers structurally presented this stage as a mind map, which is shown in the Figure 47.7 [38].

In the methods it is necessary to show, by what ways specific goals can be achieved. For example, the customer stores some sensitive information in the database, this information is available only to system administrators. In this case, the scenarios will be as follows:

1. Information from the database can be obtained with SQL-injection.
2. Information from the database can be obtained by connecting directly to the database of the site. For this, there is a need to know the data that is stored in the CMS configuration file.
3. If there are corresponding vulnerabilities, there can be realization of increasing of a normal user rights to the administrator level. We remember that sensitive information is available to them.
4. The current system administrator can have a weak password, so his account can be easily compromised and also received the access to sensitive data.

It is only a few scenarios that can be used. Using of social engineering can significantly expand the list of possible options for achieving goals.

In the recommendations it is necessary to indicate the possibilities for eliminating the detected gaps. It can be installing patches, increasing the complexity of passwords, installing SSL certificates, installing a firewall and so on.

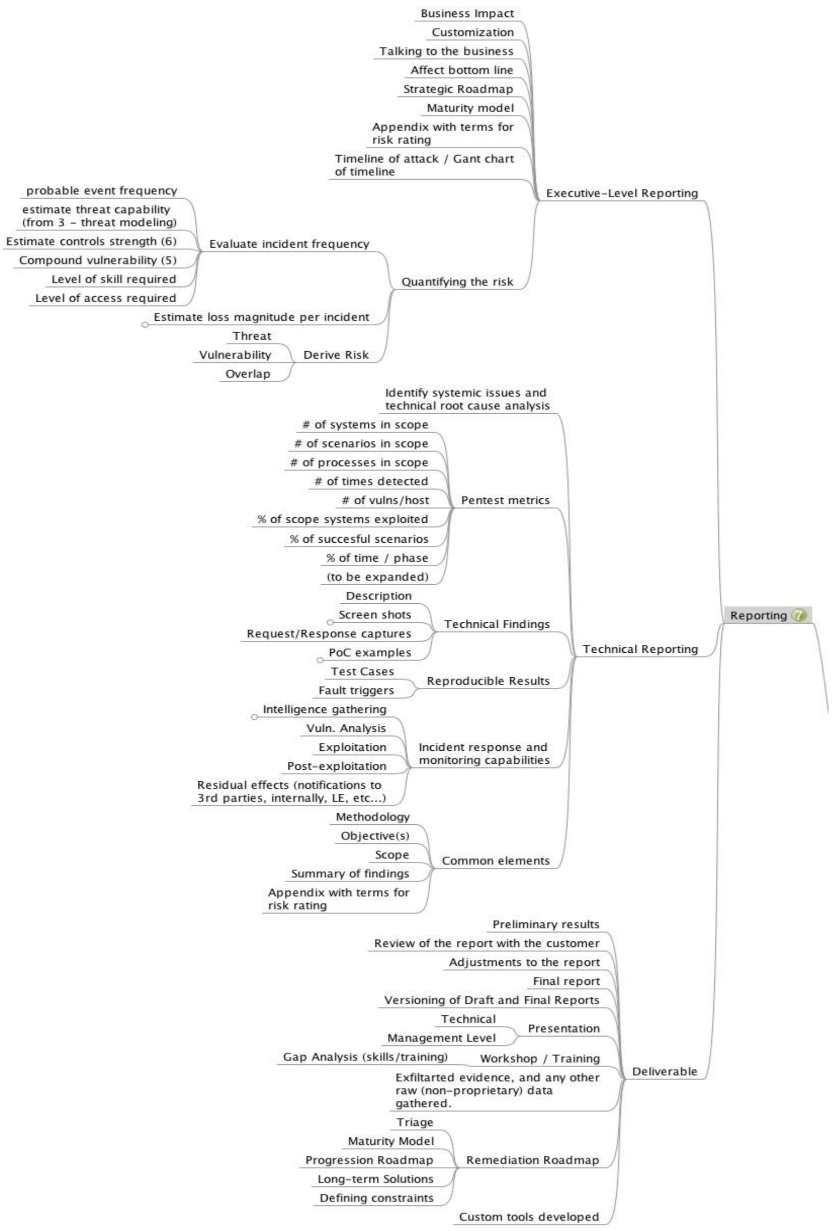


Fig. 47.7. Fragment of the mind map *"Reporting"* from PTES

47.3 The main vectors of attack

The OWASP (Open Web Application Security Project) community practice the classification of attacks vectors and vulnerabilities [45]. It is an international non-profit organization focused on analysis and improving the security of software.

OWASP created a list of the 10 most dangerous attacks vectors on Web applications, this list was called OWASP TOP-10. It collects the most dangerous vulnerabilities and attacks vectors, the most common in web applications. Some of them are critical and can lead to significant damage.

47.3.1 Injections

Almost all sites use storage - databases, which allows storing large amounts of data in an ordered form, which makes it convenient to work with them. Queries to the database are written in the SQL queries language (Structured Query Language).

Applications request data in the database when the user enters the page. A good example is the search page, when the script accesses the database with the requirement to return information about the books the user is looking for. Moving to the page of the selected book, the script will ask for information about this book, passing in the query to the database the unique identifier of this book.

The absence of data verification received from the user can lead to a violation of the query logic to the database in which this data are used. The script will execute the changed query and receive from the database those data that are not intended for it. For example, an attacker can view information about all orders on the site, although this information should be available only to the site administrator.

Injections [46] can allow not only unauthorized data retrieval, but also their modification or deletion. For example, an attacker can not only change information about his account, but also change the current account state.

47.3.2 Broken Authentication and Session Management

Applications can use cookie to store certain data in the client's browser. Such a mechanism is often used in the authorization process on the site. After authorization, the user receives the session key, which is saved in the browser. This is convenient, because during the next logging to the site, the user does not need to enter the login and password again, the browser tells the site that the user is already authorized and passes the session ID to the server [47].

An attacker can steal this key from the victim's browser and insert this key into his browser. Thus, an attacker gets access to a victim's account if the site does not control connections on the same key from different IP addresses.

However, binding only IP addresses won't solve this problem. Modern Internet providers often have ranges of addresses that users use to access the Internet. Therefore, it is advisable to check the IP belonging to the range on the server. But even such a solution does not guarantee security - an attacker can know which provider the victim is using and use IP from the same range. Moreover, the attacker can know the User Agent of the victim and use the same during the attack.

Thus, an attacker on the site impersonates someone else and can commit actions, the consequences of which can only be guessed.

47.3.3 Cross Site Scripting – XSS

Cross-site scripting, as well as SQL injection, occurs because of incorrect verification of data that comes from the user [48]. But in this case, not the logic of the query to the database changes, but the contents of the HTML document. Most often, iframe inserts or JavaScript-code inserts are added to the HTML code.

This can lead to the translation of data that is entered on the infected page to a third-party server. If such a modification is available on the payment page, where users enter their credit card information, all this information will be successfully transferred to the attackers.

Also, using this vulnerability, attackers can steal cookies from the victim's browser and try to use them in the attack that was described above.

Using JavaScript, you can also redirect users who visited an infected page. For example, they can be redirected to a phishing page for the purpose of misleading and invite them to enter a login and password for re-authorization. The user may not notice that the site addresses are different and enter their data on another site, the data from which will immediately reach the attackers and will be used to access the victim's account on the original website.

47.3.4 Insecure Direct Object References

The basis of this vulnerability is also the insufficient verification of user data [49]. For example, the site has the functionality of uploading photos, which should be available only to their owners. There is a script that displays the selected photo. Since the image information is most often stored in the database, the unique identifier of the selected photo is the primary key of the table. The reference to this object looks like this:

```
example.com/show.php?id=123,
```

where `123 -` is the photo ID.

If during object invocation the checking whether the requested object belongs to the current user is not implemented, then the user will be able to view photos of other users, although the developer of this system did not foresee such a possibility.

An example with photos may seem harmless. And now imagine that the script works on the same principle and displays personal messages, information about the order in the online store or the details of any payment. Such information can be of much greater interest to an attacker.

The peculiarity of this vulnerability is that for testing do not need too complex utilities. Part of the tests can be done by changing the contents of the address bar of the browser. To check values from the range, it is possible to use utilities that automatically substitute the desired value and analyze the response. A positive answer can be identified by the server response code or by the found keywords on the received page.

47.3.5 Security Misconfiguration

Quite often modern web systems are complex. The web server, the database server and the management system have different settings. They can be specified during the software installation. Some settings must differ on the development server and on the production server. For example, the output of error messages. In the development process, the output of such messages immediately allows developers to understand which line the error occurred in and what it consists in. On production-server output of information about errors must be disabled.

Information about errors can be used by malefactors [50]. It can contain a stack of function calls, file names and other useful information. For example, the function of creating a connection to the database can tell the database name, login and password of the database user that were used when the connection failed. This situation is quite real when the database server is overloaded or unavailable for other reasons. Systems can write error messages to a file and it is necessary that this file was protected from unauthorized access.

Also, after migration to production-server, there may be accounts with default logins and passwords, which can result to access to the system after a brute-force attack. There may be debugging files that are not related to the control system, but can display useful information for the attacker. It is necessary to monitor this.

Complex web-systems can have one entry point and in the process of work they can connect the necessary files with classes of executable code. It is necessary to ensure that these files are unavailable to users and provide secure mechanisms for interclass interaction.

47.3.6 Sensitive Data Exposure

Web applications by default transmit data over an unprotected HTTP protocol. Packages in this protocol are transmitted as a pure text. This allows attackers to intercept traffic using special programs-sniffers. If during the interception the user was authorized on the site, then his login and password will be known by attacker [51].

To create a secure connection, uses SSL certificate, which is required to authenticate and encrypt data. A protocol that uses encryption is called HTTPS. While its using in the user's browser, the safe connection icon appears as a green lock in front of the address bar.

As well as safe transfer of sensitive data, it is also necessary to ensure their safe storage. For example, for passwords are strongly recommended to use a hard-to-handle hashing mechanism. If an attacker knows the password hash, then there is a need to use the brute force or to use rainbow tables to try to find the initial value of the password.

To complicate the process of selecting the source message for the hash function, possible by using salt - some predefined value that will be concatenated with the original message at one of the stages of creating a hash.

It is possible to combine different hash functions, which also hampers the attacker work.

Salt storage in the file system is more preferable than in the database. If the database is compromised (for example, using SQL injection), the salt will remain unknown, and an attacker will have to make extra efforts to gain access to the file system.

Thus, an attacker needs to know which algorithm is hashing and what is the value of the salt. Only after that it is possible to start searching for the source password.

47.3.7 Missing Function Level Access Control

The essence of this vulnerability lies in the lack of verification of the availability of proper access to the requested object.

An example of such a vulnerability was given above when it came to *Insecure Direct Object References.* The server did not check whether the user has the right to access the requested object [52].

Most web applications verify access rights before displaying data in the user interface. However, applications must perform the same access control checks on the server when requesting any function. After all, there are still many auxiliary service queries that are often sent in the background asynchronously using AJAX technology.

Thus, it is necessary to clearly determine to which data the user can access and check his requests for the ability to access other user's data.

47.3.8 Cross-Site Request Forgery, CSRF/XSRF

The attack vector CSRF, also known as XSRF, allows an attacker to perform actions on the server on behalf of the victim where additional checks are not implemented [53].

For example, in some payment system for transferring funds to another account, there is a script with the following parameters:

```
    yourbank.com/transfer_money.jsp?transfer_amount=100
0&transfer_accou nt=12345678
```

where `transfer_amount` – amount for translation

and `transfer_account` –account number to transfer funds to.

If the victim comes to a site created by an attacker and containing the above appeal to the bank's website, a request for payment system page is secretly sent from her person. As a result, the money will go to the attacker account, and then, probably, will be promptly exchanged for Bitcoin or transferred to another irretrievable payment system, and getting them back will be difficult or impossible.

It is assumed that the victim should have previously been authenticated in the payment system and an active session should be open (for example, the site of the payment system was opened in another tab of the current browser).

47.3.9 Using Components with Known Vulnerabilities

The most sites are hacked just because they can be hacked. Such sites may not contain any data useful to the attacker (for example, accounts or bank card data). And be even more surprised when you learn that when a large number of sites are hacked, the attacker's involvement is not mandatory. An attacker can write code that will find sites with known vulnerabilities [54].

An excellent example would be the use of vulnerabilities in components of popular CMS (WordPress, Joomla и etc.). The problem and the advantage at the same time is that the components of extensions can be created by any developer. The end user of the written module does not know how safe this module is. When such a module becomes popular, it accumulates a certain number of installations, someone may want to check the module for vulnerabilities. And if it detects a vulnerability, then all websites that use this module will be at risk. An attacker can only find the sites that this module has installed, and hack them.

If the installation of the plug-in can be several thousand, the number of installations of the CMS can be measured in millions, and the vulnerability found in the CMS core also endanger all installations of this system.

After all system software (operating system, web server) is used for the system functioning, and it can also contain vulnerabilities, the exploitation of which can lead to serious consequences.

It is very important to use the latest versions of components and to monitor the emerging known vulnerabilities on the vulnerability aggregators, for example, *securityfocus.com*. No one is immune from the fact that there will be no vulnerabilities in the new versions of the updates, but when using components without known vulnerabilities the probability of successful hacking will be lower.

47.3.10 Unvalidated Redirects and Forwards

Web applications often redirect users from one page to another. In the process, improperly verifiable parameters can be used, specifying the destination page for the redirection.

Without proper checks, the attacker can use such pages to redirect the victim to a forged site that, for example, may have a very similar or indistinguishable interface, but will steal your credit card information or other sensitive confidential data [55].

This kind of vulnerabilities, as well as many others listed above, is a type of input validation errors.

Conclusions

There are numerous standards in the field of cybersecurity. One of the most well-known de facto standards for penetration testing, Penetration Testing Execution Standard is reviewed. he seven main stages of PTES with mind maps from are described.

The web-system penetration testing positioning in PTES in the context of general direction of research methods and techniques of web systems penetration testing allows to identify the potential ways of formalizing empirically-intuitive actions and results of pentester.

The most common vectors of attack vectors and vulnerabilities classified by the Open Web Application Security Project community, knowing as OWASP-10 are reviewed.

Questions and tasks for self-control

1. What is PTES?
2. What is CMM?
3. What is the main purpose of mind maps?
4. List the main stages of PTES.
5. What is correlation, how can it be applied in our conditions?
6. Name the elements of the PTES stages.
7. What are the advantages and disadvantages of SaaS?.
8. What is OWASP?
9. List attack vectors from OWASP Top-10.
10. Why there is a need to use the HTTPS protocol?

11. What is the reason to update the software in a timely manner?
12. Why SQL injections are differ in their criticality?
13. How to check if there are known vulnerabilities in the software being used?
14. What is the reason to use complex passwords?
15. What is the reason to use a hash function for passwords?
16. What are the advantages of using a hash with salt?
17. Give examples of cyberaccess.
18. What is the cause of a large number of vulnerabilities in CMS modules?
19. What data is sensitive and subject to encryption?
20. What can lead to the display of errors on the site pages?
21. How CMS that is used can be identified?

48. Tools for penetration testing of web systems and networks

48.1 Platforms with vulnerabilities for training purposes

At the present time, there is a set of systems with vulnerabilities that are created with the purpose of training and gaining experience in the search and exploitation of various kinds of vulnerabilities. Many of these systems with a brief description can be found at *www.vulnhub.com.* These systems have different severity levels of vulnerabilities exploitation; mostly will be useful for beginners in the field of information security. Several popular web platforms with vulnerabilities will be further described.

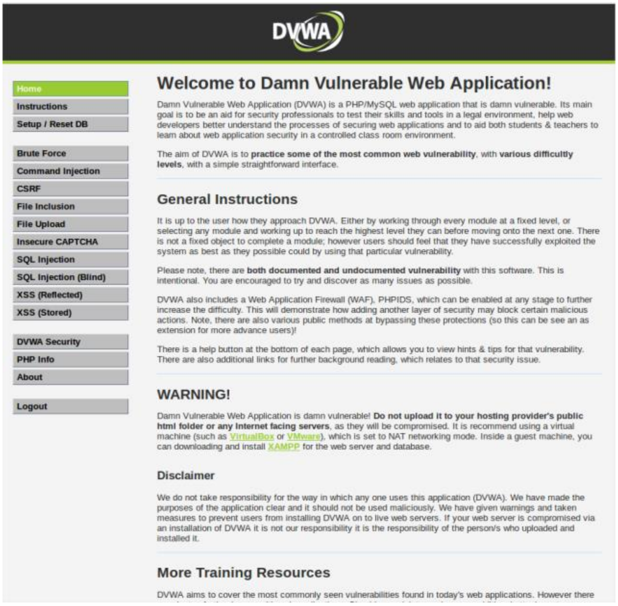
Damn Vulnerable Web Application (DVWA) is web application based on PHP/MySQL which has a huge number of vulnerabilities. The main purpose of the existence of such application is to assist information security specialists in their skills assessment, test different tools without having legal problems; to assist web developers in better understanding of the mechanism of writing secure code; and to enable students and lecturers learn more about the security of web applications in a controlled environment.

DVWA provides an opportunity to practice in the most popular web vulnerabilities at different levels of complexity. The application has a simple and intuitive interface. Application developers warn that there may be unintended and undocumented vulnerabilities within the application.

The platform can be installed without any difficulties (as most conventional CMS), enough to download an archive from the website [56], put and unpack into a web server folder, create a database, edit configuration file and run the installation script in a browser. With the successful installation the following page is displayed as shown in Figure 48.1.

It is possible to reinstall server in order to reset all settings in the section Setup/Reset DB. Moreover, it is possible to observe the status of various services that are needed for a particular task. Additional technical information can be found in the section Instructions. The menu contains links to tasks, which can be allocated into categories:

– Brute Force;
– Command Injection;
– CSRF;
– File Inclusion;
– File Upload;
– Insecure CAPTCHA;
– SQL Injection;
– SQL Injection (Blind);
– Reflected XSS;
– Stored XSS.



Welcome to Damn Vulnerable Web Application!

Damn Vulnerable Web Application (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goal is to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and to aid both students & teachers to learn about web application security in a controlled class room environment.

The aim of DVWA is to **practice some of the most common web vulnerability**, with **various difficultly levels**, with a simple straightforward interface.

General Instructions

It is up to the user how they approach DVWA. Either by working through every module at a fixed level, or selecting any module and working up to reach the highest level they can before moving onto the next one. There is not a fixed object to complete a module; however users should feel that they have successfully exploited the system as best as they possible could by using that particular vulnerability.

Please note, there are **both documented and undocumented vulnerability** with this software. This is intentional. You are encouraged to try and discover as many issues as possible.

DVWA also includes a Web Application Firewall (WAF), PHPIDS, which can be enabled at any stage to further increase the difficulty. This will demonstrate how adding another layer of security may block certain malicious actions. Note, there are also various public methods at bypassing these protections (so this can be see an as extension for more advance users)!

There is a help button at the bottom of each page, which allows you to view hints & tips for that vulnerability. There are also additional links for further background reading, which relates to that security issue.

WARNING!

Damn Vulnerable Web Application is damn vulnerable! **Do not upload it to your hosting provider's public html folder or any Internet facing servers**, as they will be compromised. It is recommend using a virtual machine (such as VirtualBox or VMware), which is set to NAT networking mode. Inside a guest machine, you can downloading and install XAMPP for the web server and database.

Disclaimer

We do not take responsibility for the way in which any one uses this application (DVWA). We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.

More Training Resources

DVWA aims to cover the most commonly seen vulnerabilities found in today's web applications. However there

Fig. 48.1. DVWA home page

For a better understanding of the application operation, there are tips on the page of each task (View Help), and also in the cases of failed attempts of hacking there is an ability to view server-side PHP code (View Source). There is an ability to choose a level of complexity (description is presented for each level of complexity) on the page DVWA Security. In addition, for complicating the task, there is an ability to enable available Web Application Firewall and IDS called PHPIDS. Page Phpinfo is a result of function phpinfo() which provides technical details of the server. Supplementary material can be found on the page About.

Another platform is Linux distribution called Damn Vulnerable Linux (DVL). DVL developers have made efforts to add to distribution all that should not be in a good Linux-system. It contains a vulnerable and poorly configured software whereupon the whole system is incredibly low protected.

The objective of the DVL project is to use this system for educational purposes training students security in information technology. The composition of Damn Vulnerable Linux distribution includes such outdated and easily vulnerable software as web-server Apache, DBMS MySQL, PHP interpreter, FTP- and SSH-servers. Furthermore, auxiliary utilities like GCC, GDB, NASM, strace, ELF Shell, DDD, LDasm, LIDa are included into DVL.

LiveDVD-image Damn Vulnerable Linux is available for free download at the project website [57].

OWASP WebGoat [58] is a platform for developing skills of websites penetration testing. The availability from anywhere of web applications has its disadvantage, it is necessary to pay extra attention to the security of the solution. And this is not a simple matter because in the formation of the resulting content involves several elements like a web server or application server, handler (HTML, PHP, Python, Perl, Ruby, JavaScript, AJAX etc.), database. Project WebGoat decided to release a platform in which all these technologies are combined and available to anyone who wishes to study. OWASP WebGoat is a cross-platform tool; it can be run on any operating system, which will operate Apache Tomcat and Java SDK. The platform has links to the learning theoretical material; a web browser is used to access the lectures.

Security professionals and developers should look into the application as a potential cracker, understand the essence of attack and see problem areas. For such situation, the developers of OWASP have created special training system WebGoat which allows in visual form study methods of web applications hacking. The base for conducting about 30 different attack types was implemented in it.

48.2 Methods of tools selection

Today there is a wide range of tools that help penetration testers to do their job efficiently. It is difficult to assume how long it would take to test hundreds or thousands of different tests manually. Each penetration tester has its own set of tools, which as it seemed to him are most convenient in the work and proved to be the best. Existing tools can be grouped into three categories according to their functionality:

– tools that perform testing on multiple vectors of attack. For example, OWASP ZAP tool combines the functions of a web-crawler, conducts testing of SQL-injections and possible XSS-attacks;

– narrowcasting tools that that test only one attack vector. An example is a sqlmap tool aimed at identifying possibilities of SQL-injection;

– tools of "targeted action" that test whether there exists a particular vulnerability in a particular system. Such tools include different exploits because they are not universal. An example is a tool that in a certain version of CMS Joomla can check the correctness of processing inputs from the address bar. If the data is processed incorrectly, it is possible to execute arbitrary code on the target system.

Due to a lot of tools, that sometimes implement the same functionality, a beginner penetration tester has a question: «What tools do I need to choose for specific tasks? ». He has to spend time reading forums with feedbacks about the tools or take the time to work with each tool and search for optimal solutions. Therefore, an idea for developing service that will help to solve the problem of choice of tools for penetration testing is suggested. The service uses neural network mechanism, the advantages and disadvantages of this approach are described at the end of sub-section.

The main element of the system under development is a matrix that defines associations of the set of tools with multiple criteria. A general view of the matrix is shown in Figure 48.2.

	CR1	CR2	CR3	...	CRN
Tool_1					
Tool_2					
Tool_3					
...					
Tool_N					

Fig. 48.2. Tools and criteria matrix

To determine the criteria that are of interest to the user, the system asks the user questions. From the set of possible criteria, the user chooses those that interest him. Thus, a vector of custom criteria will be formed. By setting the goal of selecting the most suitable utilities with a perceptron, first of all, it is necessary to determine the input vector X and the output vector D, specifying their dimensions and agree on the contents of each component [59]. In the vector X, it is logical to provide parameters that the system finds out from the user, for example, the cost, the ability to perform certain functions, the ability to conduct certain tests, etc. In the output vector D, it is necessary to encode all possible tools that are known at the moment and can be useful to a penetration tester. Thereby the output vector of the perceptron D will consist of a set of zeros and one or more unities (if there are several suitable tools). However, the tool search result is better to encode on a five-, ten- or one-hundred-point scale. Then at the stage of preparing a training set of examples using points, it will be possible to take into account the degree of the expert's confidence in the correctness of his decision and the probability of a correct response perceptron at the operational stage.

This is followed to prepare the set of training examples. As a result of the joint work of the programmer and the expert, owning experience with all tools, the required number of training examples will be created. It should be noted that a quality of the neural network system of choice directly depends on the qualification of the expert on the examples of whose work it has learned. The fact is that the neural network inherits from the expert not only his knowledge but also the gaps in his practical experience. It is clear that it will make the same mistakes that the expert did choosing tools. Therefore, to ensure a high quality of choice, the neural network should be trained on the examples of the work of a highly qualified tester or even on the results of the work of several experts.

So a result of long work of the expert testers and programmers a lot of training examples will be accumulated, consisting of many pairs of vectors *Xq* and *Dq, where (q =1,2, ..., Q)*. The task is to design the perceptron and by the training transfer to it the knowledge and experience contained in a variety of learning examples. The questions of designing perceptrons, i.e. selecting a number of hidden layers of neurons contained therein, and types of activation functions will not be considered.

As a result, the perceptron should learn how to display any vector of training set *Xq* to vector *Yq*, coinciding (or almost coinciding) with the vector *Dq*. Furthermore, when a new user with a new input vector *X* appears, the perceptron should calculate for it a new vector *Y* containing the correct choice of the tool carried out by the perceptron without the help of an expert. In other words, the perceptron should be able to generalize the experience given to it to new examples of the subject area, perform a selection of tools for new users with their requirements.

48.3 The use of tools for the main attack vectors

SQL-injection is not in vain in the first place in the vulnerability rating of OWASP Top-10. This is due to the wide spread of this vulnerability and its high level of criticality. Because of its prevalence, there are several tools that allow a user to search and exploit SQL-injections, however, each of them is unique in its way and has a number of advantages and disadvantages. For example, the advantages include a convenient window interface, and the disadvantage is an inability to detect some types of injections. Figure 48.3 shows known varieties of SQL-injections for this moment.

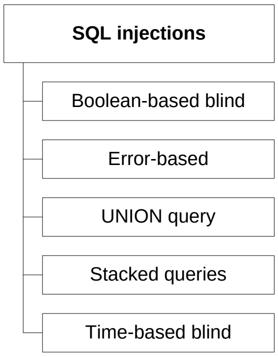


Fig. 48.3. Varieties of SQL – Injections

A complete information about the vulnerabilities related to the violation of a request logic to a database is shown in [60].

The highly specialized *sqlmap* tool [61] is well suited for searching and exploiting all the types of injections shown above. It is one of the most powerful open source tools which automates the process of searching for and exploitation of SQL-injection to retrieve data or capture a remote host. Sqlmap makes different from other tools the opportunity to exploit every vulnerability that was found. This means that sqlmap is able not only to find vulnerabilities but also to exploit them as efficiently as possible. Since the task is to exploit the vulnerability, the scanner has to be especially attentive to the details: It will not give out a million false positives (as you can see in many other applications). Any potential vulnerability is further tested for the possibility of exploitation. The scanner "from the box" has a huge functionality ranging the ability to define a database management system (DBMS), create a dump (copy) of data and ending with getting access to the system with the ability to access arbitrary files on the host and execute arbitrary commands on the server. However, the main is the possibility to make the detection of SQL-injection in the code.

This tool has several advantages:

– the possibility of automated testing – a file with addresses is passed as a parameter; a key is specified that uses the default answers for questions that arise during the tool operation;

– the possibility to test all the data that comes from the user in the script;

– the possibility to use specific technique;

– the possibility to use the DBMS console if sequential queries are supported;

– the possibility to use the OS console on which the server is installed;

– the possibility to select hash values obtained from the database;

– the possibility to save single tables or the entire databases as *csv* files;

– the advanced mechanism for searching databases, tables or even columns (for all databases at once) that can be useful for defining tables with «interesting» data as users names (users) or passwords.

– the possibility of direct connection to the database if you know the username and password;

– the possibility to check the forms at the specified URL – the tool automatically determine the names of the variables and the transfer method and test them;

– the possibility of specifying a User-Agent or using random values;

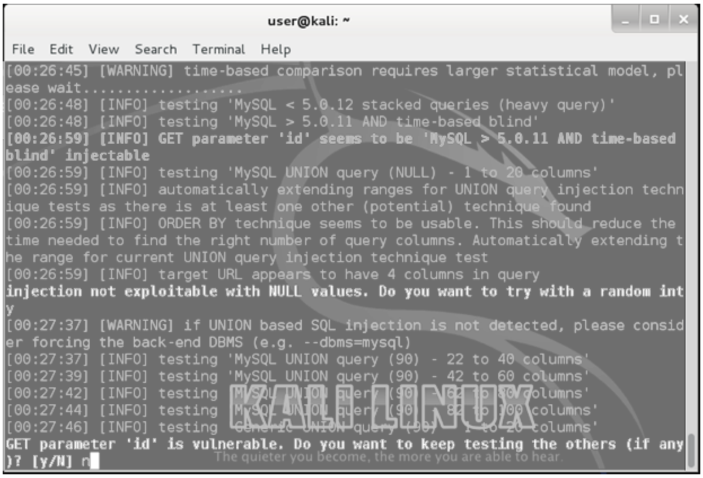
– the possibility to specify the level of «accuracy» of the check – the higher the level, the more tests the utility spends and the more time it takes to check it. The tool has five levels. GET- and POST-variables are always tested, COOKIE – from the second level, User-Agent/Referer – from the third level;

– the possibility to specify the level of detail to describe ongoing actions;

– the possibility to use all popular DBMSs (MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, SQLite, Firebird, Sybase, SAP MaxDB, DB2).

– installing a reliable TCP-connection (so-called out-of band) between the penetration tester and host that is running the database server. As the wrapper for this channel can be an interactive command line (shell), a Meterpreter session or access to a remote desktop via a VNC connection.

The program has a console interface with which is a pleasure to work due to the use of different colors. A fragment of the program is shown in Figure 48.4.



```
[00:26:45] [WARNING] time-based comparison requires larger statistical model, pl
ease wait..................
[00:26:48] [INFO] testing 'MySQL < 5.0.12 stacked queries (heavy query)'
[00:26:48] [INFO] testing 'MySQL > 5.0.11 AND time-based blind'
[00:26:59] [INFO] GET parameter 'id' seems to be 'MySQL > 5.0.11 AND time-based
blind' injectable
[00:26:59] [INFO] testing 'MySQL UNION query (NULL) - 1 to 20 columns'
[00:26:59] [INFO] automatically extending ranges for UNION query injection techn
ique tests as there is at least one other (potential) technique found
[00:26:59] [INFO] ORDER BY technique seems to be usable. This should reduce the
time needed to find the right number of query columns. Automatically extending t
he range for current UNION query injection technique test
[00:26:59] [INFO] target URL appears to have 4 columns in query
injection not exploitable with NULL values. Do you want to try with a random int
y
[00:27:37] [WARNING] if UNION based SQL injection is not detected, please consid
er forcing the back-end DBMS (e.g. --dbms=mysql)
[00:27:37] [INFO] testing 'MySQL UNION query (90) - 22 to 40 columns'
[00:27:39] [INFO] testing 'MySQL UNION query (90) - 42 to 60 columns'
[00:27:42] [INFO] testing 'MySQL UNION query (90) - 62 to 80 columns'
[00:27:44] [INFO] testing 'MySQL UNION query (90) - 82 to 100 columns'
[00:27:46] [INFO] testing 'Generic UNION query (90) - 1 to 20 columns'
GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any
)? [y/N] n
```

Fig. 48.4. Sqlmap tool in operation

Burp Suite [62] is an integrated platform for performing tests on the security of web applications. Its various tools work together effectively to support the entire testing process, from mapping the site to finding and exploiting security vulnerabilities.

Burp gives full control, allows you to combine advanced manual techniques with art-driven automatism, this makes the tester's work faster, more efficient and more enjoyable.

Burp Suite contains the following key components:

– the Burp Proxy tool that allows to inspect and modify the traffic between a browser and the target application;

– the Burp Spider tools for crawling content and functionality;

– the Burp Scanner tool for performing automated vulnerability scans of web applications;

– the Burp Intruder tool for performing automated attacks on web applications;

– the Burp Repeater tool for manipulating and resending individual requests;

– the Burp Sequencer tool for analyzing the quality of randomness in a sample of data items (session tokens);

– the possibility to save work and resume work later;

– extensibility, allowing to create own plug-ins, to perform complex tasks inside Burp.

Burp is simple and intuitive, allows beginners to get started right away. Burp is also very flexible in configuration; it contains a number of powerful functions to help the most experienced penetration testers in their work. The interface of the program is shown in Figure 48.5.

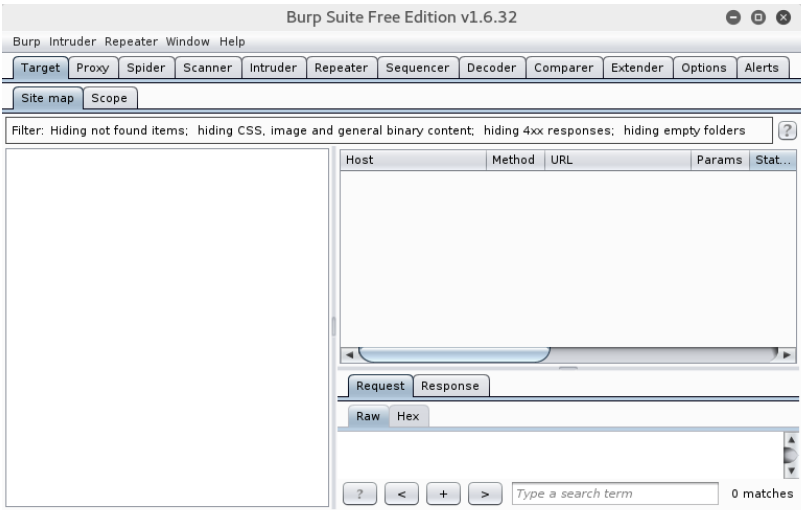

Burp Suite Free Edition v1.6.32
Burp Intruder Repeater Window Help
Target Proxy Spider Scanner Intruder Repeater Sequencer Decoder Comparer Extender Options Alerts
Site map Scope
Filter: Hiding not found items; hiding CSS, image and general binary content; hiding 4xx responses; hiding empty folders
Host Method URL Params Stat...
Request Response
Raw Hex
? < + >
Type a search term
0 matches

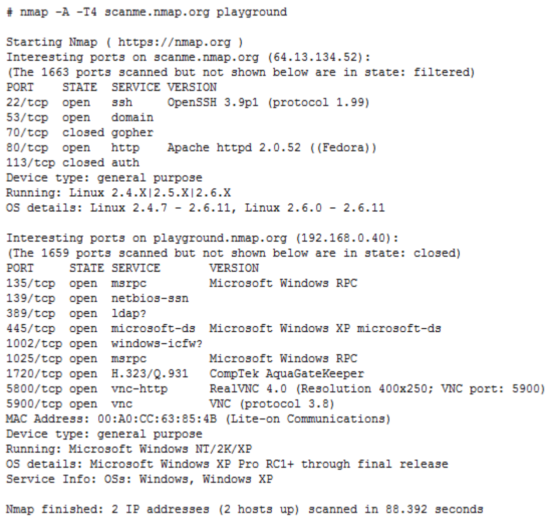
Fig. 48.5. Burp Suite interface

Burp provides ample opportunities for a MitM attack [63]. With the help of third-party applications, it is necessary to start the ARP spoofing process. Its essence lies in the fact that the victim's computer instead of sending traffic through a router starts to think that it is necessary to send traffic to the attacker now. The attacker's machine redirects traffic to the Internet, and sends replies from the Internet to the victim, i.e. the attacker's computer acts as a router.

In the process of testing, it is necessary to define the working hosts on a subnet and services running on them. Nmap [64] is a tool designed for a variety of customized IP-based network scanning with any number of objects, determining the status of the ports and their corresponding services. Nmap can scan by various methods, for example, UDP, TCP connect(), TCP SYN (half-open), FTP proxy (breakthrough via FTP), Reverse-ident, ICMP (ping), FIN, ACK, SYN and NULL-scanning. The choice of scanning option depends on the specified keys [65].

While *Nmap* is usually used for security checks, many systems and network administrators as network inventory, managing service upgrade schedules, and monitoring host or service uptime.

Nmap output data is a list of scanned targets with additional information for each, depending on the given options. The key information is the «important ports table». This table contains the port number, protocol, service name and state. The state can have a value open, filtered, closed or unfiltered. Open means that an application on the target machine is listening for connections/packets on that port. Filtered means that a firewall, filter, or other network obstacle is blocking the port so that Nmap cannot tell whether it is open or closed. Closed ports have no application listening on them, though they could open up at any time. Ports are classified as unfiltered when they are responsive to Nmap's probes, but Nmap cannot determine whether they are open or closed. Nmap reports the state combinations open|filtered and closed|filtered when it cannot determine which of the two states describe a port. The port table may also include software version details when version detection has been requested. When an IP protocol scan is requested (-sO), Nmap provides information on supported IP protocols rather than listening ports. The results of the Nmap tool is depicted in Figure 48.6.



```
# nmap -A -T4 scanme.nmap.org playground

Starting Nmap ( https://nmap.org )
Interesting ports on scanme.nmap.org (64.13.134.52):
(The 1663 ports scanned but not shown below are in state: filtered)
PORT    STATE  SERVICE VERSION
22/tcp  open   ssh     OpenSSH 3.9p1 (protocol 1.99)
53/tcp  open   domain
70/tcp  closed gopher
80/tcp  open   http    Apache httpd 2.0.52 ((Fedora))
113/tcp closed auth
Device type: general purpose
Running: Linux 2.4.X|2.5.X|2.6.X
OS details: Linux 2.4.7 - 2.6.11, Linux 2.6.0 - 2.6.11

Interesting ports on playground.nmap.org (192.168.0.40):
(The 1659 ports scanned but not shown below are in state: closed)
PORT     STATE SERVICE       VERSION
135/tcp  open  msrpc         Microsoft Windows RPC
139/tcp  open  netbios-ssn
389/tcp  open  ldap?
445/tcp  open  microsoft-ds  Microsoft Windows XP microsoft-ds
1002/tcp open  windows-icfw?
1025/tcp open  msrpc         Microsoft Windows RPC
1720/tcp open  H.323/Q.931   CompTek AquaGateKeeper
5800/tcp open  vnc-http      RealVNC 4.0 (Resolution 400x250; VNC port: 5900)
5900/tcp open  vnc           VNC (protocol 3.8)
MAC Address: 00:A0:CC:63:85:4B (Lite-on Communications)
Device type: general purpose
Running: Microsoft Windows NT/2K/XP
OS details: Microsoft Windows XP Pro RC1+ through final release
Service Info: OSs: Windows, Windows XP

Nmap finished: 2 IP addresses (2 hosts up) scanned in 88.392 seconds
```

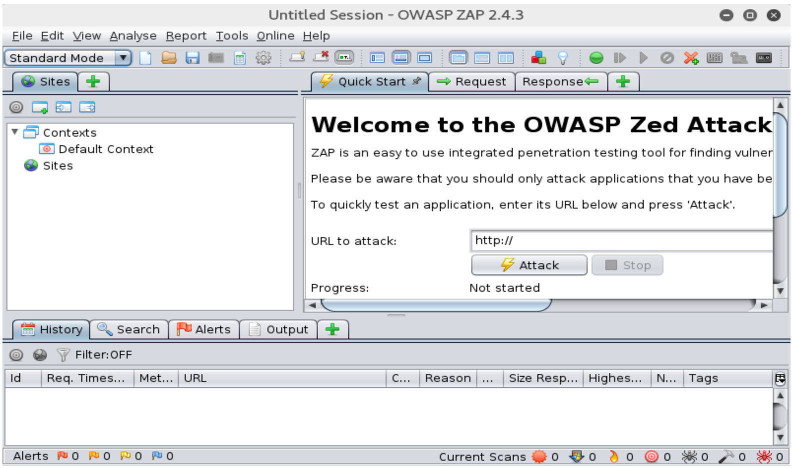
Fig. 48.6. The result of the Nmap tool

In addition to the interesting ports table, Nmap can provide further information on targets, including reverse DNS names, operating system guesses, device types, and MAC addresses. A detailed description of the functionality of this tool can be found in the source [66].

The following tool is OWASP Zed Attack Proxy [67] which can be known as OWASP-ZAP or zaproxy. This is an easy-to-use integrated tool for penetration testing and finding vulnerabilities in web applications. It is designed for use by people with different security experiences and is, therefore, ideal for developers and functional testers who are new to penetration testing. However, this program will not be useless for experienced penetration tester, it will find its place in their toolbox. Some of the features of ZAP are:

– open-source code;
– cross-platform software;
– easy to install (Java 1.7 required);
– completely free (no fee for Pro version);
– priority is ease of use;
– comprehensive information;
– translated into dozens of languages;
– based on the community involving the active promotion;
– actively developing an international volunteer team.

The main advantage of this tool is that for getting started is enough to enter the site address and click «Attack». The tool will start to follow links from the main page, thus creating a site map. Special values will be substituted in the found forms aimed at identifying the possibilities of conducting SQL-injections and XSS-attack. When the vulnerabilities appear, the numbers next to the checkboxes will appear: red means extremely serious vulnerabilities. The affected pages will also be marked in the page tree of the site. The tool interface is shown in Figure 48.7.


Untitled Session - OWASP ZAP 2.4.3

Welcome to the OWASP Zed Attack

ZAP is an easy to use integrated penetration testing tool for finding vulner

Please be aware that you should only attack applications that you have be

To quickly test an application, enter its URL below and press 'Attack'.

URL to attack: http://

Progress: Not started

Fig. 48.7. OWASP Zed Attack Proxy interface

To view all detected vulnerabilities and security concerns, go to the tab "Alerts".

For searching for objects that are not referenced from the site, the DIRB tool is used [68]. This tool is a Web-content scanner, it looks for existing (and hidden) objects. Usually, the attack is based on the object name dictionary, the response from the server is analyzed to determine the scan result.

Metasploit [69, 70] is a powerful open source framework with a thoughtful architecture, hundreds of contributors, which includes thousands of modules to automate the exploitation of a huge number of vulnerabilities. The structure of the Metasploit framework is shown in Figure 48.8.

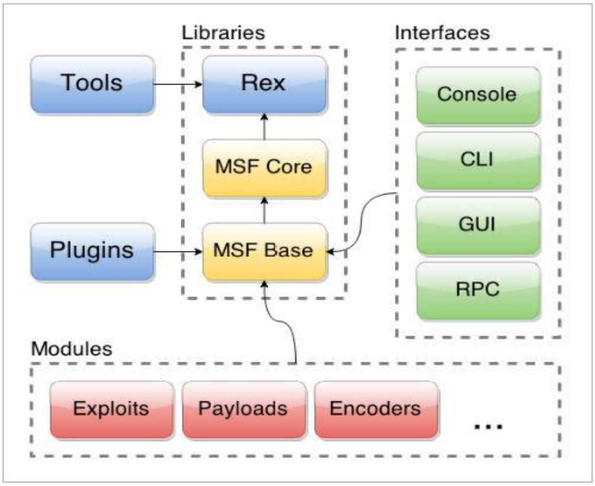


Fig. 48.8. The Metasploit structure

The main Metasploit component is a library RexIt is required for general operations: Handles sockets, protocols, text transformations, and others. The MSF Core library is based on it, which provides the “basic’” API. It is used by the MSF Base library, which, in turn, provides an API for plug-ins, a user interface (both console and graphical), and modules.

The modules need to be addressed more. They are divided into several types, depending on the provided functionality:

– *Exploit* is a code exploiting a particular vulnerability on the target system (for example, buffer overflow);

– *Payload* is a code that runs on the target system after the exploit has worked (establishes a connection, performs a shell script, etc.);

– *Post* is a code that runs on the system after successful penetration (for example, collects passwords, downloads files);

– *Encoder* is tool for obfuscation modules for masking of the antivirus and firewall;

– *NOP* is an assembler instruction that does not perform any actions. It is used to fill the void in executable files to fit the desired size.

– *Auxiliary* is modules for network scanning, traffic analysis and so on.

Currently, Metasploit is distributed in four versions:

– *Framework* – a basic version with console interface;

– *Community* – free version which includes an additional web interface and part of the functionality from commercial versions;

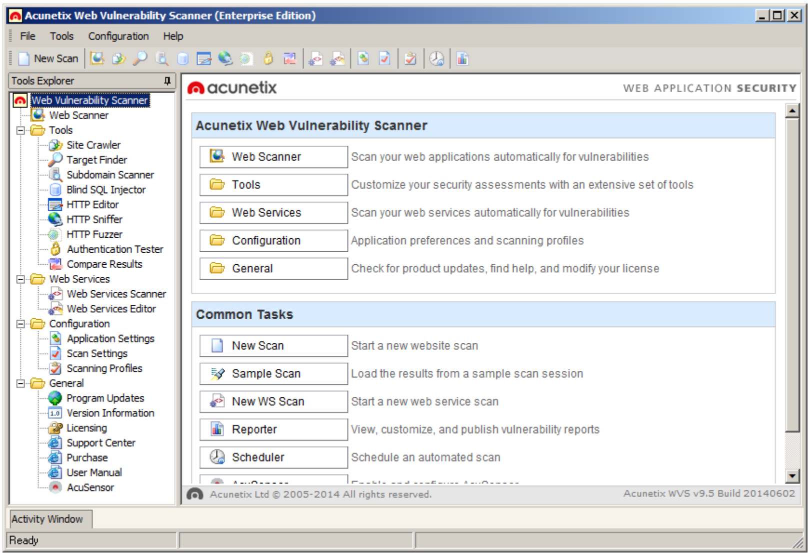
– *Express* – a version for commercial users which includes functionality that makes it easier to conduct basic audits and generate reports on them;

– *Pro* – the most advanced version, which provides advanced opportunities for attacks, allows to create a chain of tasks for auditing, draw up detailed reporting and much more.

In addition to the web interface, accessible in the Community, Pro and Express versions there are projects such as Armitage and Cobalt strike with GUI-interface for the framework.

The advantage of this tool is the availability of a large database of exploits and the ability to write your own exploits in the Ruby language.

The next powerful scanner is Acunetix. Acunetix Web Vulnerability Scanner [71] is an application for automated security control tasks of Web applications. It allows identifying vulnerabilities in the protection of a Web site before they are discovered and used by an attacker. The scanner interface is depicted in Figure 48.9.



Acunetix Web Vulnerability Scanner

Web Scanner	Scan your web applications automatically for vulnerabilities
Tools	Customize your security assessments with an extensive set of tools
Web Services	Scan your web services automatically for vulnerabilities
Configuration	Application preferences and scanning profiles
General	Check for product updates, find help, and modify your license

Common Tasks

New Scan	Start a new website scan
Sample Scan	Load the results from a sample scan session
New WS Scan	Start a new web service scan
Reporter	View, customize, and publish vulnerability reports
Scheduler	Schedule an automated scan

Acunetix Ltd © 2005-2014 All rights reserved.   Acunetix WVS v9.5 Build 20140602

Fig. 48.9. Acunetix WVS program interface

Acunetix Web Vulnerability Scanner is a vulnerabilities scanner that allows to for errors in scripts in PHP, ASP and ASP.NET languages. In addition, the program can check for errors in JavaScript scripts. Site tests can be carried out not only directly, but also via HTTP or SOCKS proxy server to ensure anonymity.

This software product is created on the principle of "all in one". It allows detecting weaknesses of protection presented in the ratings of SANS Top 20 and OWASP Top 10.

Acunetix Web Vulnerability Scanner also performs HTTP authentication testing to determine insecure settings and unreliable passwords and to verify server security settings.

Acunetix Web Vulnerability Scanner automatically performs the following checks:

– Cross-site scripting (XSS) – an intrusion of unauthorized scripts into the page and their execution by the browser;

– SQL injection - execution of SQL-queries from the browser to obtain unauthorized access to data;

– site analysis according to the database GHDB (Google hacking database) – a list of typical requests used by hackers to gain unauthorized access to a web application and sites;

– scanning of AJAX and Web 2.0 for vulnerabilities;

– port scanning and services detection;

– analysis and construction of the site structure;

– code executing;

– traversal directory;

– file inclusion;

– disclosure of the source text of the script;

– CRLF injection;

– cross frame scripting;

– detection of public backups of files and folders;

– detection of files and folders containing important information;

– detection of folders with a low level of protection, allowing you to create, modify or delete files.

The scanner identifies hosts that are running the server service and scans the host’s data for the presence of common "weak" places, such as the XSS, directory traversal, SQL input and many others. The program also tests HTTP authentication to determine insecure settings, unreliable passwords, and to verify server security settings. It informs about the presence of obsolete software on the server, which needs to be updated or corrected.

With it, you can scan virtual hosts and all types of Web servers, not just Windows-based ones WVS supports UNIX and Linux operating systems, Apache web services and PHP configuration files. It is possible to expose the OS specific checks.

You can scan many types of web pages and files, including ASP and CGI. The program supports secure proxy and HTTP (HTTPS) servers.

Settings can be set via the Configuration tab in the left pane Select Settings to change the following options:

– General – for changing the address of the updates (www.acunetix.com/wvsupdate/) and selecting the option when WVS will check for updates (when the application starts or when select "Check for updates" in the Help menu). This page sets general HTTP settings, such as file size limitation in kilobytes and the HTTP request timeout in seconds.

– Lan Settings – for setting the options for using the HTTP proxy server and/or SOCKS proxy, as well as the authentication and credentials for both servers;

– Database – for enabling or disabling the use of databases, and changing the type of database where the scan results will be stored;

– Tools Settings: Site crawler – crawls through a target website and builds the site layout using the information collected, including the site directories and files/objects. You can also use the site crawler tool to analyze the structure of a website without automatically launching the attack;

– Tools Settings: Scanner – for determining which tests are to be launched against the target website. For example, if you only want to test your website(s) for SQL injection, select the profile sql_injection. No additional tests will be performed. The scan mode will determine how both the crawler and the scanner will treat website parameters (also known as inputs), which will affect the number of security checks launched against the website;

– Tools Settings: HTTP Sniffer – can be used to manually crawl sections of your website that cannot be crawled automatically by Acunetix WVS. In these cases, you browse the sections of the website that can not be crawled automatically with a web browser, capture the HTTP traffic, and then load the crawl into the scanner and launch a security scan against it.

Default settings are:

– search is prohibited above the start folder;

– search for files below the main folder;

– search for indexes of directories, even if they are not linked by references.

In addition, the following options are possible:

– launch the HTTP analyzer for manual searching at the end of the scanning process;

– receive only the first URL address;

– log in with a specific user name and password.

If the search agent could not track all the links automatically, you should perform a manual search. After that, the agent includes the found links in the site structure. If you do not want the search agent to go through the links outside the site, then you should choose the option of getting only the first URL. To scan sites that require authentication, you should enter a user name and password.

After setting the required parameters, click the Next button or check or uncheck the option to save the scan results to the database. Then click the Finish button to start scanning.

The Scan Results consist of the following tabs:

– Alerts - Expanding on this tab displays a list of weaknesses of protection that were found during the scan: insecure settings, the passage of time or outdated version of the software. Alerts can only be an informational, low, medium and high risk. Select an item on the Alerts tab to view the details in the right pane. It also presents tips to deal with the weaknesses of protection and a list of references for more information;

– Site Structure – This tab provides the directories and files that the search agent detected, including those found during a manual search, as well as their structure in the file system. For more information (path, size in bytes, HTTP Get results, etc.), select the file or directory.

The Activity Window is at the bottom of the program interface. It depicts all actions performed by WVS. The Scan Results window shows the number of all found security weaknesses.

To compare the results of this and the previous scan, select the Compare Results item in the Tools menu.

If you enabled the database support during the installation, it is possible to create a report from the SQL or Access database files. Otherwise, it can be done as follows: expand the Configuration tab and select Settings | Database. In the small middle pane under Application options, select Database and check the box to enable database support, select the database type from the drop-down list and enter the path for the Access database or server information for the SQL database.

The Reporter tool is used to create a report using information from the database. Select New | Report at the WVS command prompt The Reporting Wizard starts. First, it prompts you to select scan options that have been saved to the database from which the report is generated. Click the Next button. Further, it is possible to select the necessary information for inclusion in the report. The report can include only certain types of alerts (be the informational, low, medium and high risk) And select whether to display detailed warnings, request/response headers and/or alert table. You can specify the name of the report, the header or text of the footer, and also select an image for the title.

The report consists of the following sections:

– Scan groups;

– Scan summary;

– Alert summary;

– Alerts details.

The scan summary provides information about the start and ends time of the scan, scan time and general information about the server (OS and version of Web services).

The report can be saved to a file or printed. It is possible to create report templates that include test options to speed up and facilitate the process of creating reports.

Acunetix WVS includes some additional tools:

– HTTP Editor that is used to create a custom variant of HTTP requests and responses scrutiny, as well as to determine the output/response format (for example, plain text or HTML). The interface also includes an encryption-decryption tool for converting between plain text and URL or Base64 codes, as well as a variable editor that allows you to edit query variables, cookies and request data.

– Target Finder explores the list or range of IP addresses for discovering HTTP / HTTPS servers.

– Authentication Tester is used to test and restore the valid passwords of the target sites, as well as to crack passwords using brute force methods.

These can be accessed through the Tools menu in the WVS command line. There is also a Troubleshooter Wizard in the sub menu Acunetix of the Programs menu.

Acunetix Web Vulnerability Scanner is easy to install, setup and use. It takes only a few minutes to get first results after installation. Such simplicity, however, does not completely reduce the severity of the program. There are many ways to customize and fit for a particular user, and additional tools provide more options. The package includes a user manual in HTML Help format, but also available in PDF and Web versions on the website [71].

Some tools, without which the work of a penetration tester would be impossible were listed above. Do not forget that tools are just an instrument, a tester must be smart enough to find possible vulnerabilities and ways to exploit them.

Conclusions

To assist information security specialists in their skills assessment, to assist web developers in better understanding of the mechanism of writing the secure code, to enable students and lecturers learn more about the security of web applications in a controlled environment as well as to test different tools without having legal problems the special vulnerable web application Damn Vulnerable Web Application has been created. DVWA provides an opportunity to practice in the most popular web vulnerabilities at different levels of complexity.

The variety of tools for penetration testing makes the choice of toolset for a particular task complex and ambiguous. And so, an idea for developing service that will help to solve the problem of choice of tools for penetration testing using criteria matrix is studied.

The pecularities of sqlmap, Burp suite, Nmap, Acunetix tools regarding the main attack vectors as well as the Metasploit framework which includes thousands of modules to automate the exploitation of a huge number of vulnerabilities are described.

Questions and tasks for self-control

1. Which platforms with vulnerabilities do you know?
2. What for the platforms with known vulnerabilities are used?
3. Why such platforms are not open-shared?
4. What is the advantage of open source platforms?
5. Why is it easier to create a vulnerable application in PHP than in other languages?
6. What is the difficulty in choosing tools?
7. What are the advantages of using neural networks choosing suitable tools?
8. What is the disadvantage of systems using neural networks?
9. What is the advantage of specific tools?
10. What tools can be used to scan networks?
11. What can lead to high-speed tools?
12. What tools can create a sitemap?
13. What is the advantage OWASP Zed Attack Proxy tool?
14. How can the result of one tool be used by another tool? Give an example.
15. List the features of the sqlmap tool.
16. List the advantages and disadvantages of using a proxy for scanning.
17. What is Metasploit?
18. What is the advantage of Metasploit?
19. What is Exploit?
20. What is a Payload in Metasploit?

References

1. Netkachov O., Popov P., Salako K. (2014) Quantification of the Impact of Cyber Attack in Critical Infrastructures. In: Bondavalli A., Ceccarelli A., Ortmeier F. (eds) Computer Safety, Reliability, and Security. SAFECOMP 2014. Lecture Notes in Computer Science, vol 8696. Springer, Cham.
http://openaccess.city.ac.uk/4363/1/Quantification%20of%20the%20Impact%20of%20cyber%20attack.pdf

2. Khani S., Gacek C., Popov P. (2015) Security-aware selection of Web Services for Reliable Composition. In: arXiv preprint arXiv:1510.02391. https://arxiv.org/ftp/arxiv/papers/1510/1510.02391.pdf

3. National Institute of Standards and Technology Special Publication 800-137, 80 pages, 2011.

4. National Institute of Standards and Technology Special Publication 800-95, 128 pages, 2007.

5. The Transport Layer Security (TLS) Protocol Version 1.2 – https://tools.ietf.org/html/rfc5246

6. IP Security (IPsec) and Internet Key Exchange (IKE) Document Roadmap – https://tools.ietf.org/html/rfc6071

7. Open wep application security testing guide V4. https://www.owasp.org/images/1/19/OTGv4.pdf

8. National Institute of Standards and Technology special publication 800-115. Technical Guide to Information Security Testing and Assessment. http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-115.pdf

9. OWASP Testing Guide – www.owasp.org/index.php/OWASP_Testing_Project

10. NIST Special Publication 800-115: Technical Guide to Information Security Testing and Assessment (NIST SP 800-115) – http://csrc.nist.gov/publications/nistpubs/800-115/SP800-115.pdf

11. Penetration Testing Execution Standard (PTES) – www.pentest-standard.org/

12. Open Source Security Testing Methodology Manual (OSSTMM) – www.isecom.org/research/osstmm.html

13. A technical community of Symantec customers, end-users, developers, and partners – www.securityfocus.com/archive/1

14. Bugtraq Mailing List – http://seclists.org/bugtraq

15. The ultimate security vulnerability datasource – www.cvedetails.com

16. Open Source Vulnerabilities Data Base (OSVDB) – http://osvdb.org

17. Secunia Research Community – http://secunia.com/communityadvisories/historic

18. Bugs Collector. Hack The World – https://bugscollector.com

19. Open bug bounty – https://www.xssposed.org

20. 18 Microsoft Security Bulletins – https://technet.microsoft.com/security/bulletin

21. Introduction to Secure Software Development Life Cycle – http://resources.infosecinstitute.com/intro-secure-software-development-life-cycle/#gref

21. Application security testing developers actually use – https://www.checkmarx.com/

22. Event Tree Analysis. – http://www.eventtreeanalysis.com/.

23. Andrews JD, Dunnett SJ. Event-tree analysis using binary decision diagrams. IEEE Transactions on Reliability, Volume: 49 , Issue: 2 , Jun 2000, pp. 230-238.

24. Twigg DW, Ramesh AV, Sharma TC. Modeling event dependencies using disjoint sets in fault trees. Proceedings of the 18th International System Safety Conference 2000, pp. 275–279.

25. Amari S, Dill G, Howald E. A new approach to solve dynamic fault trees. In: Annual IEEE reliability and maintainability symposium. Institute of Electrical and Electronics Engineers, New York, 2003, pp 374–379.

26. Vesely B. Mission Success Starts With Safety Fault Tree Analysis (FTA): Concepts and Applications – www.hq.nasa.gov/office/codeq/risk/docs/ftacourse.pdf

27. D.M. Dunlavy, B. Hendrickson, T.G. Kolda. Mathematical Challenges in Cybersecurit. Sandia Report, 2009, p. 10.

28. K.S. Trivedi. Probability and Statistics with Reliability, Queuing, and Computer Science Applications. 2nd ed., Wiley, New York, 2001, p. 1001.

29. S. Abraham, S. Nair. Cyber Security Analytics: A Stochastic Model for Security Quantification using Absorbing Markov Chains. In: Journal of Communications, vol. 9, No. 12 (December), 2014, pp.899-907.

30. J. Almasizadeh, M. A. Azgomi. A stochastic model of attack process for the evaluation of security metrics. In: Computer Networks: The International Journal of Computer and Telecommunications Networking archive, vol. 57, issue 10, 2013, pp. 2159-2180, doi 10.1016/j.comnet.2013.03.011.

31. Ye, Nong, et al. Probabilistic techniques for intrusion detection based on computer audit data. In: Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions, Vol.31, No.4, 2001, pp.266-274.

32. B.B. Madan, K.G. Popstojanova, K. Vaidyanathan, K.S. Trivedi. A method for modeling and quantifying the security attributes of intrusion tolerant systems. In: Dependable Systems and Networks-Performance and Dependability Symposium, 2004, pp. 167–186.

33. A. Sperotto, R. Sadre, , P. de Boer, A. Pras. Hidden markov model modeling of SSH brute-force attacks. In: Integrated Management of Systems, Services, Processes and People in IT, pp. 164–176.

34. Pentest standard revision and final version - http://www.pentest-standard.org/index.php?title=FAQ&diff=958&oldid=8

35. Technical Report CMU/SEI-96-TR-022 ESC-TR-96-022 November 1996. - http://leansoftwareengineering.com/wp-content/uploads/2009/02/cleanroomsei.pdf

36. The Penetration Testing Execution Standard. - http://www.pentest-standard.org/index.php/Main_Page

37. Ментальная карта PTES. - https://www.mindmeister.com/ru/70567774/penetration-testing-execution-standard

38. Chris Nickerson, Ian Amit, Wim Remes, Stefan Friedli. Fixing the Industry, one Panel at a Time. - http://www.slideshare.net/SOURCEConference/ptes-pentest-execution-standard

39. V. N. Gavva, V. A. Kolisnichenko, D. D. Uzun – Statistical practicum. – Kharkiv, National Aerospace University “KhAI”, 2000. – 147 p.

40. Software-as-a-Service; A Comprehensive Look at the Total Cost of Ownership of Software Applications // SIIA, Prepared by the Software-as-a-Service Executive Council, September 2006. - http://www.winnou.com/saas.pdf

41. David Kennedy, Jim O'Gorman, Devon Kearns, and Mati Aharoni. Metasploit. – 2011. – 328 p.

42. Pre-Engagement. - http://www.pentest-standard.org/index.php/Pre-engagement

43. National Vulnerability Database. - https://nvd.nist.gov

44. Web Application Firewall. - https://www.owasp.org/index.php/Web_Application_Firewall

45. Owasp Top 10: The Top 10 Most Critical Web Application Security Threats: Enhanced with Text Analytics and Content by Pagekicker Robot Phil 73 // Createspace. – 2014. – 54 p.

46. Injections.- https://www.owasp.org/index.php/Top_10_2013-A1-Injection

47. Broken Authentication and Session Management.- https://www.owasp.org/index.php/Top_10_2013-A2-Broken_Authentication_and_Session_Management

48. Cross Site Scripting. - https://www.owasp.org/index.php/Top_10_2013-A3-Cross-Site_Scripting_(XSS)

49. Insecure Direct Object References. - https://www.owasp.org/index.php/Top_10_2013-A4-Insecure_Direct_Object_References

50. Security Misconfiguration. - https://www.owasp.org/index.php/Top_10_2013-A5-

51. Security misconfiguration sensitive data exposure. - https://www.owasp.org/index.php/Top_10_2013-A6-Sensitive_Data_Exposure

52. Missing function level access control. - https://www.owasp.org/index.php/Top_10_2013-A7-Missing_Function_Level_Access_Control

53. Cross-Site Request Forgery, CSRF/XSRF. - https://www.owasp.org/index.php/Top_10_2013-A8-Cross-Site_Request_Forgery_(CSRF)

54. Using components with Kkown vulnerabilities. - https://www.owasp.org/index.php/Top_10_2013-A9-Using_Components_with_Known_Vulnerabilities

55. Unvalidated Redirects and Forwards. - https://www.owasp.org/index.php/Top_10_2013-A10-Unvalidated_Redirects_and_Forwards

56. Damn Vulnerable Web Application (DVWA). – http://www.dvwa.co.uk/

57. Damn Vulnerable Linux. – https://distrowatch.com/dvl

58. OWASP WebGoat Project. – https://www.owasp.org/index.php/Category:OWASP_WebGoat_Project

59. L.N. Yasnitskiy – Introduction to artificial intelligence – Moscow, «Academy», 2010 – 176 p.

60. Justine Clarke. SQL Injection Attacks and Defense. / Syngress Publishing, Inc., 2009. – 576 p.

61. Sqlmap. – http://sqlmap.org

62. Burp Suite Editions. – https://portswigger.net/burp/

63. Willie L. Pritchett , David De Smet. Kali Linux Cookbook / Packt Publishing, 2013. – 260p.

64. Nmap: the Network Mapper - Free Security Scanner. – https://nmap.org/

65. Nmap Reference Guide. – https://nmap.org/book/man.html

66. G. F. Lyon. Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning / Nmap Project, 2009. – 464 p.

67. OWASP Zed Attack Proxy Project. – https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project

68. DIRB web content scanner – http://dirb.sourceforge.net/

69. David Kennedy, Jim O'Gorman, Devon Kearns, and Mati Aharoni. Metasploit. – 2011. – 328 p.

70. Penetration Testing Software | Metasploit. – https://www.metasploit.com/

71. Acunetix Website. - http://www.acunetix.com/

49 MARKOV’S MODEL-BASED TECHNIQUE OF IOT SYSTEM AVAILABILITY CONSIDERING DDOS ATTACKS

1.1 Survey of DDoS-attacks on IoT systems classification

Internet of things (IoT) is a paradigm that involves ubiquitous presence in the environment of different things / objects that are using wireless and wire networks and unique addressing scheme are able to interact with each other and with other things / objects to create new applications / services to achieve purposes [1, 2]. This is wired and wireless sensor system, which transmit information from one device to another (M2M solutions, applications to process data from sensors, mobile electronic devices, and cloud infrastructure). IoT extends the scope of the Internet by people working at the computer in the direction of the autonomous intelligent smart devices connected to the Internet for remote monitoring and diagnosis [3-5]. According to [1], new concepts of IoT must to reduce complexity through pre-integrated modules for data acquisition, validation, and analysis; reduce risk due to compliance with the one M2M standard; must have faster time-to-value through as-a-Service (AAS) hosted models; lower total cost of ownership (TCO) by reducing capital expenses, providing scalability. Modern IoT technologies have been created under a totally different scenario [6,7]: 1) Internet and cellular networks have become the world standard, with very high levels of coverage, reliability and availability; 2) smaller and smarter devices are constantly hit the industrial and consumer markets, to better understand and present the new after-sales and remote-controlled services; 3) software development and system interoperability standards such as XML, web services and SOA are converging to create fertile ground for M2M communications technologies, that makes it easier to use them in a variety of industries.

The development of IoT is accompanied with the receipt, storage, processing and distribution of large amounts of data.

The increasing complexity and importance of IoT has led to an increase in the number of DoS- (Denial of Service), DDoS- (Distributed DoS) attacks on IoT system, the use of IoT devices for malicious purposes. All of these unauthorized influences lead to failures and failures of the critical systems that are part of the IoT. Therefore, studies aimed at improving the dependability of the system in the context of successful DoS and DDoS attacks are topical. The article examines the statistics of DoS and DDoS attacks in the world, developed the Markov model of IoT research, taking into account the impact of attacks on the server, router and IoT power supply system. We investigate the effect of successful attacks on the availability factor of the IoT system.

A DoS attack is defined as an unambiguous effort by a horrible user to get through the capital of a server or a network, thereby preventing rightful users from availing the services provided by the system [6-7].

A DDoS is defined as an attack in which multiple compromised systems are made to attack a single target to make the services unavailable for legitimate users.

A Distributed Reflective Denial of Service (DRDoS) attack is a form of Distributed Denial of Service (DDoS) that relies on the use of publicly accessible UDP servers, as well as bandwidth amplification factors, to overwhelm a victim system with UDP traffic.

The mechanism of DDoS attack on IoT systems is shown in fig. 49.1.

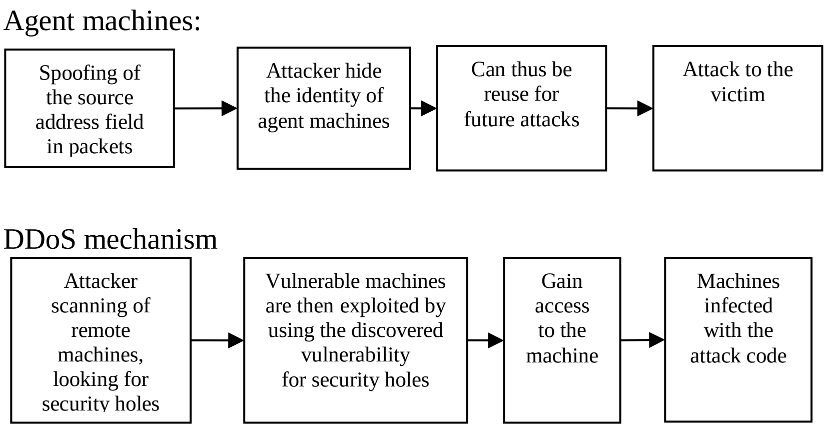

Agent machines:
Spoofing of the source address field in packets
Attacker hide the identity of agent machines
Can thus be reuse for future attacks
Attack to the victim
DDoS mechanism
Attacker scanning of remote machines, looking for security holes
Vulnerable machines are then exploited by using the discovered vulnerability for security holes
Gain access to the machine
Machines infected with the attack code

Fig.49.1 -The mechanism of DDoS attack on IoT systems

Classification of DDoS-attacks on IoT systems is shown in the tables 49.1 – 49.7.

Table 49.1 - Classification of DDoS-attacks on IoT systems by degree of automation

Manual attacks	Attacker scanning of remote machines for vulnerabilities, broke into them, installed the attack code, commanded the onset of the attack
Semi-Automatic attacks	DDoS consists of handler (master) and agent (slave) machines. Attacker deploys automated scripts for scanning and compromise of those machines and installation of the attack code. Handler machines to specify the attack type and the victim's address and to command the onset of the attack to agents, who send packets to the victim.
a) Attacks with direct communication	Agent and the handler machines need to know each other's identity in order to communicate. This is achieved by hard-coding the IP-address of the handler machines in the attack code that is later installed on the agent. Each agent then reports its readiness to the handlers, who store its IP address in a file for later communication.
b) Attacks with indirect communication	Deploy the level of indirection to increase the survivability of a DDoS network.
c) Web-based attacks	The exploit kits are developed that can install the on the remote machines and then control them from software a remote web-site.
d) P2P-based Attacks	Communication botnets are distributed and don't have central point of failure.
d1) Rational attack	In the P2P system node A wants to distribute content. To decrease the upload bandwidth burden on the node A, only a small number of nodes such as node B and node F are directly connected to it. The content is then propagated from node B and node F to additional peers such as node C, D and E. Because of the self-interested behavior in most P2P systems, a self-interested node may realize that it can save expensive upload bandwidth if it chooses not to share. If a large number of nodes are

	self-interested and refuse to contribute, the system may destabilize. In this case, if enough nodes such as B and F become self-interested, the system cannot guarantee a reasonable level of uploads and downloads.
d2) Index Poisoning Attack	Aims at the index querying process of users and makes it hard to find correct content in P2P network. The attackers simply insert large numbers of invalid peer information into the index to hinder the users from finding correct resource.
d3) Sybil attack	If an entity acts as a number of multiple identities, this entity can control a significant part of networks. Sybil attack will destroy the redundancy in P2P network.
Automatic attacks	Duration and victim's address is preprogrammed in the attack code. Using a single command – the start of the attack script.
a) Attacks with Random Scanning	Each compromised host probes random addresses in the IP address space, using a different seed. This potentially a high traffic volume since many machines probe the same addresses (Code Red).
b) Attacks with Hitlist Scanning	Probes all addresses from an externally supplied list. When it detected the vulnerable machine, it sends one half of the initial hitlist to the recipient and keeps the other half (used for a Smurf attack).
c) Attacks with Topological Scanning	Uses the information on the compromised host to select new target (E-mail worms).
d) Attacks to with Permutation Scanning	All compromised machines share a common pseudo-random permutation of the IP address space; each IP address is maped to an index in this permutation. A machine begins scanning by using an index computed from its IP addresses as a starting point. Whenever it sees an already infected machine, it chooses a new random start point.
e) Attacks with Local Subnet Scanning	A single copy of the scanning program can compromise many vulnerable machines behind a firewall (Nimbda Worm).
f) Attacks with Propagation mechanism	Based on the attack code. Worm can be propagated through file, email, web server, and so on.
f1) Attacks with Back-chaining	Is downloaded from the machine that was use to exploit the system. The infected machine then becomes a

Propagation	source for the next propagation step (Ramen worm, Morris worm).
f2) Attacks with Autonomous Propagation	Avoids the file retrieval step by injecting instructions directly into the target host during the exploitation phase (Code Red, Warhol Worm, numerous E-mail Worms).
f3) Attacks with Central Source Propagation	The attack code resides on a central server or set of servers. After compromise of the agent machine, the code is downloaded from the Central Source through a file transfer mechanism (li0n worm).

Table 49.2 - Classification of DDoS-attacks on IoT systems by impact and based on exploits

Disruptive Attacks	Completely deny the victim’s service to its clients.
a) Malformed Packet Attack	Attacker sends defective packets, such as overlapping IP fragments and packets with illegal TCP flags to a target system, so that the system crashes when it processes such packets. Attacker instructs the agents or bots to send a large volume of such packets. An application can to crash and also hide subsequent attacker’s activities.
Degrading Attacks	Consume some victim’s resources.
a) A Bandwidth Depletion Attack	Interrupts the normal working by causing congestion, exorbitant of traffic. Here Flooding basically carried out by zombies by sending large amount traffic to victim’s system in order to occupy the entire bandwidth.
b) A Resource Depletion Attack	Tie up the resources of a victim system making the victim unable to process legitimate requests for service. Attacker try to bind the victim’s resources by mainly focusing on server or on the particular process.
c) A Protocol Exploitation Attack	Attacker uses the existing bugs or some special features of protocols at the victim’s system in order to misuse the resources excessively.

Table 49.3 - Classification of DDoS-attacks on IoT systems by Attack Rate Dynamics

1) High Rate Attack	Disruption of Internet services by sending volume of packets at a point in time from distributed locations results.
2) Low Rate	Bots are used to send large volume of malicious packets

Attack	at low rate in a coordinated manner.
3) Continuous Rate Attack	After the onset is commanded, agent machines generate the attack packets with full force. This sudden packet flood disrupts the victim’s services quickly, and thus leads to attack detection.
4) Varied Rate Attack	Complex attack that uses attack tools to generate a mix of packets at high and low rates.
5) Variable Rate Attack	Vary the attack rate to avoid detection and response.
5a) Increasing Rate Attack	Have a gradually increasing rate lead to a slow exhaustion of victim’s resources. A state change of a victim could be so gradual that its services degrade slowly over a long time period, thus delaying detection of the attack.
5b) Fluctuating Rate Attack	Attack with fluctuating rate adjust the attack based on the victim’s behavior, occasionally relieved the effect to avoid detection.

Table 49.4 - Classification of DDoS-attacks on IoT systems by Spoofing techniques

Random Spoofing	Attackers can spoof random source addresses in attack packets since this can simply be achieved by generating random 32-bit numbers and stamping packets with them. Attackers may also choose more sophisticated spoofing techniques, such as subnet spoofing, to defeat some antispoofing firewalls and routers using ingress filtering (Ferguson *et al.* 1998) and route-based filtering.
Subnet Spoofing	Attacker spoofs a random address within the address space of the subnetwork. For example, a computer in a C-class network could spoof any address with the same prefix. It is impossible to detect this type of spoofing anywhere in the Internet. Subnet spoofing is useful for the attackers that compromise machines on networks running ingress filtering. A possible defense against this type of spoofing is to bind the IP address, the MAC address and the network port of each computer in the sub network.
Fixed Spoofing	The spoofed address is the address of the target. For example, an attacker performing a Smurf-attack spoofs the victim’s address so that ICMP ECHO packets will

	be reflected to the victim.
Client-Side Spoofing	A hacker impersonates himself as an authorized client and in turn gains service from an unintelligent server. A hand-on example is “r-utilities” under most of UNIX systems.
Server-Side Spoofing	A hacker impersonates himself as an authorized user and then gains data from some information provider. On the other hand, a server-side spoofing is used in a reverse way. In order to obtain private information form individual clients, a hacker masquerades himself as a real service provider and steals the trust from system users.
IP-Spoofing	Hackers created packets with spoofed source IP address, then exploited applications that use authentication based on IP address, like “r-utilities”

Table 49.5 - Classification of DDoS-attacks on IoT systems by Exploit techniques

1)Protocol Attacks	Exploit some of the design weaknesses of the TCP/IP protocol. They typically misuse the six control bits, or flags, of the TCP/IP protocol - SYN, ACK, RST, PSH, FIN, and URG - to disrupt the normal mechanisms of TCP traffic. Unlike UDP and other connectionless protocols, TCP/IP is connection-based, requiring the packet sender to establish a full connection with the intended recipient prior to sending any packets. TCP/IP relies on a three-way handshake mechanism where every request creates a half-open connection (SYN), a request for a reply (SYN-ACK), and then an acknowledgement of the reply (ACK). Attacks attempting to abuse the TCP/IP protocol often send TCP packets in the wrong order, causing the target server to run out of computing resources as it tries to understand such abnormal traffic.
a) TCP SYN Flood Attacks	The attacking clients lead the server to believe that they are asking for legitimate connections through a series of TCP requests with TCP flags set to SYN. To handle each of these SYN requests, the target server opens threads and allocates corresponding buffers to prepare for a connection. Because server resources are limited and a SYN flood often involves a massive number of connection requests, a server is unable to time out its open requests

	before new requests arrive.
b)TCP RST Attacks	An attacker interferes with an active TCP connection between two entities. The attacker sends packets with the RST Flag set to ON to host A, host B, or both. Since neither host knows that an attacker has sent these packets, they treat these packets normally, meaning that the valid TCP connection between the two hosts is terminated.
c) TCP PSH+ACK Flood Attacks	An attacker, usually using a botnet, can therefore flood a target server with many such requests, overwhelming the TCP stack buffer on the target server so it cannot process the requests or even acknowledge them - resulting in limited DDoS attack defense.
d)Sockstress Attacks	An attack tool that exploits vulnerabilities in the TCP stack, allowing an attacker to create a denial of service condition for a target server. In a normal TCP three-way handshake, a client sends a SYN packet to the server, the server responds with a SYN-ACK packet, and the client responds with an ACK, establishing a connection. Sockstress establishes a normal TCP connection with the target server but sends a "window size 0" packet to the server inside the last ACK. The TCP window is a buffer that stores received data before uploading it to the application layer. Window size set to zero means that the window size is 0 bytes. This setting tells the sender to stop sending more data until further notice.
e) Layer 7 (HTTP) Flood Attacks	Consists of seemingly legitimate session-based sets of HTTP GET or POST requests sent to a target Web server. These requests are specifically designed to consume a significant amount of the server’s resources, and therefore can result in a denial-of-service. HTTP flood is a common feature in most botnet software. To send an HTTP request, a valid TCP connection has to be established, which requires a genuine IP address. Attackers can achieve this by using a bot’s IP address, can craft the HTTP requests in different ways in order to either maximize the attack power or avoid detection. HTTP makes it difficult for network security devices to distinguish between legitimate HTTP traffic and malicious HTTP traffic, and could cause a high number of false-positive detections.

e1) HTTP Header	HTTP headers are fields which describe which resources are requested, such as URL, a form, JPEG, etc. HTTP headers also inform the web server what kind of web browser is being used. Common HTTP headers are GET, POST, ACCEPT, LANGUAGE, and USER AGENT. The requester can insert as many headers as they want and can make them communication specific. DDoS attackers can change these and many other HTTP headers to make it more difficult to identify the attack origin. In addition, HTTP headers can be designed to manipulate caching and proxy services [ieee].
e2) HTTP POST Flood	It is a type of DDoS attack in which the volume of POST requests overwhelms the server so that the server cannot respond to them all. This can result in exceptionally high utilization of system resources and consequently crash the server.
e3) HTTP POST Request	It is a method that submits data in the body of the request to be processes by the server. For example, a POST request takes the information in a form and encodes it, then post the content of the form to the server.
e4) HTTPS POST Flood	It is an HTTP POST flood sent over an SSL session. Due to the use of SSL it is necessary to decrypt this request in order to inspect it.
e5) HTTPS POST Request	It HTTPS POST Request is an encrypted version of an HTTP POST request. The actual data transferred back and forth is encrypted.
e6) HTTPS GET Flood	An HTTPS GET Flood is an HTTP GET flood sent over an SSL session. Due to the SSL, it is necessary to decrypt the requests in order to mitigate the flood.
e7) HTTPS GET Request	An HTTPS GET Request is an HTTP GET Request sent over an SSL session. Due to the use of SSL, it is necessary to decrypt the requests in order to inspect it.
e8) HTTP GET Flood	An HTTP GET Flood is a layer 7 application layer DDoS attack method in which attackers send a huge flood of requests to the server to overwhelm its resources. As a result, the server cannot respond to legitimate requests from the server.
e9) HTTP GET Request	An HTTP GET Request is a method that makes a request for information for the server. A GET request asks the server to give you something such as an image or script so

	that it may be rendered by your browsers.
f) SIP Flood Attack	The attacker can flood the SIP proxy with many SIP INVITE packets that have spoofed source IP addresses. To avoid any antispoofing mechanisms, the attackers can also launch the flood from a botnet using non-spoofed source IP-addresses. There are two categories of victims in this attack scenario. The first types of victims are the SIP proxy servers. Not only will their server resources be depleted by processing the SIP INVITE packets, but their network capacity will also be consumed by the SIP INVITE flood. In either case, the SIP proxy server will be unable to provide VoIP service. The second types of victims are the call receivers. They will be overwhelmed by the forged VoIP calls, and will become nearly impossible to reach by the genuine callers.
g) ICMP Flood (Smurf Attack)	There are three entities in these attacks: the attacker, the intermediary, and the victim. First, the attacker sends one ICMP echo request packet to the network broadcast address and the request is forwarded to all the hosts within the intermediary network.Second, all of the hosts within the intermediary network send the ICMP echo replies to flood the victim. Solutions to the smurf attack include disabling the IP-directed broadcast service at the intermediary network. Nowadays, smurf attacks are quite rare in the Internet since defending against such attacks are not difficult.
h) IGMP Flood	Is non-vulnerability based, as IGMP is designed to allow multicast. Such floods involve a large number of IGMP message reports being sent to a network or router, significantly slowing and eventually preventing legitimate traffic from being transmitted across the target network.
i) UDP Flood Attack	A does not exploit a specific vulnerability. Instead, it simply abuses normal behavior at a high enough level to cause congestion for a targeted network. It sends a large number of UDP datagrams from potentially spoofed IP addresses to random ports on a target server. The server receiving this traffic is unable to process every request and consumes all of its bandwidth attempting to send ICMP "destination unreachable" packet replies to confirm that no application was listening on the targeted ports.
2) Brute-force	Performed by initiating a vast amount of seemingly

Attacks	legitimate transactions. Since an upstream network can usually deliver higher traffic volume than the victim network can handle, this exhausts the victim’s resources.
a) Filterable Attacks	Use bogus packets or packets for non-critical services of the victim operation, and thus can be filtered by a firewall.
b) Non-filterable Attacks	Use packets that request legitimate services from the victim. Thus, filtering all packets that match the attack signature would lead to an immediate denial of the specified service to both attackers and the legitimate client.

Table 49.6 - Classification of DDoS-attacks on IoT systems by Layer of OSI Model

Network/transport level DDoS attack	At this level, mostly TCP, UDP, ICMP, and DNS protocol packets are used to launch the attacks
Application level DDoS attack	Generally consume less bandwidth and are stealthier in nature when compared to volumetric attacks. They can have a similar impact to service as they target specific characteristics of well-known applications such as HTTP, DNS, VoIP or Simple Mail Transfer Protocol (SMTP). Focus on disrupting legitimate user’s services by exhausting the resources. Overloads an application server, such as by making excessive login, database lookup or search requests. Application attacks are harder to detect than other kinds of DDoS attacks.

Table 49.7 - Classification of Encrypted DDoS Attacks on IoT systems

1) SQL Injection	Takes advantage of poor application coding. When the application inputs are not sanitized it becomes vulnerable. Attackers can modify an application SQL query to gain access to unauthorized data with administrator access, run remote commands on the server, drop or create objects in the database and more.
2) SSL attacks	SSL DDoS attacks and SSL DoS attacks target the SSL handshake mechanism, send garbage data to the SSL server, or abuse functions related to the SSL encryption key negotiation process. SSL attacks in the form of a DoS attack can also be launched over SSL-encrypted traffic, making it extremely difficult to

	identify.
3) Encrypted-based HTTP (HTTPS Flood) Attacks	HTTPS floods - floods of encrypted HTTP traffic - are now frequently being used in multi-vulnerability DDoS attack campaigns. Compounding the impact of "normal" HTTP floods, encrypted HTTP attacks add several other challenges, such as the burden of encryption and decryption mechanisms, complicating DDoS attack prevention efforts.
4) THC-SSL-DoS Attacks	The Hacker's Choice (THC), an international group of security researchers and hackers, developed this proof of concept tool to encourage vendors to patch SSL vulnerabilities and offer anti-DDoS protection. THC-SSL-DoS require only a small number of packets to cause denial of service (DoS) for a large server. It initiates a regular SSL handshake, then immediately requests renegotiation of the encryption key. The tool repeats this renegotiation request until all server resources have been exhausted.
5) Lightweight Directory Access Protocol (LDAP) Injection	An attack used to exploit web-based applications that construct LDAP statements based on user input. When an application fails to properly sanitize user input, it’s possible to modify LDAP statements using a local proxy. This could result in the execution of arbitrary commands, such as granting permissions to unauthorized queries, and content modification inside the LDAP tree. The same advanced exploitation techniques available in SQL Injection can be similarly applied in LDAP Injection.
6) Cross-Site Request Forgery (CSRF)	Malicious website will send a request to a web application that a user is already authenticated against from a different website. This way an attacker can access functionality in a target web application via the victim's already authenticated browser. Targets include web applications like social media, in-browser email clients, online banking and web interfaces for network devices.
7) Cross-Site Scripting (also known as XSS)	Is one of the most common application-layer web attacks. XSS vulnerabilities target scripts embedded in a page that are executed on the client-side (in the user’s web browser) rather than on the server-side. XSS in itself is a threat that is brought about by the internet

	security weaknesses of client-side scripting languages, such as HTML and JavaScript. The concept of XSS is to manipulate client-side scripts of a web application to execute in the manner desired by the malicious user. Such a manipulation can embed a script in a page that can be executed every time the page is loaded, or whenever an associated event is performed.
8) A man-in-the-middle attack	Is a type of cyberattack where a malicious actor inserts him/herself into a conversation between two parties, impersonates both parties and gains access to information that the two parties were trying to send to each other. A man-in-the-middle attack allows a malicious actor to intercept, send and receive data meant for someone else, or not meant to be sent at all, without either outside party knowing until it is too late.

49.2 Analysis of DDoS attacks in IoT systems statistics

Due to the increased complexity of IoT there were two issues that require quick solutions:

1) increasing the number of DoS- and DDoS- attacks on IoT systems and data center servers;

2) increasing in power consumption in the case of successful attacks.

Nowadays in the code 80% of the web-resources was discovered average risk level vulnerability Cross-Site Scripting (XSS), which allows attacker to implement in the user's browser arbitrary HTML-tags, including JavaScript language scripts, and other languages, and get the value of the identifier of attacked session, and perform other illegal actions, such as phishing attacks. Nearly 47% of web sites also contain the vulnerability associated with the lack of protection against the selection of credentials (Brute Force). Now is the large number of vulnerability of TCP connections and spurious resets (RSTs), sent with forged IP source addresses (spoofing) [8-10].

An example is using of protocols causing reflection when an attacker sending a TCP SYN packet to a well-known server with a spoofed source address; the resulting TCP SYN ACK packet will be sent to the spoofed source address. The DDoS attack prevents normal use of the computer or network by valid users. A new form of attack is a class known as the Non-Reflection DDoS attack. This new technique uses very large numbers of devices typically classified as “Things” in the terminology of the IoT, that can be harnessed from all areas of the Internet and for large number of networks. This massive number of devices that successfully generate attacks on throughput rates on the order of one Terabit-per-second (1 Tbps) or more [11].

The devices which are used to attack: DVRs, IP Cameras, CCTV, NVR, DVR devices (video surveillance); satellite antenna; network devices (such as routers, access points, WiMAX, cable and ADSL-modems, etc.); NAS (Network Attached Storage) with Internet access, video monitors, game consoles. Affected Internet devices are used for: preparation of an attack on any Internet destinations and Internet services such as HTTP, SMTP and network scanning. Vulnerable devices are then become infected with malicious software that turns them into “bots,” forcing them to obey to a central control server that can be used as a staging ground for launching powerful DDoS attacks designed to knock Web sites offline. The most attacked protocols were SSH (57%) and Telnet (42%). Is being created and use unauthorized SSH tunnel, although IoT devices must to be protected from this type of access by the implementation of secure shell commands in a web-interface without any user administrator privileges.

Malware-based botnets like Mirai, XOR and BillGates continues to expand on botnet-based attacks but is also being used in DDoS-For-Hire and extortion campaigns. Attack vectors: SYN, UDP fragment, PUSH, TCP, DNS and UDP floods [12-15]. In addition to the simple flood-attacks, widespread in the last few years the application level attacks DDoS flooding attack at Application-level (Level 7 of the OSI model), aimed at ending the provision of legitimate services to the user and exhaustion of the server resources like CPU, memory, disk/database bandwidth, sockets, input/output bandwidth. DDoS is accomplished by sending large amounts of otherwise legitimate requests to a network-aware application. It can be sending a large amount of requests to a web server, for increase the load the server process. The goal of this type of attack is to prevent other users to access the service by forcing the server to fulfill an excessive number of transactions. The network itself may still be usable, but since the web-server process cannot respond to the users, access to service is denied [16].

The fight against these attacks is more difficult, as the system firewall and Image Packaging System (IPS) regard such attacks as legitimate traffic, taking into account the construction of a full TCP 3-way handshake. Also DDoS-attacks in 2016 aimed at disabling of Load balancer mechanism of known companies-developers routers, using the vulnerability in their software or features of balancing algorithms. The fight against these attacks more difficult, as the system firewall and IPS regard such attacks as legitimate traffic, taking into account the construction of a full TCP 3-way handshake [17].

Attacks on IoT are the massive DDoS campaigns, which were tied to an IoT malware strain known as Mirai, is capable of launching multiple types of DDoS attacks, including SYN-flooding, UDP (User Datagram Protocol) flooding, Valve Source Engine query-flooding, GRE (Generic Routing Encapsulation) flooding, ACK-flooding (including a variant intended to defeat intelligent DDoS mitigation systems, or IDMSes), pseudo-random DNS label-prepending attacks (also known as DNS “Water Torture” attacks), HTTP GET attacks, HTTP POST attacks, and HTTP HEAD attacks [18].

In September 2016 attackers organized DDoS attack with capacity of 620 Gbps on site of journalist Brian Krebs, then on hosting provider OVH, follow a trend to exploit a 12-year-old vulnerability in OpenSSH to take control of thousands of poorly protected and vulnerable IoT devices being used for malicious purposes and generated 1.2 Tbps malicious traffic, which was primarily masked TCP and UDP traffic over port 53, using mechanism TCP/ACK, TCP/ACK+PSH, TCP/SYN [17-20]. Mirai exploits a version of Linux known as BusyBox, which is used in various IoT devices, botnet numbered 145607 IP wireless cameras, DVRs and home routers as proxies for malicious traffic.

The domain name system (DNS) DDoS attacks caused problems for Dyn and intermittently disrupted websites such as Netflix, Amazon, Twitter, Reddit and others. As a result, the target systems responds slowly or are completely crashed [21]. Statistics of DDoS attacks in 2015-2016 is shown in the table 49.8.

Table 49.8 - Statistics of DDoS attacks in 2015-2016

Date	Name of Botnet	Number of attacks	Using mechanisms
May 2015	MIT	30 DDoS	SYN flood

December 2015	BillGates Botnet	6 DDoS	SYN and DNS Floods, ICMP flood, TCP flood, UDP flood, SYN flood, HTTP Flood (Layer7), DNS query-of-reflection flood
January 2016-March 2016	BillGates Botnet	19 DDoS	GET –flood
March 2016	BillGates Botnet	4 DDoS	TFTP reflection
August 2013 - April 2016	MIT	74 DDoS	SYN and DNS Floods, ICMP flood, TCP flood, UDP flood, SYN flood, HTTP Flood (Layer7)
April 2016	BillGates Botnet	10 DDoS	TFTP Reflection
June 2016	XOR	6 DDoS	UDP Flood, UDP Fragment, SYN and DNS Flood, SSH brute force attempts for root login credentials (previously it was reported that infection methods include a vulnerability in ElasticSearch Java VM)
August2016	Kaiten/STD / Mirai	6 DDoS	Brute-force of Telnet and SSH ports
September 2 016	Mirai/Gafgyt (known under the name Lizkebab, BASHLITE, Bash0day, Bashdoor and Torlus) DDoS	2 DDoS	Masked TCP and UDP, TCP/ACK, TCP/ACK+PSH, TCP/SYN
October 201 6	Mirai	2 DDoS	Vulnerabilities in NTP and DNS, SYN, UDP fragment, PUSH, TCP,

			DNS and UDP floods, GRE flood
November 2016	Mirai	5 DDoS on banks	TCP/ACK, TCP/ACK+PSH, TCP/SYN

After gaining access to the network, the attacker can:

1) make the process of the invasion transparent for the personnel serving the information system for the purpose of theft of more information from the servers.

2) send invalid data to applications or network services, which causes abnormal termination or behavior of the applications or services.

3) flood a traffic of computer or the entire network until a shutdown occurs because of the overload. To block traffic, which results in a loss of access to network resources by authorized users [17-22].

Investigation of DDoS attack statistics showed that they have different principles of implementation. Even small data packets, which are formed in large number during attacks, can lead to catastrophic failures. Therefore, it is necessary to consider all possible consequences of attacks, their effect on system failure and increase in power consumption by the system under the influence of DDoS attacks.

49.3 Features of IoT systems

The number of devices connected to the IoT, is growing every day. The variants of IoT systems currently includes over fifty use cases, covering many service categories such as [1-5]: Smart metering (electricity, gas and water); Facility management services; Intruder alarms& fire alarms for homes & commercial properties; Connected personal appliances measuring health parameters; Tracking of persons, animals or objects Agriculture; Health Care / E-Health; Retail Safety and Security Automotive & Logistics Energy & Utilities Manufacturing; Smart City; Smart Home; Smart city infrastructure such as street lamps or dustbins; Smart office; Smart hotel; Connected industrial appliances such as welding machines or air compressors.

For business IoT provides significant advantages in terms of automation, energy efficiency, asset tracking and inventory management, dispatch and location, security, personal tracking and power savings.

49.3.1 Requirements to Smart Office

To IoT for office solutions (Smart Business Center - SBC) are presented such basic requirements:

a) in order to save energy in Smart Business Center (SBC) may to perform the installation of temperature control automatic systems, connection to the mobile network of intelligent systems Smart Metering accounting (electricity, gas and water), which allows you to make decisions on the use of certain energy modes in the office, as well as to save staff time through the use of remote water consumption data collection, electricity, gas, etc.

b) the possibility of using the various sensors and control units. It is necessary not just to automate certain functions (control of lighting, heating, ventilation and air conditioning - HVAC, etc.), but to integrate virtually any SBC equipment into a single system, works on the algorithm which will set the installer and designer SBC;

c) a complete feedback, which will allow to operate virtually all systems SBC, analyze the situation, make conclusions and to be able to control the SBC without external intervention (without pressing the control panel button), but only upon the occurrence of an event (for example should be provided, the emergence of the human in the corridor include of lighting, on-off ventilation and air conditioning system, power source switching to an alternative power supply, etc.);

d) SBC system should give staff full control over their offices and to provide protection against emerging new threats and threats due to the fact that new computer technologies with connection to the internet allow attackers to connect to the system. IoT architecture of any system consists of five levels: the intellectual level of the connection, the data information at the connection level, the cybernetic level, cognitive level, the configuration level. Malicious attacks on impact and vulnerability components of IoT devices, software, and a database (DB) can be applied to each of these levels. The aim of intruders can be stored data, video and audio recording, shutdown of hardware and software components of IoT, industrial espionage. List of types of malicious actions performed by the attacker: illegal use of user accounts; physical theft of office equipment and data carriers; theft software; run the executable code for the damage to the systems, for the destruction or corruption of data; modification data; identity theft; execution of actions that do not allow users to access network services and resources; execution of actions that reduce network resources and bandwidth. The basis of any SBC system - is the server on which the control software is stored. The largest number of DoS - and DDoS – attacks direct to servers. Some types of attacks aimed at disabling the router and the switch, resulting in a malfunction of computers, tablets, smart phones and a variety of IoT devices connected to the system, as well as the basic components SBC system - sensors [8]. Also on the supply system can be carried out special attacks;

f) increase the number of computers and servers resulting in significant power consumption, it is necessary to provide greater flexibility and adaptability of the infrastructure of power facilities.

49.3.2 Architecture of the SBC network

IoT solutions for the office is a network of automated and user devices, allowing staff to solve their business problems using the latest technological capabilities. Network nodes are able to receive and transmit information; can interact with other objects or be independent; may have different levels of access to its settings, depending on the security level, etc. To install SBC it is necessary as to automate some specific functions (control of lighting, ventilation and air conditioning, etc.) and introduce the virtual integration of any equipment, of SBC in the single system, which works by the algorithm which will set by the installer and designer SBC [23]. Based on the analysis of standard solutions for the implementation of IoT system is proposed the wired architecture of the network SBC. Using for IoT SBC Internet wire network devices are: router with Ethernet-ports and wireless access ability, soft switch the second or third layer, firewall, power block, server with control software, IP-camera, sensors, cables (fig. 49.2). The system can operate as a standalone or with Internet connection.

Work wired network SBC depends entirely on the power source. For the smooth functioning of the UPS is used a redundancy as of its blocks and their batteries, also alternative power sources (solar panels, etc.). The basic system is an intelligent security, the ability to backup data, the ability to software improvements, a rate recovery in the event of a malfunction or failure. All actions must interact with the subsystems of a smart home, and therefore should be focused on start-up and adjustment of the system smart home. In this case, fixing any one subsystem does not affect the operation of other subsystems.

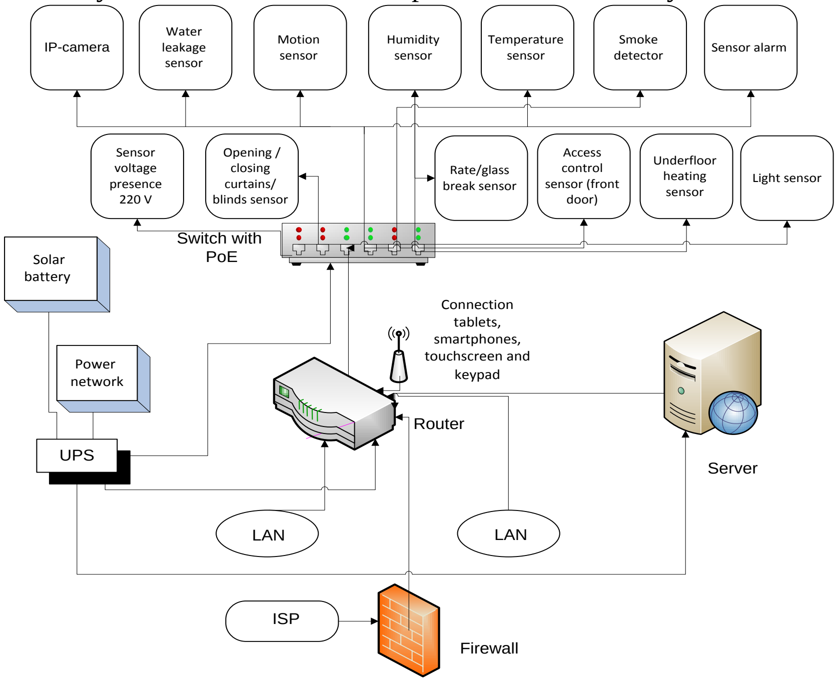


Fig.49.2- The architecture of the SBC network [23]

Router is the central subsystem of SBC, that connected Internet network by ISP and LAN, includes switch, server, interconnected cables, and also can to plug tablets, smartphones over IEEE 802.11 or 802.15.4 standards. Switch the second or third layer with PoE technology connect all Ethernet-sensors and IP-cameras in office.

Server - it is the control subsystem in the SBC, it keeps control and diagnostics programs that poll the Ethernet-sensors, and keeps and treated statistics. Additional software with security policies are installed on the server. On the server undertaken the largest number of DoS- and DDoS – attacks, brute-force attacks, Phishing attacks. Some types of attacks aimed at disabling Server, Router and Switch, resulting in malfunction of computers, tablets, smartphones and various gadgets connected to the system, as well as the sensors. All actions must interact with the subsystems of a SBC, and therefore should be focused on start-up and adjustment of the system smart home. In this case, fixing any one subsystem does not affect the operation of other subsystems.

49.3.3 Analysis of security aspects of network devices in SBC

But there is problem with security in SBC: these are the unknown or unreported vulnerabilities in the software that's being used. They could be bugs, or entirely new types of attacks [8]. Each device of the SBC is a potential entry point for a network attack by insiders, hackers, or criminals. When security is insufficient in even seemingly harmless household appliances, or other IoT products, it presents endemic vulnerabilities and risks [9]. IoT is based on using a variety of sensors and control units. SBC system gives staff complete control over their offices, but, at the same time, there are new dangers and threats due to the fact that the new computer technology with an internet connection, provide possibility of hackers to connect to the system. The IoActive IoT Security Survey revealed that nearly half (47%) of all respondents felt that less than 10% of all IoT products on the market are designed with adequate security [10].

As written at [24], the IoT facilities, available by the Internet, may disclose personal user data to another people. Lack of data security IoT brings to detection of vulnerable devices by criminals to gain access to him or other IoT facilities. The attacks report by Kaspersky Lab shows, that IP-cameras are connected wirelessly to the Internet are not necessarily secure: here is the potential for cybercriminals to passively monitor the cameras for the implementation of the code in the network, thereby replacing the image in the camera channel communication to fake shots, or put the system in offline mode. If the data packets are transmitted via a data network, it has not been encrypted, an attacker can create their own version of the software and data processing for controlling the IoT [25].

Physical security - notification of suspicious activity at the moment, which is found near the IoT-device. Alerts come with surveillance cameras, physical access control, and other sensors to detect movement and other [7, 12]. With the increasing variety and scale of apps running on the network, they need a common policy framework to move beyond the perimeter-based security model for all things connected [13]. Security and privacy are important requirements for the IoT due to the inherent heterogeneity of the Internet connected objects and the ability to monitor and control physical objects [25]. Last week was carried out a series of powerful DDoS attacks aimed to kill multiple targets. For DDoS attacks had been used malware to infect the largest number of IoT devices, connected to the Internet [6-7].

In the table 49.9 are shown security features of routers from some famous manufacturers.

Table 49.9 - Security features of routers from some manufacturers

Features of the routers	**Cisco**	**Huawei**	**D-Link**	**HP**
VPN	FlexVPN, Easy VPN remote server, Enhanced Easy VPN, Dynamic Multipoint VPN (DMVPN), Group Encrypted Transport VPN (GET VPN), V3PN, MPLS VPN	VPN	VLAN	VPN, DVPN, GDVPN, ADVPN
Means for ensuring of security and reliability	VRF-Aware Firewall and Network Address Translation (NAT) Security Group Tag Exchange Protocol (SXP), SGT over GETVPN SGT over IPSEC SGT over DMVPN SGT-based ZBFW Port/Layer 3 interface/IP/subnet-to-SGT mapping SGT export in Flexible NetFlow ACL, FPM, control plan protection, control plane policing (CoPP), QoS, role-based CLI	User authentication protocol: PAP, CHAP, MSCHAP, RADIUS, and HWTACACS User accounting protocol: RADIUS, HWTACACS, and COPS User authorization protocol: RADIUS, HWTACACS, and COPS Policy protocol:	IP-MAC-Port Binding RADIUS / TACACS / XTACACS/ TACACS+	Rich features with Comware v7 security, including IPS with high encryption. Stateful firewall, NAT, SSHv2 security features. Backup Center acts as a part of the management and backup function to provide backup for device interfaces; delivers reliability by switching traffic over to a backup interface when the primary one fails Virtual Router Redundancy Protocol (VRRP); VRRP load balancing; allows custom filters for increased performance

	access, source-based RTBH, uRPF, SSHv2	COPS and COA		and security; supports ACLs, IP prefix, AS paths, community lists, and aggregate policies

As the use of IoT business and IT - technology security experts, entrusted with the installation of IoT - applications and solutions must take into account the significant paradigm shift, as well as operational, strategic and business objectives. When faced with a variety of internal and external security issues, such as network attacks, malicious software, malware, external hackers, the decision makers should be aware that the statistics of threats from external hackers (37%), malware (33 %), internal hackers (30%), and DDoS attacks. List of types of malicious actions performed by the attacker: illegal use of user accounts; theft software; running the executable code for the damage to the systems, for the destruction or corruption of data; modification data; identity theft; execution of actions that do not allow users to access network services and resources; execution of actions that absorb network resources and bandwidth.

Manufacturers of equipment for networks proposes these methods for security increasing: provide fast VPN performance using hardware acceleration; enforce policies consistently; detect and mitigate (DDoS) and other attacks; prevent misconfigurations that attackers can exploit; identify the type of device and provide the appropriate access control; control access to IoT devices based on the user and device identity; take advantage of proven threat management with FirePOWER™ Services, support oblivious transfer (OT) protocols and applications, detect and protect against specific threats including DDoS, operational safety, and insider attacks; Advanced Malware Protection (AMP); gain visibility into the protocols, devices, and applications you specify; secure Internet connectivity with high-performance VPN, Domain Name System (DNS), Dynamic Host Configuration Protocol (DHCP), and Network Address Translation (NAT) [6,7, 14, 17, 25]. In the table 1 is shown means of security in routers by some manufacturers. Large manufacturers also use firewalls and next generation intrusion prevention systems [16]. Firewall - a set of hardware and software that filter the Internet traffic and allow only one that allowed software (filtered access to vulnerable services that control access to devices on a network, make notification to the network device attacks or Firewall itself, etc.), that have disadvantages: reduces network bandwidth; it does not protect against software bookmarks and unintentional software defects; it does not protect against users to download programs that contain viruses.

49.4 A Markov model of SBC Availability and Security

In order to describe the reliability of IoT systems in the process of their operation under the influence of various kinds of DDoS attacks, a mathematical apparatus of the Markov chains, the Hidden-Markov chains, reliable graphs, attacks trees, reliable block diagrams, and graphic security models based on the hierarchical display attack model, tree-based security models.

We choose a Markov model for assessment of SBC availability and security [25-27].

Assumptions to the development of the Markov model of SBC availability:

- stream hardware failures of the system obeys poisson distribution;
- the flow of failures of subsystems is subject to Poisson for-grabs, as the results of monitoring and diagnostics, anti-virus software testing corrected secondary error (the result of the accumulation of the effects of primary errors and defects, bookmarks), and to fix a malfunction or failure of software, eliminating or the consequences of software bookmarks and code vulnerabilities, DoS - and DDoS - attacks, the number of primary software defects permanently. Therefore, the assumption is true, that the flow of software failures obeys Poisson distribution, the failure rate is constant;
- the model does not take into account that eliminating software vulnerabilities and design faults changes the parameters of the flow of failures (and recovering). To investigate the dependability SBC use the theory of Markov models, as the failure rates of hardware and software and the availability of software vulnerabilities is constant.

Fig. 49.3 is shown a Markov graph of g of the main SBC subsystems functioning, λ - the rate of failure or attack, μ - the rate of the recovery system.

The basic states of the system:

1) Normal condition (up-state) system;

2) Failure due to faulty feeder from the stationary power supply (220 V);

3) Failure due to a malfunction of the second feeder (a solar battery);

4) Failure of the battery in the UPS;

5) Reconfigure the power subsystem;

6) Failure of the cable connecting the Router and Server;

7) Failure of the cable connecting the UPS and Switch, and / or the Router, and / or Server;

8) Failure of the cable connecting the Router and Switch;

9) Firewall Denial;

10) Refusal Server due to a fault server components, or exposure to attacks on the code server system software with vulnerabilities;

11) Failure Router as a result of failure of the router components, or the impact of the attacks on the code of the router operating system vulnerabilities;

12) Switch Failure due to a fault switch components, or exposure to an attack on the system software code switch with the presence of vulnerabilities;

13) Partial failure of the system due to the failure of cable connecting any or multiple sensors and IP cameras;

14) Partial failure of the system due to the failure of any one or more sensors and IP cameras;

15) Failure of the system.

As an index of reliability SBC we choose availability function AC(t), that is defined as the sum of the probabilities of staying the system in an up-states:

$$AC(t) = P_1(t) + P_5(t).$$

Solving the system of Kolmogorov-Chapmen equations, we can get the value of the availability function components and SBS network, the number of network failures due to software vulnerabilities, and how and with what rate the system is restored after such failures. It follows that service availability, service continuity, cyber security, data integrity, resilience and high dependability of software and hardware should be inherent in IoT networks.

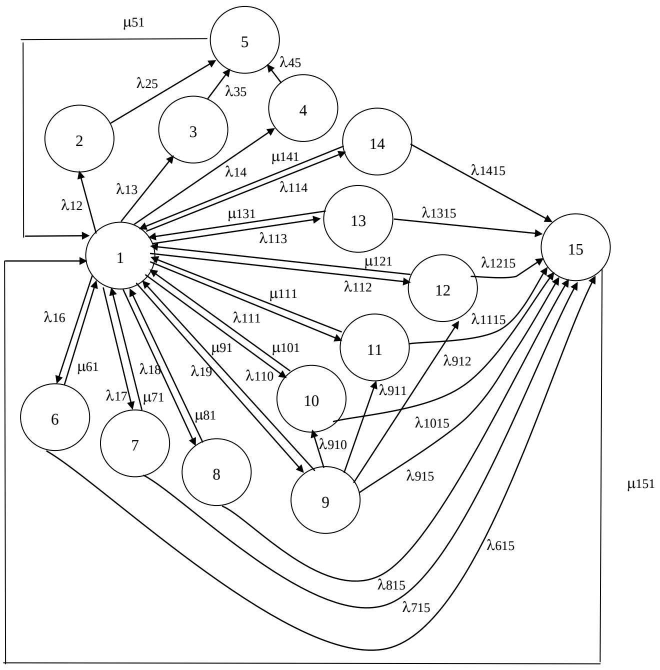


Fig. 49.3 - Graph of Markov model of SBC availability [24]

If we taking into account dependability of the alternative power sources, a graph of Markov model of SBC availability will be changed, as shown in fig.49.4. Assumptions to the development of the Markov model of SBC availability are the same.

Fig.49.4 shows a Markov graph of functioning of the main subsystems SBC, λ - the failure rate or attack, μ - the recovery rate.

The basic states of the SBC system:

1) Normal condition (up-state) system;

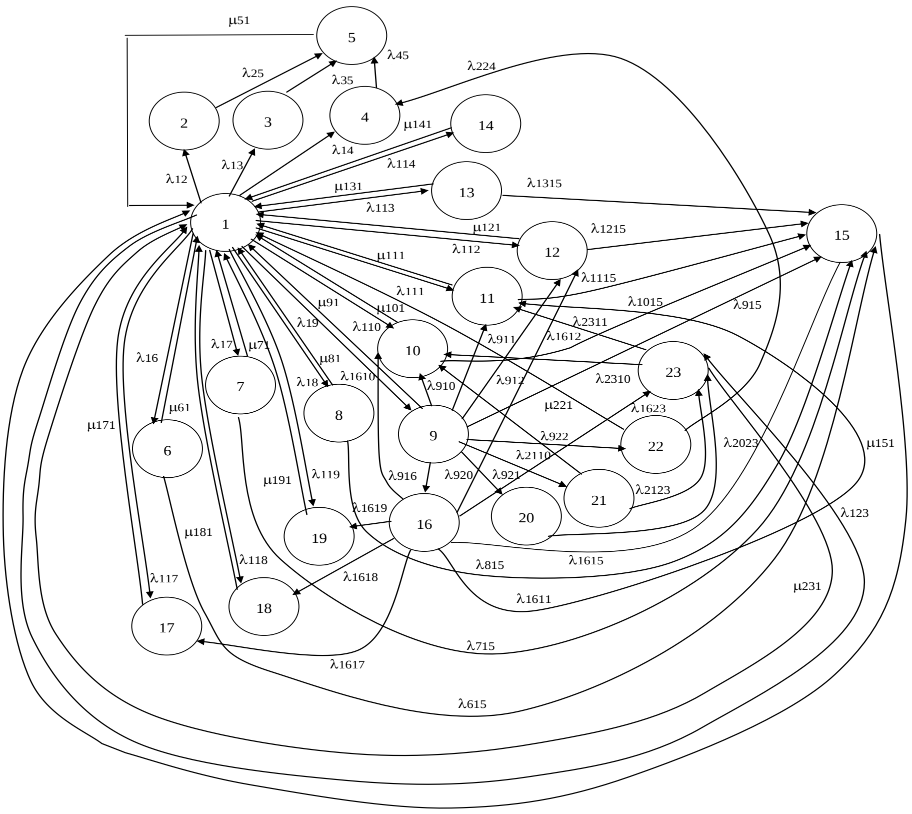


Fig. 49.4 - Graph of Markov model of SBC availability taking into account dependability of alternative power sources [26]

2) Failure due to faulty feeder from the stationary power supply (220 V);

3) Failure due to a malfunction of the second feeder (a solar battery);

4) Failure of the battery in the UPS;

5) Reconfigure the power subsystem;

6) Failure of the cable connecting the router and server;

7) Failure of the cable connecting the ups and switch, and / or the router, and / or server;

8) Failure of the cable connecting the router and switch;

9) Firewall denial;

10) Refusal server due to a fault server components, or exposure to attacks on the code server software with vulnerabilities;

11) Failure router as a result of failure of the router components, or the impact of the attacks on the code of the router operating system vulnerabilities;

12) Switch failure due to a fault switch components, or exposure to an attack on the software code switch with the presence of vulnerabilities;

13) Partial failure of the system due to the failure of cable connecting any or multiple sensors and IP cameras;

14) Partial failure of the system due to the failure of any one or more sensors and IP cameras;

15) Failure of the system SBC;

16) DDoS-attack;

17) Failure of the server hardware;

18) Failure of the router hardware;

19) Failure of the switch hardware;

20) Brute-force attack;

21) Phishing-attack;

22) Special attack on the power subsystem UPS;

23) Failure of the software of server or router.

$$AC(t) = P1(t) + P5(t).$$

At figures 49.5-49.8 is shown the changing of AC from changing the rate of the transition from one state to another in the Markov's model.

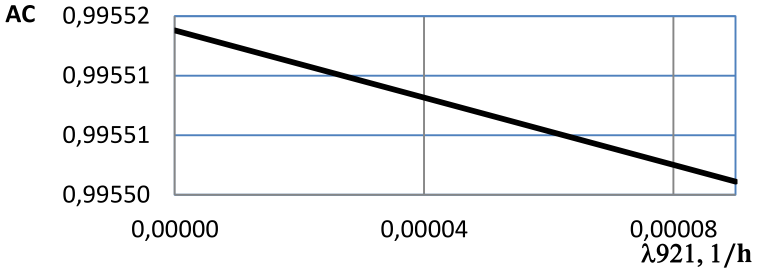


Fig.49.5 - Graph of dependence of AC of SBC and the rate of transition to a state of Phishing-attack - λ921

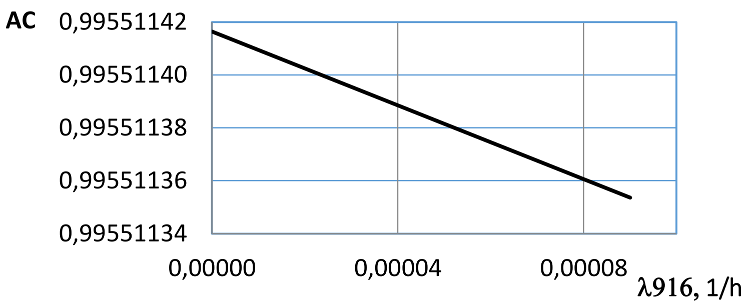


Fig.49.6 - Graph of dependence of AC of SBC and the rate of transition to a state of DDoS-attacks - λ916

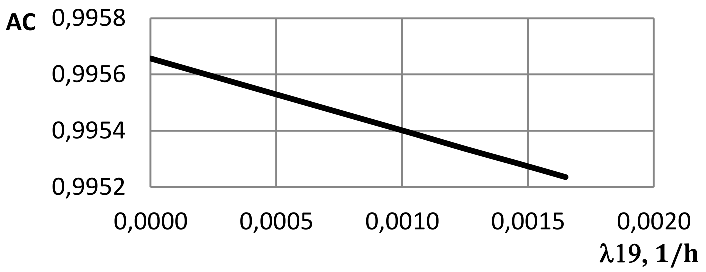


Fig.49.7 - Graph of dependence of AC of SBC and the rate of transition to a state of denial Firewall - λ19

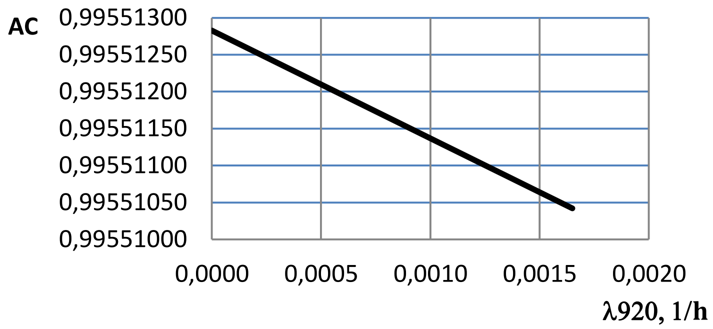


Fig.49.8 - Graph of dependence of AC of SBC and the rate of transition to a state of Brute-force attacks - λ920

The analysis of the Markov model simulation results showed that by increasing the rate of the transition to a state of denial firewall λ19 decreases the value of SBC availability function. In case of refusal the firewall different kinds of attacks can freely influence the vulnerability of the server software, router, switch. Increase the rate of the transition from a state to a state of denial firewall Brute-force attack - λ920, Phishing-attacks - λ921, DDoS-attacks - λ916 reduces the value of availability function of SBC.

Analysis of the constructed Markov model showed that the studied system is stable, the availability function quite high and not much changing with an increase in the intensities of the attacks. Recently, however, increasing number of malicious attacks and their types, that will lead to a sharp decrease in system reliability and security of SBC. It is therefore necessary to look for more advanced security techniques researched systems that suppose purpose of further research.

A Markov model of server's control unit functioning is presented on fig.49.9, considering DDoS- and spy-attacks in the case of server firewall failure, which has the following condition: Good-working state (1); The condition of the first processor of control unit is failure (2); The condition of the second processor of server's control unit is failure (3); The failure of compare unit (4); The condition of both the first and the second processors of server's control unit are failure (5); Failure condition of server's control unit (6); Failure condition of the input/output system (7); Failure condition of the ROM (8); Failure condition of the RAM (9); Failure condition of server Software (10); Failure condition of server Firewall (11); Failure condition of External server memory (12).

A system of linear differential equations of the Kolmogorov-Chapmen composed and solved in the paper with the initial conditions:

$$\sum_{i=1}^{12} \mathrm{Pi(t)} = 1,\ \mathrm{P1(0)} = 1.$$

$$\frac{dP1(t)}{dt} = \mathrm{\mu 21 \cdot P2(t) + \mu 31 \cdot P3(t) + \mu 41 \cdot P4(t) +}$$
$$\mathrm{\mu 61 \cdot P6(t) + \mu 71 \cdot P7(t) + \mu 81 \cdot P8(t) + \mu 91 \cdot P9(t) + \mu 101 \cdot P10(t) + \mu 121 \cdot P12(t) -}$$

$$P1(t) \cdot (\lambda 12 + \lambda 13 + \lambda 14 + \lambda 17 + \lambda 18 + \lambda 19 + \lambda 110);$$
$$\frac{dP2(t)}{dt} = \lambda 12 \cdot P1(t) + \mu 52 \cdot P5(t) + \lambda 112 \cdot P11(t) - P2(t) \cdot (\lambda 25 + \mu 21);$$
$$\frac{dP3(t)}{dt} = \lambda 13 \cdot P1(t) + \mu 53 \cdot P5(t) + \lambda 113 \cdot P11(t) - P3(t) \cdot (\lambda 35 + \mu 31);$$
$$\frac{dP4(t)}{dt} = \lambda 14 \cdot P1(t) - P4(t) \cdot (\lambda 46 + \mu 41);$$
$$\frac{dP5(t)}{dt} = \lambda 25 \cdot P2(t) + \lambda 35 \cdot P3(t) - P5(t) \cdot (\mu 52 + \mu 53 + \lambda 56);$$
$$\frac{dP6(t)}{dt} = \lambda 56 \cdot P5(t) + \lambda 76 \cdot P7(t) + \lambda 46 \cdot P4(t) + \lambda 86 \cdot P8(t) + \lambda 96 \cdot P9(t) + \lambda 106 \cdot P10(t) + \lambda 126 \cdot P12(t) - \mu 61 \cdot P6(t);$$
$$\frac{dP7(t)}{dt} = \lambda 17 \cdot P1(t) + \lambda 117 \cdot P11(t) - P7(t) \cdot (\mu 71 + \lambda 76);$$
$$\frac{dP8(t)}{dt} = \lambda 18 \cdot P1(t) - P8(t) \cdot (\mu 81 + \lambda 86);$$
$$\frac{dP9(t)}{dt} = \lambda 19 \cdot P1(t) + \lambda 119 \cdot P11(t) - P9(t) \cdot (\mu 91 + \lambda 96);$$
$$\frac{dP10(t)}{dt} = \lambda 110 \cdot P1(t) + \lambda 1110 \cdot P11(t) - P10(t) \cdot (\mu 101 + \lambda 106);$$
$$\frac{dP11(t)}{dt} = \lambda 111 \cdot P1(t) - P11(t) \cdot (\lambda 112 + \lambda 113 + \lambda 117 + \lambda 1110 + \lambda 119 + \lambda 1112 + \mu 111);$$
$$\frac{dP12(t)}{dt} = \lambda 112 \cdot P1(t) + \lambda 1112 \cdot P11(t) - P12(t) \cdot (\lambda 126 + \mu 121).$$

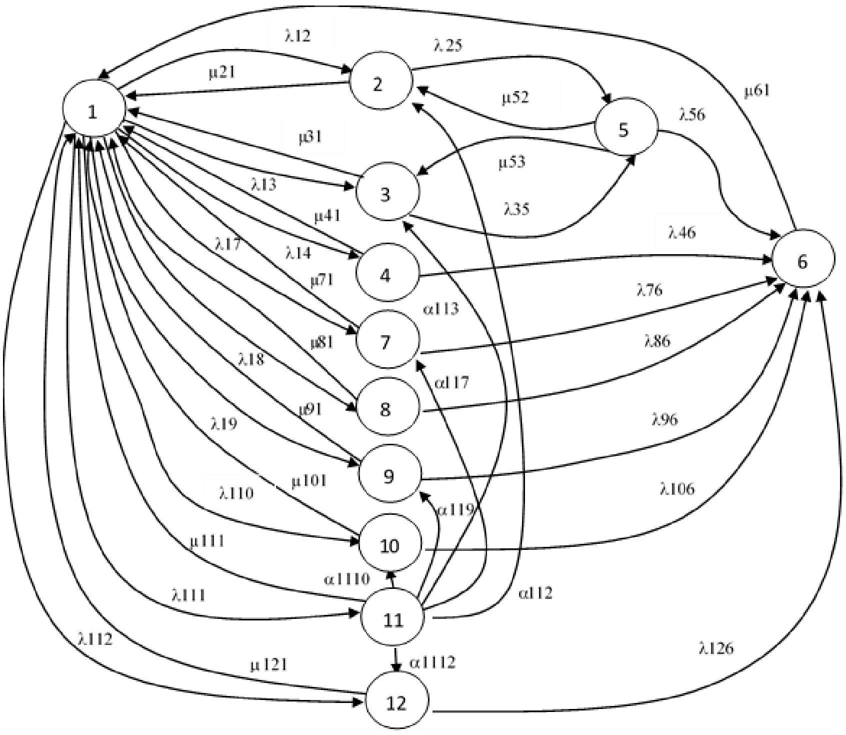


Fig. 49.9 - Graph of a Markov model server functioning in SBC taking into account impact of DDoS-attacks on its components

Conclusions

Analysis has shown, that large manufacturers of network equipment, which should be used in the organization of the wired network SBC, complement its routers, switches, additional measures to protect against possible attacks from the Internet. However, Security reports the major antivirus companies (Kaspersky Lab, McAfee, Symantec), CERT, Security lab Cisco, HP, indicates that periodically occurring attacks on the routers and other network equipment, that lead to the failure of hardware and software components of the device as a result of the availability of its software vulnerabilities [15-24]. Attack methods include the use of zero-day management target phishing attacks, as well as keylogger kernel mode, which is when you press downloads data from infected computers. They also manage to crack the weak digital signature keys to generate certificates for signing malware to malicious files presented as legitimate software.

To assess the dependability of SBC have been proposed a model that takes into account successful DDoS attacks, failures and malfunctions hardware and software of various components of SBC; energy modes of the SBC system, the rate of which is based on the analysis of statistical data of firewall functioning; failure of IoT devices management systems, following an attack impact to the router software, power supply systems, servers, etc.

The Markov model of SBC availability was developed and investigated based on the analysis of hardware failures, software components, communication lines, router, firewall, special attacks on the SBC power system, DDoS attacks on the router and the server, attacks using components IoT infrastructure.

Studies have shown that the definition of the beginning of attacks is difficult, especially if there are vulnerabilities, program bookmarks, hardware bookmarks, hardware failures or software errors in firewalls. Attacks of DDoS on the infrastructure of IoT or using IoT systems can last several hours and even several days. The large size of data packets and the high speed of their transfer to the server or router in the first moments of time can "deceive" the firewall, server, router and the entire system. Most attackers disguise their activities, and detection of attacks, especially at Application layer 7, is difficult. When the data flow increases during attacks, the server and the router exit the low-power mode and switch to the active power mode. The illusion of high availability of the IoT system arises. There is a constant processing of incoming packets until the server or router goes into a state of fail or failure.

An analysis was made of possible vulnerabilities of software servers and methods of protecting against attacks. To assess the server’s dependability in SBC have been proposed a model that takes into account successful DDoS attacks, failures of hardware and software of various components of server in SBC; have been researched of DDoS-attack impact to the server. Was researched and analyzed the function availability of server’s functioning in IoT system – SBC.

The carried out researches allow to estimate server‘s availability function in SBC at change of rate of transition from different kinds of the server components states without influence of DDoS-attacks and at influence DDoS-attacks on each basic component of the server in SBC.

Questions to self-checking

1. Which the main types of DDoS attacks on the IoT system?
2. Which main security issues there are in the network devices of IoT system?
3. What the name of the largest botnet of IoT devices?
4. What include the architecture of Smart Business Center?
5. What kind of VPN do you know and how VPN to use in IoT systems?
6. Describe principle of DDoS attack.
7. What is the reflection DDoS attack?
8. Describe the Markov model of SBC availability.
9. Why are the Kolmogorov-Chapman equations given and what do we get by solving them?
10. What requirements to Smart Office?
11. Why in assessing the availability of SBS in the Markov model it is necessary to consider the impact of attacks

References

1. O. Vermesan, p. Friess, P. Guillemin, et.al. Internet of things – from research and innovation to market deployment. river publishers series in communication, 2014. Available at: http://www.internet-of-things-research.eu/pdf/IERC_Cluster_Book_2014_Ch.3_ SRIA_WEB.pdf ,(access date: 3.08.2016). 141 p
2. Internet of Things and its future. Available at: http://www.huawei.com/ilink/en/about-huawei/newsroom/press-release/ HW_080993?dInID=23407&relatedID= 19881&relatedName= HW_076569&dInDocName=HW_076557 (access date: 3.08.2016).
3. Electricity Information sharing and analysis center. Internet of Things DDoS White Paper, http://media.wix.com/ugd/416668_9d3598f6f4c649ad9eff6f17d00f30a2.pdf (access date: 24.10.2016).

4. https://standards.ieee.org/findstds/standard/802.3az-2010.html.

5. No stars for Internet of Things security. At this week's AusCERT 2016 conference, an embedded device security specialist proposed a 'Security Star' rating for consumer IoT devices. It's a great idea, but it'll never happen. Available at: http://www.zdnet.com/article/no-stars-for-internet-of-things-security/(accepted at 3.08.2016).

6. Defending TCP Against Spoofing Attacks, https://tools.ietf.org/html/rfc4953/.

7. Internet Security Threat Report VOLUME 21, APRIL 2016. Symantec. Available at: https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf. 81 p.

8. Kaspersky security bulletin 2015. Kaspersky-Security-Bulletin- 2015_FINAL_EN.pdf, https://securelist.com/files/2015/12/ Kaspersky-Security-Bulletin-2015_FINAL_EN.pdf. (accepted at 3.08.2016). 85 p.

9. SSHowDowN – Exploit von internet. Ezra Caltum & Ory Segal, Akamai Threat Research. Verцffentlichungsdatum: https://www.akamai.com/de/de/our-thinking/threat-advisories/sshowdown-exploitation-of-iot-devices-for-launching-mass-scale-attack-campaigns.jsp (access date: 11.10.16).

10. Kaiten/STD router DDoS Malware, https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/kaiten-std-router-ddos-malware-threat-advisory.pdf. (access date: 01.10.16).

11. Attack Spotlight: 363 Gbps DDoS Attack. https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/akamai-363-gbps-ddos-attack-threat-advisory.pdf. (access date: 25.07.16).

12. Rob Wright. Risk & Repeat: Rapid7 tackles IoT threats, vulnerabilities, http://searchsecurity.techtarget.com/podcast/Risk-Repeat-Breaking-down-the-latest-Mirai-IoT-botnet-attacks (access date: 03.11.2016).

13. Rob Wright. Risk & Repeat: DNS DDoS attacks raise concerns over IoT devices,

http://searchsecurity.techtarget.com/podcast/ Risk-Repeat-DNS-DDoS-attacks-raise-concerns-over-IoT-devices (access date: 28.10.2016).

14. DDoS Attacks: Trends show a stronger threat in 2015, http://www.calyptix.com/top-threats/ddos-attacks-trends-show-stronger-threat-in-2015/ ((access date: 02.2015).

15. How DDoS Detection and Mitigation Can Fight Advanced Targeted Attacks, https://www.sans.org/reading-room/whitepapers/analyst/ddos-detection-mitigation-fight-advanced-targeted-attacks-35000/ September, 2013. A SANS Whitepaper Written by John Pescatore.

16. S. Srinivasan, R. Kumar, J. Vain. Integration of IEC 61850 and OPC UA for Smart Grid automation // 2013 IEEE Innovative Smart Grid Technologies-Asia (ISGT Asia). – P. 1-5.

17. Cisco IoT System Brochure Cisco IoT System Deploy. Accelerate. Innovate. 2015. http://www.cisco.com/c/dam/en/us/products/ collateral /se/internet-of-things/brochure-c02-734481.pdf. 52 p.

18. Cisco IoT System Security: Mitigate Risk, Simplify Compliance, and Build Trust White Paper. 2015. http://www.cisco.com/c/dam/en/us/products/collateral/se/internet-of-things/iot-system-security-wp.pdf. 4 p.

19. Cisco IOS XR for NCS 6000 Packet Timer Leak Denial of Service Vulnerability. Available at: https://tools.cisco.com/security/center/content/ CiscoSecurityAdvisory/cisco-sa-20160713-ncs6k. (accepted at 3.08.2016).

20. Cyber Risk Report 2016 Executive summary. Hewlett Packard Enterprise. Available at: http://www8.hp.com/fi/fi/software-solutions/cyber-risk-report-security-vulnerability/ (accepted at 1.08.2016).

21. McAfee Labs Threats Report June 2016. Intel security. Available at: http://www.mcafee.com /us/resources/reports/rp-quarterly-threats-may-2016.pdf (accepted at 1.08.2016). 53 p.

22. Internet Security Threat Report VOLUME 21, APRIL 2016. Symantec. Available at: https://www.symantec. com/content/dam/symantec /docs/reports/istr-21-2016-en.pdf. 81 p.

23. Executive report: SMART CITIES. Transformational ‘smart cities’: cyber security and resilience. Transformational Smart Cities - Symantec Executive Report.pdf Available at: https://eu-smartcities.eu/sites/all/files/blog/files/Transformational%20Smart %20Cities%20-%20Symantec%20Executive %20Report.pdf. (accepted at 1.08.2016). 20 p.

24. O. Netkachov, P. Popov, K. Salako. Model-Based Evaluation of the Resilience of Critical Infrastructures Under Cyber Attacks // Proceeding of 9th International Conference (CRITIS 2014). – P. 231-243.

25. Reliability and Security Issues for IoT-Based Smart Business Center: Architecture and Markov Model. Kharchenko Vyacheslav, Kolisnyk Maryna, Piskachova Iryna, IEEE; Computer of science, MCSI 2016, Greece, Chania, 2016. Paper ID: 4564699.

26. Markov Model of the Smart Business Center Wired Network Considering Attacks on Software and Hardware Components. Vyacheslav Kharchenko, Maryna Kolisnyk, Iryna Piskachova, Nikolaos Bardis. International journal of computers and communications ISSN: 2074-1294, Volume 10, 2016, P. 113-119. //http://www.naun.org/cms.action?id=11961.

27. A Markov Model of IoT System Availability Considering DDoS Attacks and Energy Modes of Server and Router./ Maryna Kolisnyk, Vyacheslav Kharchenko, Iryna Piskachova and Nikolaos Bardis// CEUR-WS Proceeding of ICTERY 2017 Conference.

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ К РАЗДЕЛУ 49

AAS - As-a-Service
ACK – Acknowledgement
ACL – Access List
AMP - Advanced Malware Protection
CHAP – Challenge-Handshake Authentication Protocol
COPS – Check-Off Password System
CSRF - Cross-Site Request Forgery
DDoS - Distributed Denied of Access
DMVPN - Dynamic Multipoint VPN
DHCP - Dynamic Host Configuration Protocol
DNS - Domain Name System

DoS – Denied of Access
FTP – File Transfer Protocol
GRE – Generic Routing Encapsulation
HP – Hewlett Packard
HTTP –Hypertext Transfer Protocol
HVAC - Heating, Ventilation and Air Conditioning
HWTACACS - Hardware TACACS
ICMP – Internet Control Message Protocol
IGMP – Internet Group Management Protocol
IoT – Internet of Things
IT – Internet Technologies
LDAP -Lightweight Directory Access Protocol
NAS - Network Attached Storage
NAT – Network Address Translation
PAP – Password authentication protocol
PSH - Push
RADIUS - Remote Authentication Dial-In User Service
SBC – Smart Business Center
SSH – Secure Shell
SYN – Synchronization
TACACS - Terminal Access Controller Access Control System
TCO - Total Cost of Ownership
TCP- Transmission Control Protocol
TFTP – Trivial File Transfer Protocol
UDP – User Datagram Protocol
XTACACS – Extended TACACS
XSS - Cross-Site Scripting
VRRP - Virtual Router Redundancy Protocol
VPN – Virtual Private Network

АННОТАЦИЯ

В разделе дана классификация DDoS-атак, рассмотрены особенности реализации безопасности в маршрутизаторах разных производителей как компонентов системы IoT. Предложены Марковские модели для оценки готовности и безопасности IoT систем, которые учитывают отказы аппаратных средств и программного обеспечения, а также воздействие DDoS-атак на компоненты IoT систем.

В розділі дана класифікація DDoS-атак, розглянуті особливості реалізації безпеки в маршрутизаторах різних виробників, як компонентів системи IoT. Представлено Марковські моделі оцінки готовності та безпеки ITT систем, які враховують відхилення засобів та програмного забезпечення, а також вплив DDoS-атак на компоненти IoT систем.

In the section the classification of DDoS-attacks is given, Features of security implementation in routers of different manufacturers as components of the system IoT. Markov models are proposed for assessing the availability and security of IoT systems that take into account hardware and software failures, as well as the impact of DDoS attacks on IoT system components.

PART 13
Big Data Analysis Based Assessment of Software Reliability and Security

ABBREVIATIONS

BDA	Big Data Analysis
SWS	Software system
ERP	Enterprise Resource Planning
SCADA	Supervisory Control And Data Acquisition
IDC	International Data Corporation
TOAStRaS	Technology - oriented assessment of software reliability and security
RCM	Requirements complexity metric
XP	Extreme Programming
UML	Unified Modelling Language
LC	Life cycle
SRM	Software reliability methods
SRS	Software requirements specification
ER	Entity relationships
RDI	Requirements defectiveness index
RCI	The Requirements correctness index
OP-D	Operation and debugging
NHPP	Non-homogenous Poisson process
ODC	Orthogonal defects classification
HCT	Hierarchical clustering technique
OTRAP	Optimal Testing Resource Allocation Problem
NLPP	Nonlinear programming problem
RFC	Metric of response for a class
WMC	Metric of weighted methods per class
LCOM	Metric of lack of cohesion in Methods
LOC	Metric of lines of code
NPM	Metric of number of public methods
CA	Metric of coupling axipetal
CE	Metric of coupling external
CBO	Metric of coupling between object classes
NOC	Metric of number of children

DIT	Metric of depth of inheritance tree
KLOC	Initial code dimension in thousands of lines
SQL	Structured query language
CI	Critical infrastructures
CAE	Claims, Arguments and Evidence
PIA	Preliminary Interdependency Analysis
ERTMS	European Railway Traffic Management System specifications
XML	Extensible Mark up Language
ASV	Assessment of software vulnerabilities
IASV	Integrated assessment of software vulnerabilities

50. Big Data Analysis Based Assessment of Software Reliability and Security

The number of software systems (SWS) necessary for mankind constantly increases. Along with it mankind's dependence on SWS reliability and security considerably grows. According to estimates of National Institute of Standards and Technologies, as a result of insufficient reliability of SWS only direct economic loss of the USA makes annually about sixty billion dollars. At the same time threats of the successful hackers' attacks and invasions grow quicker, than protection from them. Level of SWS security constantly decreases. The technologies of cyber protection change slowly at the enterprises. At the same time hackers' methods and tools develop very actively. Results of the Positive Technologies' researches demonstrate that in 2013 86% of corporate systems of the large companies had the vulnerabilities allowing to receive total control over crucial resources. These are communication and payment service, transport and power systems, e-mail, storages of personal information and documents, ERP and SCADA systems. In 82% of cases it was enough for hacker to carry out the attack to possess a low or average qualification. The successful hackers' attacks cause enormous financial damage to the enterprises. According to International Data Corporation (IDC) analysis, the amount of world economy losses from cyber crime equal 445 billion dollars in 2014. In 2016 annual losses of world economy equal up to 560 billion dollars. Therefore the increase of SWS reliability and security is very important and serious task of the modern program engineering. SWS reliability and security properties are integrated by the term dependability. For the increase of SWS dependability, the first step to be made is to evaluate the quantitative indices of Reliability and Security. The second one is taking measures for their improvement. Hence, the SWS dependability assessment methods require development and widespread of implementation in best practices of the software companies.

50.1. Big data based assessment. Motivation and goals

Experts of software corporations need experimental data to review and prediction SWS dependability indicators. Processes of data accumulation run intensely in modern digital world. From 2005 to 2020, the digital universe will grow by a factor of 300, from 130 to 40,000 Exabytes, or 40 trillion gigabytes (more than 5,200 gigabytes per each man, woman, and child in 2020). From now until 2020, the digital universe will be increasing almost twice every two years [1]. This data includes data necessary for reliability evaluation from SWS demands management systems, configuration control systems, defects control systems, and prototypes, initial software code, metrical data, test-cases, temporary rows of defects' identification, code analyzers reports, system logs, users' reports on SWS failures, vulnerabilities etc. They may include audio, visual, graphical, text and numerical data either structured, or partially structured, or non-structured. However, such data is not normalized and convenient for direct application. Big data should be used for its processing.

Big data [2] is an umbrella term that often refers to a process of applying computer analytics to massive quantities of data in order to discover new insights and improve decision-making. It often describes data sets that are so large in volume, so diverse in variety, and moving with such a velocity that it is difficult to process it using traditional data processing tools. Big data construes a basis for business intelligence. Business intelligence refers to the set of technologies and applications that transform crude data into operational insights that may improve decision-making and business performance in development of SWS with the minimum quantity defects and vulnerabilities.

The SWS development is a manufacturing sector. It stores more data than any other sector. As a result, manufacturers have a lot to gain from better use of data to boost efficiency, drive quality, and improve the way products are designed, composed, and distributed. According to estimation made in [3], better use of data in manufacturing may yield up to a 50 percent decrease in product development time and assembly costs. In fact, IDC estimates that manufacturing companies that take full advantage of their data are poised to achieve a $371 billion data dividend within four years. Companies that use data-based decision-making report about 5% to 6% boost in productivity [4]. Using big data, companies may also better track and manage global supply chains, and reduce product defects and vulnerabilities. In works [5, 6] the achieved results and numerous problems of research directions for engineering big data analytics software are reviewed. Summarizing the aforesaid, it is necessary and expedient to use big data analysis as a basis for the SWS dependability assessment methods.

Data located in the big data storages is a great and insufficiently used resource for SWS dependability evaluation, prediction and management. At the same time big data based software dependability assessment may be applied using methods of data filtering and processing, as described in [7], such as: Data cleaning, Classification, Clustering, Frequent Pattern Mining, Probabilistic & Statistical Methods, Anomaly & Outlier Detection, Feature Extraction, Selection and Dimension Reduction, Mining with Constraints, Mining Unstructured and Semi Structured Data, Mining Complex Datasets. Review of processed data enables to evaluate and predict dependability indexes and adopt efficient solutions aimed to improve SWS dependability. The solutions enable to answer such questions as which artefacts should be verified, applicable sequence and duration of verification, what modules should be exposed to re-factoring, what test-cases should be applied, where the required dependability should be admitted as obtained, number of technical support experts to be assigned, etc.

Necessity arises to find similar data with already known dependability indexes in big data storages. They include research data [8], open data of major public bodies [9], corporative data [10], data of internet communities of developers [11]. Accumulation and systematizing of required data enable an individual software corporation to form a data field for dependability assessment and prediction. The essential problem is to find data for dependability evaluation of particular SWS being under development in huge storages.

This chapter proposes and discusses a common approach and techniques to solve some tasks related to Big Data Analysis (BDA) based assessment of software dependability. Road map of the chapter is the following:

– Firstly in paragraph 50.2 we describe methodology of technology oriented assessment of software dependability including a concept, principles and tasks. We called this methodology TOAStRaS (Technology Oriented Assessment of Software Reliability and Security);

– Secondly in paragraph 50.3 we describe metric-based method of software requirements correctness improvement including motivation in TOAStRaS context, principles of software reliability management, standards and terms applicable at the stage of requirements construction, overview of requirement metrics needed for complexity evaluation, development and application of requirements complexity metric (RCM), algorithm of requirements correctness improving, case study and conclusions;

– Thirdly in paragraph 50.4 we describe complexity-based prediction of faults number for software modules ranking before testing including motivation in TOAStRaS context, overview of related works, approaches to reliability assessment and management, checking of assumption for the complexity-based prediction technique, the experiment performance technique, results analysis and conclusions;

– Fourthly in paragraph 50.5 we describe BDA based assessment of software reliability including motivation in TOAStRaS context, software systems similarity principle, software agent for search of similar programs, technique of search of similar programs, case study, results analysis and conclusions;

– Fifthly in paragraph 50.6 we describe BDA based assessment of software security including motivation in TOAStRaS context, analysis of software vulnerabilities, detection method of software vulnerabilities and features of the assessment;

– In summary of this chapter in paragraph 50.7 we describe common conclusions and next steps of the research.

50.2. Methodology TOAStRaS of technology-oriented assessment of software dependability

Strategic aim of the research consists is the SWS dependability improvement in restricted resources conditions by means of embedding methods and tools of dependability evaluation and control into lifecycle processes. The work is aimed to develop concepts, principles, tasks and elements of methodology for technology-oriented evaluation of the SWS dependability. The proposed approach is titled as Methodology of Technology Oriented Assessment of Software Reliability and Security (TOAStRaS). This methodology is based on general concept and a few principles (Figure 50.1).

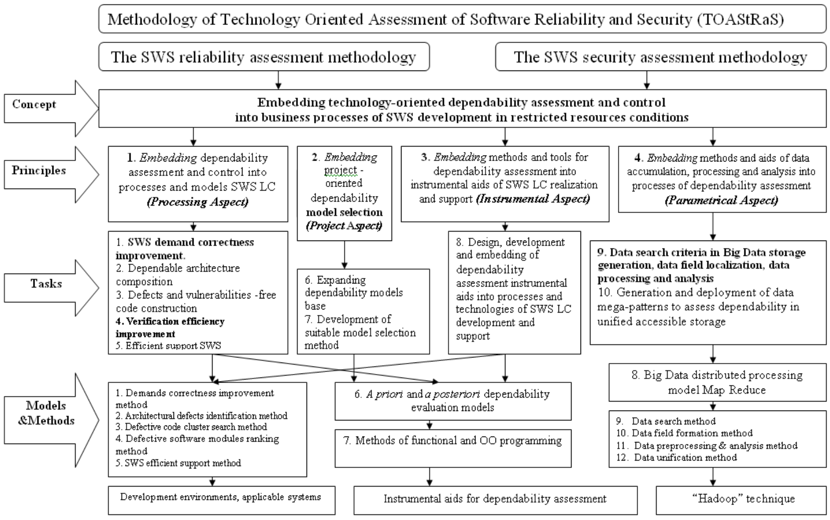

Methodology of Technology Oriented Assessment of Software Reliability and Security (TOAStRaS)
The SWS reliability assessment methodology
The SWS security assessment methodology
Concept
Embedding technology-oriented dependability assessment and control into business processes of SWS development in restricted resources conditions
Principles
1. Embedding dependability assessment and control into processes and models SWS LC (Processing Aspect)
2. Embedding project - oriented dependability model selection (Project Aspect)
3. Embedding methods and tools for dependability assessment into instrumental aids of SWS LC realization and support (Instrumental Aspect)
4. Embedding methods and aids of data accumulation, processing and analysis into processes of dependability assessment (Parametrical Aspect)
Tasks
1. SWS demand correctness improvement. 2. Dependable architecture composition 3. Defects and vulnerabilities -free code construction 4. Verification efficiency improvement 5. Efficient support SWS
6. Expanding dependability models base 7. Development of suitable model selection method
8. Design, development and embedding of dependability assessment instrumental aids into processes and technologies of SWS LC development and support
9. Data search criteria in Big Data storage generation, data field localization, data processing and analysis 10. Generation and deployment of data mega-patterns to assess dependability in unified accessible storage
Models &Methods
1. Demands correctness improvement method 2. Architectural defects identification method 3. Defective code cluster search method 4. Defective software modules ranking method 5. SWS efficient support method
6. A priori and a posteriori dependability evaluation models
7. Methods of functional and OO programming
8. Big Data distributed processing model Map Reduce
9. Data search method 10. Data field formation method 11. Data preprocessing & analysis method 12. Data unification method
Development environments, applicable systems
Instrumental aids for dependability assessment
“Hadoop” technique

Figure 50.1 - The structure of methodology TOAStRaS

The tasks of the research are, as follows:

1. Development of concepts, principles, tasks and elements of methodology for technologically oriented evaluation of the SWS dependability;
2. Development of metric-based method for software requirements correctness improvement;

3. Development of complexity-based prediction of faults number for software modules ranking before testing:
4. Development of search method in big data storage to find experimental data of similar SWS to evaluate dependability indexes;
5. Big data analysis based assessment of software security.

50.2.1 Concept

We offer embedding technology-oriented dependability assessment and control into business processes of SWS development in restricted resources conditions. Embedding technology-oriented dependability assessment includes the SWS reliability assessment and the SWS security assessment. The TOAStRaS methodology is based on big data search and analysis. Big data search focused on a similarity of properties and characteristics of the estimated SWS and the found systems. The found systems have known characteristics of the sizes, structures, complexity, architecture and configuration, direct or indirect indicators of reliability and security which are known by results of the expert estimation, testing, verification and application. The SWS developers form predictive estimates of software reliability and security on the basis of these known indicators.

The general approach generates several questions to estimation of such SWS properties as reliability and security:

a) How to formulate and prove sufficiency of signs of similarity? Signs of similarity are various for reliability and security of SWS.

b) How to describe and narrow a set of the analyzed storages and data sources;

c) How to carry out search, comparison and selection of similar systems;

d) How to process information of similar systems and to calculate necessary assessment reliability and security? Indicators differ for reliability and security significantly. Defects and faults are important indicators for reliability estimation. However it is necessary to know quantity and types of vulnerabilities for security estimation.

50.2.2 The Principles

Principle 1. *Embedding evaluation procedures and* dependability *management procedures into SWS life cycle processes and models.*

The principle means embedding into processes of requirements analysis, architectural design, realization, verification, and maintenance, as specified in the ISO/IEC 15288:2015, Systems engineering — System life cycle processes.

The principle further supposes embedding models, as enlisted: V-model, XP, SCRUM, Rapid Application Development, Dynamic Systems

Development Method, Rational Unified Process, Microsoft Solutions Framework, Kanban Development, Cleanroom Software Engineering, Waterfall model into SWS life cycle processes. This principle implements *Processing Aspect* of solving problems of SWS dependability evaluation and control. It enables to evaluate and control dependability of generated artefacts at various stages of SWS life cycle.

Principle 2. *Embedding project-oriented selection, complexion and parameterization of models to evaluate achieved* dependability *indexes at the final stage of the SWS development.* Implementation of this principle supposes applied models' base expansion and their allowances; development of models selection, complexion and parameterization methodology. This principle reflects the *Project Aspect*.

Principle 3. *Embedding* dependability *evaluation methods and instrumented aids into instrumental tools of implementation and supporting processes of SWS life cycle*. This principle supposes design and development of software dependability evaluation aids for their flexible embedding in design environment and control systems in artefacts operating within SWS life cycle. The principle also supposes usage of existing standard software applications for functional dependability evaluation. This principle reflects *Instrumental Aspect*.

Principle 4. *Embedding methods and aids of data search, accumulation, storage, processing and analysis applied in similar projects and systems into SWS under development,* dependability *evaluation processes using big data storages, principles and technologies.* Realization of this principle supposes search of necessary data in *big data* storages, generation of broad corporative data field for dependability evaluation, data accumulation and storage in cloud storage, *big data* processing and review applying specialized techniques, using resulting data for dependability assessment *(Software dependability big data*), and, finally, making data usage convenient *(Software dependability big data usability).* This principle implements *Parametrical Aspect*.

A sequence of tasks is proposed below for realization of above principles.

50.2.3 The tasks

The following tasks should be solve to implement TOAStRaS methodology.

Tasks implementing principle 1:

Task 1 – Requirements correctness improvement (method taking into account priorities and complicatedness of requirements embedded into SWS LC);

Task 2 – Building dependable architecture (UML-models, mathematical models, method of “bottleneck” and security gap search in architecture, embedding into SWS LC processes);

Task 3 – Defects and vulnerabilities - free code generation (simulating models, defective and insecure clusters search method within code, embedding into SWS LC processes);

Task 4 – Achievement of efficient verification (method of software modules ranking taking into account their complicatedness, embedding into SWS life cycle processes);

Task 5 – Efficient support («rear bench technique» - bringing changes only into essential functions, putting support off obsolete SWS functions).

Tasks implementing principle 2:

Task 6 – Expanding applicable models’ base and their allowances;

Task 7 – Development model selection method based upon setting priorities for allowances and numeric evaluation of their practical implementation extent. The proposed evaluation technique enables to obtain more precise numeric evaluation for various models meeting the needs of individual project.

Task implementing principle 3:

Task 8 – Development and embedding instrumental aids of dependability assessment into technologies of SWS life cycle development and support.

Tasks implementing principle 4:

Task 9 – forming search criteria, methods and software aids in *big data* storage of SWS experimental data similar to SWS being developed; processing and review of this data; generation of corporative data field for dependability evaluation based on this data;

Task 10 – Creating data mega-patterns to assess dependability (unified data pattern and necessary context data); placement of this data in unified data storage for common access.

We offer metrics-based method of software requirements correctness improvement within the framework of proposed concept for the solution *of the first task* (paragraph 50.3).

We offer complexity-based prediction of faults number for software modules ranking before testing: technique and case study within the framework of proposed concept for the solution *of the fourth task* (paragraph 50.4).

We offer big data search technique for similar system to assess reliability of SWS being in development within the framework of proposed concept for the solution *of the ninth task* (paragraphs 50.5 and 50.6).

In paragraph 50.7 we offer common conclusions and next steps of the researches.

50.3. Metric-based method of software requirements correctness improvement

Nowadays humanity utilizes a tremendous amount of software tools and software systems for different domains and applications. Customers impose more complex and diverse requirements. Software produces more and more considerable influence on our life and our safety. In its turn, complexity of SWS increases [12]. Consequently, number of defects imposed by developers increases, the SWS reliability decreases, design costs rise. Huge losses and the decrease in the efficiency of human activity caused by the imperfection of SWS and design faults is one of the key challenges in modern IT engineering encouraging increased software reliability [13].

Hence, to implement TOAStRaS problems of reliability assessment and management should be analysed and discussed. Secondly, in view of importance of software requirements correctness from the point of view of the SWS reliability we propose technique for assessment of complexity and improving of requirements' correctness.

50.3.1. Principles of Software Reliability Management. State of the Art

Researches worldwide developed a number of models, methods and software application to evaluate and manage the SWS reliability [14-22]. However, these methods don't take into due account variety and specific features of SWS development business processes and require great expenditures for adaptation and implementation. The above factors prevent to implement widely available model methods and reliability aids in software engineering routine practice, making such implementation impossible in a number of cases. Therefore, the first principle of the software reliability methods (SRM) provides adaptability, simplicity and low costs contributing to easy implementation of the assessment technique into business processes of software corporations.

The SWS lifecycle (LC) SWS consists of a sequence mutually dependent processes. These are stages of requirements' construction, design, codification, testing and the SWS maintenance. At these stages experts of various profile and qualifications create unique artefacts. They are requirements' specification, architecture and detailed SWS design, initial codes, testing procedures and versions, ready software product. The developed SWS reliability is pre-determined by correctness of each individual artefact, depends on its complexity and forms at each stage of its LC. However, existing reliability models don't take into sufficient account complexity and evaluate reliability at each LC stage fragmentarily. Therefore, the second SRM principle provides thorough complexity evaluation and artefacts reliability management throughout the entire SWS LC.

Different production data accumulates throughout the SWS LC stages. It includes user demands in textual, tabulated, graphical and verbal formats, architecture and design descriptions, algorithms, initial software code, testing scenarios, incoming data massive, video files of the SWS behaviour under various conditions, technical documentation, etc. This multi-type structured and non-structured data of various SWS versions and assemblies are stored and actively used by developers during the long term of operation. This data, also called as big data may be measured in terabytes. Methods and applications for big data processing and analysis to improve SWS reliability are not sufficiently developed. Therefore, the third SRM principle provides for development big data analysis methods and application for SWS LC.

Competitive market compels software corporations to reduce SWS development costs with sharp need to provide the demanded reliability level. The developers often face the problem how to spend limited resources in the most efficient manner? What artefacts subsets should be verified in the available resources don't allow to afford complete verification? What particular pages of e.g. 200 available should be checked to identify maximum of errors, controversies, inaccuracies or incompleteness? What of thousands modules should be tested to detect maximum number of defects? So, the fourth SRM principle consists in maximum possible reliability improvement under restricted resources condition.

Each SWS development starts from collection and analysis of requirements. This process includes clarifying and documenting what should be done by an application and what features it should possess. This process involves all the concerned parties – investors, purchasers, users, from the one side, and managers, analysts, designers, codifiers, testers, i.e. SWS developers, at the other side. In the course of extended joint work errors may occur with any party category at both sides. As a result, incorrect requirements contain defects of various types, such as erroneous data, unclearness, ambiguities, incompleteness, doubling, and contradictions. Imperfect requirements may cost very much. As source [23] marks, requirements defects are 20-50 times more expensive to correct once the defects got into development process. Inspection of one full-scale software project showed that problems with requirements led to US$ 600.000.000 development budget exceeding, 8 years time delay and reduced functionality. Another research [24] showed that it takes about 30 minutes to correct error detected at the stage of requirement processing, whereas it takes 5 – 17 hours to correct an error detected in the course of system testing. As identified in the [25] eliminating requirements' defects are 100 times expensive than those detected in the course of requirements development, if indicated by the customers. Evidently, methods directed to identify as much defects, as possible, in requirements saves plenty time and costs. Therefore, method of improvement of SWS requirements correction is suggested. Below standards and terms applicable at the stage of requirements construction are described.

50.3.2. Standards and Terms Applicable at the Stage of Requirements Construction

Standards provided in [26, 27] lost their actuality and are inapplicable. Actual standard is IEEE/ISO/IEC 29148-2011 "Systems and software engineering – Life cycle processes – Requirements engineering" [28]. Terms definitions are enlisted in compliance with the said standard.

SWS project (software system project) – a full set of developed agreed artefacts including requirements specification, architecture and detailed design, initial code, test procedures and versions, technical documentation, etc.

Requirements are a document specifying users' needs and aims, condition, features and capabilities of the software product.

Concerned parties are individuals, groups and/or organizations taking an active part in the project.

Requirements analyst is a person in developers' team in charge of the requirements extraction, analysis, definition, approval and their management.

Requirements development is a process aimed to determine project scope, users' class and representatives, requirements' extraction, analysis, specification and approval.

Requirements analysis is process of categorizing data concerning requirements, requirements evaluation to determine desirable quality, requirements representing in various formats, detailed requirements extraction from requirements of higher levels, requirements priority discussion, etc.

Software requirements specification (SRS) – result of a system requirements documenting in structured, accessible and controllable format. According to source [29], requirements specification is a set of technical documentation containing detailed SWS under development and its functionality, requirements to external interfaces, non-functional requirements, glossary, processes and subject zone models, other SWS data, as may be required by parties in concerned.

Requirements attributes – description data of requirements with unique requirements numbers, data sources, logical groundings, priorities, persons in charge, SWS versions numbers, etc.

Requirement's elements – any elements composing the requirement – constant values, variables, arithmetic and logical operations, etc.

Requirement complexity – difficulties met in understanding and realization of a requirement related with great quantity of components and links with other requirements and business rules.

Business rule – policy, guiding principles or provisions which determine or restrict certain aspects of business activities.

Requirement priority– numerical and/or quality evaluation of requirement attribute, taking into account importance and term of its realization. According to source [29], priorities construe a way to solve conflict between the requirements competitive struggle for restricted resources.

Requirement metric – mathematical representation of numerical evaluation of a particular requirement's feature.

50.3.3. Method of SWS Requirements Correctness Improvement

Modern SWS include plenty multi-type requirements. They include functional requirements, requirements to user and software interfaces, hardware interfaces, communications interfaces, requirements to capacity, data safety storage, system safety, etc. Therefore, the SWS specification is a bulky document of hundreds pages. Theoretical approach supposes total review of all these requirements. In practice, "even in medium-size specification experts review thoroughly only the first part, some of the most stubborn may look through half, but it is unlikely that anyone reaches the end" [29]. Entire SWS requirement review is impossible due to lack of time, financial and human resources. In this view, the offered method or requirements correctness improvement consists in extracting the most priority and complicated requirements from their entire set to be thoroughly reviewed to detect the highest possible quantity of defects in prevailing restricted resources situation. Priorities are assigned to requirements after their collection and documenting jointly by the manager, analyst and customers' representatives. Requirements complexity is determined by means of metrics.

The method is based on the following procedures:

1. To define the priorities of requirements fulfilment;
2. To estimate complexity of requirements by means of a specific metric;
3. To sort requirements in the account of complexity estimates for each priority;
4. To verify requirements possessing a high priority and complexity grade;
5. To define requirements correctness index and make the reasonable decision on completion or continuation of requirements verification.

50.3.4. Overview of Requirement Metrics

Publications [23, 29, 31, 32] and standards [28, 30] have been studied for analysis of existing metrics. These standards contain numerical evaluations for about 50 metrics. They include metrics for completeness; precise detailed requirements; non-elementary requirements; requirements not coordinated with other requirements; undetectable and untestable requirements; missed or skipped requirements; non-obvious requirements; defective requirements detected for 1 hour of checking; detailed requirement value; metrics for deleting, adding or modification of the requirement. Their analysis shows that such metrics are designated for quality control, requirements review and analysis efficiency. However, there is no metric evaluating requirement complexity in these standards.

Authors of work [23] apply requirement defect metric as a number of defects per documentation page, provide its interpretation and permissible value as 0.2 defects per page of the requirements' testing description, i.e. one defect per 5 pages. Unclear definition since a number of requirements per page is unknown.

Authors of work [29] stress upon correctness and traceability as the most important feature of requirements without outlining evaluation formulae. The authors define correctness as a precise description of desirable functionality. Traceability is defined as a possibility of analysis both in backward direction to initial sources of requirements and forward to design elements, initial code, tests, scope of application and other SWS elements. The elements include requirements of different types, business rules, architecture and design components, initial code modules, test versions, reference files, traceability is of great importance for requirements management. It enables to determine all the workable artefacts which should be modified once the requirement changes. As per complexity, the authors mark complexity of requirements detection process, complexity of their statement in foreign language, complexity of understanding and technical implementation of requirements with complicated Boolean logics (combination of operators AND, OR, NOT) having, however, refrained from providing numerical evaluation aids for these features.

Authors of [31, 32] propose more advance set of metrics. In work [31] they are numerical metrics of unique, understandable, misinterpreted, modified requirements and requirements under testing. The authors also propose two newer metrics, closer in sense, but different in evaluation manner:

1) correctness $MC = \frac{R_c}{R_t} * 100\%$, with R_c - number of requirements received the same true interpretation by more than 1 referee; R_t - total quantity of requirements;

2) quality $MQ = \frac{R_t - (R_f + R_d + R_m)}{R_t} * 100\%$, with R_f - number of identified and corrected requirements with defects, R_d - number of deleted requirements with defects, R_m - number of modified requirements.

Authors of work [32] added metrics of ambiguity, logical linkage and capability to check the requirements starting from the initial source. Yet, the work does not contain complexity metrics.

Thus, the review of publications and standards showed the lack of metrics enabling to evaluate requirement complexity. The referred enlisted metrics describe construction process and various requirements features, saving the complexity. With such metric missing the most complicated requirements cannot be identified and selected for inspection and review to detect higher defects quantity with prevailing restricted resources situation. The [29] notes that the requirement complexity is evaluated by expert in a subjective manner. Analysis results and author's personal experience in SWS development enable to state that requirements complexity is not numerically evaluated in the software engineering. Evaluations are not applied to improve requirements correctness and the SWS reliability. Therefore, it is necessary to develop a metric for precise numerical evaluation of requirements' complexity.

50.3.5. The Metric for Requirement Complexity Evaluation

The new metric should take into account internal and external complexity inherent with the requirement. Internal complexity is determined by quantity of components. External complexity is represented by links connecting a particular requirement with other requirements, business rules, etc. These links are illustrated at figure 50.2 representing an entity relationships (ER) diagram of logical contacts of the requirement with other elements.

The ER – diagram shows that the requirement may include one element, or more; may affect, or does not affect several dependent requirements; may depend, or not, on some affecting requirements; may depend (or not) on some affecting business rules; may depend, or not, on some affecting entities, strictly specified for each SWS being developed. The higher is quantity of elements in requirement and links of the requirement with other requirements, business rules and other entities, the higher is the requirement and its complexity rate by metric.

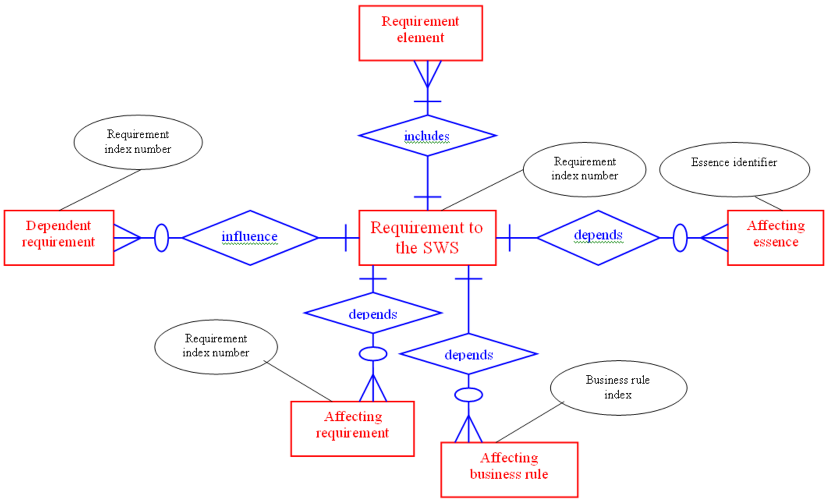


Figure 50.2 - ER – diagram of logical interconnections of the requirement with other components

Table 50.1 contains proposed requirement complexity metric structure and examples of its evaluation. Data required to evaluate requirement complexity by metric is, as follows:

1. Individual requirement index within the system being under development;
2. Number of elements composing the requirement (e.g, to calculate tax rate (variable 1st element) multiplied (operation 2nd element) by 18% rate (constant value, 3rd element) from income value (variable, 4th element) deducting (variable, 5th element) maternity leave (variable, 6th element) and amount payable under sick leave (variable, 7th element);
3. Number of affecting business rules (e.g. global – currency exchange rate, regional – applicable tax rate, corporate – terms to earn quarter bonus in the corporation);
4. Number of affecting and dependent requirements;
5. Number of other affecting entities (e.g. hardware capacity, network traffic speed).

Table 50.1 - Requirement Complexity Metric Structure

Requirement Complexity Characteristics					RCM 6=2+3+4+5
Unique requirement index	Internal requirement complexity	External requirement complexity			
	Number of components composing the requirement	Number of affecting business rules	Number of affecting and dependent requirements	Number of affecting and dependent entities	
-1-	*-2-*	*-3-*	*-4-*	*-5-*	*-6-*
F63	2	1	5	1	9
F64	3	0	2	2	7
F65	2	0	2	0	4
F66	5	0	1	2	8
F67	2	0	1	0	2
F68	4	3	0	1	8
Mean requirements' complexity					6,3

Mathematical representation of the requirement complexity evaluation by metric (requirement complexity metric, RCM) looks, as below:

$$RCM = EQ + IBRQ + IDRQ + IDEQ \qquad (50.1)$$

EQ – the elements quantity (column 2), *IBRQ* - the influencing business rules quantity (column 3), *IDRQ* – quantity of the affecting and dependent requirements (column 4), *IDEQ* - quantity of the influencing and dependent entities (column 5).

Such is the proposed requirement complexity evaluation metric. Calculation of metric consists in summing up the numerical characteristics in internal and external complexity of a particular requirement. The metric enables to evaluate the requirement complexity in complexity units. Obtained values enable to sort the requirements by their complexity level.

50.3.6. RCM Metric Application

The proposed *RCM* metric has been applied for functional requirements complexity evaluation within a particular corporate system of accounting and analysis. This medium-size system has been developed in Ukraine for the Danube Institute of National University “Odessa Maritime Academy” in 2016. The realized system consisted of 65 components and a database of 26 interlinked tables. Total size of the system amounted to 40000 lines of initial code. System specification encompassed 120 functional requirements. Requirements analysts calculated complexity rates applying the RCM metric and basing on Table 50.1. The rates in question have been obtained in the course of collection, review and documenting the requirements. The evaluation process was rather easily performed in MS Excel and did not demand additional time or additional software and depended only on organization aspect and personal qualities of experts.

Requirements mean complexity RC_{avg} has been calculated being 6,3 complexity units. Total system complexity amounted to $120 \cdot 6{,}3 = 756$ complexity units. The obtained value has been applied to compare complexity of existing and previously developed systems enabling to update development terms and budget. It was also revealed that 15 % requirements had complexity characteristics exceeding 10 units ($RC>10$). The analysis showed that those requirements had been hard enough to be realized. Detailed review of the requirements in question enabled the experts to form a complete idea of the complexity of the system being developed and to involve necessary resources. Certain complicated requirements have been modified or decomposed into easier. Some requirements, after discussion with the customers have been deleted from specification. After these activities total complexity of the system reduced to 83 units, i.e. by 11%.

50.3.7. Algorithm of SWS requirement correctness improving

The offered method is based on logical admission that the more is requirement complexity, the more actual and potential defects it contains, such as contr5adictions, discrepancies, incompleteness, inaccuracies which at further stages of system design may likely become defects. Really, the more elements are assembled into a requirement, the higher is chance of errors in elements. The more entities are interconnected with the requirement, the harder it is to describe the requirement clearly, correctly, to codify it properly and to modify it in the course of the SWS maintenance.

The method may be applied once requirement construction process is completed and priorities of their realizations are assigned. Method utilized the proposed RCM metric to evaluate the requirements' complexity. *The method's incoming data* contains characteristics of each requirement.

1. Unique requirement index;
2. Detailed text description of the requirement;
3. Priority of realization of the requirement *Pr* (high $Pr=1$, medium $Pr=2$, low $Pr=3$).

This data is accumulated in the course of requirements construction and does not demand any additional activities and costs. Method procedure includes seven (7) steps:

Step 1. Full set of requirements should be sorted in the order from higher to lower.

Step 2. Subset requirements with $Pr = 1$ R_{pr1} should be derived from R_{full} set. Complexity values should be calculated using metric RCM according to expression (50.1). The requirements should be additionally sorted according to complexity decreasing criterion. Obtained *List 1* should contain rising rank to each requirement starting from 1 and further to n.

Step 3. Requirements R_{pr1} should be verified. Real and potential requirements' defects should be detected and eliminated (contradiction, mismatching, incompleteness, inaccuracies, in ambiguity).

Step 4. Resources permitting, derive requirements subset R_{pr2} with priority $Pr = 2$. Complexity values should be calculated using metric RCM according to expression (50.1). The requirements should be additionally sorted according to complexity decreasing criterion. Obtained *List 2* should contain rising rank to each requirement starting from $n+1$ and further to k. Further proceed to steps 3, 5, 6, 7. With lack of resources proceed to Steps 6 and 7.

Step 5. Resources permitting, derive requirements subset R_{pr3} with priority $Pr = 3$. Complexity values should be calculated using metric RCM according to expression (50.1). The requirements should be additionally sorted according to complexity decreasing criterion. Obtained *List 3* should contain rising rank to each requirement starting from $k+1$. Further proceed to steps 3, 6, 7. With lack of resources proceed to Steps 6 and 7.

Step 6. Requirements Correctness Evaluation. Having completed the verification quantity of requirements should be counted by categories, including total verified R_{total}; defective, fixed R_{fix}; defective deleted R_{del}, potentially defective, modified R_{mod}; skipped R_{gap}. The *requirements defectiveness* index **RDI** may be calculated using the formula (50.2) below

$$RDI = \frac{R_{fix} + R_{del} + R_{mod} + R_{gap}}{R_{total}} \tag{50.2}$$

The requirements correctness index, **RCI,** should be calculated using the formula (50.3) below:

$$RCI=1-RDI \tag{50.3}$$

Step 7. Requirements correctness improvement. Comparing index values of *RCI, RDI* with known under previous projects (if available) and/or permissible corporative values. Referring to the comparative results conclusion should be drawn about achieved requirements' correctness and finalizing or further proceeding with their construction stage. E.g. permissible corporative value for $RCI_{norm} = 0.92$ with $RDI_{norm} = 0.08$, i.e. not more than 8 defective of 100 total requirements are permitted. Should $RCI \geq 0.92$, the requirements are correct enough and fit for further development stages. Otherwise

(RCI < 0.92), the requirements are not correct and demand re-proceeding and repeated performance of the procedure under offered methodology.

This step finalizes the method. Table 50.2 shows an example of ranking the requirements.

Table 50.2 - Requirement ranking example

Individual Requirement Index	Requirement Priority	Requirement Complexity Grade by Metric (1, ∞)	Requirement Rank (1, ∞)
F63	1	9	1
F64	1	7	2
F65	2	4	4
F66	2	8	3
F67	3	5	6
F68	3	8	5

The top rank for requirements with priority 1 is caused with the highest evaluated complexity rate (complexity rate = 9, rank = 1). As complexity rate decreases the rank decreases as well (complexity rate = 8, rank = 2). The highest rank for requirements with priority 2 should be equal to next after last assigned rank. The top rank for priority 2 requirements is equal to the rank with index following the last assigned rank. It is assigned to the requirement with the highest complexity index (complexity index = 7, rank = 3). Figure 50.3 shows the UML-diagram of activities under the method. The UML diagram represents logical consistence of actions within the method.

After the method is completed the final data is obtained as a through *ranked list of requirements and defectiveness and correctness indexes*. The through ranked list enables to perform individual verification of the top priority and most complex requirements and to increase quantity of detectable potential and actual defects. Defectiveness and correctness indexes indicate either requirements' fitness to further stages of processing, or necessity of their reprocessing with further repeated verification. The method enables to reduce defectiveness and to improve correctness of requirements with prevailing restricted resources situation.

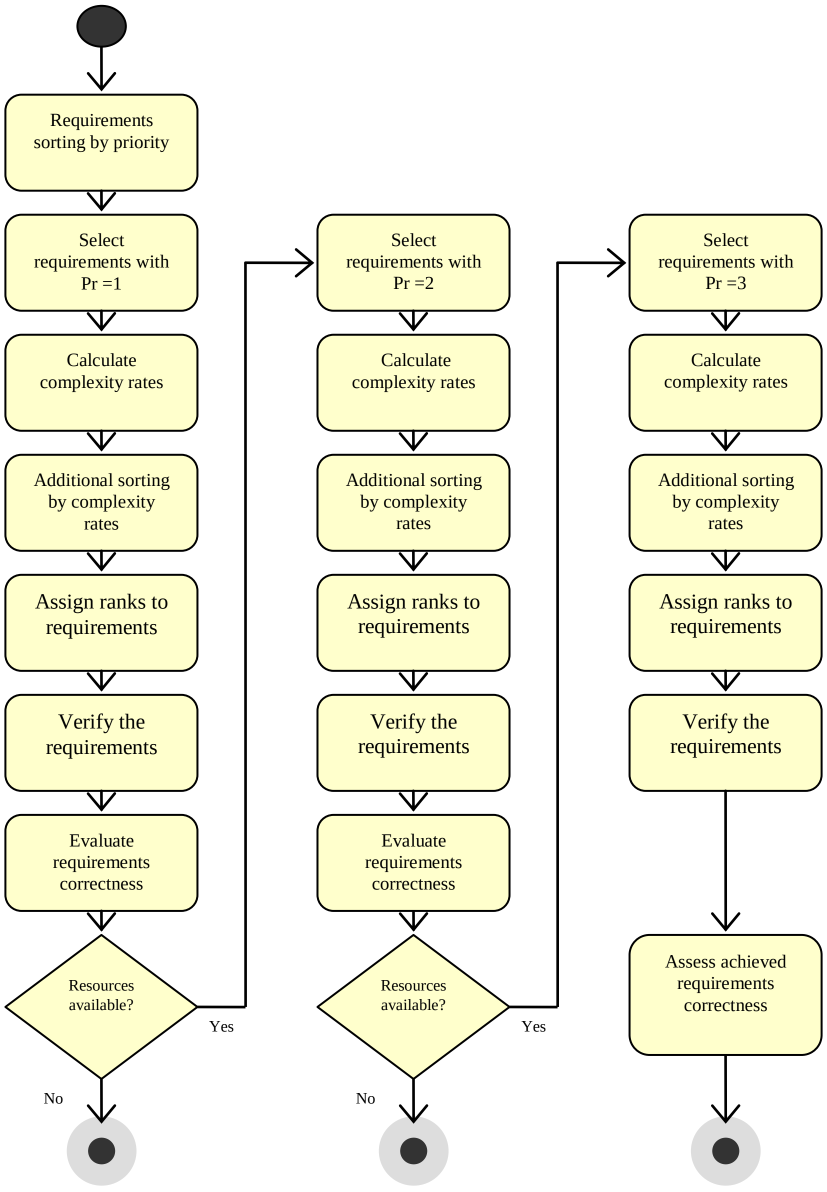

Requirements sorting by priority
Select requirements with Pr =1
Calculate complexity rates
Additional sorting by complexity rates
Assign ranks to requirements
Verify the requirements
Evaluate requirements correctness
Resources available?
Yes
No
Select requirements with Pr =2
Calculate complexity rates
Additional sorting by complexity rates
Assign ranks to requirements
Verify the requirements
Evaluate requirements correctness
Resources available?
Yes
No
Select requirements with Pr =3
Calculate complexity rates
Additional sorting by complexity rates
Assign ranks to requirements
Verify the requirements
Assess achieved requirements correctness

Figure 50.3 - UML - diagram of activity of SWS requirements correctness improving technique

50.3.8. Case study and conclusions

The offered method has been applied in the course of inspection of functional requirements of corporate system, as referred above. Through ranked script of the requirements of top priority and the highest complexity has been applied. Methodology enabled to increase quantity of detectable actual and potential defects by 9% in average in comparison with total verification of the requirements disregarding their complexity, improve requirement correctness to the same extent and to avoid sensible costs which might be incurred to eliminate such defects at further stages of development.

The most important SRM principles have been outlined in this work:

1. *Compatibility* of models methods and technologies with SWS development business processes;
2. *Complexity evaluation* for SWS development process and artifacts reliability management throughout the SWS LC;
3. Development of methods and applications for analysis of *big data* SWS LC;
4. Maximum possible reliability improvement with *restricted resources* situation.

Requirement complexity evaluation metric is offered within the SRM framework. The metric calculation consists of summing up quantitative characteristics of internal and external requirement complexity. The metric evaluates requirement complexity in complexity units. The obtained evaluation enables to determine total system complexity and adjust terms and costs of development. Modifications or decomposition of complex requirement enables to reduce total system complexity.

Method of improving requirements correctness has been offered within the SRM framework. The method applies an admission that more complex requirements contain more actual and potential defects. The method applies RCM metric to evaluate requirements complexity. The method applies a double sorting procedure basing on requirements' priority and complexity evaluation. The method is advantageous comparatively to the total verification due to possibility to select the most complicated requirements with top priority by means of generating their through ranked script. The method enables to improve requirements' correctness and defectiveness. The method enables to evaluate achieved indexes of correctness and defectiveness. The application of the method in the course of requirements verification provided substantial technological and economical effect.

The method has been developed taking into account two principles: the principle of artefacts complexity evaluation and the principle of maximum possible reliability improvement in restricted resources situation. Further researches are planned in two directions. The first lies in achieving and providing organic building-in with different SWS development business processes by means of making method patterns with defining roles and activities of the developers. The second direction lies in development of methods and applications for big data analysis occurring at all stages of the SWS LC.

50.4. Complexity-based prediction of faults number for software modules ranking before testing

Software system reliability (hereinafter referred to as the System) substantially depends on its complicatedness. Complicatedness of multiple and mutually related requirements leads to formation of erroneous, incomplete, contradicting specifications. System's architecture complicatedness causes fatal errors (critical faults) generations, which may result to project ruination. Complicated system implementation leads to errors' increase in the initial code. Various sources show that testing process reveals 10 – 20 errors per each 1,000 code lines as an average. Changes in initial code are required at the stage of system operation contributing to newer errors generation. Integrated approach is therefore essential to access and ensure reliability basing on system complicatedness evaluation throughout its lifetime. Actual software indicators/ metrics should be measured thoroughly and continuously to ensure reliability and handle development and verification procedures. Increasing systems' complicatedness, from the one side and limited resources, from the other side, generate contradictions between required systems' reliability and that achieved once development is completed. It is therefore critically important to develop and to research reliability assessment methods basing on system's complicatedness analysis, both in general and its components taking into note limited testing resources

50.4.1 Motivation of the complexity-based prediction in TOAStRaS context

The TOAStRaS metodology assumes embedding technology-oriented dependability assessment and management into business processes of SWS development in restricted resources conditions. Embedding technology-oriented dependability assessment includes the SWS reliability and security assessment.

Reliability and security estimates assume the quantitative predicting of faults and vulnerabilities. The complexity of the source code SWS is the significant cause of faults and vulnerabilities. The most complex modules contain the greatest number of the faults. The complexity of a code is estimated on the basis of metrics by means of specific software metrics tools. The complexity estimates can be used for modules ranking. Such ranking will help to reveal modules with great number of the probable faults. It is necessary to range modules prior to their verification. In this case limited verification resources will be used most effectively.

It will allow to reveal the maximum faults number, to increase reliability and security of SWS. The rise of SWS reliability and security means increasing of dependability SWS at source code verification stage by using limited resources. The special software tool for modules ranking will provide embedding technology-oriented dependability assessment and management into business processes of SWS development. The most effective verification will provide dependability management at this development stage.

Thus, the first principle of methodology TOAStRaS will be realized. It consists of the embedding evaluation and dependability management procedures into SWS life cycle processes. This principle means embedding into realization and verification processes of SWS development, as it specified in the ISO/IEC 15288:2015, Systems engineering — System life cycle processes. The first principle implements processing aspect of solving problems of SWS dependability evaluation and management. The third principle will also be realized. It consists of the embedding prediction methods and instrumented aids into instrumental tools of supporting processes of SWS life cycle. The third principle reflects an instrumental aspect.

50.4.2 Related works

Systems' reliability should be continuously controlled throughout their entire lifecycle. Therefore the process of required reliability progress should be controlled at all stages. Any process management demands indicative values evaluation. Multiple number of models, methods and tools are developed to evaluate various reliability indicators at various lifecycle stages.

Works [33-37 propose various existing models and methods to increase precision of evaluation of reliability indicators. In work [33] the authors analysed the impact of debugging time upon reliability evaluation and forecast. The debugging time causes an overestimation of the perceived software quality up to 15% in studied dataset; similarly, it causes the underestimation of testing time required to obtain a given quality for a software product.

Work [34] proposes a method based on matrix allowing known reliability models. Applying matrix enables to 1) form an allowance vector for software system under development to take into account development process peculiarities; 2) to choice appropriate model basing on allowances vector; and 3) calculate model parameters. However, the allowances matrix requires adding newer models and methods.

The most of expenditures are associated with the testing stage. Here the bulk of defects are also revealed. Authors of work [35] prove that applying combined operation and debugging tests (OP-D) ensures, in general, addressing both faults occurring with high frequency at the operation time and those occurring with lower frequency. The objective of proposed OP-D technique lies in reliability improving only by means of operation testing revealing, at the same time, as many bugs, as possible, as may be achieved via debugging application. However, the proposed OP-D technique is unable to take into account rigid restrictions in testing resources.

Work [36] has been devoted to the basic theory of the of software systems' dynamics and established the theoretical basis for the reliability assessment of and proposed a new universal method for such assessment. This method takes into account effects of secondary faults, and improves the accuracy of software reliability indices more than twice. Proposed models and methods require experimentally obtained data of faults revealing time in the course of testing. The more data becomes available, the more precise are reliability evaluating indicators, shorter is the testing period, and the less are resources required to achieve required reliability level.

Work [37] describes software reliability model with complexity index based on the non-homogenous Poisson process (NHPP). The method of software reliability assessment has been developed on the basis of generalized NHPP model and the testing sufficiency criteria. Software application for software failures prediction using artificial neural networks has been developed. However, reliability forecasting by means of neural networks binds developers to use software known not so well and to study network characteristics. Complicatedness, reduced studying rate and high level of operation margin prevent this method from implementation into routine engineering practice.

It is statistically established that revealing and elimination of faults at the earlier stages of development are 10 – 100 times cheaper, than same actions performed at the ready product testing stage. Reliability indicators should be evaluated prior testing commencement to save expenditures. Developers should be aware of faults quantity in the software system. This knowledge enables to draw up testing process and to link it with available resources.

However, this data is insufficient. It is more important to know which modules contain the highest quantity of faults. It enables to minimize testing group efforts and to maximize faults revealing which plays an essential; part in restricted resources situation. Works [38] through [45] propose certain methods of code complicatedness evaluation by means of metrics.

Work [38 describes analytic model to establish relation between faults quantity in the initial code and complicatedness indexes basing upon developed model. Application and statistical analysis of the proposed method showed that discrepancy between actual faults quantity and estimated one amounted to 11%. The work researches methods of faults localization in software modules. As a result, 9% discrepancy between obtained indicators and actual quantity of faults has been found. However, the problems associated with choice and ranging of the fault prone software modules still has not been solved.

The method proposed in [39] allows to combine techniques so as to maximize the number of faults revealed for the tested software from those expected to be available. As for fault types the method refers to well-known orthogonal defects classification (ODC). As for testing techniques, the method applies techniques of functional, statistical, robustness and stress testing. The final result is a forecast of faults quantity of each ODC category each technique is capable to detect for a particular application. However, it still remains a question which modules should be tested if limited monetary and time resources make it impossible to carry on total testing.

Solutions for the problem in question are proposed in works [40] through [45]. Authors of work [40] developed three individual models forecasting fault-proneness in data set: one with ascending stepwise logistic regression, one with descending stepwise logistic regression and one without stepwise selection in logistic regression. The authors concluded that descending stepwise regression provides the best model. The level of false alarm rate is too high in all the models, while it should be contrary.

Complexity metrics in predicting fault-prone software modules have been intensively studied in the work [41]. The binary logistic regression method is applied in studying using as an example commonly available data on five commercial products. The study shows, (1) models generated using more data sets may improve the prediction accuracy but not the recall rate; (2) reducing the cut-off value may improve the recall rate, but the number of false positives will increase, resulting in higher maintenance efforts.

The authors of work [42] studied, whether metrics available in the early lifecycle (i.e. requirement metrics), combined with metrics available in the late lifecycle (i.e. code metrics), may be used to identify fault prone modules using genetic algorithm technique. However, applying multiple various metrics increases complicatedness of the method. It also requires running software, which is not well known.

Authors of research [43] have empirically evaluated performance of Hierarchical Clustering Technique (HCT) in predicting fault-prone modules using open source software and metrics. The proposed technique has shown 85% accuracy. However, developers should study thoroughly MatLab hierarchy clusterization algorithms to use this method in practice.

Research [44] addresses the problem of predicting fault prone modules using data mining techniques. In this study the authors applied different data mining rule-based classification techniques on several commonly available datasets. The newly proposed algorithm is an enhanced existing algorithm in terms of effectiveness (i.e. generating less number of rules) and accuracy (i.e. improving the results). Despite of rules reduction the method itself and its automatization possibilities are too complicated to enable its application.

The authors of work [45] have studied various metrics (requirement metrics, design metrics and code metrics) and techniques to identify fault prone modules. The proposed metrics are aimed to provide higher prediction results than existing techniques. However, various metrics combined with their application algorithms substantially increase complicatedness of this method.

The testing stage, aimed to improve reliability of a software system, is the most expensive and time-consuming one. Moreover, since dimensions of software systems have increased significantly during the past decades, effective utilization of limited testing resource has become even more important than before. A software system is typically composed by a number of modules. Each of them needs to be assigned with some testing resource before the testing stage commences. Hence, a natural question is how to allocate the testing resource to modules so that the reliability of a software system is could be maximized. Such a problem was formally defined by Ohtera and Yamada as the Optimal Testing Resource Allocation Problems (OTRAPs). In the works [46-48] it was proved that an optimal allocation scheme may lead to significant improvement in terms of the reliability of a software system. The available resource should be allocated among modules in a way enabling maximum number of faults to be removed from each module to achieve higher software reliability.

The optimization problems are formulated in work [46] as nonlinear programming problems (NLPP), which are solved by means of software reliability improving model based on a non-homogenous Poisson process which incorporates log-logistic testing-effort function. Work [47] suggests solving the OTRAPs by means of Multi-Objective Algorithms known as Hierarchy Particle Swarm Optimization Algorithm. Experimental results show that the proposed algorithm has overcome the drawbacks of the existing algorithm, and is more efficient. The main goal of the article [48] is to examine the resource allocation plan for fault detection and correction process of the software to save costs during testing and operational phases. The authors developed a model for fault detection and correction process in pursue of the said aim. Methods proposed in works [46-48] are complicated enough. Their application requires highly qualified personnel.

Analysis of described methods shows situation, as follows. Methods [33] through [37] don't permit to plan and evaluate testing resources, since they require testing data. Methods [38] through [48] require data on projects developed previously. Differences between systems being currently developed and those already existing are inevitably accompanied by substantial margin in evaluation. Allowances methods in realistic development processes don't work, thus reducing precision in reliability indicators' evaluation. Methods are complicated in application. Input data processing requires specific not commonly known software and efforts of skilled highly qualified personnel. All these items increase systems' development costs and reduce popularity of the methods. Methods don't take into account rigid restrictions of the testing resources. Analysis of described defects and drawbacks enables to offer general approaches to reliability evaluation and management throughout systems' lifetime taking into consideration their complicatedness and limited resources assigned for development basing on comparatively simple and more precise methods.

The rest of the work is composed, as follows. Paragraph 50.4.3 describes general approaches to reliability assessment and management. Paragraph 50.4.4 presents the check assumption and the description of the complexity-based prediction technique. Paragraph 50.4.5 describes the case study in terms of the experimental verification methods. Paragraph 50.4.6 describes results analysis. Conclusions are represented in paragraph 50.4.7.

50.4.3 Approaches to reliability assessment and management

There are two key principles of reliability evaluation and reliability management. The first one is technology-oriented. It means that proposed evaluation methods and aids are based on complicatedness recording and may be applicable with various systems development methodologies. Adjustable software may ensure their flexibility and simplify their implementation into business – software corporate processes. The second aspect of reliability management is based on resource considerations and enables to calculate and put foundation under resources allocation for reliability evaluation and ensuring. It comprises determining costs associated with achieving required reliability and comparative analysis of expenditures and incomes derived from faults detection in all the created products (specifications demands, project, initial code, etc.) throughout the entire development stage and risks and losses reduction at the operation stage. The technological- and resources-oriented approach to reliability evaluation and reliability management supposes following methods to be applied.

Complicatedness evaluation method with appropriate metrics enables to identify the most complicated and the most fault-prone requirements. These activities may contribute to design faults identification and elimination at the earliest stages. Their analysis will contribute to earlier faults identification and elimination in specifications. The project complicatedness should be also evaluated using metrics at the development stage. Such an approach enables to analyse project, to identify most complicated subsystems, components, interfaces, carry on their decomposition and improve links. Such activities may contribute to earlier identification and elimination of design faults.

One of proposed ways to reduce complicatedness and to improve program code reliability lies in its optimization (also known as refactoring). A large number of authors of publications in software systems development state that the initial complicatedness cannot be reduced. From their point of view, simplifying any software constructions leads to other ones becoming more complicated. However, the problem of code complicatedness quantification both at pre-optimization and post-optimization stages remains still unsolved.

Once the initial code is written, the testing procedure should be scheduled and resources should be allocated. Here total faults number in the system should be evaluated basing on code metrics. Since lack of resources is a common situation they should be spent in a most efficient manner. Selection and ranking modules with the highest fault number method should be developed. Testing of such modules will enable to reveal maximum faults with minimum number of modules subject to testing.

Conditions analysis for development, testing and allowance for reliability models enables to select a suitable model for evaluating achieved reliability indicators. Profile of system application should be evaluated at the operation stage by means of multiple metrics. Such a measure enables to improve efficient resources allocation to monitor the system to improve its reliability.

The proposed approach is directed to achieve required reliability in the most cost-saving manner. With testing being the most expensive and extended development stage initially, within the framework of general approach, method of improving testing efficiency should be developed. It determines the researches goal – complexity-based faults number prediction technique development and verification for ranking software modules prior testing.

50.4.4 The complexity-based prediction technique of faults number for software modules ranking: check assumption

The proposed method is based on an assumption that the most complicated modules (classes, components) contain the bulk quantity of faults. This assumption is logical and is often applied in scientific publications, requiring, however, practical confirmation. Five sets of experiment data have been used for verification [8]. The data in question represent complicatedness indicators by metrics and number of faults revealed in the course of modules' testing for five various object-oriented systems. Systems characteristics are shown in Table 50.3. Total volume of explored data exceeds 1.000.000 initial code lines, contains 4.330 modules and 4.449 faults. Comparative analysis of data grouped in the Table 50.3 shows that the researched system very substantially in their characteristics. Table 50.3 contains complicatedness measures per one module in metrics. Multiple metric choice $\mathrm{M} = \left\{\mathrm{RFC}, \mathrm{WMC}, \mathrm{LCOM}, \mathrm{LOC}, \mathrm{NPM}, \mathrm{CE}, \mathrm{CBO}\right\}$ is explained in the work [38].

Table 50.3 - Data on systems being under research

Systems characteristics	Abbreviated systems names and versions				
	Luc 2.4	Xer 1.4	Ant 1.7	Xal 2.7	Cam 1.4
Modules quantity	340	588	746	910	1 746
Faults quantity	632	1 596	338	1213	670
Faults free modules ratio	60%	74%	22%	99%	17%
Fault modules ratio	40%	26%	78%	1%	83%
Number of faults in a module	1.86	2.71	0.45	1.33	0.38
Faults density per 1,000 lines	6.14	11.30	1.62	2.83	3.42

Table 50.4 data analysis shows that complicatedness of software systems differs substantially in a number of metrics. Significant total volume and specified differences in the said SS enable to regard this sample as a representative sample. Actions described below are proposed to identify probable correlations between complicatedness and faults quantity in an individual module. Total modules' multitude SET_{mod} for each system has been indexed seven (7) times (according to applied metrics number) in decreasing sequence of complicatedness numerical indicator per each metric.

Table 50.4 - Complicatedness measures in metrics per one module

Complicatedness dimensions	Abbreviated systems names and versions				
	Luc 2.4	Xer 1.4	Ant 1.7	Xal 2.7	Cam 1.4
RFC metric	25	19	34	29	21
WMC metric	10	10	11	11	9
LCOM metric	69	75	89	126	73
LOC metric	303	240	280	471	112
NPM metric	7	8	8	9	7
CE metric	5	3	6	7	6
CBO metric	11	6	11	12	11

A certain part has been derived from each multitude (10 %, … 50 %) of the most complicated modules with highest complicatedness level under each metric. As a result seven basic sub-collections have been obtained $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{RFC}}$, $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{WMC}}$, $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{LCOM}}$, $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{LOC}}$, $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{NPM}}$, $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{CE}}$, $\mathrm{SUBSET}_{\mathrm{mod}}^{\mathrm{CBO}}$. Intersection of the sets by seven metrics formed newer modules' sub-collections $\mathrm{SUBSET}_{\mathrm{mod}}^{7}$, by six metrics $\mathrm{SUBSET}_{\mathrm{mod}}^{6}$, by five metrics $\mathrm{SUBSET}_{\mathrm{mod}}^{5}$, by four metrics $\mathrm{SUBSET}_{\mathrm{mod}}^{4}$, by three metrics $\mathrm{SUBSET}_{\mathrm{mod}}^{3}$, by two metrics $\mathrm{SUBSET}_{\mathrm{mod}}^{2}$, by one metric $\mathrm{SUBSET}_{\mathrm{mod}}^{1}$. Actual quantity of faults per one module has been counted for each enlisted sub-collection. The results are displayed in Table 50.5.

Table 50.5 - Average actual faults quantity per module

Modules complicatedness	Abbreviated systems names and versions				
	Luc 2.4	Xer 1.4	Ant 1.7	Xal 2.7	Cam 1.4
$\mathrm{SUBSET}_{\mathrm{mod}}^{7}$	13,5	20,9	3,1	3,0	4,6
$\mathrm{SUBSET}_{\mathrm{mod}}^{6}$	5,4	13,2	1,9	2,3	2,0
$\mathrm{SUBSET}_{\mathrm{mod}}^{5}$	3,1	14,2	1,1	2,0	1,1
$\mathrm{SUBSET}_{\mathrm{mod}}^{4}$	2,1	5,8	1,4	1,7	1,2
$\mathrm{SUBSET}_{\mathrm{mod}}^{3}$	2,0	4,3	1,0	1,7	1,9
$\mathrm{SUBSET}_{\mathrm{mod}}^{2}$	2,3	3,8	1,0	1,5	1,6
$\mathrm{SUBSET}_{\mathrm{mod}}^{1}$	1,5	2,8	0,4	1,7	0,4

Data contained in Table 50.5 shows that the most complicated software system modules have the greatest number of faults. As modules complicatedness decreases, the average fault number in module reduces confirming thus the validity of allowance method.

The proposed method should be run once the initial code is written, prior its testing commences. Complicatedness indicators of individual modules in metrics form the method's input data. Should the developers lack the corporative and practically checked set of metrics commonly available and most informative for faults forecasting object-oriented metrics considered in the work [38] may be applied as starters. The set of applicable metrics is indicated as $\mathrm{M} = \{\mathrm{M}_1, ..., \mathrm{M}_n\}$. The method supposed proceeding in five steps, as described below.

Step 1. Complicatedness indicators calculation split into metrics for individual modules in a system being developed using standard or corporate software. ...

Step 2. Total modules set $\mathrm{SET_{mod}}$ should be indexed n times as the complicatedness indicator descends with each metric $\mathrm{SET_{mod}^{M_1}}, ..., \mathrm{SET_{mod}^{M_n}}$.

Step 3. A certain part of the most complicated modules by a specific metric should be drawn from each indexed set. Dimensions of such a part should be determined referring to testing resources' restrictions. The less are these resources, the less is the part of drawn modules. As a result *n* basic sub-collections have been obtained for *n* metrics $\mathrm{SUBSET_{mod}^{M_1}}, ..., \mathrm{SUBSET_{mod}^{M_n}}$.

Step 4. Once intersection for basic sub-collections $\mathrm{SUBSET_{mod}^{M_1}}, ..., \mathrm{SUBSET_{mod}^{M_n}}$ is determined, intermediate sub-collections should be formed with top complicatedness indicators by all the metrics $\mathrm{SUBSET_{mod}^{n}} = \mathrm{SUBSET_{mod}^{M_1}} \cap ... \cap \mathrm{SUBSET_{mod}^{M_n}}$. All the modules within this sub-collections should be ranked as *n*. Choosing all and any *n-1* metrics intersections for subset $\mathrm{SUBSET_{mod}^{n-1}}$ should be determined with all the modules to be ranked as *n – 1*. Similar procedure should be applied until subset for any single metric is built as $\mathrm{SUBSET_{mod}^{1}}$ with all the modules in it ranked as *1*. Number of ranks should be the same as metrics number.

Step 5. Determining the sum of subsets for $\mathrm{SUBSET_{mod}^{n}}, \mathrm{SUBSET_{mod}^{n-1}}, ..., \mathrm{SUBSET_{mod}^{1}}$ the resulting ranked modules sample should be formed as $\mathrm{SUBSET_{mod}^{R}} = \mathrm{SUBSET_{mod}^{n}} \cup \mathrm{SUBSET_{mod}^{n-1}} \cup ... \cup \mathrm{SUBSET_{mod}^{1}}$.

The resulting sample should include modules with top complicatedness indicators simultaneously by *n* metrics (*rank 7* in our example), *n-1* metrics (*rank 6*), … and, finally, one metric (*rank 1*). Modules ranking in resulting sample is necessary to establish sequence of their testing. Top rank modules are the most complicated, supposed to contain the most of faults, and are subject to be tested at the first turn. The proposed method should be applied once the initial code is written and prior its testing starts. Complicatedness indicators of individual modules within the developed system form the input data with output data being the resulting ranked sample of modules with faults. Basic sample dimensions should be determined by restrictions imposed by testing resources.

50.4.5 Case study: the experiment performance technique

The method has been verified by means of commonly accessible experimental data [8] and specially developed software. Number of selected modules has been calculated as well, as faults number in these modules for each system. As this data has been being calculated the quantity of involved metrics has been varied as well, as the modules number with top complicatedness indicators corresponding to these metrics. Obtained data has been analysed. The first subtask of the analysis was to define relations between faults number and modules' complicatedness and was estimated simultaneously by a number of metrics. The second subtask was to define relation between faults number and that of selected modules. Selected ratio values encompassed 10%, …, 50% including the most complicated modules. The parts dimensions were governed by probable restrictions in testing resources. Since the researched systems substantially varied in their characteristics, relative percentage indicators have been calculated.

99% modules within one system under research contained faults making it remarkably distinguished among others. Data processing and analysis showed that modules selection based on their complicatedness had not given increase in a quantity of faults in them. Average data for four systems is represented in Table 50.6. Data is allocated in two lines. The first line displays modules ratio from their total quantity. The lower ratio shows faults ratio contained in these modules in relation to their total quantity.

Values highlighted as examples in the last column in Table 50.6 should be interpreted, as follows. Base sample 20% of the most complicated modules per each metric forms a resulting ranked modules sample 41,8% ratio of their total number, containing 74,8% ratio of total faults' quantity.

The data in question should be applied in a manner, as follows. Tests are developed for each selected module. These tests take a certain amount of labour expressed in person-hours. Time consumed by running all such tests for all selected modules enables to calculate testing expenditures required to reveal 74,8% of the total faults quantity. At the further stage, restrictions imposed by labour, time, financial, hardware and software resources should be considered. Should there be a lack in resources required to test selected modules, basic sample dimensions should be reduced.

Factor $k = \frac{d_m}{m}$, with m being modules ratio in their total quantity and faults ratio of their total number contained in appropriate modules is proposed to reduce the number of analysed indicators and to facilitate the method's efficiency analysis. The higher is the *k* value, the more is the number of faults within the selected modules.

Table 50.6 **-** Average calculated data on four systems under research

Modules' ratio	Indicators	Rank 7	Rank 6	Rank 5	Rank 4	Rank 3	Rank 2	Rank 1	$\sum$
10 %	mod., %	2,0	1,3	2,5	2,5	3,8	3,3	8,0	23,3
	faults, %	15,0	6,5	7,0	8,0	6,5	4,8	7,8	55,5
11 %	mod., %	2,0	1,8	2,8	3,0	4,3	3,0	8,8	25,5
	faults, %	14,3	7,3	7,8	6,8	7,3	4,8	8,0	56,0
13 %	mod., %	2,3	1,8	2,8	3,5	4,0	3,8	8,8	26,8
	faults, %	17,3	8,8	8,3	7,0	6,8	5,8	8,5	62,3
14 %	mod., %	2,5	2,8	3,3	4,3	4,5	4,3	10,3	31,8
	faults,%	18,3	11,3	7,3	8,3	5,8	6,5	9,3	66,5
17 %	mod., %	2,5	3,0	3,5	5,3	5,0	4,8	11,0	35,5
	faults, %	19,8	13,0	7,3	9,5	6,3	5,8	9,0	70,5
20 %	mod., %	4,5	4,0	4,5	5,5	6,0	5,5	11,8	41,8
	faults, %	23,5	13,3	8,3	7,8	6,5	6,8	8,8	74,8
25 %	mod., %	6,3	5,0	5,5	7,0	7,0	6,3	13,5	50,5
	faults, %	30,0	12,0	9,0	9,0	5,8	5,3	9,0	80,0
33 %	mod., %	9,0	7,5	8,0	8,8	6,5	8,3	14,5	62,5
	faults, %	36,0	14,8	9,3	8,5	5,3	6,5	7,0	87,3
50 %	mod., %	17,8	13,3	10,5	9,8	9,0	9,3	11,5	81,0
	faults, %	47,0	19,3	10,8	5,8	4,3	4,8	4,5	96,3

Basing on the k application, Table 50.6 data was transformed and represented in Table 50.7 format. The k values in Table 50.7 show how the selected and ranked modules share exceeds the part of faults they contain.

Table 50.7 - Method's average efficiency factors k for four systems

Modules' ratio	k for rank 7	k for rank 6	k for rank 5	k for rank 4	k for rank 3	k for rank 2	k for rank 1
10 %	7,50	5,20	2,80	3,20	1,73	1,46	0,97
11 %	7,13	4,14	2,82	2,25	1,71	1,58	0,91
13 %	7,67	5,00	3,00	2,00	1,69	1,53	0,97
14 %	7,30	4,09	2,23	1,94	1,28	1,53	0,90
17 %	6,58	4,33	2,07	1,81	1,25	1,21	0,82
20 %	5,22	3,31	1,83	1,41	1,08	1,23	0,74
25 %	4,80	2,40	1,64	1,29	0,82	0,84	0,67
33 %	4,00	1,97	1,16	0,97	0,81	0,79	0,48
50 %	2,65	1,45	1,02	0,59	0,47	0,51	0,39

For example, k = 7,5 means that testing of 1% of the selected modules enables to reveal 7,5% of faults. Minimum k value is 0,39. Average calculated k value is 2,34. The following factor interpretation is proposed. If $k \le 1$ the modules' selection is inefficient from the point of view of their complicatedness evaluated by metrics. If $1 \le k \le 1{,}5$ modules are selected with minor efficiency. Finally, if k > 1,5 the modules are selected efficiently. E.g. if 13 % modules are selected their testing may be efficient if they are characterized by high complicatedness levels simultaneously by seven, six, five, four, three, and two metrics. If 20 % modules are selected, testing of modules with high complicatedness values by seven, six, and five metrics will be efficient.

50.4.6. Results analysis

Analysis of Table 50.7 data enabled to identify two tendencies. The first tendency may be followed in each horizontal data line. Efficient modules selection will be maximum for modules with top complicatedness indications involving simultaneously all the involved metrics. The k decreases as modules' complicatedness reduces. The k <1 in the Table 50.7 last column. Thus, modules selection by a single metric is not advantageous for faults detection. The second tendency may be followed in each data column. Selection of the least part of the most complicated modules is the most efficient. The k value decreases as the selected module number increases. Consequently, the less is selected modules number, the higher is effect or modules complicatedness evaluation by metrics.

Thus the method's efficiency depends on metrics' quantity determining simultaneously high complicatedness and on modules sample dimensions. The coefficient value increases with the metrics' number increase and sample dimensions decrease. Reverse statement is valid, too. Maximum coefficient value amounted to 7,5 with average value 2,34 and minimum value 0,39.

Selecting modules and tests for them enables to estimate testing expenditures to detect a certain quantity of faults. Since restricted resources prevent testing all the modules, the unrevealed faults may cause risk of losses. Risks calculation and comparing them with testing expenditures may enable managers to find adequate and economically grounded solutions. Tables 50.6 and 50.7 data may be helpful not only for efficient testing efforts allocations, but also for efficient application of other code verification methods, e.g. inspections, statistical analysis, etc.

The proposed set of known metrics for code complicatedness evaluation is an initial, or starting, one. Software corporations' experts may apply their unique corporate metrics set, proven at practice. Discussable aspect is using obtained results not only for code testing but its applicability for other verification methods, such as survey, control, inspections, audits, statistical analysis. It is not still clarified, whether selected modules contain fault-free modules, what is their quantity, how it depends on complicatedness level.

50.4.7 Conclusions

Elements are developed for technological and resources oriented approach to reliability management at all the lifecycle stages in restricted resources environment. The technique of complexity-based prediction of faults number for ranking software modules is proposed within the framework of general approach. Supposed statement that the most complicated modules contain the bulk quantity of faults is proved experimentally. The method is applicable after initial code is written prior its testing commencement. The method's input data consists of complicatedness indicators for individual modules of the system being in development by metrics. The output data represents a sample of ranked modules of certain dimensions with a certain number of faults. Sample dimensions are only restricted by testing resources.

The proposed method is aimed to efficient testing with restricted resources. Verification of the method using a representative sample of experimental data demonstrated its efficient operability. Proposed efficiency factor depends on number of metrics by which modules have simultaneous high complicatedness rates and on modules sample dimensions. The achieved results enable to control process of achieving required reliability with restricted resources by means of allocating the testing efforts to a certain number of modules with the highest faults quantity.

Sampling such modules and selecting tests for them enables to estimate *apriori* testing costs by means of summing up testing time for all the selected modules. Maximization of revealed faults number and improved systems' operating reliability may reduce expenditures and increase developers' revenues. Risks of faults triggering at the operation stage may be mitigated. Testing expenditures will be efficient investments into the systems' reliability.

The proposed method is simple enough to be applied in practice. The method does not demand any additional data except the code complicatedness evaluation for the system being in development. The method may be completely automated. Revealed application restriction concerns systems with 99% faulty modules ratio. Only in such a case method application may be inefficient. Prospective direction of further researching may be implementation of the proposed method into business processes involving various methodologies of software systems developments. Testing expenditures calculation model and method of comparing expenditures with unrevealed faults risk evaluation should be developed. Software implementation of the proposed method should be described.

50.5. BDA based assessment of software reliability

Let us suppose that a certain software corporation is developing a software system. Initial software code is partly developed. SWS development process is restricted in time and funding. Scope and costs of works associated with forthcoming tests should be evaluated to meet the imposed restrictions. Reliability estimated indexes should be calculated. For example, quantity of defects in particular modules, total quantity of defects in developed code and defects' density should be assessed prior the testing process commencement. Corporation may use experimental data on defects identified in previously developed SWS for such preliminary assessment. They may be further referred to as historical data. As another example, the case may be when limited funds, terms or personnel resources prevent the corporation from accumulation and reasonable usage of such data. Even with available data the system being designed may have nothing common or similar with previously developed systems.

With available historical data found in big data storage the corporation may use them for verification or updating preliminary assessment obtained on the basis of historical data. In any case, it may be feasible to search SWS experimental data from other developers applicable for reliability assessment of own SWS in big data storages with free access. These data storages are well known in global data field. They may be, for instance, software reliability data depositories [8] of international conferences, NASA data portal [9], services of code testing and statistic analysis [10], software sport services [11] and other sources.

The abovementioned big data storages contain gigabytes of codes and experimental reliability data on multiple SWS by different developers. This data may be called “associated”. It may consist of artefacts – requirements, initial code, tests, artefacts evaluations and processes of their development under various metrics (e.g. known object-oriented metrics RFC, WMC, LCOM, LOC, NPM, CE, CBO, CA, NOC, DIT). This data includes actual quantity of defects identified in modules, temporal series of defects detection. This data enables to evaluate, forecast or verify reliability indexes of systems at various stages of development.

Thus, there are both necessity and possibility to evaluate, predict and control reliability of SWS being developed using big data storages. To do it, it is necessary to find a system similar by a number of criteria to particular system in development. Therefore a method should be developed to perform big data based search for similar programs.

50.5.1 Motivation BDA based assessment of software reliability in TOAStRaS context

The fourth principle of the TOAStRaS metodology assumes the embedding of methods and aids of data search, accumulation, storage, processing and analysis applied to SWS under development on the basis of similar projects and systems. Dependability evaluation processes using big data storages, principles and technologies. Realization of this principle supposes search of necessary data in big data storages, generation of broad corporative data field for dependability evaluation, data accumulation and keeping in cloud storage, big data processing and review of the applying specialized techniques, using resulting data for dependability assessment, and, finally, improving of data usability. This principle implements parametrical aspect.

SWS development consists of a number of technological stages (requirements outlining, design, code writing and verification). Artefacts are generated at each of such stages. The artefacts contain defects and affect the SWS reliability. Majority of defects is located in the initial code. Initial codes with various SWS possess different objective features, such as structure, dimensions, complicatedness, and programming languages. In view of such differences the task of search for suitable associated data for reliability indexes assessment and forecasting may be formulated, as follows: SWS should be found in big data storage with initial code of the greatest resemblance with that of the SWS under development in structure, dimension, complicatedness and programming language.

In view of the above the necessity arises to formulate a SWS initial code similarity principle.

50.5.2 Software systems similarity principle

Structure, dimensions and complicatedness of the SWS may be assessed by means of metrics. The proposed SWS code similarity principle bases upon metrics enlisted below:

1. Initial code dimension in thousands of lines (KLOC);
2. Total quantity of code modules;
3. Complicatedness assessment metrics for code modules;
4. Total, average and maximum evaluation for each metric.

It is worthwhile to mention here, that numerical evaluations of these metrics reflect not only system’s dimensions and complicatedness, but also number of system’s modules/classes and their links, i.e. system structure. The possibility to assess complicatedness of system being developed and associated system by means of uniform set of metrics guarantees resemblance of programming languages of the systems in question.

In general, proximity of ratings under certain metrics for system under development and system taken for comparison provides similarity of dimensions, complicatedness, structure and programming language. System similar to the system under development should be defined as a system having minimum deviations of ratings by nominated metrics. Selection a similar SWS can be made via relative deviations of each of appropriate metrics for system under development and system taken for comparison.

1. Relative deviation of initial code dimensions

$$RD_{size} = \frac{|\ KLOC_d - KLOC_f\ |}{KLOC_d} \bullet 100\% \qquad (5.4)$$

The bottom index "d" corresponds to rating of the system under development. The bottom index "f" corresponds to rating of a system taken for comparison.

2. Code modules quantity relative deviation

$$RD_{mod} = \frac{|\ MC_d - MC_f\ |}{MC_d} \bullet 100\% \qquad (5.5)$$

with MC – number of modules.

3. Summarized rate relative deviation

$$RD_{sum} = \frac{|\ Sum_d - Sum_f\ |}{Sum_d} \bullet 100\% \qquad (5.6)$$

4. Average rate relative deviation

$$RD_{avg} = \frac{|\ Avg_d - Avg_f\ |}{Avg_d} \bullet 100\% \qquad (5.7)$$

5. Maximum rate relative deviation

$$RD_{max} = \frac{|\ Max_d - Max_f\ |}{Max_d} \bullet 100\% \qquad (5.8)$$

for each metric of complicatedness.

At the next stage calculated deviations should be grouped into three groups. The first group indicates dimensions deviation rate, the second group indicates structure deviation rates, and the third group indicates code complication deviation rates. Average deviation rate should be calculated for each group. Deviations within a group are feasible to apply with unequal priority indexes for SWS similarity assessment. Under certain circumstances, priority indexes for SWS similarity assessment may be either dimension, or structure, or complicatedness of the system. Common general average deviation rate for all the rates should be also calculated. This value is feasible to apply for indexes with equal significance.

So, search for comparative SWS being similar or the most proximal to the SWS under development requires to know code dimensions, number of modules, estimated complicatedness evaluated applying unified set of metrics, calculated metrical rates' deviation and, finally, system selection with minimum deviations.

The authors state a hypothesis that a similar system may be found in big data storage among available multiplicity. This hypothesis, however, should be checked. *Big data storage* contains experimental data of multiple various systems. For example, storage [8] contains data relevant to metrics and defects of sixty one SWS. Manual processing of such data may take too much time and labour. Specialized software Agent for search of similar programs could be helpful in the aspect of automation of such a process.

50.5.3 Software agent for search of similar programs

The agent performs the following functions:

1. Downloading of metrical rates of system under development as a reference point for comparison with other systems.
2. Downloading of other SWS data (metrical indexes and defects quantity from big data storage into local corporative or cloud storage. This step is necessary to generate a corporative data field for multiple reliability assessment.
3. Data transformation into appropriate format for processing (*.db, *.xml, *.xlsx, etc.);
4. Transformed SWS data import into agent memory.
5. Data processing – calculation of deviation rates within a group and total average deviation of all the rates.
6. Entering deviations for each system into resulting account.
7. Accounted deviations sorting to ground similar SWS choice.

The agent creates the resulting account with group and average deviations for multiple involved SWS. After the deviation values are sorted the SWS with minimum deviations form metrics of SWS under development are placed into the top of account. The account enables to make a well-grounded choice of SWS with the highest similarity index to the SWS under development. Experimental data on defects of the chosen similar SWS may be used to assess and predict similarity of the SWS under development. The proposed agent is a program for processing flat (not linked) tables and for calculation of statistic indexes.

50.5.4 Technique of similar programs search

The procedure of similar programs search based on big data consists of seven steps.

Step 1. Calculate metrical rates for SWS structure, dimensions and complicatedness under development.

Step 2. Activate the agent, consistently download identical metrical rates and data of defects of other SWS from big data storage.

Step 3. Transform downloaded data of other SWS into appropriate format for processing.

Step 4. Calculate internal deviation rates and general average deviation rate.

Step 5. Record deviation rates for each SWS into resulting account. Sort indexes in the account.

Step 6. Select similar system with minimum deviation rates in the account.

Step 7. Use actual data on defects of the selected SWS to assess reliability of the SWS under development.

50.5.5 Case study

The above declared hypothesis stating that a system similar to the SWS under development in structure, dimensions and program language may be found in big data storage requires experimental checking. Such a check was performed in a manner described below. Metrical data and defects data for twenty one SWS have been randomly selected and downloaded from big data storage [8] into local computer disc. One of these systems has been taken as a reference point. Other twenty systems have been explored for similarity of their features (structure, dimensions and complicatedness) to the reference system. Programming language similarity of the systems in question has been supported by unified set of metrics for complicatedness assessment. They are rather common metrics of object-oriented code complicatedness assessment RFC, WMC, LCOM, LOC, NPM, CE, CBO, CA, NOC, DIT.

The agent designed by authors transformed data from *.txt or *.csv format into *.dbf format. Further calculations of relative deviations for metrical rates had been performed by means of SQL instructions for each system. Data processing applying the agent took about two working hours. Group and average deviation rates have been stored in resulting account, as shown in Table 50.8.

Indexes in the account have been sorted in increasing order. Reference SWS has number 1. Naturally, deviation rates in corresponding line are zero. SWS No 2 follows directly after it with minimum deviation from reference SWS (highlighted line in Table 50.8). System with increasing deviation rates are placed downwards. The resulting account enabled to choose SWS with the highest level of similarity to the reference SWS.

Therefore implementation of the TOAStRaS methodology, in particular, prediction of SWS reliability can be based on processing information, which is extracted from big data storages by using the software agent. Data processing for twenty SWS by means of the agent took about two hours. The search of similar programs represents practical value for a project manager and personnel of the SWS testing group. The agent may be adapted by software companies to take into account specifics of developed SWS.

Table 50.8 - Metric rates relative deviations of SWS compared with reference system

№ SWS	Metrical rates deviations, %			
	Structure	Dimensions	Complicatedness	Average rate
1	0,0	0,0	0,0	0,0
2	**5,1**	**9,0**	**5,4**	**6,5**
3	12,2	20,6	41,8	24,9
4	12,7	51,3	35,7	33,2
5	12,8	19,0	28,7	20,2
6	14,4	42,1	46,7	34,4
7	22,4	87,7	34,9	48,3
8	23,6	57,4	33,6	38,2
9	24,5	82,3	29,2	45,3
10	27,4	56,0	43,9	42,4
11	28,6	40,5	40,5	36,5
12	28,6	51,9	44,5	41,7
13	29,9	68,0	20,9	39,6
14	30,3	70,9	26,0	42,4
15	30,9	83,6	45,1	53,2
16	31,4	78,9	38,6	49,6
17	46,0	24,7	44,3	38,3
18	71,8	52,5	24,5	49,6
19	74,6	59,0	25,4	53,0
20	270,1	815,7	52,5	379,4
21	350,6	660,9	66,9	359,5

As a result the technique has been suggested to search and analyse similar programs. The similarity principle is based on complexity and structure SWS metrics and metrics of program language similarity. Calculation formulas to assess group and average deviation rates describing the SWS similarity have been suggested.

Case study allowed to obtain some experimental results. A system has been identified with minimum (5,1 – 9,0 %) and average 6,5% relative deviation of metrical rates among twenty explored systems. Obtained results confirm the allegation that systems with known reliability indexes similar to the SWS under development may be found from great quantity of experimental data kept in big data storage to assess, verify and predict its reliability.

50.6. BDA based assessment of software security

Security is so important component of dependability property as reliability. The main threat of SWS security is formed by vulnerabilities. A vulnerability is a weakness which allows an attacker to reduce a system's information assurance.

Vulnerability is the intersection of three elements: a system susceptibility or flaw, attacker access to the flaw, and attacker capability to exploit the flaw. To exploit vulnerability, an attacker must have at least one applicable tool or technique that can connect to a system weakness. In this frame, vulnerability is also known as the attack surface. Vulnerability management is the cyclical practice of identifying, classifying, remediating, and mitigating vulnerabilities. This practice generally refers to software vulnerabilities in computing systems. Using vulnerability as a method of criminal activity or to create civil unrest falls under US Code Chapter 113B on terrorism. One of the key concepts of information security is the principle of defence in depth: i.e. to set up a multilayer defence system that can: 1) prevent the exploit, 2) detect and intercept the attack, 3) find out the threat agents and prosecute them.

A security risk may be classified as vulnerability. The use of vulnerability with the same meaning of risk can lead to confusion. The risk is tied to the potential of a significant loss. Then there are vulnerabilities without risk: for example when the affected asset has no value. A vulnerability with one or more known instances of working and fully implemented attacks is classified as an exploitable vulnerability — a vulnerability for which an exploit exists. The window of vulnerability is the time from when the security hole was introduced or manifested in deployed software, to when access was removed, a security fix was available/deployed, or the attacker was disabled.

Security bug (security defect) is a narrower concept: there are vulnerabilities that are not related to software: hardware, site, personnel vulnerabilities are examples of vulnerabilities that are not software security bugs. Constructs in programming languages that are difficult to use properly can be a large source of vulnerabilities.

Vulnerability is a software defect that allows an attacker to violate an explicit (or implicit) security policy to achieve some impact (or consequence). Vulnerabilities give hackers an opportunity of plunder, damage, illegal modification, access blocking and information destruction. The main security risk is posed by vulnerabilities appropriate to SWS and the used configuration components: operating systems, Web and databases servers.

The task of vulnerabilities detection is very important. Loss of SWS operability, stealing, distortion and information destruction caused by the hackers' attacks result in the companies' enormous damage. According to the research conducted by request of FBI and the National institute of Standards and Technology of the USA, the losses caused by vulnerabilities equal from 22 to 60 billion dollars annually [49].

50.6.1 Vulnearbilies and software security in TOAStRaS context

The TOAStRaS methodology assumes embedding methods and aids of data search, accumulation, storage, processing and analysis applied into SWS under development on the basis of similar projects and systems. Similar projects and systems possess definite number and types of vulnerabilities. These are data which allow to evaluate information security of SWS under development. Thus the fourth principle of TOAStRaS methodology dealing with the parametrical aspect will be realized.

The last development and achievement in the field of software security are described in scientists' numerous researches conducted in Centre for Software Reliability, City University London. In the papers [50, 51] authors report recent results on modelling the impact of cyber-attacks on the resilience of complex industrial systems. Authors use a hybrid model of the system under study in which the accidental failures and the malicious behaviour of the Adversary are modelled stochastically, while the consequences of failures and attacks are modelled in detail using deterministic models. This modelling approach is demonstrated on a complex case study - a reference power transmission network, enhanced with a detailed model of the computer and communication network used for monitoring, protection and control compliant with the international standard IEC 61850. Authors studied the resilience of the modelled system under different scenarios: i) a base-line scenario in which the modelled system operates in the presence of accidental failures without cyber-attacks; ii) several different scenarios of cyber-attacks.

The evaluation of the security, reliability and resilience of critical infrastructures (CI) faces a wide range of challenges ranging from the scale and tempo of attacks to the need to address complex and interdependent systems of systems in work [52]. Model-based approaches and probabilistic design are fundamental to the evaluation of CI and we need to know whether we can trust these models. This paper presents an approach authors are developing to justify the models used to assure CI using structured assurance cases based on Claims, Arguments and Evidence (CAE). The modelling and quantitative evaluation of the properties are supported by the Preliminary Interdependency Analysis (PIA) method and platform applied to a case study –

a reference power transmission network enhanced with an industrial distributed system of monitoring, protection and control.

Safety cases are the development foundation for safety-critical systems and are often quite complex to understand depending on the size of the system and operational conditions. The recent advent of security aspects complicates the issues further. The paper [53] describes an approach to analysing safety and security in a structured way and creating security-informed safety cases that provide justification of safety taking into particular consideration the impact of security. The paper includes an overview of the structured assurance case concept, a security-informed safety methodology and a layered approach to constructing cases. The approach is applied to a Security Gateway that is used to control data flow between security domains in a separation kernel based operating system in avionics environment. Authors show that a clear and structured way of presenting a safety case combining safety and security alleviates understanding important interactions taking into account the impact and, hence, increases safety.

Traditionally, safety and security have been treated as separate disciplines, but this position is increasingly becoming untenable and stakeholders are beginning to argue that if it’s not secure, it’s not safe. In paper [54] authors present some of the work we have been doing on “security-informed safety”. Authors’ approach is based on the use of structured safety cases. Authors discuss the impact that security might have on an existing safety case. Authors also outline a method they have been developing for assessing the security risks associated with an existing safety system such as a large-scale critical infrastructure.

The paper [55] reports on the results of a security analysis of the European Railway Traffic Management System specifications (ERTMS). ERTMS is designed to be fail-safe and the general philosophy of ‘if in doubt, stop the train’ makes it difficult to engineer a train accident. However, it is possible to exploit the fail-safe behaviour of ERTMS and create a situation that causes a train to halt. Thus, denial of service attacks are possible, and could be launched at a time and place of the attacker’s choosing, perhaps designed to cause maximum disruption or passenger discomfort. Causing an accident is more difficult but not impossible. According to an experts’ opinion it is crucial to take following measures to avoid vulnerabilities at a development stage of SWS:

1. To think over architectural concepts from the viewpoint of gaps absence in protection system against unauthorized invasions;
2. To develop sufficient mechanisms of protection;
3. To execute protection mechanisms realization in coding unmistakably;
4. To eliminate vulnerabilities in a detection case quickly.

These activities are principal for SWS security assurance.

50.6.2 Analysis of software vulnerabilities

The knowledgebase [56] houses the public Vulnerability Notes Database, the Vulnerability Card Catalog, and the Special Communications Database. The knowledgebase is a collection of internet security information related to incidents and vulnerabilities. The knowledgebase houses the public Vulnerability Notes Database as well as two restricted-access components:

1. Vulnerability Card Catalog contains descriptive and referential information regarding thousands of vulnerabilities reported to the Coordination Center.
2. Special Communications Database contains briefs that provide advance warning and important information about vulnerabilities, intruder activity, or other critical security threats.

From the vulnerabilities origin viewpoint which occurs at stages of SWS life cycle, it is possible to divide them into three categories:

1. Vulnerabilities of a design stage;
2. Vulnerabilities of a stage of realization;
3. Vulnerabilities of an operational phase.

The most critical are the vulnerabilities of a design stage caused by SWS architectural drawbacks. It is very problematic for the developers to eliminate such vulnerabilities. Such vulnerabilities are often copied in the following versions of the system. It makes new systems vulnerable automatically, despite extension of the functional capabilities. Vulnerabilities of realization stage arise when the developer enters different defects into an algorithm, which is correct from the view point of security. Vulnerabilities of an operational phase are linked to the components of a system program configuration: operating system, Web and database servers.

Many specialized software and systems are developed for automatic search of SWS vulnerabilities. However vulnerabilities search systems in SWS architecture weren't widely adopted in a business environment of developers at design stage. In our opinion, it is connected to insufficient development of the formal models, methods and tools of defects and vulnerabilities search in SWS architecture and their deployment in business processes of the software companies. There are a few organizations which exploit such models, methods and tools. These are the laboratories executing certification of SWS in accordance with the security requirements.

The analysis systems of SWS security in the implementation and operation stages gained the greatest spread among the interested persons. These are the static analysis systems of the source code (BOON [57]), system of expert audit automation of the source code (Flawfinder [58], ITS4 [59], RATS [60]), system of vulnerabilities automatic search (FindBugs [61] and Klocwork [62], system of source code tangling (VMProtect [63], CodeVirtualizer (Themida) [64], Safengine Protector [65], Dotfuscator [66]).

Systems of tangling are directed to that the source text recovered from the binary code possessed excessive logic which isn't connected to logic of operation of the initial algorithm. In this case the definition of an algorithm for a study and the following attack requires the considerable additional actions which don’t allow to solve problems of the reverse engineering in reasonable terms. The most widespread systems of vulnerabilities search in executable code are those of the attacks simulation which model different unauthorized invasion in SWS. Such systems became widely known in view of the relative simplicity and low cost. These are the examples of such systems: SATAN, Internet Scanner, Cisco Secure Scanner. The Internet Scanner system is one of the most known search tools. It reveals more than 900 different vulnerabilities of different categories: Denial of Service, Brute Force, FTP, LDAP, SNMP, RPC, NIS, NFS, DNS, etc. The Whisker system was created especially for scanning of Web servers and allows to reveal vulnerable CGI scripts. It should be noted the listed systems support security of SWS at the sufficient level only in case of the regular up-dating.

The listed systems of vulnerabilities search generally use the following methods of vulnerabilities detection in source texts and the binary code:

1. Experts’ estimation method of software security on the basis of manual vulnerabilities search;
2. A method of automatic vulnerabilities search of the given patterns using specialized software;
3. A fuzzing method which treated as the analysis of boundary values. This method consists in determination of tolerance ranges of input values and values verification which are within or out of the range. Besides, the fuzzing method assumes preparation of different input data for SWS security check.
4. Cryptography techniques and the tangling of the source code complicating or making impossible disassembling of the executed code;
5. Systems of hackers’ attacks simulation.

In addition to the listed above methods we offer another one for vulnerabilities detection in “new” SWS (which means a system under development).

50.6.3 Detection method of software vulnerabilities

Detection method of software vulnerabilities consists in search of SWS, similar to the new system for which the quantity and types of vulnerabilities are already known. The known vulnerabilities of similar system will allow evaluating of new SWS security. Search of similar SWS is executed in big data storages. These storages contain requirements, the source code, estimates of

SWS properties on different metrics, information on detection time, quantity and types of defects and vulnerabilities etc. It is necessary to define attributes of SWS similarity for search of similar systems. Vulnerabilities get to SWS at a stage of architecture creation. Therefore, architectural attributes need to be considered in case of SWS similarity determination. The text or graphic architecture descriptions are necessary for systems similarity establishment.

The member of a development team determines architecture similarity by method of expert assessment from the viewpoint of two equivalent signs:

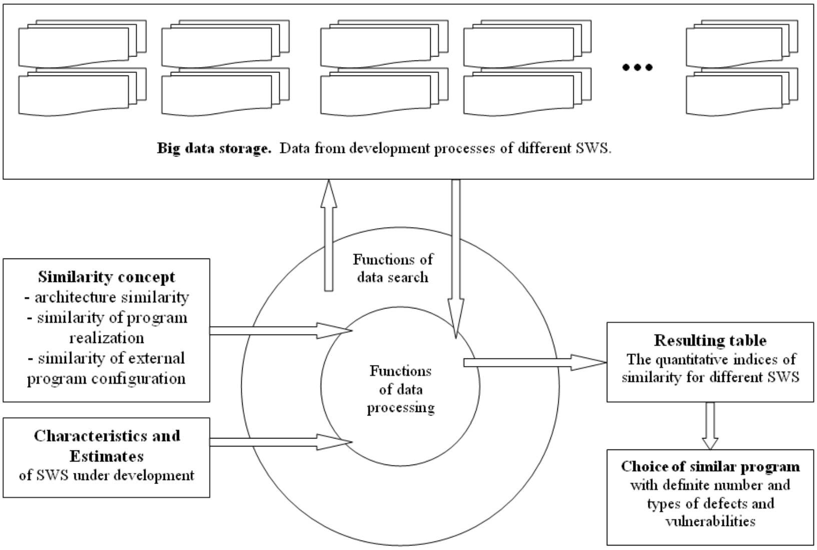
1. Identical or simile quantity and types of systems structural elements (functional and interface units, classes, etc.), their interaction;
2. Identical or simile interaction of system elements with external IT - environment.

The expert estimates architecture similarity of new and some SWS from big data storage numerically by means of deviation assessment in percentage on each sign.

Vulnerabilities get to SWS at a writing stage of the source program code owing to incorrect of protective procedures realization. Therefore, it is also necessary to consider metric attributes (estimates of SWS properties by means of metrics). It is necessary to compare estimates of the source code on structural, size and complexity metrics for definition of systems similarity. We offer to use known object-oriented complexity metrics RFC, WMC, LCOM, LOC, NPM, CE, CBO, CA, NOC, DIT for estimation of SWS attributes. We offer to use relative deviations (concerning attributes estimation of new SWS) offered in paragraph 50.5.2. SWS taken from big data storage will have the minimum deviations. Such system will resemble new SWS the best. The described in paragraph 50.5.3 software agent for similar programs search estimates program realization similarity for new SWS and those taken from big data storage numerically. The estimation is realized by means of the relative deviation by structure, size and complexity of a program code.

Vulnerabilities in SWS at an operational phase are caused by a system program configuration. Therefore information of the configuration where SWS are exploited is necessary to define systems similarity. These are text or graphic descriptions of a program configuration (an operating system, Web and database servers, browsers, etc.). The similarity of systems configuration is determined by the member of a development team by method of an expert assessment by means of deviations concerning attributes of new SWS.

Further we suggest to calculate a resultant deviation as mean value of deviations in architecture, program realization and configuration of SWS. The resultant deviation characterizes the similarity of the compared systems numerically. A system with the minimum deviations will resemble new SWS in the best way. Figure 50.4 shows the chart of search process of similar programs by means of a special software agent.



Big data storage. Data from development processes of different SWS.
Functions of data search
Similarity concept
- architecture similarity
- similarity of program realization
- similarity of external program configuration
Functions of data processing
Characteristics and Estimates of SWS under development
Resulting table
The quantitative indices of similarity for different SWS
Choice of similar program with definite number and types of defects and vulnerabilities

Figure 50.4 - The chart of search process of similar programs by means of a special software agent

50.6.4 Features of the assessment

After similar SWS with definite number and types of vulnerabilities has been found, we will be able to estimate security of the new SWS numerically. In our opinion, for this purpose it is necessary to obtain up-to-date information about the found vulnerabilities and to execute the search of generally available exploits for them.

Up-to-date information about vulnerabilities can be obtained from generally available sources. These are databases of vulnerabilities:

- *Open Source Vulnerability Database* [67] provides information both in an XML format, and in the form of SQL dumpings. The database provides a possibility of storage of the dates connected to vulnerability. These are dates of opening, exposure, an "exploit" and a decision outputs. Dates of the publication and modification of record about vulnerability are also available in the database.

- *Common Vulnerabilities and Exposures* [68] supply the uniform general dictionary of vulnerabilities. Other bases operate CVE ids. The information about vulnerabilities is delivered in the format of XML files;
- *National Vulnerability Database* [69] allows to load XML files with the information about vulnerabilities. Data from NVDB are the most detailed. This database includes metrics of vulnerabilities importance. However, at present the base saves only publication and modification dates of record about vulnerability.

Databases of vulnerabilities differ on the information filling. For obtaining full information about certain vulnerability it is necessary to examine all listed bases. The obtaining of the all information necessary for the estimation of SWS security is possible from the bases. It includes an assessment of vulnerability importance on a ten-point scale, threat type, the date of the publication, the date of decision issue and the date of exploit advent.

Exploits are program tools which automate the attacks using the vulnerabilities found at software. Exploits are written by researchers of information security and published on specific websites. References to exploits are stored in public databases, such as: SecurityFocus [70], Rapid7 [71], Exploit-db [72], The Exploit Database [73].

Existence and availability of exploits considerably increases army of hackers. School pupils, students, ex-employees and unfair competitors can easily become them. Program scanners are developed and available for automated search of exploits. Scanners check lists of exploits on specific websites and find required exploit in a short term. If any exploits exist, it considerably increases the number of the attacks, risks of possible hackers' invasion and significantly reduces security. In this case the software information vulnerability will increase. We suggest the following formula for numerical *assessment of software vulnerabilities (ASV)*

$$\mathrm{ASV} = \sum_{i=1}^{n} S_i \cdot D_i \cdot 10^{m_i} \qquad (50.9)$$

where *n* is a number of vulnerabilities in similar system, D_i is the number of risk days for *i th* vulnerability with releases of updates, S_i is criticality degree for *i th* vulnerability, m_i is a quantity of the existing exploits for *i th* vulnerability.

The formula (50.9) takes the following form taking into account vulnerabilities of a system program configuration in which SWS is used

$$\mathrm{IASV} = \sum_{i=1}^{n} S_i \cdot D_i \cdot 10^{m_i} + \sum_{j=1}^{k} S_j \cdot D_j \cdot 10^{m_j} \qquad (50.10)$$

where *k* is a number of vulnerabilities in the used configuration (these are vulnerabilities of an operating system, the Web server, the database server, etc.), D_j is the number of risk days for j^{th} vulnerability in the used configuration, S_j is criticality degree for j^{th} vulnerability on ten-point scale, m_j is a quantity of the existing exploits for j^{th} vulnerability.

The formula (50.10) allows to get an *integrated assessment of software vulnerabilities (IASV). The IASV* grade characterizes information security of SWS. If the *IASV* grade is lower, the information security of SWS is higher and the system is protected from unauthorized invasion better. The calculated on formula (50.10) *I*ASV grade can be used for security level comparing of different systems.

50.7. Common conclusions and the following steps

The chapter describes the proposed approach which is titled as Methodology of Technology Oriented Assessment of Software Reliability and Security (TOAStRaS). In this chapter a concept, tasks and some solutions for the TOAStRaS methodology are formulated. As a concept we offer embedding technology-oriented dependability assessment and management into business processes of SWS development in restricted resources conditions. Embedding technology-oriented dependability assessment includes the SWS reliability assessment and the SWS security assessment. The TOAStRaS methodology is based on four principles. The first principle consists of embedding evaluation procedures and dependability management procedures into SWS life cycle processes and models. The second principle consists of embedding project-oriented selection, complexion and parameterization of models to evaluate achieved dependability indexes at the final stage of the SWS development. The third principle consists of embedding dependability evaluation methods and instrumented aids into instrumental tools of implementation and supporting processes of SWS life cycle. The fourth principle consists of embedding methods and aids of data search, accumulation, storage, processing and analysis applied in similar projects and systems into SWS under development, dependability evaluation processes using big data storages, principles and technologies. These principles implement the processing aspect, the project aspect, the instrumental aspect and the parametrical aspect. The realization of the TOAStRaS methodology assumes the solution of ten tasks. Some tasks are solved in this chapter.

In particular, in TOAStRaS methodology context for the solution of the first task we offer metrics-based method of software requirements correctness improvement within the framework of proposed concept. For the solution of the fourth task we offer complexity-based prediction of faults number for software modules ranking before testing: technique and case study. For the solution of the ninth task we offer big data search technique for similar program system to assess dependability in reliability and security of SWS under development context.

The offered metrics-based method of software requirements correctness improvement has been applied in the course of functional requirements inspection within corporate systems. Through ranked script of the requirements of top priority and the highest complexity has been applied. Methodology enables to increase quantity of detectable actual and potential faults by 9% in average comparing with total verification of the requirements disregarding their complexity, improve requirement correctness to the same extent and to avoid significant costs which might be incurred to eliminate such faults at further stages of development.

The method uses an admission that more complex requirements contain more actual and potential faults. Requirement complexity evaluation metric is offered within the SRM framework. The metric calculation consists of summing up quantitative characteristics of internal and external requirement complexity. The metric evaluates requirement complexity in complexity units. The obtained evaluation enables to determine total system complexity and adjust terms and costs of development. Modifications or decomposition of complex requirement enables to reduce total system complexity.

The method applies RCM metric to evaluate requirements complexity. The method uses a double sorting procedure basing on requirements' priority and complexity evaluation. The method is comparatively advantage to the total verification due to possibility to select the most complicated requirements with the highest priority by means of generating ranked list of software modules. The method enables to improve requirements' correctness and defectiveness. And also it provides an evaluation of correctness and defectiveness achieved indexes. The application of the method in the course of requirements verification provides substantial technological and economical effect.

The method has been developed taking into account two principles: the principle of artefacts complexity evaluation and the principle of maximum possible reliability improvement in restricted resources situation. Further researches to be held in two directions. The first lies in achieving and providing organic building-in with different SWS development business processes by means of making method patterns with defining roles and activities of the developers. The second direction lies in development of methods and applications for big data analysis occurring at all stages of the SWS LC processes.

The technique of complexity-based prediction of faults number for ranking software modules is proposed for technological and resources oriented approach to reliability management at all the lifecycle stages in restricted resources environment. Supposed statement that the most complicated modules contain the bulk quantity of faults is proved experimentally. The method is applicable after initial code has been written but before its testing commencement. The method's input data consists of complicatedness indicators for individual modules of the system under development by complexity metrics of object-oriented code. The output data represents a sample of ranked modules of certain dimensions with a certain number of faults. Sample dimensions are only restricted by testing resources.

The proposed method is aimed to efficient testing with restricted resources. Verification of the method using a representative sample of experimental data has demonstrated its efficient operability. Proposed efficiency factor depends on number of metrics by which modules have simultaneous high complicatedness rates and on modules sample dimensions. The achieved results enable to control process of achieving required reliability with restricted resources by means of allocating the testing efforts to a certain number of modules with the highest faults quantity.

End-point analysis enabled to identify two tendencies. The first tendency consists of efficient modules selection will be maximum for modules with top complicatedness indications involving all the involved metrics simultaneously. The second tendency consists of selection of the least part of the most complicated modules is the most efficient. Thus the method's efficiency depends on metrics quantity determining simultaneously high complicatedness. Selected modules and tests for them enables to estimate testing expenditures to detect a certain quantity of faults. Since restricted resources prevent testing all the modules, the unrevealed faults may cause risk of losses. Risks calculation and comparing them with testing expenditures may enable managers to find adequate and economically grounded solutions. The method results may be helpful not only for efficient testing efforts allocations, but also for efficient application of other code verification methods, e.g. inspections, statistical analysis, etc.

The proposed set of known metrics for code complicatedness evaluation is an initial, or starting, one. Software corporations' experts may apply their unique corporate metrics set, proven at practice. The obtained results are applied not only for code testing but they also can be applicable for other verification methods, such as inspections, control, audits and statistical analysis. It is not still clarified, whether selected modules contain fault-free modules, their quantity and their connection with the complicatedness level.

The set of such modules and selected tests for them enable to estimate *apriori* testing costs by means of summing up testing time for all the selected modules. Maximization of revealed faults number and improved systems operating reliability may reduce expenditures and increase developers revenues. Risks of faults triggering at the operation stage may be reduced. Testing expenditures will be efficient investments into the systems reliability.

The proposed method is simple enough to be applied in practice. The method does not demand any additional data except the code complicatedness evaluation for the system under development. The method may be totally automated. Revealed application restriction concerns systems with 99% faulty modules ratio. The inefficient application of this method may occur in such only. Prospective direction of further research may be connected with the implementation of the proposed method into business processes involving various methodologies of software systems development.

The implementation of the TOAStRaS methodology, in particular, prediction of SWS reliability and security can be based on the information, which is extracted from big data storages by using the software agent. It took about two hours to process data taken from twenty SWS by means of the agent. The search of similar programs represents practical value for a project manager and personnel of the SWS testing group. The agent may be adapted by software companies to take into account particular features of the developed SWS.

As a result the technique has been suggested to search and analyse similar programs. The similarity principle is based on complexity and structure SWS metrics and metrics of program language similarity. Calculation formulas have been suggested to assess group and average relative deviation rates describing the SWS similarity.

Case study allowed to obtain some experimental results. A system has been identified with minimum (5,1 – 9,0 %) and average 6,5% relative deviation of metrical rates among twenty explored systems. Obtained results confirm the supposition that systems with known reliability indexes similar to the SWS under development may be found in a great quantity of experimental data kept in big data storage to assess, verify and predict its reliability.

The implementation of the TOAStRaS methodology, in particular, BDA based assessment of software security has been offered.

We suggest to calculate a resultant deviation as an average value of deviations in architecture, program realization and configuration of SWS. The resultant deviation characterizes the similarity of the compared systems numerically. A system with the minimum deviations will resemble new SWS in the best way.

After similar SWS with definite number and types of vulnerabilities has been found, we will be able to estimate security of the new SWS numerically.

We suggest a formula which allows to get an integrated assessment of software vulnerabilities. The integrated assessment characterizes the information security of SWS. If the integrated assessment is lower, the information security of SWS is higher and the system is protected from the unauthorized invasion better. The integrated assessment can be used for security level comparing of different software systems.

Future research will be aimed at the extension of the existing big data methods and creation of new ones which can be used for software reliability and security estimation. Further research will be also focused on the formal definition of similar SWS including functionality and applied technology. The software agent and the technique of similar program search can be integrated with procedures of SWS development for the obtaining integrated assessment of software dependability. The application of big data convolution technique to multi-dimensional matrix of SWS metrical rates draws particular attention.

It is necessary to reduce dimensionality of metric estimates array to find out an influence of different SWS characteristics on the faults and vulnerabilities. The found dependences will allow to predict SWS reliability and security level. Further research will be focused on the process of software dependability management on the basis of the suggested assessment methods.

Questions and tasks for self-control

1. What is the software dependability?
2. What is big data analysis based assessment of software reliability?
3. What is big data analysis based assessment of software security?
4. Explain the connection between security and big data analysis based assessment of software reliability.
5. What is methodology TOAStRaS?
6. On what principles is the methodology based?
7. What aspects does the methodology realize?
8. List the tasks of the methodology.
9. Give the definition of the main purpose of metric-based method of software requirements correctness improvement.
10. What are software requirements specification and priority?
11. What is requirement complexity metric?
12. On which procedures is metric-based method of software requirements correctness improvement based?
13. Define the algorithm of SWS requirement correctness improving.

14. What are input and output data for the method of requirement correctness improving?
15. What opportunity does the method for increase in efficiency of requirements construction give?
16. What is the main purpose of complexity-based prediction technique of faults number for software modules ranking?
17. What are input and output data for the complexity-based prediction technique of faults number?
18. What steps are supposed to proceed complexity-based prediction technique?
19. What experimental results have been obtained using complexity-based prediction technique?
20. What opportunity does the technique for increase in efficiency of code verification give?
21. What big data storage of reliability and security information do you know?
22. What is the software systems similarity principle?
23. What functions are performed by the agent for search of similar programs in big data storage?
24. What additional functions can the agent perform in big data storage?
25. List steps of technique of similar programs search in big data storage.
26. What experimental results have we obtained using of technique for search of similar programs in big data storage?
27. What systems for the of SWS security analysis at stages of SWS life cycle do you know?
28. Define the main purpose of offered method of software vulnerabilities detection on the basis of the similar programs principle.
29. Define features of the assessment of software vulnerabilities detection using the offered method.
30. Give a formula for calculating the assessment of software vulnerabilities.
31. Give a formula for calculating integrated assessment of software vulnerabilities.
32. What further steps are expected in future researches?

References

1. Gantz, J., Reinsel D.: The digital universe in 2020: Big Data, Bigger Digital Shadow s, and Biggest Grow in the Far East, http://www.emc.com/collateral/analyst-reports/idc-the-digital-universe-in-2020.pdf

2. BSA | The Software Alliance. What’s the Big Deal With Data? https://www.bsa.org

3. ductivity. McKinsey Global Institute, http://www.mckinsey.com/~/media/McKinsey/dotcom/Insights20and%20pubs/MGI/Research/Technology%20and%20Innovation/Big%20Data/MGI_big_data_full_report.ashx

4. Economist Intelligence Unit. The Deciding Factor: Big Data & Decision Making. Point Of View, http://bigdata.pervasive.com/Solutions/Telecom-Analytics.aspx

5. Carlos, O., Adrian, P.: Research Directions for Engineering Big Data Analytics Software. Florida Institute of Technology. Published by the IEEE Computer Society, pp. 13--19, http://barbie.uta.edu/~hdfeng/bigdata/Papers/Research%20Directions%20for%20Engineering%20Big%20Data%20Analytics%20Software.pdf (2015)

6. Rademakers, F.: ROOT for Big Data Analysis. Workshop on the future of Big Data management, London, UK (2013)

7. Leskovec, J., Rajaraman, A., Jeffrey D.: Mining of Massive Datasets. Stanford Univ., Milliway Labs., p. 495 (2014)

8. Tera-PROMISE Home, http://openscience.us/repo/defect/ck/

9. NASA'S DATA PORTAL, https://data.nasa.gov/

10. Software Testing and Static Code Analysis, http://www.coverity.com/

11. Topcoder | Deliver Faster through Crowdsourcing, https://www.topcoder.com/

12. E. Eric. Domain-Driven Design - Tackling Complexity in the Heart of Software. Addison Wesley, 560 p. (2013)

13. Frischknecht Ch. Top 10 Software Fails Of 2014. Access mode: http://www.tricentis.com/blog/2014/12/18/top-10-software-fails-of-2014 (2014)

14. G. Carrozza, R. Pietrantuono, S. Russo. Defect Analysis in Mission-critical Software Systems: a Detailed Investigation. J. Softw. Evol. and Proc. ; 2012/07/12, Vol.2, pp. 1-28. DOI: 10.1002/smr (2012)

15. Cotroneo, D., Pietrantuono, R., Russo, S. Testing techniques selection based on ODC fault types and software metrics. The Journal of Systems and Software, Vol.86, pp. 1613-1637 (2013)

16. Cotroneo, D., Pietrantuono, R., Russo, S. Combining Operational and Debug Testing for Improving Reliability. The Journal of Systems and Software, Vol. 62, no. 2, pp. 408- 423 (2013)

17. V.S.Kharchenko, V.V.Sklar, O.M.Tarasyuk. Methods for modeling and evaluation of the quality and reliability of the software. Kharkov: Nat. Aerospace. Univ."KhAI", 159 p. (2004)

18. Yaremchuk, S., Kharchenko, V. Complexity-based Prediction of Faults Number for Software Modules Ranking Before Testing: Technique and

Case Study. Proceedings of the 12th International Conference, ICTERI 2016. Kyiv, Ukraine, Vol. 1614, pp. 461–474 (2016)

19. V. Yakovyna, D. Fedasyuk, O. Nytrebych, Iu. Parfenyuk, V. Matselyukh. Software reliability assessment using high-order Markov chains. International Journal of Engineering Science Invention. — Vol. **3**, Issue 7. – pp. 1–6 (2014)

20. Yakovyna V. S. Software failures prediction using RBF neural network. Researches of the Odessa national polytechnical university. — Vol. **2**(46). – pp. 111–118 (2015)

21. D. A. Maevsky, S. A. Yaremchuk, L. N. Shapa. A method of a priori software reliability evaluation. Reliability: Theory & Applications. — Vol. **9**. – № 1(31). – pp. 64 – 72 Access mode: http://www.gnedenko-forum.org/Journal/2014_1.html (2014)

22. S. A. Yaremchuk, D. A. Maevsky. The Software Reliability Increase Method. Studies in Sociology of Science. Vol. **5**, No. 2 pp. 89 – 95. Access mode: http://www.cscanada.net/index.php/sss/article/view/4845 (2014)

23. 12. Eric J. Braude. Software Engineering An Object-Oriented Perspective. John Wiley and Sons, 575 p. (2011)

24. Kelly, John C., Joseph S. Sherif, Jonathon Hops. An Analysis of Defect Densities Found During Software Inspections. Journal of Systems and Software Vol. 17(2), pp. 111-117 (1992)

25. Hofmann, Hubert F., Franz Lehner. Requirements Engineering as a Success Factor in Software Projects. IEEE Software Vol. **18**(4):58-66 (2001)

26. IEEE 830-1998: IEEE Recommended Practice for Software Requirements Specifications. Los Alamitos, CA: IEEE Computer Society Press (1998)

27. IEEE 1233-1998: IEEE Guide for Developing System Requirements Specifications. Los Alamitos, CA: IEEE Computer Society Press (1998)

28. IEEE/ISO/IEC 29148-2011: Systems and software engineering – Life cycle processes – Requirements engineering (2011)

29. Wiegers K., Beatty J. Software Requirements, 3rd Edition. Microsoft Press, 673 p. (2013)

30. IEEE 1061-1992: A Software Quality Metrics Methodology. Los Alamitos, CA: IEEE Computer Society Press (1992)

31. Shahid Iqbal, M. Naeem Ahmed Khan. Yet another Set of Requirement Metrics for Software Projects. International Journal of Software Engineering and Its Applications Vol. **6**, No. pp.19-28 (2012)

32. Mohd. Haleem, Mohd.Rizwan Beg, Sheikh Fahad Ahmad. Overview of Impact of Requirement Metrics in Software Development Environment. International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Vol. **2**, No 5, pp. 1811-1815 (2013)

33. Cinque, M., Gaiani, C., Stradis, D., Pecchia, A., Pietrantuono R., Russo, S.: On the Impact of Debugging on Software Reliability Growth Analysis: A Case Study. Computational Science and Its Applications (ICCSA), vol. 8583 of the series Lecture Notes in Computer Science, pp 461--475 (2014)

34. Kharchenko, V. S., Tarasyuk, O. M., Sklyar, V.V.: The Method of Software Reliability y Growth Models Choice Using Assumptions Matrix. Proceedings of 26-th Annual Int. Computer Software and Applications Conference, COMPSAC, Oxford, England, pp. 541--546 (2002)

35. Cotroneo, D., Pietrantuono, R., Russo, S.: Combining Operational and Debug Testing for Improving Reliability. The Journal of Systems and Software, vol. 62, no. 2, pp. 408-- 423 (2013)

36. Mayevsky, D. A.: A New Approach to Software Reliability. Lecture Notes in Computer Science. Software Engineering for Resilient Systems. Berlin, Springer, no. 8166, pp. 156--168 (2013)

37. Yakovyna V. S.: Software failures prediction using RBF neural network, http://pratsi.opu.ua/app/webroot/articles/111-118.pdf

38. Yaremchuk, S. O.: Models, methods and technology of a priori estimation reliability indices of accounting and analytical information systems. Dissertation for scientific degree of candidate of technical sciences in specialty Information technology. – Odessa National Polytechnic University, Odessa, Ukraine, 210 p. (2015) (In Ukrainian)

39. Cotroneo, D., Pietrantuono, R., Russo, S.: Testing techniques selection based on ODC fault types and software metrics. The Journal of Systems and Software, no. 86, pp. 1613--1637 (2013)

40. Mausa, G., Grbac, T. G., Basic, B. D.: Multivariate Logistic Regression Prediction of Fault-Proneness in Software Modules. MIPRO 2012/CTI, pp. 813--818 (2012)

41. Yu1, L., Mishra, A.: Experience in Predicting Fault-Prone Software Modules Using Complexity Metrics. Quality Technology & Quantitative Management, vol. 9, no. 4, pp. 421--433 (2012)

42. Sandhu, P. S., Khullar, S., Singh, S., Bains, S. K., Kaur, M., Singh, G.: A Study on Early Prediction of Fault Proneness in Software Modules using Genetic Algorithm. International Journal of Computer, Electrical, Automation, Control and Information Engineering, vol. 4, no. 12, pp. 1891--1896 (2010)

43. Kaur, S., Mahajan, M., Sandhu, P. S.: Identification of Fault Prone Modules in Open Source Software Systems using Hierarchical based Clustering. ISEMS, Bangkok, pp. 238--244 (2011)

44. Najadat, H., Alsmadi, I.: Enhance Rule Based Detection for Software Fault Prone Modules. International Journal of Software Engineering and Its Applications, vol. 6, no. 1, pp. 75--86 (2012)

45. Kaur, I.: A Compound Metric for Identification of Fault Prone Modules. IOSR Journal of Computer Engineering (IOSR-JCE), vol. 17, Issue 6, Ver. V, pp. 31--35 (2015)

46. Ahmad, N., Khan, M., Islam, S.: Optimal Allocation of Testing Resource for Modular Software based on Testing-Effort Dependent Software Reliability Growth. ICCCNT-12, Coimbatore, India (2012)

47. Pavithra, M.: Optimal Testing Resource Allocation Problems in Software System using Heuristic Algorithm. Bonfring International Journal of Software Engineering and Soft Computing, vol. 2, no. 4, pp. 1--9 (2012)

48. Nasar, M., Johri, P.: Testing and Debugging Resource Allocation for Fault Detection and Removal Process. International Journal of New Computer Architectures and their Applications, no. 4, pp. 193--200 (2014)

49. 15th Annual CSI/FBI Computer Crime and Security Survey. Executive Summary. – CSI, FBI, 2010. – 17 c.

50. Popov, P. T., Netkachov, A. & Salako, K. (2014). Quantification of the Impact of Cyber Attack in Critical Infrastructures. *Lecture Notes in Computer Science*, 8696, pp. 316-327. doi: 10.1007/978-3-319-10557-4_35

51. Netkachov, O., Popov, P. T. & Salako, K. (2014). Model-based Evaluation of the Resilience of Critical Infrastructures under Cyber Attacks. *Lecture Notes in Computer Science*, 8985, pp. 231-243. doi: 10.1007/978-3-319-31664-2_24

52. Netkachova, K., Bloomfield, R. E., Popov, P. T. & Netkachov, O. (2015). *Using Structured Assurance Case Approach to Analyse Security and Reliability of Critical Infrastructures*. Paper presented at the SAFECOMP 2015 Workshops, ASSURE, DECSoS, ISSE, ReSA4CI, and SASSUR, 22-09-2015, Delft, Netherlands.

53. Netkachova, K., Müller, K., Paulitsch, M. & Bloomfield, R. E. (2015). *Security-Informed Safety Case Approach to Analysing MILS Systems*. Paper presented at the International Workshop on MILS: Architecture and Assurance for Secure Systems, 19-21 January 2015, Amsterdam, The Netherlands.

54. Bloomfield, R. E., Netkachova, K. & Stroud, R. (2013). Security-Informed Safety: If it's not secure, it's not safe. Paper presented at the 5th International Workshop on Software Engineering for Resilient Systems (SERENE 2013), 03rd - 04th October 2013, Kiev, Ukraine.

55. Gashi, I., Bloomfield, R., Bloomfield, R. E. & Stroud, R. (2012). How secure is ERTMS? Paper presented at the Workshop on Dependable and Secure Computing for Large-scale Complex Critical Infrastructures (DESEC4LCCI), 25 September 2012, Herrenkrug, Germany.

56. Software engineering institute of Carnegie Mellon University. http://www.cert.org/vulnerability-analysis/knowledgebase/index.cfm

57. Wagner David, Foster Jeffrey S., Brewer Eric A., Aiken Alexander. A first step towards automated detection of buffer overrun vulnerabilities // In: Network and Distributed System Security Symposium. San Diego, CA, February 2000. P. 3-17.

58. David A. Wheeler. Flawfinder. http://www.dwheeler.com/flawfinder/

59. Viega J., Bloch J. T., Kohno Y., Mcgraw G. Its4: a static vulnerability scanner for C and C++ code // In: Computer Security Applications. ACSAC '2000. 16th Annual Conference. 2000. P. 257-267.

60. Fortify Software, Inc. RATS – Rough Auditing Tool for Security. https://www.fortify.com/ssa-elements/threat-intelligence/rats.html

61. David Hovemeyer, William Pugh. Finding bugs is easy // SIGPLAN Not. 2004. Vol. 39, No. 12. P. 92-106.

62. Klocwork. Systems of the source code analysis. http://www.klocwork.com/products/

63. VMProtect Software. Vmprotect software protection. http://vmpsoft.com/

64. Oreans Technologies. Code virtualizer: Total obfuscation http://www.oreans.com/codevirtualizer.php

65. Safengine Protector. http://www.safengine.com/en-us/products/protector

66. Dotfuscator Community Edition 4.0. http://msdn.microsoft.com/ru-ru/library/ms227240%28v=vs.90%29.aspx

67. Nginx security advisories. http://nginx.org/en/security_advisories.html

68. Common Vulnerabilities and Exposures. http://www.cve.mitre.org/

69. National Vulnerability Database. https://nvd.nist.gov/

70. Vulnerabilities. http://www.securityfocus.com/

71. Rapid7 Defines the Next-Generation Platform for Security and IT Professionals. https://www.rapid7.com/

72. Offensive Security's Exploit Database Archive. https://www.exploit-db.com/

73. The Exploit Database. https://www.offensive-security.com/community-projects/the-exploit-database/

51 SEMI-MARKOV'S MODELS OF IAAS CLOUDS AVAILABILITY AND SECURITY

51.1 Availability Models for IaaS Clouds with Multiple Pools

51.1.1 Taxonomy Metamodel for Availability Analysis of IaaS Cloud

Cloud Infrastructure is one of the most widely used model of Cloud Computing. Therefore, modern large cloud service providers (CSPs) such as Amazon, Microsoft, Google, Rackspace need approaches and models for the quantification of reliability level. In particular, Infrastructure as a Service (IaaS) Cloud provider's data centers try to ensure quality of service (QoS) by using different approaches for determining of their *availability* level. Significance of issue ensuring of availability of the IaaS Cloud can hardly be exaggerated. Nowadays large CSPs try to use multiple pools of physical machines (PMs) in order to maintain normal operation of cloud's components on quite a long period of time. However, with a larger number of PMs number of states for stochastic model of an IaaS Cloud also increase; the model ought to include a large number of parameters while still being tractable [1]. Therefore, researchers ought to focus to create taxonomy of the cloud infrastructure *metamodel* in order to solve availability task for IaaS Cloud.

Note that the architecture of an IaaS Cloud is not tied to a real cloud implementation [2]. Suppose that researchers have used a simple cloud infrastructure with certain number of PMs. To reduce power consumption, cooling and infrastructure costs, PMs are grouped into three pools such as: hot, warm and cold pools. Assume that hot pool consists from turned on and running PMs; warm pool contains turned on, although not ready physical machines; cold pool consists from turned off PMs. Moreover, this architecture has certain number of virtual machines (VMs), which are deployed on PMs. Deployment of VMs on base PMs allows to reduce power consumption and to maintain enough high performance of the cloud implementation. In difference from other, proposed concept for maintaining of availability for a cloud infrastructure bases on use of two additional systems, namely *Technical State Control System (TSCS)* and Resource Provisioning Decision Engine (RPDE) [3]. Our IaaS Cloud should be used TSCS, which is working in monitoring and diagnostic modes. In this case, these modes as regarded as an organization form of constant control of the significant parameters that the determinate not only the PMs performability, but also affect cloud infrastructure readiness to make effective intended use [4]. It's obviously, that monitoring and diagnostic sub-systems provide repair facilitated by information which is needed to repair and migration of PMs from one pool to other. As described in [2], RPDE tries to find a PM that can accept the job provisioning.

Figure 51.1 shows the portions of the *taxonomy metamodel* for availability analysis of IaaS Cloud. Researchers in order to deal with the complexities of metamodeling should work in the paradigm of four models, such as scalability, performance, flexibility (elasticity), power consumption. Each model ensures the overall metamodel by input parameters, namely initial number of PMs for each pool (scalability model), power consumption for each PM (power consumption model), management metrics values, search rates (flexibility model) and failure rates, repair rates, migration rates, number of repair facilities (reliability model). In other words, output parameters of these models are input parameters for *metamodel*. At the same time values of design and temporal parameters of such models can be experimentally measured. The stages of metamodeling are colorfully shown by this figure. According to the illustrated Fig. 51.1, we will try to create *analytical models* with considering states and stochastic changing of all times failures, repairs and migrations of PMs.

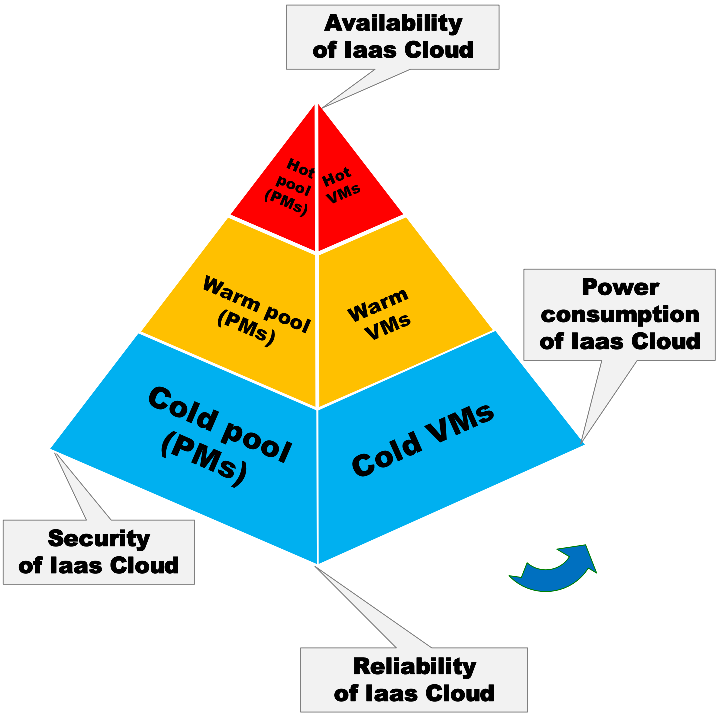


Fig. 51.1. Taxonomy metamodel for availability analysis of IaaS Cloud

On this basis, we will construct *Semi-Markov model*s for availability analysis of an IaaS Cloud with three pools. Therefore, it is proposed to describe various options of interactions of PMs at availability-model level.

51.1.2 Analytical Availability Models for IaaS Cloud with Three Pools

Let's consider two interesting analytical models of an IaaS Cloud. Figure 51.2 shows a *Semi-Markov (SM) model for availability analysis* of the IaaS Cloud with three pools (hot, warm and cold) and three PMs in each pool.

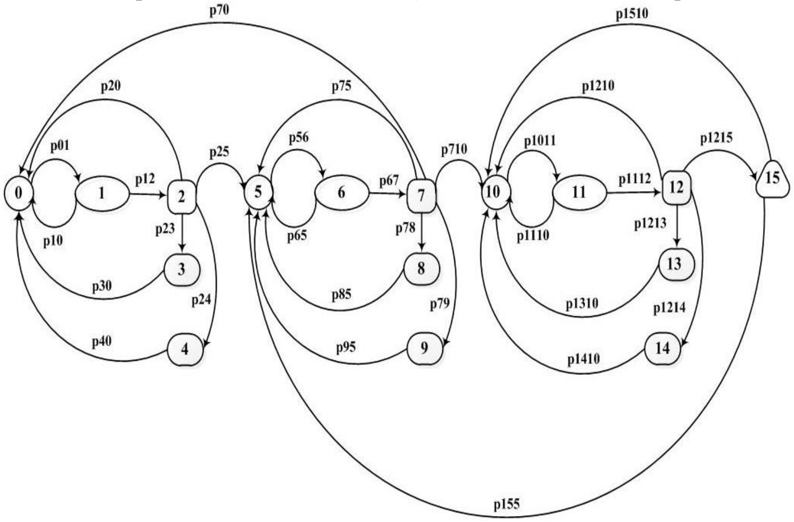


Fig. 51.2. SM model for the availability analysis of the IaaS Cloud with three PMs in each pool

In our modeling we use the following assumptions and limitations:

- hot, warm, and cold pools contain identical PMs [5]. If a hot PM fails the failed PM is replaced by available (non-failed) PM from warm or cold pools, respectively;
- we assume that periodic *technical state control* (CTS) of hot PMs is operated during a time interval, which lasts τ_c;
- to analyze the availability of the IaaS Cloud we also assume that all times to failure of all PMs are *exponentially distributed*. Typically, mean time to failure (MTTF) of warm PMs ($1/\lambda_w$) is higher than MTTF of hot PMs ($1/\lambda_h$) by a factor of two to four [3]. At the same time MTTF of cold PMs

is a very lower than $1/\lambda_w$. However, for process of SM modeling we will use only MTTF of hot PMs, considering quite high *availability* level of warm and cold PMs;

- moreover, in real situations providers haven't enough time for repair of failed hot PMs, as well as they haven't enough number of repair facilities. Therefore, we also assume that all times to repair are *not exponentially distributed*. In this occasion we have preferred to use *Erlang-k distribution*, where $k = 2,\ 3$ [6]. Parameter $1/\mu$ is mean time to repair (MTTR) of a PM;
- available PMs can migrate from warm and cold pools to hot pool. We also assume that all times to migration (migration delays) of PMs are exponentially distributed. For modeling we have used mean time to migration (MTTM) of PMs from warm ($1/\gamma_{wh}$) and cold ($1/\gamma_{ch}$) pools to hot pool;
- the migrations of PMs to hot pool are implemented when providers can search non-failed warm or cold PMs with mean time to searches (MTTSs) $1/\delta_w$ and $1/\delta_c$;
- we consider that IaaS Cloud becomes *unavailable* when the SM model enters the state S_{15}.

Suppose that this infrastructure is operated during a time interval $t \in [0,T]$ and at the initial moment $t = 0$ the IaaS Cloud is ready for using (state S_0). The transition from state S_0 to state S_1 occurs at *fixed nonrandom time* $\tau_c < T$, where parameter T is operation time of IaaS Cloud between two *periodic controls of technical state*. The state S_1 is state of CTS. Note that the periodic CTS includes monitoring and diagnostic operations of hot PMs and conduct by means of using TSCS. If third hot PM is available the SM model returns from state S_1 to state S_0.

Otherwise when the TSCS detects a failure, model goes to state S_2 with rate λ_h. In state failure of the third hot PM, model tries to search non-failed warm PM (transition from state S_2 to state S_3) with rate δ_w or cold PM (transition from state S_2 to state S_4) with rate δ_c. When warm or cold PMs are available, model transforms from state S_3 to state S_0 or from state S_4 to state S_0 respectively.

If the warm and cold pools are empty, repair facility tries to recover the failed hot PM, that is model goes from state S_2 to state S_0 with repair rate μ . When recovery the third failed PM is impossible, model transforms from state S_2 to state S_5 with overall failure rate $3\lambda_h$. It means that next steps of modeling as regards states of S_5 – S_9 for second hot PM and states of S_{10} – S_{12} for first hot PM are repeated. Note that in this case we can maintain that transition from state S_7 to state S_{10} and transition from state S_{12} to state S_{15} are implemented with overall failure rates $2\lambda_h$ and λ_h respectively. We also consider, that the model will transition from state S_{12} to state S_{15}, when the last hot PM fails.

To solve this task in the following we are inclined to use method of transformation of the SM models into embedded Markov chains [6]. For this type of *models*, the transitions of process from state i to state j occur through unit time. Therefore, the transitions of this *SM process* are interpreted as follows. Since CTS performs within fixed deterministic period of time T , consequently transition from state S_0 to state S_1 is given by:

$$Q_{01}(t) = \begin{cases} 0, t < T, \\ 1, t \geq T. \end{cases}$$

The transition from state S_1 to state S_0 is then given by:

$$Q_{10}(t) = \begin{cases} 0, t < \tau_c, \\ 1, t \geq \tau_c. \end{cases}$$

The other similar transitions can be got as follows:

$$Q_{56}(t) = Q_{1011}(t) = \begin{cases} 0, t < T, \\ 1, t \geq T, \end{cases}$$

$$Q_{65}(t) = Q_{1110}(t) = \begin{cases} 0, t < \tau_c, \\ 1, t \geq \tau_c. \end{cases}$$

At the same time, probabilities of *sudden failures* of hot PMs at random times for transitions from state S_1 to state S_2, from state S_6 to state S_7 and from state S_{11} to state S_{12} are given by:

$$Q_{12}(t) = Q_{67}(t) = Q_{1112}(t) = 1 - e^{-\lambda_h t}.$$

Implementations of transitions from state S_2 to state S_0, from state S_7 to state S_5, from state S_{12} to state S_{10} and from state S_7 to state S_0, from state S_{12} to state S_5, from state S_{15} to state S_{10} depend from time to repair of the hot PMs. Therefore, in these cases, distribution functions of repair time are given by:

$$Q_{20}(t)=Q_{75}(t)=Q_{1210}(t)=1-(1+\mu t)e^{-\mu t},$$

$$Q_{70}(t)=Q_{125}(t)=Q_{1510}(t)=1-\left(1+\mu t+\frac{(\mu t)^2}{2}\right)e^{-\mu t}.$$

For our *SM availability model*, we assume that distribution functions of search time of non-failed warm and cold PMs respectively are given by:

$$Q_{23}(t)=Q_{78}(t)=Q_{1213}(t)=1-e^{-\delta_w t},$$

$$Q_{24}(t)=Q_{79}(t)=Q_{1214}(t)=1-e^{-\delta_c t}.$$

Similarly, *distribution functions of migration time* for warm and cold PMs respectively are given by:

$$Q_{30}(t)=Q_{85}(t)=Q_{1310}(t)=1-e^{-\gamma_{wh}t},$$

$$Q_{40}(t)=Q_{95}(t)=Q_{1410}(t)=1-e^{-\gamma_{ch}t}.$$

Then *steady-state availability* [6] of the cloud can be computed as

$$A=\pi_0+\pi_5+\pi_{10}, \qquad (51.1)$$

where π_0, π_5, π_{10} are steady-state probabilities for states S_0, S_5, S_{10}.

On the other hand, the *steady-state availability* A (51.1) is given by [7]:

$$A=\lim_{t\to\infty} A(t),$$

where $A(t)$ – *instantaneous availability* of the cloud infrastructure.

In the overall case steady-state probabilities of *SM availability model* are given by:

$$\pi_0=\frac{\overline{t_0}}{U},\ \pi_5=\beta\frac{\overline{t_5}}{U},\ \pi_{10}=\alpha\beta\frac{\overline{t_{10}}}{U},$$

$$U=\overline{t_0}+\overline{t_1}+p_{12}\left(\overline{t_2}+p_{23}\overline{t_3}+p_{24}\overline{t_4}\right)+\beta\Big[\overline{t_5}+\overline{t_6}+p_{67}\left(\overline{t_7}+p_{78}\overline{t_8}+p_{79}\overline{t_9}\right)+$$
$$+\alpha\left(\overline{t_{10}}+\overline{t_{11}}\right)\Big]+\varepsilon\left(\overline{t_{12}}+p_{1213}\overline{t_{13}}+p_{1214}\overline{t_{14}}+p_{1215}\overline{t_{15}}\right),$$

where

$$\alpha = \frac{p_{67} p_{710}}{1 - p_{1110} - \xi p_{1112}}, \ \beta = \frac{p_{12} p_{25}}{1 - p_{65} - \alpha p_{1112} p_{125} - \nu p_{67}},$$

$$\xi = p_{1210} + p_{1213} + p_{1214} + p_{1215}, \ \nu = p_{75} - p_{78} - p_{79}, \ \varepsilon = \alpha\beta p_{1112},$$

$$p_{12} = p_{67} = p_{1112} = 1 - e^{-\lambda_h \tau_c},$$

$$p_{23} = \delta_w \int_0^\infty (1 + \mu t) e^{-(3\lambda_h + \delta_w + \delta_c + \mu)t} dt, \ p_{24} = \delta_c \int_0^\infty (1 + \mu t) e^{-(3\lambda_h + \delta_w + \delta_c + \mu)t} dt,$$

$$p_{25} = \lambda_h \int_0^\infty (1 + \mu t) e^{-(3\lambda_h + \delta_w + \delta_c + \mu)t} dt,$$

$$p_{75} = \frac{1}{2} \int_0^\infty \mu^2 t \left(\mu^2 t^2 + 2\mu t + 2\right) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(2\lambda_h + \delta_w + \delta_c + 2\mu)t} dt,$$

$$p_{78} = \delta_w \int_0^\infty (1 + \mu t) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(2\lambda_h + \delta_w + \delta_c + 2\mu)t} dt,$$

$$p_{79} = \delta_c \int_0^\infty (1 + \mu t) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(2\lambda_h + \delta_w + \delta_c + 2\mu)t} dt,$$

$$p_{710} = \lambda_h \int_0^\infty (1 + \mu t) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(2\lambda_h + \delta_w + \delta_c + 2\mu)t} dt, \ p_{65} = p_{1110} = e^{-\lambda_h \tau_c},$$

$$p_{75} = \frac{1}{2} \int_0^\infty \mu^2 t \left(\mu^2 t^2 + 2\mu t + 2\right) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(2\lambda_h + \delta_w + \delta_c + 2\mu)t} dt,$$

$$p_{78} = \delta_w \int_0^\infty (1 + \mu t) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(2\lambda_h + \delta_w + \delta_c + 2\mu)t} dt,$$

$$p_{79} = \delta_c \int_0^\infty (1 + \mu t) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(2\lambda_h + \delta_w + \delta_c + 2\mu)t} dt,$$

$$p_{710} = \lambda_h \int_0^\infty (1 + \mu t) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(2\lambda_h + \delta_w + \delta_c + 2\mu)t} dt, \ p_{65} = p_{1110} = e^{-\lambda_h \tau_c},$$

$$p_{1210} = \frac{1}{2} \int_0^\infty \mu^2 t \left(\mu^2 t^2 + 2\mu t + 2\right) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(\lambda_h + \delta_w + \delta_c + 2\mu)t} dt,$$

$$p_{1214} = \delta_c \int_0^\infty (1 + \mu t) \left(1 + \mu t + \frac{(\mu t)^2}{2}\right) e^{-(\lambda_h + \delta_w + \delta_c + 2\mu)t} dt,$$

$$p_{1215}=\lambda_h\int_0^{\infty}(1+\mu t)\left(1+\mu t+\frac{(\mu t)^2}{2}\right)e^{-(\lambda_h+\delta_w+\delta_c+2\mu)t}dt\,,$$

$$p_{125}=1-p_{1210}-p_{1213}-p_{1214}-p_{1215},$$

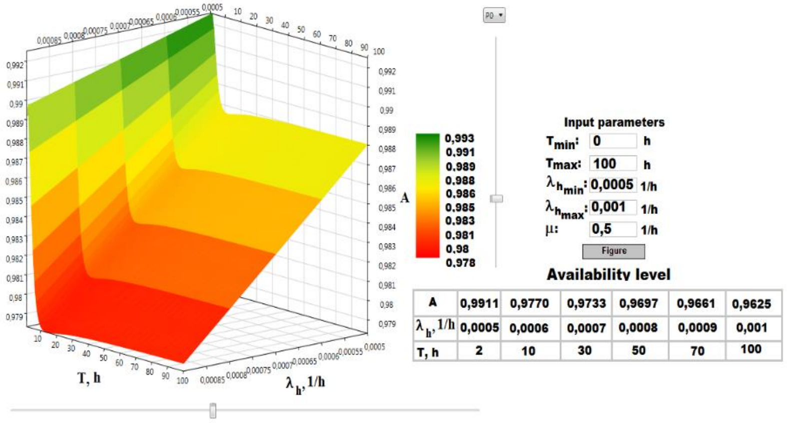
$$\overline{t_0}=\overline{t_5}=\overline{t_{10}}=T\,,\ \overline{t_2}=\frac{3\lambda_h+\delta_w+\delta_c+2\mu}{(3\lambda_h+\delta_w+\delta_c+\mu)^2}\,,\ \overline{t_3}=\overline{t_8}=\overline{t_{13}}=\frac{1}{\gamma_{wh}}\,,$$

$$\overline{t_4}=\overline{t_9}=\overline{t_{14}}=\frac{1}{\gamma_{ch}}\,,\ \overline{t_{15}}=\frac{3}{\mu}\,,\ \overline{t_1}=\overline{t_6}=\overline{t_{11}}=\frac{1}{\lambda_h}\left(1-e^{-\lambda_h\tau_c}\right),$$

$$\overline{t_7}=\int_0^{\infty}(1+\mu t)\left(1+\mu t+\frac{(\mu t)^2}{2}\right)e^{-(2\lambda_h+\delta_w+\delta_c+2\mu)t}dt\,,$$

$$\overline{t_{12}}=\int_0^{\infty}(1+\mu t)\left(1+\mu t+\frac{(\mu t)^2}{2}\right)e^{-(\lambda_h+\delta_w+\delta_c+2\mu)t}dt\,.$$

Plots depending of *steady-state availability* A from failure rates λ_h of hot PMs and operation time T (repair rates μ are constant values) are shown in Fig. 51.3, Fig. 51.4. The values of steady-state availability A are greatly increased by means of increasing of *repair rate* μ and reduction of *failure rate* λ_h of hot PMs, as depicted in Fig. 51.3 and Fig. 51.4.



Input parameters

T_{min}: 0 h

Tmax: 100 h

$\lambda_{h_{min}}$: 0,0005 1/h

$\lambda_{h_{max}}$: 0,001 1/h

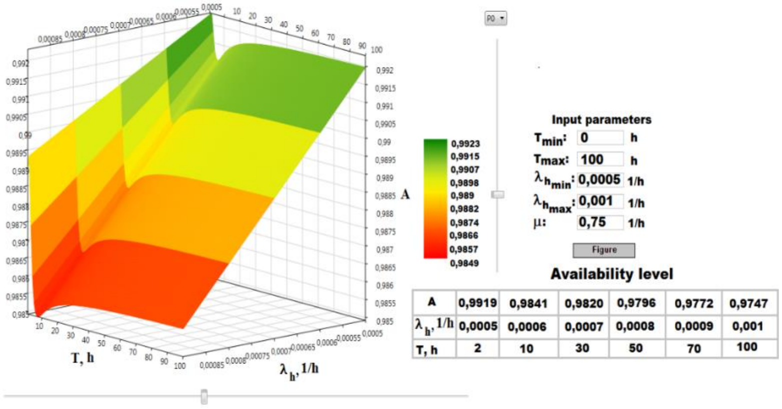
μ: 0,5 1/h

Availability level

A	0,9911	0,9770	0,9733	0,9697	0,9661	0,9625
λ_h, 1/h	0,0005	0,0006	0,0007	0,0008	0,0009	0,001
T, h	2	10	30	50	70	100

Fig. 51.3. Depending of steady-state availability $A(\lambda_h, T)$ for $T=100$ h, $\mu=0{,}5$ 1/h

Let's continue our researches by means of creation more scalable stochastic model for IaaS Cloud. Because with a larger number of PMs in a data center, the overall Cloud Service *availability* increases, leading to lower cost of service downtime [3]. Therefore, within a unified methodological approach we will try to create an improved *SM availability model* of infrastructure with a larger number of PMs.



Input parameters

T_{min}:	0	h
T_{max}:	100	h
$\lambda_{h_{min}}$:	0,0005	1/h
$\lambda_{h_{max}}$:	0,001	1/h
μ:	0,75	1/h

Availability level

A	0,9919	0,9841	0,9820	0,9796	0,9772	0,9747
λ_h, 1/h	0,0005	0,0006	0,0007	0,0008	0,0009	0,001
T, h	2	10	30	50	70	100

Fig. 51.4. Depending of steady-state availability $A(\lambda_h, T)$ for $T = 100$ h, $\mu = 0{,}75\,1/\text{h}$

Additional researches have shown that CSPs wish to increase number of PMs in order to minimize downtime cost and damage business reputation [2], [2], [8]. Perhaps inspired by using stochastic approaches for solution various serious tasks of determining the optimal PM capacity configuration of IaaS Cloud [2], we have been proposing next *SM availability model*.

Assume that our infrastructure contains similar *three pools with ten PMs* in each pool. This SM model for *availability analysis* of the IaaS Cloud is shown in Fig. 51.5. Also suppose that all times to failure of PMs are *exponentially distributed* and *Erlang-k distribution*, where $k = 2$ is general distribution for all times to repair. In spite of the fact, that both models are *SMs models*, we have to take into consideration some interesting features of their implementation.

Unlike first SM model, second *SM availability model* of the IaaS Cloud includes modeling kernel from five states. The states S_0, S_1, S_5 for second model (Fig. 51.5) are the same as the first model (Fig. 51.2). But the difference between kernels of first model and second model is that states S_4, S_9 for first model are states of search of the cold PM, whilst these states for second model are states of failure of the warm PMs. For second model the following group assumptions can take place:

- model contains hot, warm, and cold pools. Every pool consists of ten identical PMs [9]. If a hot PM fails the failed PM is replaced by available (non-failed) PM from warm or cold pools too;
- upon failure of the warm PM, the failed PM is replaced by available (non-failed) PM from cold pool.
- we also assume that periodic technical state control of hot PMs is operated during a time interval, which lasts τ_c;
- to analyze performance and availability of the IaaS Cloud we also assume that all times to failure of all hot and warm PMs are exponentially distributed;
- we also consider that all times to repair are not exponentially distributed. In this case we have used *Erlang-k distribution*, where $k = 2{,}3$ [6]. Parameter $1/\mu$ is MTTR of a PM;
- cloud infrastructure becomes *unavailable* when the SM model enters the state S_{50}.

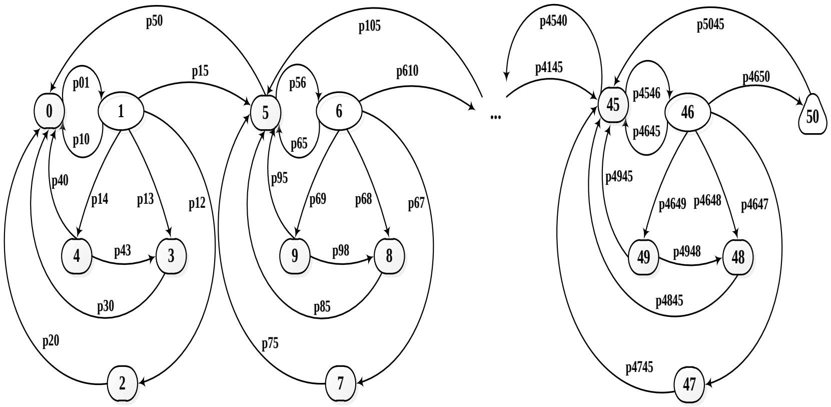


Fig. 51.5. SM model for the availability analysis of the IaaS Cloud with ten PMs in each pool

Therefore, the transitions of *modeling kernel* for second SM model can be written as follows:

$$Q_{01}(t)=Q_{56}(t)=Q_{1011}(t)=Q_{1516}(t)=Q_{2021}(t)=Q_{2526}(t)=Q_{3031}(t)=$$
$$=Q_{3536}(t)=Q_{4041}(t)=Q_{4546}(t)=\begin{cases}0,t<T,\\1,t\geq T,\end{cases}$$

$$Q_{10}(t)=Q_{65}(t)=Q_{1110}(t)=Q_{1615}(t)=Q_{2120}(t)=Q_{2625}(t)=Q_{3130}(t)=$$
$$=Q_{3635}(t)==Q_{4140}(t)=Q_{4645}(t)=\begin{cases}0,t<\tau_c,\\1,t\geq \tau_c,\end{cases}$$

For other functions we can write the following:

$$Q_{12}(t)=Q_{67}(t)=Q_{1112}(t)=Q_{1617}(t)=Q_{2122}(t)=Q_{2627}(t)=Q_{3132}(t)=$$
$$=Q_{3637}(t)==Q_{4142}(t)=Q_{4647}(t)=1-e^{-\delta_w t},$$

$$Q_{13}(t)=Q_{68}(t)=Q_{1113}(t)=Q_{1618}(t)=Q_{2123}(t)=Q_{2628}(t)=Q_{3133}(t)=$$
$$=Q_{3638}(t)==Q_{4143}(t)=Q_{4648}(t)=1-e^{-\delta_c t},$$

$$Q_{14}(t)=Q_{69}(t)=Q_{1114}(t)=Q_{1619}(t)=Q_{2124}(t)=Q_{2629}(t)=Q_{3134}(t)=$$
$$=Q_{3639}(t)==Q_{4144}(t)=Q_{4649}(t)=1-e^{-\lambda_w t},$$

$$Q_{20}(t)=Q_{75}(t)=Q_{1210}(t)=Q_{1715}(t)=Q_{2220}(t)=Q_{2725}(t)=Q_{3230}(t)=$$
$$=Q_{3735}(t)==Q_{4240}(t)=Q_{4745}(t)=1-e^{-\gamma_{wh} t},$$

$$Q_{30}(t)=Q_{85}(t)=Q_{1310}(t)=Q_{1815}(t)=Q_{2320}(t)=Q_{2825}(t)=Q_{3330}(t)=$$
$$=Q_{3835}(t)==Q_{4340}(t)=Q_{4845}(t)=1-e^{-\gamma_{ch} t},$$

$$Q_{43}(t)=Q_{98}(t)=Q_{1413}(t)=Q_{1918}(t)=Q_{2423}(t)=Q_{2928}(t)=Q_{3433}(t)=$$
$$=Q_{3938}(t)==Q_{4443}(t)=Q_{4948}(t)=1-e^{-\delta_c t},$$

$$Q_{40}(t)=Q_{95}(t)=Q_{1410}(t)=Q_{1915}(t)=Q_{2420}(t)=Q_{2925}(t)=Q_{3430}(t)=$$
$$=Q_{3935}(t)==Q_{4440}(t)=Q_{4945}(t)=1-(1+\mu t)e^{-\mu t},$$

$$Q_{50}(t)=Q_{105}(t)=Q_{1510}(t)=Q_{2015}(t)=Q_{2520}(t)=Q_{3025}(t)=Q_{3530}(t)=$$
$$=Q_{4035}(t)==Q_{4450}(t)=Q_{5045}(t)=1-\left(1+\mu t+\frac{(\mu t)^2}{2}\right)e^{-\mu t},$$

$$Q_{15}(t)=1-e^{-\lambda_1 t}, \tag{51.2}$$

$$Q_{610}(t)=1-e^{-\lambda_2 t}, \tag{51.3}$$

$$Q_{1115}(t)=1-e^{-\lambda_3 t}, \tag{51.4}$$

$$Q_{1620}(t)=1-e^{-\lambda_4 t}, \tag{51.5}$$

$$Q_{2125}(t)=1-e^{-\lambda_5 t}, \tag{51.6}$$

$$Q_{2630}(t) = 1 - e^{-\lambda_6 t}, \tag{51.7}$$

$$Q_{3135}(t) = 1 - e^{-\lambda_7 t}, \tag{51.8}$$

$$Q_{3640}(t) = 1 - e^{-\lambda_8 t}, \tag{51.9}$$

$$Q_{4145}(t) = 1 - e^{-\lambda_9 t}, \tag{51.10}$$

$$Q_{4650}(t) = 1 - e^{-\lambda_{10} t}. \tag{51.11}$$

Failure rates are defined for $j = 1,2,...,n_h$ ($n_h = 10$) PMs nodes [5], [6]:

$$\lambda_j = (n_h - i)\lambda_0,\ i = 0,1,..k \text{ (for } k = n_h - 1\text{)}, \tag{51.12}$$

where λ_0 – basic failure rate value for all PMs.

By replacing the λ_j expression (51.12) to the λ_s values in the equations (51.2), (51.3), …, (51.11), we will be finished description of second model.

As can be seen in Fig. 51.3 and Fig. 51.4 in case with three PMs in each pool, IaaS Cloud has quite high of *availability level*. Overall feature for both SM models is identical modeling kernels.

51.2 Approach for Availability and Security Analysis for IaaS Clouds with Three Pools of Physical Machines

51.2.1 Taxonomy Model for Availability and Security Analysis of IaaS Cloud

In the previous part of paper, we offered to use Semi-Markov models in order to analyze IaaS Cloud availability level as *monolithic system* of PMs. However, we should remember, that virtual resource is significant component of infrastructure and we need build *availability* models for infrastructure, which consider numbers of PMs. We will not try describing concrete architecture for IaaS Cloud, too. At the same time, we will consider that user has access to physical and virtual resources of the IaaS Cloud.

Results of analysis are showed, that several famous scientists usually prefer to use Markov models in order to solve different tasks for IaaS Cloud. In fact Continuous Time Markov Chains are main toolkit and predominate among the different mathematical models of *availability* and reliability for IaaS Cloud [9]. However researchers have to take into consideration that this type of models need to describe different events for IaaS Cloud, including sudden and hidden failures, repairs and monitoring services, software vulnerabilities and attack patterns [10]. We can't afford to ignore issues, that to relate to the monitoring of technical and information states of different components for cloud infrastructures. Our researches show, that due to combination of deterministic and stochastic durations into working cycle of IaaS Cloud, this *availability and security model* can be presented by us as Semi-Markov model. We will also try to consider solutions for IaaS Cloud on base of benefits added by description of Semi-Markov process with special states. It gives us a way to conduct quite deep analysis of IaaS Cloud behavior in different negative situations, involving accidents of data centers, failures of PMs or even DoS and DDoS attacks. At the same time, we won't theorize approaches and techniques that related to the availability and security of cloud infrastructure. We will only consider concrete situation for the IaaS Cloud with definite number of PMs.

The approach uses the means of failure detection that two main components could employ. First component is RPDE by which the pools implement capture the resource provisioning decision process [3]. As second component that are functioned in system, namely, TSCS, which is working in monitoring and diagnostic modes, detection and prevention of hacker attack, including DoS and DDoS attacks. Furthermore, CSPs will get possibilities for effective repair and *migration* of physical resources. Figure 51.6 shows the taxonomy model for availability and security analysis of IaaS Cloud.

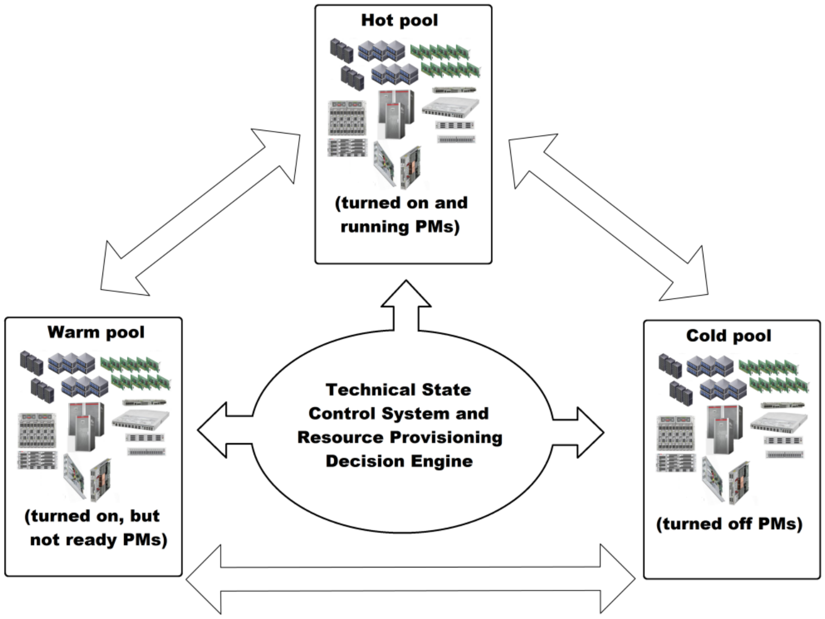

Hot pool
(turned on and running PMs)
Warm pool
(turned on, but not ready PMs)
Cold pool
(turned off PMs)
Technical State Control System and Resource Provisioning Decision Engine

Fig. 51.6. Taxonomy model for availability and security analysis of IaaS Cloud

As this taxonomy (Fig. 51.6) shows, that CSP has to consider different regimes and operational time of TSCS in order to ensure high *availability and security level* of the IaaS Cloud. Moreover, researchers must build mathematical availability and security models for IaaS Cloud considering different types of physical machine’s failures and attack patterns. In [2], the authors presented data about several typical cloud services’ downtime. In particular, they calculated that in 2011 year Amazon cloud services downtime equals about 8,5 days and the corresponding *availability* was about 97,67%. This information teaches us, how to analyze, what is causing the downtime of cloud services. Results of analysis have shown that hidden failures of PMs are one the most important causes of the IaaS Cloud downtime.

Turning to building of models, we will look at specific type of models that is Semi-Markov models with special states. Let's look now at overall methods of solution tasks to assess the IaaS Cloud availability level. Research has shown that in order to solve different tasks we propose to enhance the State Space Modeling Taxonomy [10], [11] with new type of Semi-Markov models. Semi-Markov regenerative process is described by this type of models. In fact we can characterize these models, that relate to the Semi-Markov birth-death processes.

So far, we have tried to use state-space models for monolithic cloud computing system, but not for separate components of IaaS Cloud. It was serious problem, because this model couldn't give accurate assessments for whole system. Now we can obtain the *availability* allocation for all IaaS Cloud and for concrete PM using Semi-Markov modeling approach, by which cloud provider determines possibilities of their own resources in working environment. As an illustrative example, sudden and hidden failures for IaaS Cloud with three pools of PMs and TSCS will be considered by us.

51.2.2 Analytical Availability and Security Models for an IaaS Cloud Considering Hidden Failures of PMs

In this section we present results of analytical and stochastic modelling in order to determine *availability and security level* for IaaS Cloud with three pools of PMs. The case study chosen is a model of IaaS Cloud. Figure 51.7 shows finite graph for Semi-Markov model of the IaaS Cloud with three PMs.

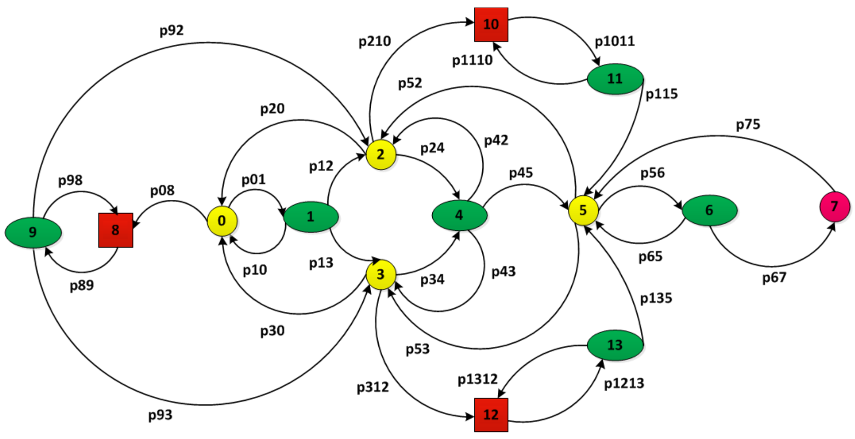


Fig. 51.7. Semi-Markov availability model of the IaaS Cloud with three PMs

In Fig. 51.7, if three PMs fail, the cloud computing system becomes unavailable. State S_7 is unavailable state of IaaS Cloud. Obviously, the IaaS Cloud becomes *available*, when the model enters states S_0, S_2, S_3, S_5. In state S_0 three PMs are operational. At the same time states S_2 and S_3 are states with two operational PMs. Then state S_5 is state with one operational PM. From now *available states* are yellow, whereas *unavailable states* are red and states of TSCS are green.

Suppose our IaaS Cloud works throughout a particular time with operated duration $t \in [0, T]$. We use the following assumptions and limitations for our modeling process.

- Hot, warm, and cold PMs are identical PMs [5]. Cloud Service Provider can do replacement of failed hot PM by available warm or cold PM, respectively.
- Solution for this model can be obtained when replacement process of PMs is instantaneous. It means that we consider immediate transitions.
- CSP provider can perform technical state control of hot PM. The duration of this interval is τ_c.
- Overall effect several types of possible failures in PMs with an aggregated MTTF is considered here [12]. We also use assumptions that all times to failures for all PMs are *exponentially distributed*. Despite the fact that hot, warm and cold PMs have different operating time in order to simplify modeling process, we will suppose that MTTF $1/\lambda_s$ can be represented as

equal values. Apparently, it is reasonable to consider case for three hot, warm and cold PMs ($n_h = n_w = n_c = 1$), when MTTF $1/\lambda_s = 1/\lambda_{sh} = 1/\lambda_{sw}$, where sudden failure rates λ_{sh} for hot PM and sudden failure rates λ_{sw} for warm PM.

- IaaS Cloud provider haven't enough time in order to perform repair operations for failed PMs. Therefore, we need to take into consideration that all times to repair are not exponentially distributed. We use *Erlang-k distribution* in preference to exponential distribution, where $k = 2$ [12]. We also assume that MTTR of warm PM $1/\mu_w$ is higher than MTTR of hot PM $1/\mu_h$ by a factor of two.
- Specific feature of architecture for IaaS Cloud is implementation of migration process of PMs. We consider the migration operations of physical machines as operations to restore the working capacity of warm and hot PMs with repair rates μ_w and μ_h, respectively.
- We assume that hot and warm PMs can fail due to the occurrence of hidden failures, for example, DoS and DDoS attacks, with rates $\lambda_h = \lambda_{hh} = \lambda_{hw}$. In this case MTTF equals $1/\lambda_h = 1/\lambda_{hh} = 1/\lambda_{hw}$ for hot and warm PMs, respectively. Hidden *unavailable* hot or warm PM will repair after next CTS with rate λ_h^*.
- IaaS Cloud becomes unavailable when the SM model enters the state S_7.

According to the Fig. 51.2, our IaaS Cloud is processing workload into the states S_0 (at the initial moment $t = 0$), S_2, S_3 and S_5, that is *available* sub-set space S_A^1 for IaaS Cloud was created by states $S_A^1 = \{S_0, S_2, S_3, S_5\}$. Other states for IaaS Cloud can be described as: a) CTS sub-set space $S_{CTS}^1 = \{S_1, S_4, S_6, S_9, S_{11}, S_{13}\}$; b) *unavailable* sub-set space $S_{UA}^1 = \{S_7, S_8, S_{10}, S_{12}\}$. In order to solve this task we will employ analytical and stochastic method based on *embedded Markov Chains* [13], [14]. Then *steady-state probability* vector $\pi = \{\pi_0, \pi_2, \pi_3, \pi_5\}$ is solution of this task.

Turning to solution, we will interpret Semi-Markov process as follows. As our research shows CTS performs deterministic period of time T, therefore transitions from states S_i to states S_j are given by:

$$Q_{01}(t) = Q_{24}(t) = Q_{34}(t) = Q_{56}(t) = Q_{89}(t) = Q_{1011}(t) = Q_{1213}(t) = \begin{cases} 0, t < T, \\ 1, t \geq T. \end{cases}$$

At the same time transitions for TSCS from states S_j to states S_i can be written as:

$$Q_{10}(t)=Q_{42}(t)=Q_{43}(t)=Q_{65}(t)=Q_{98}(t)=Q_{1110}(t)=Q_{1312}(t)=\begin{cases}0, t<\tau_c,\\ 1, t\geq\tau_c.\end{cases}$$

Let we turn now to sudden failures for hot, warm and cold PMs. In this case distribution function for transitions from state S_1 to state S_2, from state S_1 to state S_3, from state S_4 to state S_5 and from state S_6 to state S_7 are given by:

$$Q_{12}(t)=Q_{13}(t)=Q_{45}(t)=Q_{67}(t)=1-e^{-\lambda_s t}.$$

Now we will move on to *hidden failures* for hot and warm PMs. Distribution functions for transitions from state S_0 to state S_8, from state S_2 to state S_{10}, from state S_3 to state S_{12} can be written as:

$$Q_{08}(t)=Q_{210}(t)=Q_{312}(t)=\begin{cases}1-e^{-\lambda_h t}, t<T,\\ 0, t\geq T.\end{cases}$$

Next distribution functions for *hidden unavailable* hot or warm PM after next CTS are given by:

$$Q_{92}(t)=Q_{93}(t)=Q_{115}(t)=Q_{135}(t)=1-e^{-\lambda_h^* t}.$$

Distribution functions of transitions from state S_2 to state S_0, from state S_3 to state S_0 and from state S_7 to state S_5 can be written as:

$$Q_{20}(t)=Q_{30}(t)=Q_{75}(t)=1-(1+\mu_h t)e^{-\mu_h t}.$$

Simultaneously, distribution functions of transitions from state S_5 to state S_2, from state S_5 to state S_3 are given by:

$$Q_{52}(t)=Q_{53}(t)=1-(1+\mu_w t)e^{-\mu_w t}.$$

Taking the total probability relation $\sum_{i=0}^{13}\pi_i=1$ and taking the steady-state probability vector, we can compute the required result as

$$A=\pi_0+\pi_2+\pi_3+\pi_5, \qquad (51.13)$$

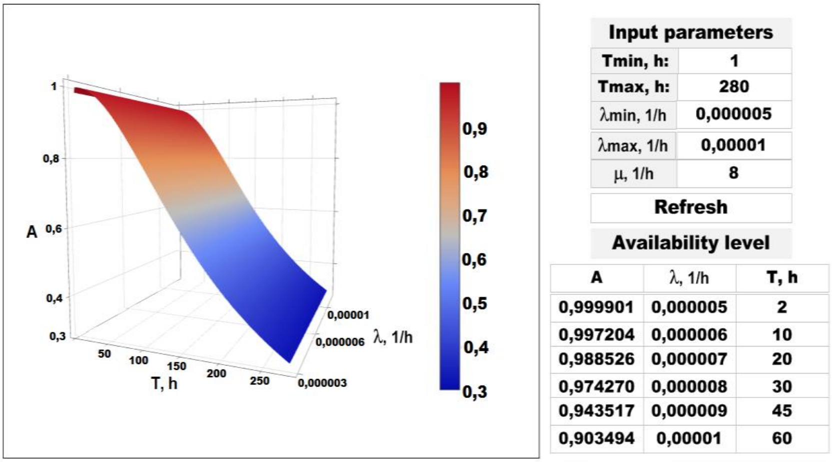
where π_0,π_2,π_3,π_5 are steady-state probabilities for states S_0,S_2,S_3,S_5.

We can describe situation that relate to the states S_0, S_2, S_3, S_5 as situation, when IaaS Cloud can perform operational required functions. But we also consider others states, when IaaS Cloud can't perform a certain amount of useful work. Overall results of IaaS Cloud modeling based on embedded Markov Chains are shown in Fig. 51.8 and Fig. 51.9.

Analysis of modeling results confirmed that value of steady-state availability A is increased by means of increasing of repair rate μ of hot PMs and reduction of hidden failure rate λ_h of hot PMs, as it results by comparing Fig. 51.8 with Fig. 51.9.

Next, we will illustrate how to use our stochastic approach in order to describe the behavior of IaaS Cloud with three pools of PMs. Figure 5 shows finite graph for second type of Semi-Markov model of the IaaS Cloud with nine PMs. Note that the model was solved for only one type of hidden failures. It is serious lack, because in real situation we can observe different types of hidden failures. For example, familiar attack patterns may be developed and occurred as hidden failures of physical machines of the IaaS Cloud or rejuvenation operation for software applications.

In considering the second model, we consider that two different types of hidden failures of PMs are made possible. Previous study have shown that we can interpret two branches of hidden failures for PMs using the following probability transitions: 1) p_{03}, p_{711}, p_{1325}, p_{819}, p_{1429}, p_{2750}, p_{2239}, p_{3564}, p_{5371}, p_{4261}, p_{5775}, p_{7385}, p_{2447}, p_{4668}, p_{6782}, p_{8189}, p_{8893}, p_{9297},(first branch for PMs of hot pool); 2) p_{04}, p_{816}, p_{2243}, p_{717}, p_{1437}, p_{3558}, p_{1332}, p_{2754}, p_{5378} (second branch for PMs of warm pool).

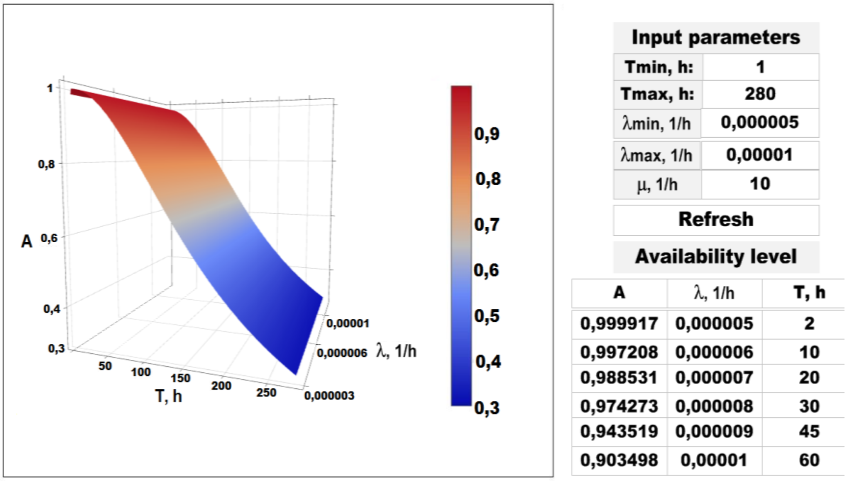


Input parameters	
Tmin, h:	1
Tmax, h:	280
λmin, 1/h	0,000005
λmax, 1/h	0,00001
μ, 1/h	8

Availability level

A	λ, 1/h	T, h
0,999901	0,000005	2
0,997204	0,000006	10
0,988526	0,000007	20
0,974270	0,000008	30
0,943517	0,000009	45
0,903494	0,00001	60

Fig. 51.8. Depending of steady-state availability $A(\lambda_h, T)$ for $T = 250$ h, $\mu = 8$ 1/h



Input parameters	
Tmin, h:	1
Tmax, h:	280
λmin, 1/h	0,000005
λmax, 1/h	0,00001
μ, 1/h	10

Availability level

A	λ, 1/h	T, h
0,999917	0,000005	2
0,997208	0,000006	10
0,988531	0,000007	20
0,974273	0,000008	30
0,943519	0,000009	45
0,903498	0,00001	60

Fig. 51.9. Depending of steady-state availability $A(\lambda_h, T)$ for $T = 250$ h, $\mu = 10$ 1/h

According to the Fig. 51.10, available sub-set space S_A^2 for second model of IaaS Cloud was created by states $S_A^2 = \{S_0, S_7, S_8, S_{13}, S_{14}, S_{22}, S_{24}, S_{27}, S_{35}, S_{42}, S_{46}, S_{53}, S_{57}, S_{67}, S_{73}, S_{81}, S_{88},$

$S_{92}\}$. Other states for second Semi-Markov availability model can be described as *unavailable*. Then steady-state probability vector $\pi = \{\pi_0, \pi_7, \pi_8, \pi_{13}, \pi_{14}, \pi_{22}, \pi_{24}, \pi_{27}, \pi_{35}, \pi_{42}, \pi_{46}, \pi_{53}, \pi_{57}, \pi_{67}, \pi_{73}, \pi_{81}, \pi_{88}, \pi_{92}\}$ is solution of this task.

Next, we will also move on to hidden failures for hot and warm PMs. Distribution functions for first branch of transitions can be written as:

$$Q_{ij}(t) = \begin{cases} 1 - e^{-\gamma_h t}, t < T, \\ 0, t \geq T. \end{cases}$$

At the same time distribution functions for second branch of transitions is given by:

$$Q_{ij}(t) = \begin{cases} 1 - e^{-\gamma_w t}, t < T, \\ 0, t \geq T, \end{cases}$$

where hidden failure rates of warm PMs γ_w is lower than hidden failure rates of hot PMs γ_h by a factor of two to four [2].

Distribution functions of transitions for repair time of the hot and warm PMs are given by:

$$Q_{ji}(t) = 1 - (1 + \delta_h t)e^{-\delta_h t},$$

$$Q_{ji}(t) == 1 - (1 + \delta_w t)e^{-\delta_w t},$$

where repair rate of hot PMs δ_h is higher than repair rate of warm PMs δ_w.

Overall equation for *steady-state availability* of IaaS Cloud can be written as:

$$A = \pi_0 + \pi_7 + \pi_8 + \pi_{13} + \pi_{14} + \pi_{22} + \pi_{24} + \pi_{27} + \pi_{35} + \pi_{42} + \pi_{46} + \pi_{53} + \\ + \pi_{57} + \pi_{67} + \pi_{73} + + \pi_{81} + \pi_{88} + \pi_{92}, \qquad (51.14)$$

where π_i – *steady-state probability* of random event, when most of all components, that is PMs of hot and warm pools are available to solve different tasks of Cloud Computing and operational.

The modeling results for IaaS Cloud with three pools of PMs based on *embedded Markov Chains* are obtained and shown availability level of the cloud infrastructure how QoS assessment. mission of the cloud infrastructures in the critical energy infrastructures domain.

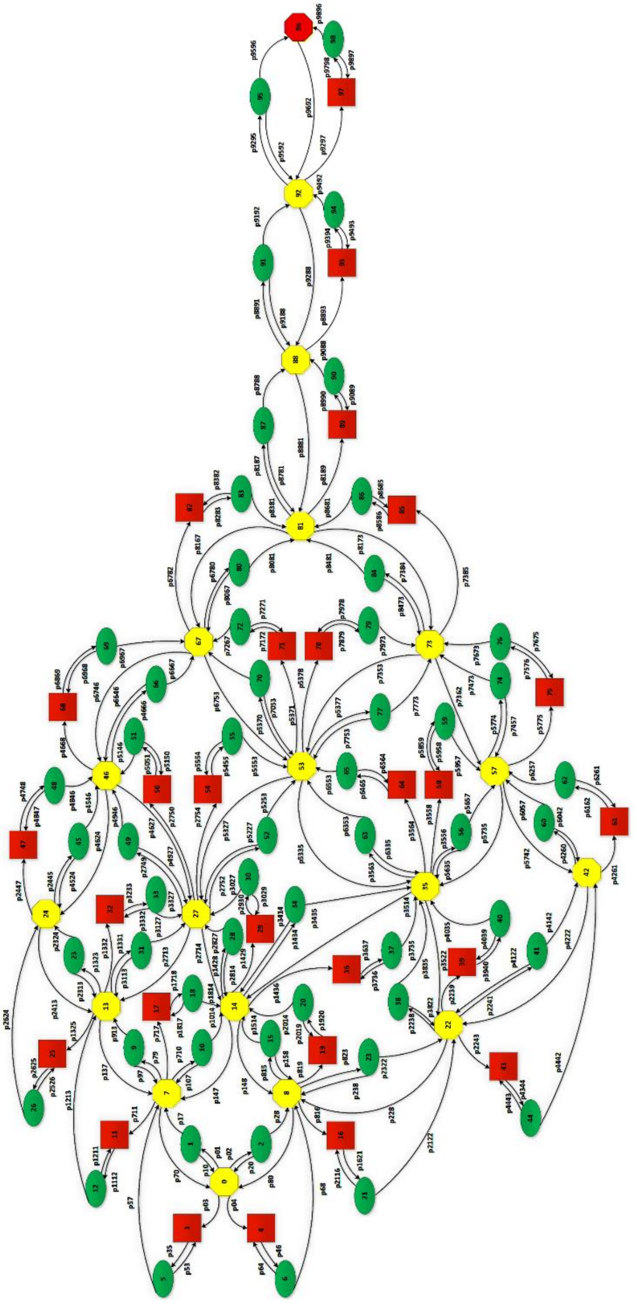


Fig. 24.10. Semi-Markov availability and security model of the IaaS Cloud with ten PMs

Finally, in spite of the fact that second Semi-Markov model (Fig. 51.10) is more complex than third model (Fig. 51.7), nevertheless we can use similar approach in order to get solution based on embedded Markov Chains. Nowadays we can allege that it is one of the most notable advantages of Markovian modeling all among the famous mathematical methods and stochastic approaches, on which scientific researches for Cloud Computing area is based.

Conclusion

In the chapter we have looked at different types of analytical and stochastic models, which we can use in order to determine availability and security assessments for IaaS Cloud with multiple pools of PMs. In this part we performed Semi-Markov modeling for the IaaS Cloud for two concrete situations. As an illustration, we obtained cloud infrastructures availability assessments for the following samples: a) when IaaS Cloud consists of three PMs and we can observe sudden physical machines failures; b) when IaaS Cloud consists of three PMs, but physical machines failures are sudden or hidden failures, that created serious complexity for this task; c) when we can simulate attack pattern upon cloud infrastructure’s resource based on Semi-Markov modelling of DoS and DDoS queries. Several optimization problems, that related to the architecture of IaaS Cloud can be solved based on our stochastic approach and Semi-Markov models with special states described in this part.

Questions for the Self-Control

This work is aimed to answer a series of related questions, including:

1. How is the availability models changed considering Markovian basis and properties?
2. What is the difference between Markov and Semi-Markov models?
3. What type of Markov models will you use in order to perform Semi-Markov availability modelling for cloud infrastructures? do time-out settings affect system dependability and latency and allow to interplay between them?
4. What is fundamental link between physical and virtual machines failures?
5. How does cloud infrastructures availability level determine considering steady-state probable link between pools of physical machines?
6. How do mean time between failures and mean time to repair of physical machines influence cloud infrastructures availability level?
7. How do physical machines pools affect cloud infrastructure availability and security?

8. How embedded Markov chains can be used to solve Semi-Markov modeling task for cloud infrastructures?

9. How can cloud computing technologies be used to avoidance negative effects of DoS and DDoS attacks?

10. What software tools would you be able to prefer to use in order to solve Semi-Markov availability and security modelling tasks for cloud infrastructures?

References

[1] Khazaei, H., Misic, J., Misic, V.B., Beigi-Mohammadi, : Availability analysis of cloud computing centers. In Globecom 2012 – Communications Software, Services and Multimedia Symposium (GC12 CSSM), 2012.

[2] Longo, F., Ghosh, R., Naik, V.K., Trivedi, K.S.: A scalable availability model for infrastructure-as-a-service cloud. Dependable Systems and Networks, International Conference on, pp. 335–346, 2011.

[3] Ghosh, R.: Scalable stochastic models for cloud services. Diss. Duke of University (2012).

[4] Ivanchenko, O., Kharchenko, V., Skatkov, A.: Management of critical infrastructures Based on Technical Megastate. In an International JournaI "Information and Security", vol. 28, no. 1, pp. 37–51 (2012).

[5] Ghosh, R., Longo, F., Frattini, F., Russo, S., Trivedi, K.S.: Scalable analytics for IaaS cloud availability. In IEEE Transactions on Cloud Computing, 2(1), pp. 57–70 (2014).

[6] Ivanchenko, O., Lovyagin, V., Maschenko, E., Skatkov, A., Shevchenko, V.: Distributed critical systems and infrastructures. National Aerospace University named after N. Zhukovsky "KhAI", Kharkiv (2013).

[7] Abbadi, I.M.: Toward Trustworthy Clouds' Internet Scale Critical Infrastructure. In: Bao, F., Weng J. (eds.) Information Security Practice and Experience. LNCS, vol. 6672, pp. 71–82. Springer, Heidelberg (2011).

[8] Ghosh, R., Trivedi, K.S., Naik, V.K., Kim, D.S.: End-to-end performability analysis for infrastructure-as-a-service cloud: An Interacting Stochastic Models Approach. In IEEE PRDC, Tokyo (2010).

[9] Matos, R., Araujo, J., Oliveira, D., Maciel, P., Trivedi, K.S.: Sensitivity analysis of a hierarchical model of mobile cloud computing. Simulation Modelling Practice and Theory, no. 50, pp. 151–164 (2015).

[10] Xiaodan, L., Xiaolin, C, Board, J., Kishor S. Trivedi. A Novel Approach for Software Vulnerability Classification. Annual Reliabilty and Maintainability Symposium (RAMS’2017).

[11] Trivedi, K.S., and Sahner, R.: SHARPE at the age of twenty two. ACM Sigmetrics Performance Evaluation Review, vol. 36, no. 4, pp. 52–57 (2009).

[12] Cai, B.L., Zhang, R.Q., Zhou, X.B., Zhao, L.P., Li, K.Q.: Experience Availability: Tail-Latency Oriented Availability in Software-Defined Cloud Computing. In Journal of Computer Science and Technology, vol. 32, no. 2, pp. 250–257 (2017).

[13] Lanus, M., Yin, L., and Trivedi, K.S.: Hierarchical Composition and Aggregation of State-Based Availability and Performability Models. In IEEE Trans. Reliability, vol. 52, no. 1, pp. 44–52 (2003).

[14] Ivanchenko, O, Kharchenko, V. Semi-markov availability models for an Infrastructure as a Service Cloud with multiple pools. In Proc. International Conference on ICT in Education, Research, and Industrial Applications, pp. 349–360 (2016).

PART 14

Security and Resilience Assurance Cases

ABBREVIATIONS

ISO	International Organization for Standardization
IEC	International Electrotechnical Commission
OSI	The Open Systems Interconnection model
CMM	Capability Maturity Model
CC	Common Criteria for Information Technology Security Evaluation
PP	Protection Profiles
ST	Security Targets
EAL	Evaluation Assurance Levels
NIST	National Institute of Standards and Technology
ACL	Access Control List
SSP	Sensitive Security Parameters
CSP	Critical Security Parameters
FIPS	Federal Information Processing Standards
EFP	Environmental Failure Protections
EFT	Environmental Failure Testing
HDL	Hardware Description Language
PAL	Protocol Assurance Levels
IA	Identity Assurance
LoA	Level of Assurance
IAL	Identity Assurance Level
ENISA	European Network and Information Security Agency
eIDAS	electronic IDentification, Authentication and trust Services
TOE	Target of Evaluation
IT	Information Technology
LV	Linguistic Variable
TER	Technical evaluation report

52. Characteristics of international standards in the field of security assurances

Currently, the International Organization for Standardization (ISO) adopted about 20 International Standards, that standardizes security requirements of IT security. In fact this regulations already constitute a system of standards for security assurances. Today the standardization objects are:

- product;
- system;
- process;
- environment.

For these objects developed and standardized techniques, guidelines and assessment. Documents analysis showed, that assurance requirements nominated to administrative, organizational and technical controls.

Central place in this system occupy international standard ISO/IEC 15408 Common Criteria for Information Technology Security Evaluation (CC) [1-3]. Due to this standard security assurance concept is widely implemented in the design and evaluation practice of IT security. Security assurances are becoming a subject of evaluation.

This approach was very productive so developers began to distribute it to other objects of standardization in the field of IT-security.

The approach that was enshrined in Common Criteria has been used in the standards that define requirements for specific technologies. For example, in the international standard ISO/IEC 29128:2011 Information technology – Security techniques – Verification of cryptographic protocols [8] are formulated requirements for verification of cryptographic protocols, which can be viewed as assurance requirements for crypto protocols. International standard ISO/IEC 19792:2009 Information technology – Security techniques – Security evaluation of biometrics [9] establishes the assurance requirements for biometric safeguards. Standard ISO/IEC 19791:2009 Information technology – Security techniques – Information technology – Security techniques – Security assessment of operational systems [10] standardizes protection profile of operating systems, where also introduces assurance requirements for operational systems.

The security assurance requirements can be presented in various forms, but they retain their essence – their execution creates confidence in true implementation of functional safety requirements of other requirements that imposed to the product, system, process or environment. The security assurance requirements are the basis of trust to the security of these facilities.

52.1. The security assurance requirements model of international standard ISO/IEC 15408

Central place in this system occupy international standard ISO/IEC 15408 Common Criteria for Information Technology Security Evaluation (CC) [1-3]. Due to this standard security assurance concept is widely implemented in the design and evaluation practice of IT security. Security assurances are becoming a subject of evaluation.

ISO/IEC 15408-3 Part 3 – Security Assurance Requirements. This part produces a catalogue of establishes set of assurance components that can be used as a standard way of expressing the assurance requirements for IT products and systems. The Part 3 catalogue is organized into the same class - family - component structure. Part 3 also defines evaluation criteria for Protection Profiles (PPs) and Security Targets (STs). Part 3 presents the seven Evaluation Assurance Levels (EALs), which are predefined packages of assurance components that make up the CC scale for rating confidence in the security of IT products and systems.

Assurance is a ground for confidence that the IT-product meets its security functional requirements. Assurance reduces the uncertainty associated with vulnerabilities of the IT-product, and thus the potential vulnerability is reduced leading to a reduction in the overall risk.

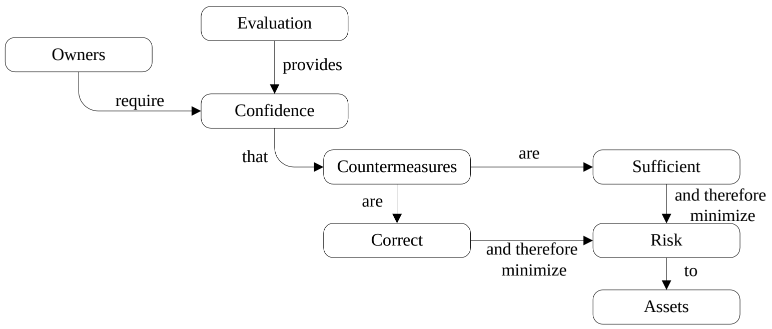


Fig.52.1. Evaluation concepts and relationships

Assurance does not “add” any safeguards or services to the IT-product. The assurance actually contributes to the confidence that one has in the mechanism strength by reducing the uncertainty (or probability) of a threat. ISO 15408 provides assurance through active investigation. Active investigation is an evaluation of the IT product in order to determine its security properties.

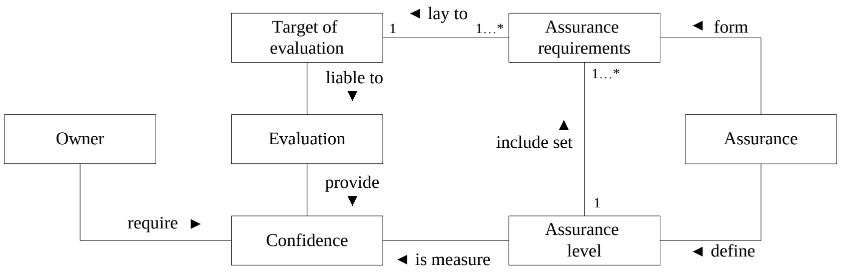


Fig. 52.2. Ontological model of "confidence" and "assurance" concepts

The standard standardizes the security assurance requirements by their organization in the requirements hierarchy. The hierarchical structure consists of classes, families and components of security assurance requirements. On the set of security assurance requirements components enters hierarchical dependency relationships. Each security assurance requirement is based on a unified pattern.

Security assurance requirement are grouped into classes. Class is the most general grouping of security requirements, and all members of a class share a common focus. There are 8 assurance classes within Part3 of the CC. These are as follows: Configuration management, Guidance documents, Vulnerability assessment, Delivery and operation, Life cycle support, Assurance maintenance, Development, and Test. Two additional classes contain the assurance requirements for PPs and STs.

On the set of security assurance components defined the security assurance levels scale – evaluation assurance level (EAL – set of assurance requirements drawn from CC Part 3, representing a point on the CC predefined assurance scale, that form an assurance package). There are 7 security assurance levels. First level is the lowest level of security, seventh is the highest. The higher assurance level is, then more security assurance requirements it is contains. Each higher security assurance level includes all security assurance requirements lower level directly or enhances them. Brief explanations of EAL [4]:

Table 52.1. Evaluation Assurance Level

Common Criteria Evaluation Assurance Level (EAL)	Process rigor required for development of an IT product
EAL1: Functionally Tested	This where the applicable where threat to security is not serious, however some confidence in current operation is required. In the evaluation, there is no assistance from TOE developer. The requirements are: Configuration Management, Delivery and Operation, Development, Guidance documents and

	Tests.
EAL2: Structurally Tested	This assurance level is applicable where low to moderate level of independently assured. security is required. Here, it requires some cooperation from the developer. It will definitely require no more than good vendor commercial practices. To add to the previous requirements are developer testing, vulnerability analysis, and more extensive independent testing.
EAL3: Methodically Tested and Checked	It is applicable where moderate level of independently assured security is required. The cooperation from the developer is requires. It places additional requirements on testing, development environment controls and configuration management. The additional requirement is the Life Cycle support.
EAL4: Methodically Designed, Tested, and Reviewed.	This is applicable where moderate to high level of independently assured security is required. It is to ensure that there is some security engineering added to commercial development practices. This currently the highest level likely for retrofit of an existing product. There are additional requirements on design, implementation, vulnerability analysis, development and configuration management
EAL5: Semiformally Designed and Tested.	It is applicable where high level of independently assured security is required. It requires rigorous commercial development practices and moderate use of specialist engineering techniques with additional requirements on specification, design, and their correspondence.
EAL6: Semiformally Verified Design and Tested.	This evaluation level is applicable where assets are valuable and risks are high and do requires a rigorous development environment. The additional requirements are on analysis, design, development, configuration management, and vulnerability/covert channel analysis.
EAL7: Formally Verified Design and Tested.	This is applicable where assets are highly valuable and risks are extremely high. However, practical use is functionally limited for amenability to formal analysis. The assurance is gained through application of formal methods. The additional requirements for these is testing and formal analysis.

In order to methodological support of general criteria were developed standards ISO/IEC TR 15446:2009 Information technology – Security techniques – Guide for the production of Protection Profiles and Security Targets [5] and ISO/IEC 15446 Common Methodology for Information Technology Security Evaluation. Evaluation methodology [6].

The first document provides guidance relating to the construction of PP and ST that are intended to be compliant with the third edition of ISO/IEC 15408. Protection Profiles and Security Targets are special designs, that describes functional requirements and security assurance requirements, which apply to the evaluation object. The evaluation object can be IT-product or IT-system. Essentially, Protection Profile can be viewed as security assurance case. And Security Target can be viewed as embodiment of functional security requirements and security assurance requirements.

The second document – ISO/IEC 18045 determines methodology for evaluation of the security assurance requirements. By using this document it is possible to develop a program and method of security assurance evaluation [7].

Thus, the standard ISO/IEC 15408 regulatory ratified that formation of confidence in security and providing trust to IT-product achieved by nomination and evaluation of security assurance requirements.

Consider some of presentation forms of security assurance requirements in various international standards, which standardizes requirements for security techniques.

52.2. Security requirements for cryptographic modules

Since the 1990s, NIST is working on standardization of security requirements for cryptographic modules (National standards USA FIPS 140-1:1994 (http://csrc.nist.gov/publications/fips/fips140-1/fips1401.pdf), FIPS 140-2: 2001 (http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf), FIPS 140-3:2005-2012 (http://csrc.nist.gov/groups/ST/FIPS140_3/). In 2012, based on this standard was adopted international standard ISO/IEC 19790: 2012 Information technology — Security techniques — Security requirements for cryptographic modules [11] (last revision was in 2015). In 2014 was adopted an international standard ISO/IEC 24759:2014 Information technology – Security techniques – Test requirements for cryptographic modules [12]. These standards standardizes security requirements for cryptographic modules and assessment methods such claims.

The standard specifies four security levels. Below are a summary of the levels and the basic requirements of each:

Table 52.2. Security Level FIPS 140-2

Security Level	Description

Security Level 1	– At least one approved security function or approved sensitive security parameter establishment method; – Operation in a non-modifiable, limited, or modifiable operating environment; – No physical security mechanisms are required above production-grade components; – Any Non-invasive mitigation methods or mitigation of other attacks, which are implemented, are documented.
Security Level 2	– Adds the requirement for tamper evidence, which includes the use of tamper-evident coatings or seals or pick-resistant locks on removable covers or doors; – Role-based authentication; – Software cryptographic module to be executed in a modifiable environment that implements role-based access controls or, at the minimum, a discretionary access control with a robust mechanism of defining new groups and assigning restrictive permissions through access control lists (ACLs), and with the capability of assigning each user to more than one group, and that protects against unauthorized execution, modification, and reading of cryptographic software; – Security level 2 continues to be the highest security level attainable by a pure software module.
Security Level 3	– Additional requirements to mitigate the unauthorized access to SSPs held within the cryptographic module; – Physical security mechanisms required at Security Level 3 are intended to have a high probability of detecting and responding to attempts at direct physical access, use or modification of the cryptographic module and probing through ventilation holes or slits; – Identity-based authentication mechanisms; – Manually established plaintext CSPs must be encrypted, utilize a trusted channel or use a split knowledge procedure for entry or output; – Mechanisms to protect a cryptographic module

	against a security compromise due to environmental conditions outside of the module's normal operating ranges for voltage and temperature; – Any Non-invasive mitigation methods that are implemented in the module must be tested against metrics for security level 3 that are defined in the standard; – Additional life-cycle assurances, such as automated configuration management, detailed design, low-level testing, and operator authentication using vendor-provided authentication information
Security Level 4	– Multi-factor authentication for operator; – The module includes special environmental protection features designed to detect voltage and temperature boundaries and zeroes CSPs; – Any Non-invasive mitigation methods that are implemented in the module must be tested against metrics for security level 4 that are defined in the standard; – Design verification by the correspondence between both pre- and post-state conditions and the functional specification.

The security requirements are organized into 11 requirement areas; many of these areas have changes from those specified in FIPS 140-2:

Table 52.3. Requirement areas from FIPS 140-2

Functional areas	Description
Cryptographic module specification	This area includes the cryptographic module specification general requirements, the types of cryptographic modules, the cryptographic boundary and the modes of operations.
Cryptographic module interfaces	This area includes the cryptographic module interfaces general requirements, the types of interfaces, the definition of interfaces and trusted channel requirements.
Roles, services, and authentication	This area includes general requirements for roles, services, and authentication.
Software/Firmware security	This area includes general requirements for providing software security.
Operational	This area includes general requirements for the

environment	operational environment, Operating system requirements for limited or non-modifiable operational environments, and Operating system requirements for modifiable operational environments.
Physical security	This area includes requirements for embodiments, general requirements, requirements for each physical security embodiment and Environmental failure protection/testing requirements.
Non-invasive security	This area includes requirements for protection of cryptographic modules from physical non-invasive attack.
Sensitive security parameter management	This area includes general requirements, Random bit generators, Sensitive security parameter generation, Sensitive security parameter establishment , Sensitive security parameter entry and output , Sensitive security parameter storage, and Sensitive security parameter zeroization.
Self-tests	General requirements for self tests, pre-operational self-tests, and conditional self-tests.
Life-cycle assurance	This area includes the life-cycle general requirements, Configuration management, Design ,Finite state model , Development ,Vendor testing, Delivery and operation, End of life and Guidance documents.
Mitigation of other attacks	This area includes the general requirements for protection against other types of attacks.

Table 52.4. Requirements for cryptographic modules according to international standard

	Security Level			
Requirement Area	1	2	3	4
1.Cryptographic Module Specification	Specification of cryptographic module, cryptographic boundary, approved security functions, and normal and degraded modes of operation. Description of cryptographic module, including all hardware, software, and firmware components. All services provide status information to indicate when the service utilizes an approved cryptographic algorithm, security function or process in an approved manner.			
2. Cryptographic Module Interfaces	Required and optional interfaces. Specification of all interfaces and of all input and output data paths.		Trusted channel.	
3.Roles, Services, and Authentication	Logical separation of required and optional roles and services.	Role-based or identity-based operator authentication.	Identity-based operator authentication.	Multi-factor authentication.
4.Software/ Firmware Security	Approved integrity technique, defined SFMI, HFMI and HSMI. Executable code.	Approved digital signature or keyed message authentication code-based integrity test.	Approved digital signature based integrity test.	
5. Operational Environment	Non-Modifiable, Limited or Modifiable, Control of SSPs.	Modifiable. Role-based or discretionary access control. Audit mechanism	▨	▨
6. Physical Security	Production-grade	Tamper evidence.	Tamper detection and	Tamper detection and

	components	Opaque covering or enclosure.	response for covers and doors. Strong enclosure or coating. Protection from direct probing. EFP or EFT.	response envelope. EFP. Fault injection mitigation.
7. Non-Invasive Security	Module is designed to mitigate against non-invasive attacks specified in Annex F.			
	Documentation and effectiveness of mitigation techniques specified in AnnexF.		Mitigation Testing. (level 3)	Mitigation Testing. (level 4)
8. Security Parameter Manageme nt	Random bit generators. SSP generation, establishment, entry and output, storage and zeroisation.			
	Automated SSP transport or SSP agreement using approved methods.			
	Manually established SSPs maybe entered output in plaint ext form.		Manually established SSPs maybe entered output in either encrypted form, via a trusted channel or using split knowledge procedures.	
9. Self-Tests	Pre-operational software/firmware integrity, bypass, and critical functions test.			
	Conditional: cryptographic algorithm, pair-wise consistency, software/firmware loading, manual entry, conditional bypass and critical functions test.			
10. Life-Cycle Assurance				
Configuratio n Management	Configuration management system for cryptographic module, components, and documentation. Each uniquely identified and tracked throughout lifecycle.		Automated configuration management system.	
Design	Module designed to allow testing of all provided security related services.			
Finite State	Finite state model.			

Model			
Development	Annotated source code, schematics or HDL.	Software high-level language. Hardware high-level descriptive language.	Documentation annotated with pre-conditions upon entry into module components and post conditions expected to be true when components is completed.
Testing	Functional Testing.	Low-level Testing	
Delivery and Operation	Initialization procedures.	Delivery Procedures.	Operator authentication using vendor provided authentication information.
Guidance	Administrator and non-administrator guidance.		
11. Mitigation of other attacks	Specification of mitigation of attacks for which no testable requirements are currently available.		Specification of mitigation of attacks with testable requirements.

Assurance requirements concerning to life cycle applies to cryptographic modules and these requirements match with appropriate requirement class of international standard ISO/IEC 15408.

52.3. The security assurance requirements of cryptographic protocols

In 2011 was issued an International Standard ISO/IEC 29128:2011 Information technology – Security techniques – Verification of cryptographic protocols [13]. Standard sets forth requirements areas of evaluation cryptographic protocols, to which the appropriate criteria for evaluating security assurance are set. Such areas are Protocol Specification, Adversarial model, Security property and Selfassessment evidence.

So ISO/IEC 29128 prescribes four Protocol Assurance Levels (PALs) based on the accuracy of their expressions (Table 52.5).

Table 52.5. Protocol assurance levels

Protocol assurance levels	Protocol Assurance Level 1 (PAL1)	Protocol Assurance Level 2 (PAL2)	Protocol Assurance Level 3 (PAL3)	Protocol Assurance Level 4 (PAL4)

Protocol Specification	PPS_SEMIFORMAL Semiformal description of protocol specification	PPS_FORAMAL Formal description of protocol specification	PPS_MECHANIZED. Formal description of protocol specification in a tool-specific specification language, whose semantics is mathematically defined	
Adversarial model	PAM_INFORMAL Informal description of adversarial model	PAM_FORMAL Formal description of adversarial model	PAM_MECHANIZED. Formal description of adversarial model in a tool-specific specification language, whose semantics is mathematically defined	
Security property	PSP_INFORMAL Informal description of security property	PSP_FORMAL Formal description of security property	PSP_MECHANIZED Formal description of security property in a tool-specific specification language, whose semantics is mathematically defined	
Selfassessment evidence	PEV_ARGUMENT Informal argument that the specification of the cryptographic protocol satisfies the given objectives and properties with respect to the adversarial model	PEV_HANDPROVEN Mathematically formal paper-and-pencil proof that the specification of the cryptographic protocol satisfies the given objectives and properties with respect to the adversarial model	PEV_BOUNDED Tool-aided bounded verification that the specification of the cryptographic protocol satisfies the given objectives and properties with respect to the adversarial model	PEV_BOUNDED Tool-aided unbounded verification that the specification of the cryptographic protocol satisfies the given objectives and properties with respect to the adversarial model

When the descriptions are more accurate the protocol is handled as PAL4. When the descriptions are vague the protocol is treated as PAL1. If the possibility of attack is not discovered in the outputs from these tools the tools are regarded as safe in the respective PALs. What must be noted here is that these assessments do not mean that a safe PAL4 protocol is safer than a safe PAL1 protocol, but rather that the possibility of attack was not found in the former in a more accurate assessment. Therefore it is not necessarily possible to compare the safety of PAL4 with that of PAL1. Generally, a high-PAL protocol is used to assess systems that handle important information.

Also the standard sets the correspondence between assurance levels EAL ISO/IEC 15408 and PAL ISO/IEC 29128, that submitted in Table 52.6.

Table 52.6. Correspondence of levels of EAL and PAL

ISO/IEC 15408 EAL	ISO/IEC 29128 PAL
EAL1	PAL1
EAL2	
EAL3	PAL2
EAL4	
EAL5	PAL3
EAL6	
EAL7	PAL4

In 2009, was developed the standard ISO/IEC 19792:2009 Information technology – Security techniques – Security evaluation of biometrics [14]. In this standard the security assurance concept are the base of evaluation security equipment and biometric protection systems methodology [15].

52.4. The Models of Identity assurance

Rapid development of so-called digital economy, e-commerce, e-services, etc. sharply raised the question of confidence in the electronic identification of interaction subject. The question of identity assurance became the key in the implementation of these technologies [16-19].

Identity assurance is at the core of most government and business transactions. It is also a critical underpinning of a number of strategic government and broader public sector initiatives such as online or multi-channel service delivery, integrated case management and improving outcomes for citizens through better information sharing and citizen-centric services. All of these initiatives are dependent on knowing, with a high degree of certainty [19]:

– who is attempting to access government information and services (including what organizations they work for and what roles and privileges they have); and,

– who the information or service, at issue, is about.

There are several IA frameworks (standards) that prescribe different types of IA levels. Those schemes are not always directly compatible, i.e. there is no simple and permanent mapping between one set of levels to the other set of levels. It is not only that they may use different numbering scheme (e.g. 1 to 4 vs. 0 to 3), or that there may be a different number of levels: they may be also built around different paradigms, which may render them incomparable.

52.5. The model of ISO/IEC 29115

To determine the assurance level of authentication the basic is international standard ISO/IEC 29115 [21]. International standard specifies that electron transaction between information and telecommunications systems or in the middle of system has satisfying requirements, which depend on the level of confidence in the identification of the parties involved in the transaction. According to standard, Assurance refers to the confidence placed in all the process, management activities and technologies used to establish and manage the identity of an entity for use in authentication transactions.

The standard specifies four authentication levels, relating to technical measures, process and management and defines the criteria for evaluation of assurance level (Fig. 52.3).

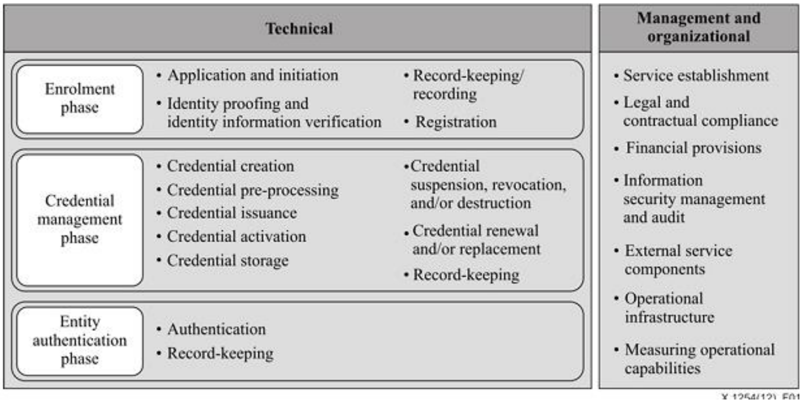

Technical
Management and organizational
Enrolment phase
• Application and initiation
• Identity proofing and identity information verification
• Record-keeping/ recording
• Registration
Credential management phase
• Credential creation
• Credential pre-processing
• Credential issuance
• Credential activation
• Credential storage
• Credential suspension, revocation, and/or destruction
• Credential renewal and/or replacement
• Record-keeping
Entity authentication phase
• Authentication
• Record-keeping
• Service establishment
• Legal and contractual compliance
• Financial provisions
• Information security management and audit
• External service components
• Operational infrastructure
• Measuring operational capabilities
Y.1254(12)_F01

Fig. 52.3. Overview of the entity authentication assurance framework

Each levels of assurance (LoA) for entity authentication describes the degree of confidence in the process leading up to and including the authentication process itself, thus providing assurance that the entity that uses a particular identity is in fact the entity to which that identity was assigned. The assurance level can be represented by function

LoA = F(Process, Management Activities, Technical controls).

Assurance levels are defined as shown in Table 52.7 [21].

Table 52.7. Levels of assurance ISO/IEC 29115

Level	Description
1 - Low	At LoA1, there is minimal confidence in the claimed or asserted identity of the entity, but some confidence that the entity is the same over consecutive authentication

	events. This LoA is used when minimum risk is associated with erroneous authentication. There is no specific requirement for the authentication mechanism used; only that it provides some minimal assurance. A wide range of available technologies, including the credentials associated with higher LoAs, can satisfy the entity authentication assurance requirements for this LoA. This level does not require use of cryptographic authentication methods (e.g., cryptographic-based challenge-response protocol).
2 - Medium	At LoA2, there is some confidence in the claimed or asserted identity of the entity. This LoA is used when moderate risk is associated with erroneous authentication. Single-factor authentication is acceptable. Successful authentication shall be dependent upon the entity proving, through a secure authentication protocol, that the entity has control of the credential. Controls should be in place to reduce the effectiveness of eavesdroppers and online guessing attacks. Controls shall be in place to protect against attacks on stored credentials.
3 - High	At LoA3, there is high confidence in the claimed or asserted identity of the entity. This LoA is used where substantial risk is associated with erroneous authentication. This LoA shall employ multifactor authentication. Any secret information exchanged in authentication protocols shall be cryptographically protected in transit and at rest (although LoA3 does not require the use of a cryptographic-based challenge-response protocol).
4 - Very High	At LoA4, there is very high confidence in the claimed or asserted identity of the entity. This LoA is used when high risk is associated with erroneous authentication. LoA4 is similar to LoA3, but it adds the requirements of in-person identity proofing for human entities and the use of tamper resistant hardware devices for the storage of all secret or private cryptographic keys. Additionally, all PII and other sensitive data included in authentication protocols shall be cryptographically protected in transit and at rest.

The requirements of the international standard is the basis for the national standards development and the requirements of security authentication level. The most developed are requirements of NIST [22], approach that has been used in the UK [23], requirements model that has been used in European Union[20] and Canada [24]. Brief overview of the models and requirements.

52.6. US approach (NIST SP 800-63-2)

The majority of US Identity Assurance Level (IALs) draw on works of the US NIST (National Institute for Standards and Technology Special Publication 800-63-2) standard, further developed by Liberty Alliance and by Kantara (in its Identity Assurance Framework). Those works, compatible with the US NIST approach, are also being standardised in the UK (BS ISO/IEC 29115), and adopted by ENISA (Mapping IDABC Authentication Policy to SAML v2.0, ENISA Report 2008).

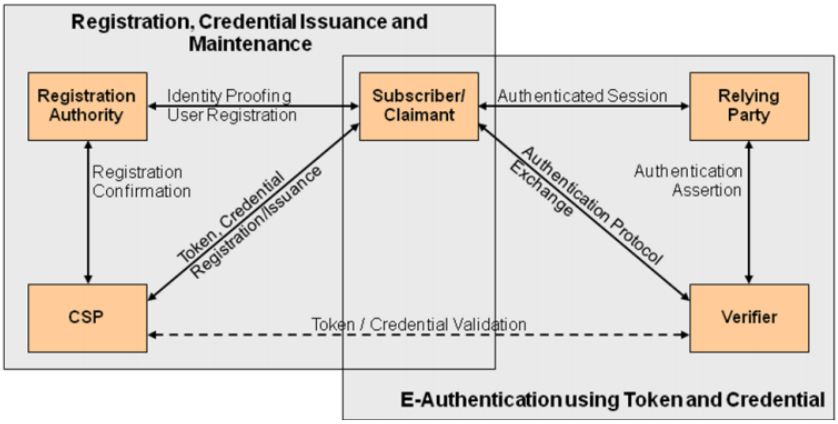

Registration, Credential Issuance and Maintenance
Registration Authority
Identity Proofing
User Registration
Subscriber/ Claimant
Authenticated Session
Relying Party
Registration Confirmation
Token, Credential Registration/Issuance
Authentication Protocol Exchange
Authentication Assertion
CSP
Token / Credential Validation
Verifier
E-Authentication using Token and Credential

Fig. 52.4. Model of Identity Assurance Level [22]

In general, they define four IA levels, numbered 1 to 4 (from the weakest to the strongest one). Kantara IALs are defined on the basis of risk mitigation supplemented by particular technology choices. US NIST IALs are defined entirely through technology that must be used at each level. ENISA introduces the notion of appropriate processes that encompass not only tokens, but also the quality of issuance and authentication.

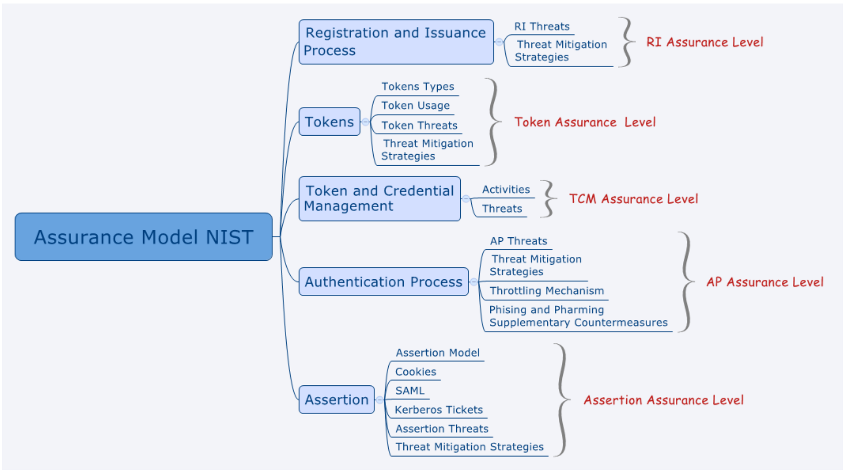


Fig. 52.5. The elements of assurance model NIST

For reference, those IA levels are commonly associated with the following technologies and processes (definitions after NIST and BS, technology after NIST, Kantara and ENISA):

Table 52.8. NIST Identity assurance Level

Assurance Level	Description
Level 1	Low:Little or no confidence in the asserted identity. Although there is no identity proofing requirement at this level, the authentication mechanism provides some assurance that the same claimant is accessing the protected transaction or data. Challenge-response with password or any method applicable for higher levels is satisfactory.
Level 2	Medium: Some confidence in the asserted identity Level 2 provides single factor remote network authentication. A wide range of available authentication technologies can be employed at Level 2, but password over encrypted communication channel is satisfactory
Level 3	High: High confidence in the asserted identity Level 3 provides multi-factor remote network authentication. At this level, identity proofing procedures require verification of identifying materials and information. Level 3 authentication is based on proof of possession of a key or a one-time password through a cryptographic

	protocol. A minimum of two authentication factors is required. Three kinds of tokens may be used: “soft” cryptographic tokens, “hard” cryptographic tokens and “one-time password” device tokens.
Level 4	Very High: Very high confidence in the asserted identity Level 4 is intended to provide the highest practical remote network authentication assurance. Level 4 authentication is based on proof of possession of a key through a cryptographic protocol. Only “hard” cryptographic tokens are allowed. The token shall be a hardware cryptographic module validated at FIPS 140-2 Level 2 or higher overall with at least FIPS 140-2 Level 3 physical security

While the US approach has been widely adopted as a general concept, it faces some significant problems of its own. Permanently binding levels to particular technologies may not be appropriate, as legal practices, technology development, quality of operation as well as fraudulent activities may undermine some technological choices. It may be beneficial to separate the principle of IA levels from the implementation that should follow current best practices.

52.7. UK Government approach (Identity assurance program)

The UK government has introduced its own set of IALs, described e.g. in the "Cabinet Office Identity Assurance Strategy". This set of four (numbered 0 to 3) is designed around the notion of the burden of proof and due diligence. Intuitionally, by applying a given IA level, the relying party can demonstrate in a court of law that it has applied an appropriate level of due diligence in identifying a person, regardless of the technology used.

This particular design of IA levels can be considered beneficial, as it provides a direct reference to legal and social practices. However, it introduces its own problems, as vendors as well as operators are left without clear guidelines regarding the exact technology and processes that should be implemented.

The four levels of IA are specified below:

Table 52.9. UK Gov Identity Assurance

Assurance Level	Description
Level 0 (anonymous)	The relying party is not concerned with the actual identity of a person, such as e.g. in casual petty cash transactions.

	This does not preclude that the relying party may concerned with the entitlement of a person (e.g. whether the person has a valid ticket) or with the continuation of the relationship (e.g. whether goods are collected by the same person who has deposited them).
Level 1 (self-asserted)	The relying party accepts the statement of the person himself when it comes to his identity (e.g. his business card), in anticipation that the person is generally trustworthy. This does not preclude the relying party from seeking further assurance if and when the situation changes (e.g. before signing a contract), or when there is a reasonable doubt about the self-asserted identity (e.g. details for the business card seem suspicious).
Level 2 (balance of probabilities)	It is a level that satisfies the standard of proof for civil cases. It requires the relying party to demonstrate that there is a sufficient probability that a person’s identity is genuine, against the probability that it is not. Usually, this level requires credentials that are issued by an identity provider. In practical situations, credentials such as an utility bill or credit card suffice, specifically if the person can demonstrate the ownership of credentials (e.g. knows the valid PIN).
Level 3 (beyond reasonable doubt)	It is a level that satisfies the standard of proof for criminal cases. It requires the relying party to demonstrate that there is no doubt regarding the person identity, i.e. that there is enough evidence to support the identity versus no evidence against it. Quality credentials are usually required, and an ability to demonstrate the undeniable link between credential and the person is desired (e.g. in a form of biometrics).

52.8. The model of EU (Commission Implementing Regulation 2015/1502)

In September 2015, the European Commission adopts document 2015/1502, which sets out minimum technical specifications and procedures for assurance levels for electronic identification means [20]. This normative document was adopted in order to implement the requirements of Regulation 910/2014 eIDAS. During the formation of these requirements were taken into account the requirements of the international standard ISO/IEC 29115, and requirements of regulations eIDAS and other features that come with the need to take into account the requirements of national eID schemes of EU members. Also took into account the results of the STORK project , conducted in the EU to ensure the interoperability of electronic trust services.

Requirements for identity proofing and verification should take into account different systems and practices, while ensuring sufficiently high assurance in order to establish the necessary trust. Assurance levels low, substantial and high for electronic identification means issued under a notified electronic identification scheme shall be determined with reference to the specifications and procedures set out in the Regulation. The specifications and procedures shall be used to specify the assurance level of the electronic identification means issued under a notified electronic identification scheme by determining the reliability and quality of following elements (Fig. 52.6):

(a) enrolment;

(b) electronic identification means management;

(c) authentication;

(d) management and organisation.

The elements of technical specifications and procedures shall be used to determine how the requirements and criteria shall be applied for electronic identification means issued under an electronic identification scheme.

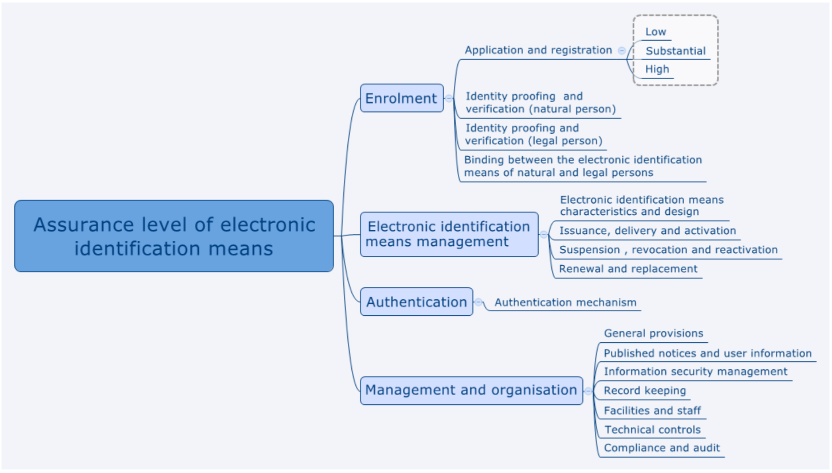

Assurance level of electronic identification means
Enrolment
Application and registration
Low
Substantial
High
Identity proofing and verification (natural person)
Identity proofing and verification (legal person)
Binding between the electronic identification means of natural and legal persons
Electronic identification means management
Electronic identification means characteristics and design
Issuance, delivery and activation
Suspension , revocation and reactivation
Renewal and replacement
Authentication
Authentication mechanism
Management and organisation
General provisions
Published notices and user information
Information security management
Record keeping
Facilities and staff
Technical controls
Compliance and audit

Fig. 52.6. Assurance level eID means

For each element defined appropriate subelements to make the evaluation. Then, for each category carefully designed and formulated evaluation criterias. Examples of such criterias listed in the Table 52.10.

Table 52.10. Sample criteria for assurance evaluation of Electronic identification means characteristics and design

Assurance level	Elements needed
Low	1. The electronic identification means utilises at least one authentication factor. 2. The electronic identification means is designed so that the issuer takes reasonable steps to check that it is used only under the control or possession of the person to whom it belongs.
Substantial	1.The electronic electronic identification means utilises at least two autentication factors from different categories. 2.The electronic electronic identification means is designed so that it can be assumed to be used only if under the control or possession of the person to whom it belongs.
High	Level substantial, plus: 1.The electronic electronic identification means protects against duplication and tampering as well as against attackers with high attack potential. 2.The electronic electronic identification means is designed so that it can be reliably protected by the person to whom it belongs against use by others.

Thus, final security authentication level is a collection of all security levels by every element.

52.9. Pan-Canadian Assurance Model

The Government of Canada has introduced an assurance security model within Identity Management and Authentication framework [24].

The Canadian model includes two assurance categories – Identity Assurance and Credential Assurance.

Identity Assurance is the level of confidence (certainty) that the client is really who they claim to be. An identity assurance mitigates the risks (or uncertainties) associated with false or inaccurate claims around an individual's truthful identity.

Credential Assurance is the level of confidence (certainty) that the client has maintained control over what has been entrusted to them. A credential assurance mitigates the risk (or uncertainties) associated with a compromised, lost or stolen credential.

An assurance level reflects the degree of certainty that given by one party (the assurance provider) or is required by a relying party. The model introduces standardized description of assurance level.

Table 52.11. Standardized Assurance Level Descriptions [24]

Assurance Level[3]	Assurance Level Description[4]
None	No confidence required: No harm to any party in the event of authentication error, therefore, authentication is typically not required nor desired.
Level 1	Little confidence required. Harm from an authentication error would be nil to minimal.
Level 2	Some confidence required. Harm from an authentication error would be minor to moderate.
Level 3	High confidence required. Harm from an authentication error would be moderate to serious.
Level 4	Very high confidence required. Harm from an authentication error would be serious to catastrophic.

And introduces Credential Assurance Level (CAL) (Tab. 52.12), and Identity Assurance Levels (IAL) (Tab. 52.13).

Table 52.12 . Credential Assurance Level Descriptions

Credential Assurance Level	Credential Assurance Level (CAL) Description
None	No confidence required that the client maintained control over the entrusted credential and the credential has not been compromised
Level 1	Little confidence required that the client maintained control over the entrusted credential and the credential has not been compromised
Level 2	Some confidence required that the client maintained control over the entrusted credential and the credential has not been compromised.
Level 3	High confidence required that the client has maintained control over the entrusted credential and the credential has not been compromised.
Level 4	Very high confidence required that the client has maintained control over the entrusted credential and the credential has not been compromised.

Table 52.13. Identity Assurance Levels

Identity Assurance Level	Identity Assurance Level (IAL) Description
None	No confidence required that the client is or is not who they claim to be.
Level 1	Little confidence required that the client is or is not who they claim to be.
Level 2	Some confidence required that the client is or is not who they claim to be.
Level 3	High confidence required that the client is or is not who they claim to be.
Level 4	Very high confidence required that the client is or is not who they claim to be.

52.10. The Identity Assurance Framework of British Colambia

Total Canadian government model is the basis for developing appropriate models of assurance for all regions of Canada. Below is a brief overview of British Columbia models.

Identity assurance (IA) is the set of standards and processes that allow a relying party to determine, at a satisfying level of certainty, the particular identity. For example, while both the bank and the e-commerce web site may want to identify their customers, their needs for certainty may differ. Each relying party may define its own requirements regarding the certainty of identity assurance, but there is a benefit of standardising those requirements into a set of identity assurance levels (IALs). [24]

Identity Assurance is a measure of the confidence that an identity claim or assertion is true. An Identity Assurance Level is a relative measure (e.g., low, medium, high, very high) of the strength of assurance that can be placed in an identity claim. A lower level of assurance means less certainty in an identity claim, while a higher level of assurance indicates a higher degree of certainty.

Figure 52.7, below, illustrates this relationship as an equation. Essentially, the levels of assurance created are a minimum function of this equation which consists of:

1. the rigour of the original registration and evidence of identity process;
2. the strength of the credential used for authentication; and,
3. the authentication event, itself.

The model introduces assurance equation:

Identity Assurance = min(Identification Level, Credential Strength Level, Authentication Level).

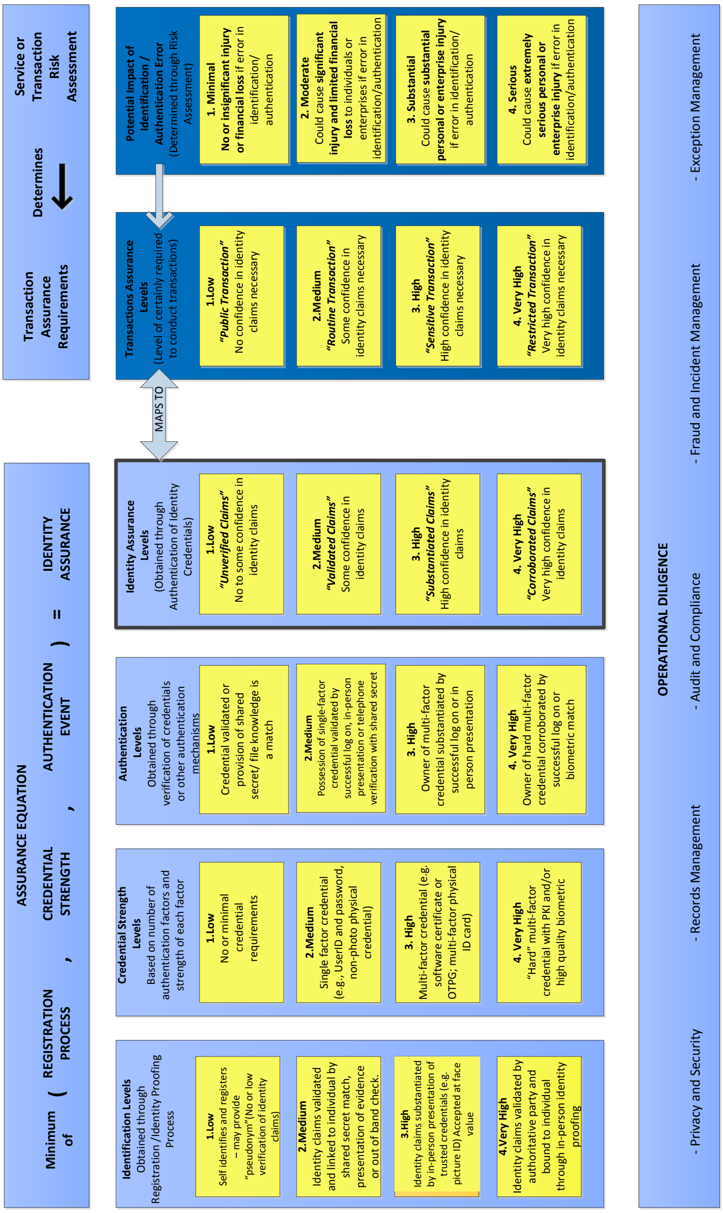

Service or Transaction Risk Assessment
Determines
Transaction Assurance Requirements
Potential Impact of Identification / Authentication Error (Determined through Risk Assessment)
1. Minimal No or insignificant injury or financial loss if error in identification/ authentication
2. Moderate Could cause significant injury and limited financial loss to individuals or enterprises if error in identification/authentication
3. Substantial Could cause substantial personal or enterprise injury if error in identification/ authentication
4. Serious Could cause extremely serious personal or enterprise injury if error in identification/authentication
Transactions Assurance Levels (Level of certainly required to conduct transactions)
1.Low "Public Transaction" No confidence in identity claims necessary
2.Medium "Routine Transaction" Some confidence in identity claims necessary
3. High "Sensitive Transaction" High confidence in identity claims necessary
4. Very High "Restricted Transaction" Very high confidence in identity claims necessary
MAPS TO
ASSURANCE EQUATION
Minimum of (REGISTRATION PROCESS , CREDENTIAL STRENGTH , AUTHENTICATION EVENT) = IDENTITY ASSURANCE
Identification Levels Obtained through Registration /Identity Proofing Process
1.Low Self identifies and registers – may provide "pseudonym"(No or low verification of identity claims)
2.Medium Identity claims validated and linked to individual by shared secret match, presentation of evidence or out of band check.
3.High Identity claims substantiated by in-person presentation of trusted credentials (e.g. picture ID) Accepted at face value
4.Very High Identity claims validated by authoritative party and bound to individual through in-person identity proofing
Credential Strength Levels Based on number of authentication factors and strength of each factor
1.Low No or minimal credential requirements
2.Medium Single factor credential (e.g., UserID and password, non-photo physical credential)
3. High Multi-factor credential (e.g. software certificate or OTPG; multi-factor physical ID card)
4. Very High "Hard" multi-factor credential with PKI and/or high quality biometric
Authentication Levels Obtained through verification of credentials or other authentication mechanisms
1.Low Credential validated or provision of shared secret/ file knowledge is a match
2.Medium Possession of single-factor credential validated by successful log on, in-person presentation or telephone verification with shared secret
3. High Owner of multi-factor credential substantiated by successful log on or in person presentation
4. Very High Owner of hard multi-factor credential corroborated by successful log on or biometric match
Identity Assurance Levels (Obtained through Authentication of Identity Credentials)
1.Low "Unverified Claims" No to some confidence in identity claims
2.Medium "Validated Claims" Some confidence in identity claims
3. High "Substantiated Claims" High confidence in identity claims
4. Very High "Corroborated Claims" Very high confidence in identity claims
OPERATIONAL DILIGENCE
- Privacy and Security
- Records Management
- Audit and Compliance
- Fraud and Incident Management
- Exception Management

Fig. 52.7. Identity Assurance Framework

Assurance levels are dynamically created through an authentication event (whether through a successful log-on or a successful in-person or telephone verification) and are dependent on the original identification or registration process and the strength of the credential used to authenticate the identity claim.

On the other hand, the level of identity assurance interconnected with Transaction assurance Level, that depends from Potential Impact of identification/Authentication Error.

Thus, assurance model takes into account various aspects: technical, organizational and managerial.

52.11. Conclusions

1. The difference between UK and US approaches is fundamental, and cannot be easily resolved by simply 'shifting' UK IALs up by one. The US approach is based on technology-driven risk assessment, while the UK approach is based on legal practices. Both can diverge at any time, either as a result of technological developments or changes in legal practices.
2. Further, IALs are only the part of the issue, because the successful IA should harmonise three components: the quality of credentials (e.g. whether they are tamper-resistant), the quality of the process of its issuance (e.g. whether the sufficient vetting has been performed) and the quality of its verification (e.g. whether the on-line verification is mandatory). Thus e.g. 'level 3' assurance may require designated 'level 3' credentials issued by a 'level 3 certified' identity provider, verified by the 'level 3' process. Currently those components can be connected only by technological practice and convenience, not by requirements implied by the UK Gov IALs.
3. Technology-based IALs bring the risk of over-reliance on technology. There are always alternative, non-technological, means of establishing identity, well established in legal practices. For example, the identity of a person can be established 'beyond reasonable doubt' (i.e. at level 3) by a corroborated witness statement, in the absence of other means of identification. Similarly, the balance of probabilities can be satisfied by a variety of procedures, not always requiring any credential at all (e.g. 'known personally' clause used by financial institutions).
4. The levels defined in the assurance model are consistent with emerging industry and public sector models (NIST, Liberty Alliance, etc.), although they have been adapted slightly to suit the Pan-Canadian context. The key difference from other models is that Pan-Canadian levels are first defined generically. These generic standards can then be applied to various categories of assurance, in the case of the Pan-Canadian Model specifically, to establish Identity Assurance Levels and Credential Assurance Levels. The primary benefit of articulating generic, standardized assurance levels is that the assurance model

can be easily extended and applied to other risk categories as they are identified (currently out of scope of this document).

Table 52.14 is a comparison of the Pan-Canadian assurance levels to the other major models.

Table 52.14. Comparison of Assurance Levels in Other Models

Identity Assurance Level	Pan-Canadian Model	Liberty Alliance	OMB M04-04	eID Interoperability for PEGS[14]
None	No confidence required	N/A	N/A	N/A
Level 1	Little confidence required	Little or no confidence in the asserted identity	Little or no confidence in the asserted identity's validity	Minimal Assurance
Level 2	Some confidence required	Some confidence that an asserted identity is accurate	Some confidence in the asserted identity's validity	Low Assurance
Level 3	High confidence required	High confidence in asserted identity	High confidence in the asserted identity's validity	Substantial Assurance
Level 4	Very high confidence required	Very high confidence in asserted identity	Very high confidence in the asserted identity's validity	High Assurance

52.12. Questions and tasks for self-control

1. What is NIST?
2. The main objects of international standartisation.
3. The scope of the international standard ISO/IEC 15408.
4. The Evaluation concepts and relationships.
5. Ontological model of “confidence” and “assurance” concepts.

6. The characteristic of Evaluation Assurance Level.
7. The scope of international standard ISO/IEC 24759 and NIST FIPS 140-2.
8. The Security Level FIPS 140-2.
9. The characteristics of Requirements for cryptographic modules according to international standard.
10. The security assurance requirements of cryptographic protocols.
11. The characteristics of Protocol assurance levels.
12. The scope of models of Identity assurance.
13. The Overview of the entity authentication assurance framework ISO/IEC 29115.
14. The characteristic of Levels of assurance ISO/IEC 29115
15. US approach to Identity Assurance. The Model of Identity Assurance Level.
16. The main elements of assurance model NIST, Identity assurance Level.
17. UK Government approach to Identity assurance. UK Gov Identity Assurance Levels.
18. The relationship between EAL and LoA.
19. The model of EU of Identity Assurance. Assurance level eID means.
20. Pan-Canadian Assurance Model. Main element of model. Level Descriptions.
21. Main elements of the comparative analysis of identity assurance models.

53. Model of IT-security assurance evaluation process

The relevance of ensure execution of requirements to the process and results of the assurance evaluation determine the necessity of scientific-methodological apparatus elements development, as which can act methods, means, techniques, approaches, methodologies of tasks solution in the field of assurance evaluation.

Since security assurance evaluation is an activity, including processes of interaction subjects examination and processes of actions execution for the evaluation in the course of the examination, then to the assurance evaluation is expedient to approach from the position of the process approach[25], i.e. to consider as a process. The common features of any process as well as different process definitions are considered in [26]. Based on these results, we make the following definition.

Definition 53.1. Evaluation process of security assurance requirements is a set of interrelated operations and actions, aimed at research, testing, analysis and evaluation of the examination object. That is achieved by converting input material and information flows in the output flows, of interest to the examination subjects, in order to determine the extent of correspondence of examination object characteristics to the specified requirements and to determine the possibility of using the estimated object as a trustee in terms of IT security.

In developing the evaluation assurance process model the following aspects were taken into account:

– functional, which specifies what the process elements do;

– Informational, which shows an informational entity that is formed or used by process;

– organizational, which describes when and who performs the actions, the work, the process operations, including the physical mechanisms of transmission and preservation of objects;

– causal, which refers to coordination and dependence of actions, and to the subjects of these activities and resources.

The bases of a formal model of IT security assurance evaluation process are the following axiomatic designs.

A1. The assurance evaluation process is set of actions $A = \{A_n | n = \overline{1, N}\}$ for evaluation of assurance requirements.

Actions describe the work that is performed by experts and other participants of evaluation process. A set of actions can be represented as a graph G^A, which describes the specific types of relations on the set of actions. The set of actions constitute an evaluation methodology.

A2. A set of relations $D = \{D_m | d = \overline{1, M}\}$ of different types, defined on the set A.

The set of actions A is forming process $Z(A, d)$, if $A \in \overline{A}$ and $d(A) \in D$ is defined on this set. The d may be "part-whole", «causal dependence», « existential dependence» relation and etc. Evaluation assurance actions set $\overline{A}$ is defined in the regulations [28]. For the process formation on the set A it is necessary to specify the dependency relations. In this case, the actions set can be considered as an actions sequence.

A3. appointment of a process is formed by the purpose *TRG* and expected results *REZ* .

$$Purpose = < TRG, REZ >. \qquad (53.1)$$

The process is implemented and executed to achieve a specific goal and the results of interest to the participants. The goal stands as a factor which determines the relations on the actions set. In the context of IT security assurance evaluation the main purpose is establishing the degree of satisfaction of a given assurance requirements set and accordance of the examination object to the certain level of security assurances.

Main goal can be decomposed into subgoals, in particular to a set of properties $P = \{P_j | j = \overline{1, J}\}$, which must have the examination object to satisfy the assurance requirements. The set of properties can be represented as a graph $G^P = \{P, D\}$, where D – set of relations established on the set of properties. The set of properties constitute the evaluation program *Pr* [29].

As a result of evaluation process *REZ* perform the set of conclusions (verdicts) of experts $V = \{V_i | i = \overline{1, I}\}$ relating to the manifestation degree of certain properties of IT security assurances. These verdicts are issued as a report, that is the material demonstration of the result of IT security assurance evaluation. Assurance requirements analysis, which are fixed by regulations [30], led to make the conclusion, that requirements are qualitative and not quantifiable. This representation complicates its analysis and formal representation. One solution to this problem is the use of the apparatus of linguistic variables, allowing to set up variables formal values as a verbal expressions. By using this approach [31] for each property linguistic variable L is introduced and determined its term-set β_L, i.e. a set of values that it can take. Thus, each verdict is a reflection of the value of linguistic variable, which it adopted in the examination course. Each verdict V_i contains the value of linguistic variable of *i*-th property, and the report can be presented as a set of linguistic variables $V=\{L_i | i = \overline{1, I}\}$.

Thus, appointment of IT security assurance evaluation process will formulate the objectives tree G^P (which is described in the evaluation program) and the set of verdicts V, presented as a report:

$$Purpose = < G^P, V >. \tag{53.2}$$

A4. A set of inputs $IN = \{I^{in}, M^{in}\}$ an outputs $OUT = \{I^{out}, M^{out}\}$ of process Z with a predetermined conversion operator $F : IN \rightarrow OUT$.

Generally, process inputs and outputs are the material and informational flows. Material flow M is continuous or discrete set of material objects $M = \{m_q | q = \overline{1,Q}\}$, which is distributed in time. Informational flow I is continuous or discrete set of informational objects $I = \{i_n | n = \overline{1,N}\}$. The assurance evaluation process will include two flows: material – *TOE* examination object, and material-informational – a set of evidence E.

In domestic regulatory documents [30] as an examination object for compliance with the assurance requirements perform technical measures for protection from unauthorized access, which include software, hardware, and software and hardware, that is created as a separate product production. Also it has the necessary design and/or operational documentation and provides by itself or with other means protection against risks of unauthorized access to information, which is processed in the information and telecommunications system. The International Standard [1] define target of evaluation as a set of software, software and hardware, and hardware, which may be accompanied by guidance.

In general, the examination object (target of evaluation) *TOE* can be expressed as:

$$TOE = <Hw, Sw, D>, \tag{53.3}$$

where $Hw=\{Hw_i | i = \overline{1,I}\}$ – hardware components of the *TOE*; $Sw=\{Sw_j | j = \overline{1,J}\}$ – software components; $D=\{D_z | z = \overline{1,Z}\}$ – documentation package for *TOE*.

Evaluation of *TOE* conducted on obtained evidences E. The main sources of evidence are received from the customer's expertise (examination object developer) documents and materials, oral statements and written responses of employees of the expertise customers organization (examination object developer), results of observations to these employees, examination object and its parts. The resulting set of evidence can be formally expressed as a set of $E = \{e_y | y = \overline{1,Y}\}$, and if there are inheritance relations (hierarchical dependencies) – as a graph G^E (or table).

A5. A set of participants-subjects of the process. In general, this is

$$PS = (Own, Man, Per, Sup, Cus), \tag{53.4}$$

where *Own* – process owner; *Man* – process manager; *Per* – process performer; *Sup* – process supplier; *Cus* – process customer. The set of process participants are forming a team *Process_team*, if on the set *PS* are defined roles *role* and authorities *authority* of process subjects, which characterizes relations between process participants, i.e.

$$Process\_team\ (PS,\ role,\ authority). \qquad (53.5)$$

In the [41] as an examination subjects are defined: legal and natural persons who are the expertise customers; the competent State authority; subdivision of the competent State authority, enterprises, institutions and organizations that conduct an examination (examination organizers); state authorities; natural persons – performers of expert work (experts). In this occasion, the subjects can be represented as a set *PS*. In [42] the author deeply analyzed the international standard ISO/IEC 15408 [1,2] and identified the subjects involved in the evaluation process. The results of this analysis can be represented as a graph G^{PS}.

Among the set of the examination subjects (evaluation), we are interested in experts *B*, i. e. persons performing actions and making decision concerning the assurance evaluation properties (verdicts). Examination organizers does selection of experts. One of the expert selection methods is the choice by the formal features, such as: post, education, academic degrees and titles, experience, competence, successful participation in previous examinations, etc. These features may be used for comparing the experts. For this purpose they must be represented as a set $B_i = \{b_{ix} | x = \overline{1, X}\}$, where X – number of formal features, i – i-th expert.

A6. A set of financial *F*, temporal *T*, labor force *L*, economical *E*, material *Mat* and other resources, required for the implementation of process *Z*:

$$Resource = \{F, T, L, E, Mat\}.$$

Based on the entered designs A1-A6 there is the following formal assurance evaluation process model:

$$Z = <Pur\ (G^P, V),\ d(A),\ F(TOE, E) : IN \to OUT,\ Process\_team\ (B),$$
$$Resource >.$$

Graphical interpretation of this model is shown in Figure 53.1.

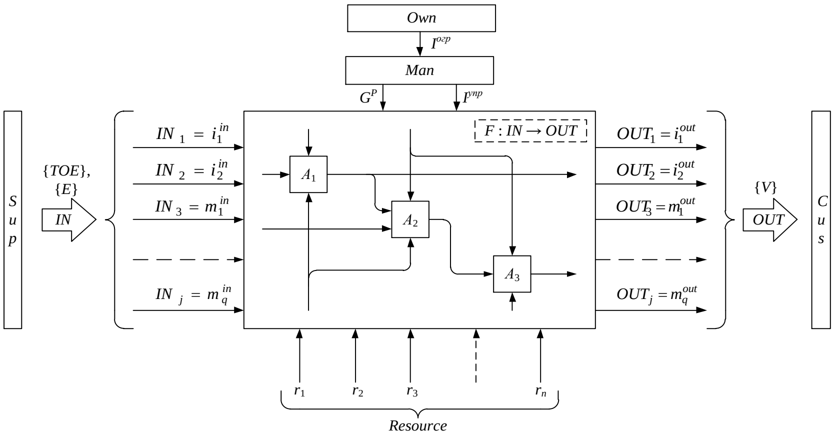


Fig. 53.1. Graphical interpretation of the model of IT security assurance evaluation process

This model can be considered as the base and used for further studies of the IT security assurance evaluation process.

53.1. Analytical requirements models for assurance evaluation results

In the context of the proposed assurance evaluation process model (53.7) there are some requirement definitions imposed on examination results and presented as analytical models [42, 44]. Analytical models will be offered only to the requirements of repeatability, reproducibility and comparability because of the requirements specificity. Objectivity and impartiality requirements will be clarified verbally.

53.2. Description of the objectivity requirement

Definition 53.2. Objectivity is a property supposed that the evaluation requirements results must be actual, i.e. not to be undergone to sense influence or expert (evaluator) opinions.

Objectivity can be ensured, if evaluation results obtained by evidence analysis and the expert is confident in its reliability. Objectivity and impartiality of the evaluation results are intersected closely.

53.3. Description of the impartiality requirement

Definition 53.3. Impartiality is a property which provides the assurance requirements evaluation not prejudiced towards any specific evaluation result.

Impartiality can be ensured, if expertise organizers and experts that conduct evaluation are independent. The experts should not communicate with the organization - the expertise customer and with object developers. Experts may be only those people (organizations), for which can be documented their non-participation in any of the stages of evaluation object development, including advisory services.

Thus, the impartiality provided by the examination organization procedure, which is enshrined in the regulations.

53.4. Analytical model of repeatability requirement

Definition 53.4. Repeatability is a property which provides the identity of the evaluation results during a re-evaluation of the same TOE held by the same program and methodology of security requirements evaluation by the same expert (evaluator).

Analytical model of repeatability requirement will be presented as:

$Z_{(t_0,t_1)}$		$Z_{(t_2,t_3)}$	– examination number
(t_0, t_1)	$\neq$	(t_2, t_3)	– time of the examination
TOE_1	=	TOE_2	– target of evaluation
Pr_1	=	Pr_2	– evaluation program
A_1	=	A_2	– evaluation methodology
B_1	=	B_2	– an expert
E_1	=	E_2	– set of evidences
	⇓		
V_1	=	V_2	– evaluation results

Within the proposed model, there is the problem of comparing individual elements of the model of assurance evaluation process. To solve this problem, we introduce the formal terms of a comparison of some model elements.

Condition 53.1. Two targets of evaluation TOE_1 and TOE_2 are identical, if the condition of equality of the respective sets of components is satisfied:

$$\text{IF } (\{Hw_i | i = \overline{1,I}\}_{TOE_1} = \{Hw_i | i = \overline{1,I}\}_{TOE_2}, \{Sw_j | j = \overline{1,J}\}_{TOE_1} =$$

$$= \{Sw_j | j = \overline{1,J}\}_{TOE_2}, \{D_z | z = \overline{1,Z}\}_{TOE_1} = \{D_z | z = \overline{1,Z}\}_{TOE_2})$$

$$\text{THEN } TOE_1 = TOE_2$$

When the comparing of two TOE equality condition of at least one pair of sets are not satisfied, the TOE are non-identical.

Condition 53.2. Two evaluation programs Pr_1 and Pr_2 are identical, if equality conditions of graph $G^P_{\mathrm{Pr}_1} = G^P_{\mathrm{Pr}_2}$ and their adjacency matrix $(p_{ij})_{\mathrm{Pr}_1} = (p_{ij})_{\mathrm{Pr}_2}$ are satisfied:

$$IF\ (G^P_{\mathrm{Pr}_1} = G^P_{\mathrm{Pr}_2},\ (p_{ij})_{\mathrm{Pr}_1} = (p_{ij})_{\mathrm{Pr}_2}, i = \overline{1,m}, j = \overline{1,n})\ THEN\ Pr_1 = Pr_2. \quad (53.9)$$

If equality conditions are not satisfied, then the programs are different.

Condition 53.3. Two methodologies A_1 and A_2 are identical, if equality condition of graph $G^A_{A_1} = G^A_{A_2}$ and their adjacency matrix $(a_{ij})_{A_1} = (a_{ij})_{A_2}$ are satisfied:

$$\text{IF } (G^A_{A_1} = G^A_{A_2} \text{ and } (a_{ij})_{A_1} = (a_{ij})_{A_2}, i = \overline{1,m}, j = \overline{1,n}) \text{ THEN } A_1 = A_2.$$

If equality conditions are not satisfied, then the methodologies are different.

Condition 53.4. Two experts B_1 and B_2 are identical, if equality conditions of set of their formal features are satisfied:

$$\text{IF } (B_{1x}|x = \overline{1,X}\} = \{B_{2x}|x = \overline{1,X}\}) \text{ THEN } B_1 = B_2. \quad (53.11)$$

If equality conditions of set are not satisfied, then the experts are different.

Condition 53.5. Two sets of evidences E_1 and E_2 are identical, if equality conditions of set are satisfied:

$$\text{IF } (\{e_y|y = \overline{1,Y}\}_{E_1} = \{e_y|y = \overline{1,Y}\}_{E_2}) \text{ THEN } E_1 = E_2, \quad (53.12)$$

or equality of graphs and adjacency matrices:

$$\text{IF } (G^E_{E_1} = G^E_{E_2} \text{ и } (e_{ij})_{E_1} = (e_{ij})_{E_2}, i = \overline{1,m}, j = \overline{1,n}) \text{ THEN } E_1 = E_2. \quad (53.13)$$

If equality conditions are not satisfied, then the sets of evidences are different.

Condition 53.6. Two reports V_1 and V_2 are identical, if the equality condition of the sets of values of linguistic variables corresponding reports $\{L_i|i = \overline{1,I}\}_{V_1}$ and $\{L_i|i = \overline{1,I}\}_{V_2}$ are satisfied:

$$\text{IF } (\{L_i|i = \overline{1,I}\}^{V_1} = \{L_i|i = \overline{1,I}\}_{V_2}) \text{ THEN } V_1 = V_2. \quad (53.14)$$

If equality conditions are not satisfied, then the evaluation reports are different.

Based on the conditions 53.1-53.6 and definition 53.4, we introduce a formal condition for the requirement of repeatability.

Condition 53.7. Assurance evaluation results are considered to be repeatable, if *TOE* was subject to two examinations (1.15) and (1.16)

$$Z_{(t_0,t_1)} = < Pur\,(Pr_1, V_1),\ d(A_1),\ F(TOE_1, E_1) : IN \rightarrow OUT,$$
$$Process\_team\,(B_1), Resource >, \qquad (53.15)$$

$$Z_{(t_2,t_3)} = < Pur\,(Pr_2, V_2),\ d(A_2),\ F(TOE_2, E_2) : IN \rightarrow OUT,$$
$$Process\_team\,(B_2), Resource >, \qquad (53.16)$$

at the time of (t_0, t_1) and (t_2, t_3) accordingly, satisfied the conditions: TOE identity $TOE_1 = TOE_2$ (53.8), evaluation programs identity $Pr_1 = Pr_2$ (1.9), evaluation methodologies identity $A_1 = A_2$ (53.10), experts identity $B_1 = B_2$ (53.11), the sets of evidences identity $E_1 = E_2$ (53.12 or 53.13), and the identity conditions of examination results is satisfied $V_1 = V_2$ (53.14).

Thus, the repeatability condition says that the examination results are invariant with respect to time.

53.5. Analytical model of reproducibility requirement

Definition 53.5. Reproducibility is a property which provides the identity of the evaluation results during a re-evaluation of the same TOE held by the same program and methodology of security requirements evaluation by the different experts (evaluators).

Analytical model of reproducibility requirement will be presented as:

$Z_{(t_0,t_1)}$		$Z_{(t_2,t_3)}$	– examination number
(t_0, t_1)	$\neq$	(t_2, t_3)	– time of the examination
TOE_1	=	TOE_2	– target of evaluation
Pr_1	=	Pr_2	– evaluation program
A_1	=	A_2	– evaluation methodology
B_1	$\neq$	B_2	– an expert
E_1	=	E_2	– set of evidences
	$\Downarrow$		
V_1	=	V_2	– evaluation results

Based on the conditions 53.1-53.6 and definition 53.5, we introduce a formal condition for the requirement of reproducibility.

Condition 53.8. Assurance evaluation results are considered to be reproducible, if *TOE* was subject to two examinations (1.17) and (1.18)

$$Z_{(t_0,t_1)} = < Pur\ (Pr_1, V_1),\ d(A_1),\ F(TOE_1, E_1) : IN \rightarrow OUT,\ Process\_team\ (B_1), Resource >, \tag{53.17}$$

$$Z_{(t_2,t_3)} = < Pur\ (Pr_2, V_2),\ d(A_2),\ F(TOE_2, E_2) : IN \rightarrow OUT,\ Process\_team\ (B_2), Resource >, \tag{53.18}$$

at the time of (t_0, t_1) and (t_2, t_3) accordingly, satisfied the conditions: TOE identity $TOE_1 = TOE_2$ (53.8), evaluation programs identity $Pr_1 = Pr_2$ (53.9), evaluation methodologies identity $A_1 = A_2$ (53.10), experts non-identity $B_1 \neq B_2$ (53.11), the sets of evidences identity $E_1 = E_2$ (53.12 or 53.13), and the identity condition of examination results is satisfied $V_1 = V_2$ (53.14).

Thus, the reproducibility condition says that the examination results are invariant with respect to time and experts.

53.6. Analytical model of comparability requirement

Definition 53.6. Comparability is a property which provides matching the evaluation results obtained during the evaluation of the TOE held by different program and evaluation methodology by another (or the same) expert (evaluator).

Analytical model of comparability requirement will be presented as:

$Z_{(t_0,t_1)}$		$Z_{(t_2,t_3)}$	– examination number
(t_0, t_1)	≠	(t_2, t_3)	– time of the examination
TOE_1	=	TOE_2	– target of evaluation
Pr_1	≠	Pr_2	– evaluation program
A_1	≠	A_2	– evaluation methodology
B_1	≠ (=)	B_2	– an expert
E_1	≠ (=)	E_2	– set of evidences
	⇓		
$V_1(L_i)$	<=>	$V_2(L_i)$	– evaluation results

Based on the conditions 53.1-53.6 and definition 53.6, we introduce a formal condition for the requirement of comparability.

Condition 53.9. Assurance evaluation results are considered to be comparable, if *TOE* was subject to two examinations (1.19) and (1.20)

$$Z_{(t_0,t_1)} = < Pur\ (Pr_1,\ V_1),\ d(A_1),\ F(TOE_1,\ E_1) : IN \rightarrow\ OUT,\ Process\_team\ (B_1), Resource >, \qquad (53.19)$$

$$Z_{(t_2,t_3)} = < Pur\ (Pr_2,\ V_2),\ d(A_2),\ F(TOE_2,\ E_2) : IN \rightarrow OUT,\ Process\_team\ (B_2),\ Resource >, \qquad (53.20)$$

at the time of (t_0, t_1) *and* (t_2, t_3) accordingly, satisfied the conditions: TOE identity $TOE_1 = TOE_2$ (53.8), evaluation programs non-identity $Pr_1 \neq Pr_2$ (53.9), evaluation methodologies non-identity $A_1 \neq A_2$ (53.10), experts identity or non-identity $B_1 = (\neq)\ B_2$ (53.11), the sets of evidences identity or non-identity $E_1 = (\neq)\ E_2$ (53.12 or 53.13), and the matching condition of examination results is satisfied $V_1(L_i) <=> V_2(L_i)$.

The matching condition determines the matching of results, i.e. features or characteristics of matching results that provides matching. Because the reports can be presented in various forms, such characteristics can be obtained in the analysis of evaluation results. For example, in the design of evaluation program *Pr* for each property P_i linguistic variable L_i, was introduced, then as comparison criteria may be the corresponding values of linguistic variables.

53.7. Conclusions

1. The formal model of assurance evaluation process developed from the standpoint of the process approach was introduced. The process model of assurance evaluation was obtained for the first time and was used as a base for the examination processes study.

2. Requirements definitions to the assurance evaluation results were formulated (objectivity, impartiality, repeatability, reproducibility, comparability). Based on the assurance evaluation process model the analytical models of requirements of the repeatability, reproducibility and comparability were developed. Within the analytical models the identity conditions of objects evaluation, evaluation programs, evaluation methodologies, experts, evidences set and evaluation results were formulated. These conditions are clear and are based on strict equality definitions of the corresponding sets or graphs that allows to objectively verifying (proving, justifying) requirements for examination results in practice.

3. It is shown that the requirements of objectivity and impartiality of the evaluation results provided by the examination organization (constitutional provisions), construction of expertise system at the state level. Execution of

requirements of repeatability and reproducibility of evaluation results means that evaluation results have to be invariant to the time and experts. For the requirements of comparability it is necessary to introduce conditions (criteria) of comparison. Presenting the assurance evaluation results in the form of verbal description complicates the introduction of such criteria. It is therefore necessary to study the comparability nature and developing the ways of comparing the verbal evaluation results.

53.8. Questions and tasks for self-control

1. Model of IT-security assurance evaluation process. Main Definitions, basic elements of process.
2. What is TOE, PP and ST?
3. Give a formal description of evaluation process.
4. Give the Graphical interpretation of the model of IT security assurance evaluation process.
5. Give definition of the objectivity requirement.
6. Analytical model of repeatability requirement. Definition, conditions.
7. Analytical model of reproducibility requirement. Definition, conditions.
8. Analytical model of comparability requirement. Definition, conditions.

54. Method of functional-linguistic approach to the security assurance evaluation

The proposed model of assurance evaluation process and conditions of evaluation process requirements that developed to the analytical models, are creating the preconditions for the evaluation assurance method development. The idea that is the foundation of this method is set out in the functional-linguistic approach concept. The method includes the description of the developed ways to build an object-oriented and process-oriented ontological models and recommendations of using known means of modeling and elements of mathematical apparatus for the purpose of functional-linguistic approach to the security assurance evaluation.

54.1. Conception of Functional-Linguistic Approach to the Security Assurance Evaluation

During the analysis of assurance evaluation domain authors advanced a proposal to evaluate not the TOE, but its inherent assurance properties. [45]. These properties are detected during the requirements. To prove that a particular property is inherent (available) to the TOE, it is using evidences, as which can be TOE, its parts, documentation, test results. Thereby, IT-product evaluation consists of manifestation degree assessment of assurance properties inherent to the product. This is the key idea of the approach to the assurance evaluation proposed by authors.

Functional-linguistic approach structure is shown in Figure 54.1. Assurance evaluation is implemented in four phases.

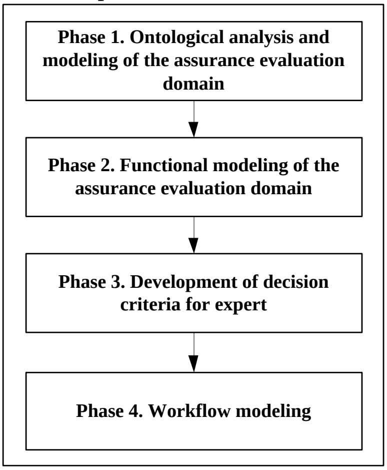

Phase 1. Ontological analysis and modeling of the assurance evaluation domain
Phase 2. Functional modeling of the assurance evaluation domain
Phase 3. Development of decision criteria for expert
Phase 4. Workflow modeling

Fig. 54.1. Functional-linguistic approach to the assurance evaluation

In the *phase 1* the ontological analysis and modeling of the domain evaluation are carried out [45].

The main tasks of the ontological assurance requirements analysis are :

– exact, clear and unambiguous description of the subject area of IT security assurance evaluation, and selection of the basic concepts;

– a clear definition of the evaluation content;

– determining the required depth of evaluation;

– converting a set of assurance static requirements in a dynamic framework for its using by people from different spheres of activity and understanding of the evaluation area;

– representation of requirements of assurance evaluation sphere in a form that allows to create its electronic analogue.

Analysis includes the research of the assurance requirements set ($R=\{r_1, r_2,...,r_i\}$, $i = \overline{1,N}$) advanced to the TOE, and detection the assurance properties set ($P=\{p_1, p_2,...,p_j\}$, $j = \overline{1,L}$) the TOE must possess. The assurance properties set *P* defines dependences and relations among properties. Analyses results are shown in form of ontological graphs, that exactly and unambiguously (in accepted notation) describe the domain (notably the main concepts and relations among them). Complex coverage of domain modeling is ensured by ontological graphs of two types: object-oriented and process oriented.

In the *phase 2* the functional modeling of the assurance evaluation process is implemented [45]. Functional modeling goal is the formalized presentation of the evaluation process. As modeling language IDEF0 notation was selected [42]. IDEF0 notation makes it possible to defined evaluation steps unambiguously (as a directed graph), determine for each step the assurance property which is evaluated, evidences which is necessary for the evaluation of this assurance properties, evaluation subjects and regulations. If it necessary to evaluate complex property, each step (diagram box) can be decomposed for the detail description of subproperties evaluation.

In the *phase 3* for each property p_j the linguistic variable $\Omega p_j = <\beta, T(\beta), G, M>$ and its term-set $T(\beta)$ is defined. The application of linguistic variables can be explained by impossibility of quantitative characteristics usage for most assurance properties. Therefore for making decision on inherent degree of assurance properties it is convenient to use mathematical techniques of fuzzy inference conclusion on the ground of production rules basis [44]. Application of linguistic variables and fuzzy logic operations are provide the requirements implementation of objectivity and repeatability of assurance evaluation results.

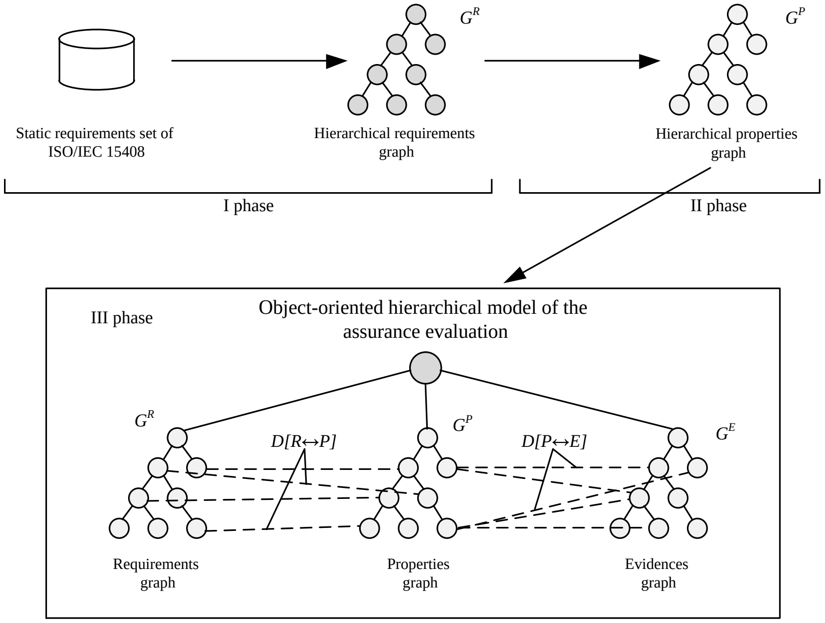
In the *phase 4* workflow diagrams in IDEF3 notation are constructed . It makes it possible to define the order and priority of evaluation actions implementation. Each diagram box represents the separate evaluator action. Each box is followed by the node, which defines the rule for chose the next action according to evaluator decision about inherent degree of evaluated property. The number of options depends on the number of values that can take the linguistic variable which describes the evaluated property. The diagrams defines the points at which expert must make a decision and reach a verdict on the degree of property manifestation. Applications of IDEF3 diagrams ensure the requirements implementation of evaluation results repeatability.

Selecting the verdict depends on what values take linguistic variables in the evaluation of properties (in fact it is an expert choice which depends on the degree of properties manifestation).

So, approach realization makes it possible to implement requirements to assurance evaluation process and evaluation results.

54.2. Method of the Object-Oriented Assurance Ontological Modeling

The process-oriented assurance ontological modeling is implemented into 3 phases (Figure 54.2) .



G^R
G^P
Static requirements set of ISO/IEC 15408
Hierarchical requirements graph
Hierarchical properties graph
I phase
II phase
III phase
Object-oriented hierarchical model of the assurance evaluation
G^R
G^P
G^E
$D[R\leftrightarrow P]$
$D[P\leftrightarrow E]$
Requirements graph
Properties graph
Evidences graph

Fig. 54.2. Object-oriented ontological model of the assurance evaluation domain

For ontological modeling it is necessary to introduce the formal notation for each graph node. Formal note will be of form $R_i^{\{j\}}$, where i – graph node number of current level, $\{j\}$ – set, which values are graph node numbers according to level from top level to level previous to current, it is a decomposition way for node R_i of current level. The power of requirements (properties, evidences) set shown by hierarchical ontology graph can be defined by formula:

$$W = \sum_i \sum_h \sum_l G^i \cdot S_{h,l}\,, \tag{54.1}$$

where G^i – ontological graph of the i-th set, $i = \overline{1,3}$; $S_{h,l}$ – point degree of graph, equal to the number proceed from it lines, $h = \overline{1,H}$ – levels quantity of the ontological graph, $l = \overline{1,L_h}$ – point number on corresponding (h) level of the ontological graph.

Formal description form of the object-oriented ontological model by assurance evaluation domain is:

$$\Omega_O = < G^R, G^P, G^E, D >, \tag{54.2}$$

where G^R – the object-oriented ontological graph of assurance requirements set; G^P – the object-oriented ontological graph of assurance properties set; G^E – the object-oriented ontological graph of evidences; $D = \{D[R\leftrightarrow P], D[P\leftrightarrow E]\}$ – the relations set kind of “requirement-property” and “property-evidence”.

The process-oriented assurance ontological modeling is implemented into 3 phases:

Phase 1. The object-oriented hierarchical graph of the assurance requirements (G^R) is constructed. Depth degree (specification level) of requirements is defined. Depth degree of requirements is defined by the power of the TOE assurance requirements set. Dependence relations on assurance requirements set are detected. Their mode (part, existential, causal, intraclass, interclass etc.) are defined. Formal description form of the assurance requirements graph is:

$$G^R = <R, Q_R>, \tag{54.3}$$

where $R=\{r_1, r_2,...,r_i\}$, $i = \overline{1,N}$ – assurance requirements set, $Q_R=\{Q_f[r_i \leftrightarrow r_j]\}$, $f = \overline{1,F}$ – relations (dependences) set among assurance requirements.

Results of the requirements analysis recorded in the table (Table 54.1) and presented as a graph (Figure 54.3).

Table 54.1. Results of the requirements analysis

$R_i^{\{j\}}$	Requirement	Dependences between requirements

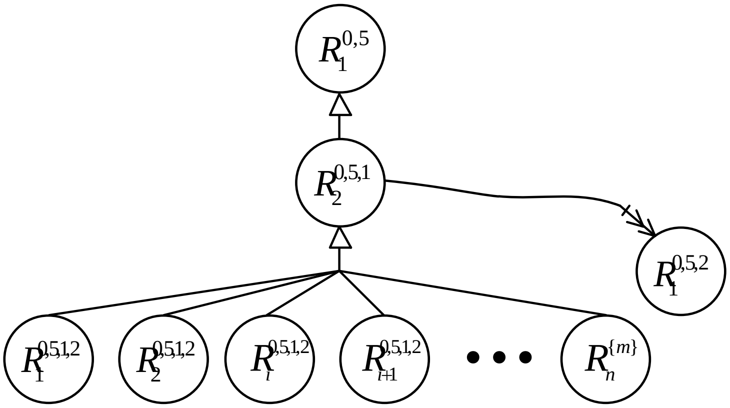


Figure 54.3 – Graph of assurance requirements

Phase 2. The object-oriented hierarchical graph of the assurance properties *(G^P)* is constructed. Dependences (relations) between requirements graph and properties graph *($D[R\leftrightarrow P]$)* are detected. The set of properties dependences Q_P grounded on Q_R dependences analysis is defined. Dependences can be repeated or arise as new ones. Complex assurance properties are defined. Complex is a property for evaluation of which it is necessary to check or examine the subproperties set. Formal description form of the assurance properties graph is:

$$G^P = <P, Q_P>, \tag{54.4}$$

where $P=\{p_1, p_2,...,p_i\}$, $i = \overline{1,N}$ – assurance properties set, $Q_P=\{Q_s[p_i \leftrightarrow p_j]\}$, $s = \overline{1,S}$ – relations (dependences) set among assurance properties.

The results of the properties analysis are recorded in the table (Table 54.2) and presented as a graph (Figure 54.4).

Table 54.2. The results of the properties analysis

$R_i^{\{j\}}$	Requirement	$P_i^{\{j\}}$	Property	Dependences	
				Inside-graphs	Between-

					graphs

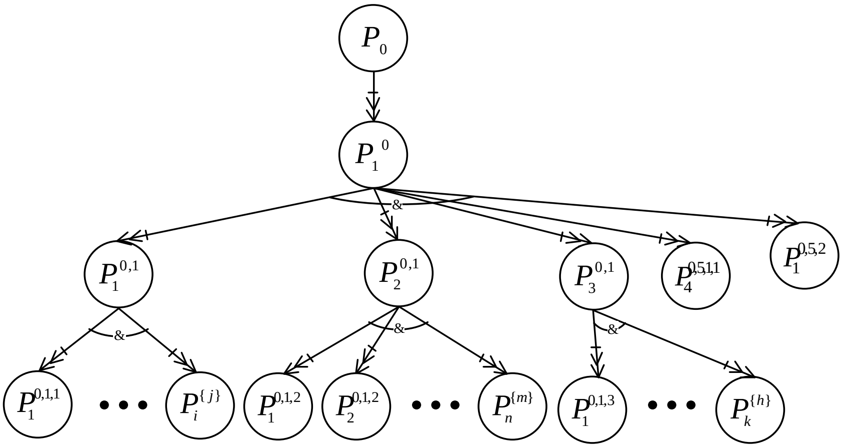


Figure 54.4 – Graph of assurance properties

Phase 3. The hierarchical graph of the evidences set *(G^E)* is constructed. Evidences are getting from the TOE decomposition. For each elementary property $p_i \in P$ the set of evidences $Ep_i=\{e_1, e_2, \ldots, e_i\}$, $i = \overline{1,N}$ is defined. Dependences between graphs G^P and G^E are shown in the form of relations kind of “property - evidence” $D[P\leftrightarrow E]$. Formal description form of the evidences graph is:

$$G^E = <E, Q_E>, \tag{54.5}$$

where $E=\{e_1, e_2,\ldots,e_z\}$, $z = \overline{1,Z}$ – evidences set, $Q_E=\{Q_y[e_i \leftrightarrow e_j]\}$, $y = \overline{1,Y}$ – relations set among evidences.

Results of the evidences analysis are recorded in the table (Table 54.3) and presented as a graph (Figure 54.5).

Table 54.3. Results of the evidences analysis

$E_i^{\{j\}}$	Evidence	Dependences

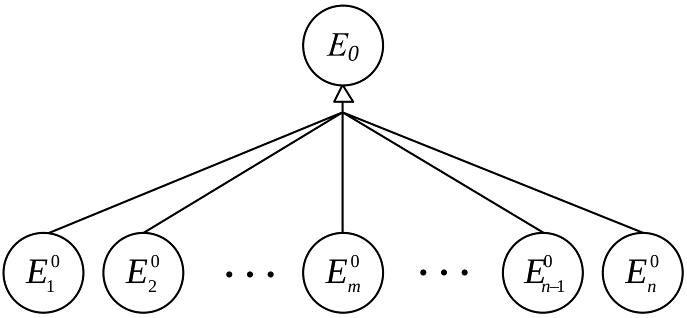


Figure 54.5 – Graph of evidences

The results of the analysis of correspondence between graphs G^P and G^E are recorded in the Table 54.4.

Table 54.4. The results of the analysis of correspondence between graphs G^P and G^E

Property ($P_i^{\{j\}}$)	The set of evidences { $E_i^{\{j\}}$ }, needed to properties evaluation $P_i^{\{j\}}$	Dependences between graphs G^P and G^E

For complex properties the correspondence between evidences and properties can be presented in a table (Table 54.5) or matrix form (54.6).

Table 54.5. Correspondence between evidences and properties

Complex property, P				
p_j / e_i	p_1	p_2	$\cdots$	p_j
e_1	$a_{1,1}$	$a_{1,2}$	$\cdots$	$a_{1,j}$
e_2	$a_{2,1}$	$a_{2,2}$	$\cdots$	$a_{2,j}$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
e_i	$a_{i,1}$	$a_{i,2}$	$\cdots$	$a_{i,j}$

$$A = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,j} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,j} \\ \cdots & \cdots & \cdots & \cdots \\ a_{i,1} & a_{i,2} & \cdots & a_{i,j} \end{pmatrix}. \qquad (54.6)$$

In the correspondence table in the cell we put "1" if to evidence e_i (table rows), inherent property p_j (table columns), i.e. for the property evaluation p_j, which is recorded in the column, it is necessary to use an evidence e_i. In the cell we put "0" if between evidence and property there is no relation (correspondence).

Completing of correspondence table orders (organizes) the knowledge about the TOE and allows to specify characterization of the evaluated properties. That affect the objectivity of the assurance properties evaluation.

Thus, a requirement and an evidence (one or set) are determined uniquely for each assurance property.

54.3. Method of the Process-Oriented Assurance Ontological Modeling

Process-oriented assurance ontology is constructed on the ground of ISO/IEC 18045 requirements [28]. The main reason of this ontology working out is the necessarily of relations identification between properties and evaluation actions. Construction a process-oriented ontology is working out taking into account the results obtained in the construction of an object-oriented ontological modeling. As inputs for process-oriented assurance ontological modeling the assurance requirements graph G^R, the assurance properties graph G^P and the relations set between them $D[R\leftrightarrow P]$ are used.

Thereby, formal description form of the process-oriented ontological model by assurance evaluation domain is:

$$\Omega_P = < G^R, G^P, G^A, D, G^B >, \qquad (54.7)$$

where G^R – ontological graph of the assurance requirements set; G^P – ontological graph of the assurance properties set; G^A – ontological graph of the evaluation actions set; $D = \{D[R\leftrightarrow P], D[R\leftrightarrow A], D[A\leftrightarrow P]\}$ – relations set kind of “requirement - property”, “requirement - action” and “action - property”; G^B – ontological graph of interested parties that participate in the evaluation process.

Modeling is implemented into 4 phases:

Phase 1. Evaluation assurance actions ontological graph (G^A) defined in the ISO/IEC 18045 is constructed. Dependence relations set (Q_A) on actions set A is defined. Formal description form of the actions graph is:

$$G^A = <A, Q_A>, \qquad (54.8)$$

where $A=\{a_1, a_2,\ldots,a_i\}$, $i = \overline{1,N}$ – evaluation assurance actions set, $Q_A=\{Q_s[a_i \leftrightarrow a_j]\}$, $s = \overline{1,S}$ – dependence set among evaluation assurance actions.

Results for analysis of assurance evaluation actions are recorded in the table (Table 54.6) and presented as a graph (Figure 54.6).

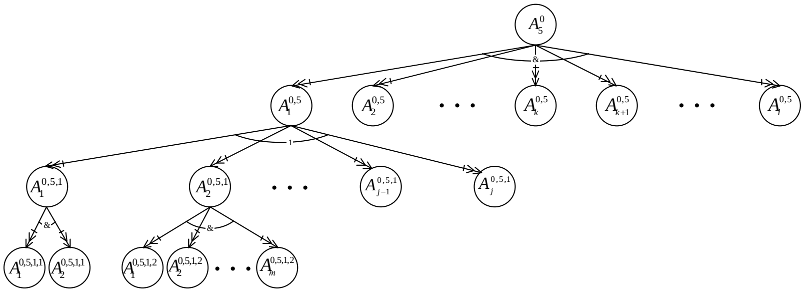


Fig. 54.6. Graph of analysis of assurance evaluation actions

Dependences set ($D[R\leftrightarrow A]$) between ontological graphs of actions (G^A) and requirements (G^R) is constructed. Interdependences between structural components of assurance requirements and assurance actions by ISO/IEC 18045 are shown in Figure 53.1.

The results of the analysis of relations between the ontological graphs actions G^A and requirements G^R are recorded in the table (Table 54.6).

Table 54.6. The results of the actions analysis

$A_i^{\{j\}}$	Actions	Dependences	
		Inside-graphs	Between graphs G^A and G^R

Phase 3. Dependences set ($D[A\leftrightarrow P]$) between ontological graphs of evaluation actions (G^A) and properties (G^P) are defined. These dependences are identified in indirect way, because it is impossible to do this directly. Correspondence model between actions and properties is shown in Figure 54.7.

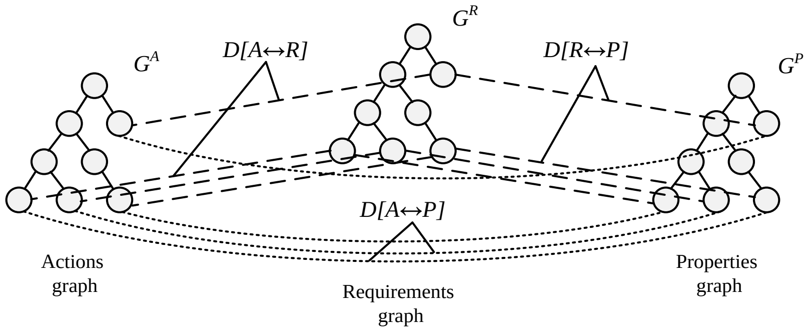


Fig. 54.7. Correspondence model between actions and properties

The results of the dependences analysis between the ontological graphs of assurance evaluation actions G^A and assurance properties G^P are recorded in the table (Table 54.7).

Table 54.7. The results of the dependences analysis between assurance properties and actions

$A_i^{\{j\}}$	Actions	$R_i^{\{j\}}$	Requirements	$P_i^{\{j\}}$	Propertie s	Dependence s between G^A and G^P

Phase 4. Interested parties ontology (G^B) of the assurance evaluation process is constructed. Formal description form of the parties graph is:

$$G^B = <B, Q_B>, \quad (54.9)$$

where $B=\{b_1, b_2,...,b_i\}$, $i = \overline{1,N}$ – interested parties set, $Q_B=\{Q_f[b_i \leftrightarrow b_j]\}$, $f = \overline{1,F}$ – relations set among parties.

The results of interested parties analysis are recorded in the table (Table 54.8) and presented as a graph (Figure 54.8).

Table 54.8. The results of interested parties analysis

$B_i^{\{j\}}$	Interested parties	Dependences

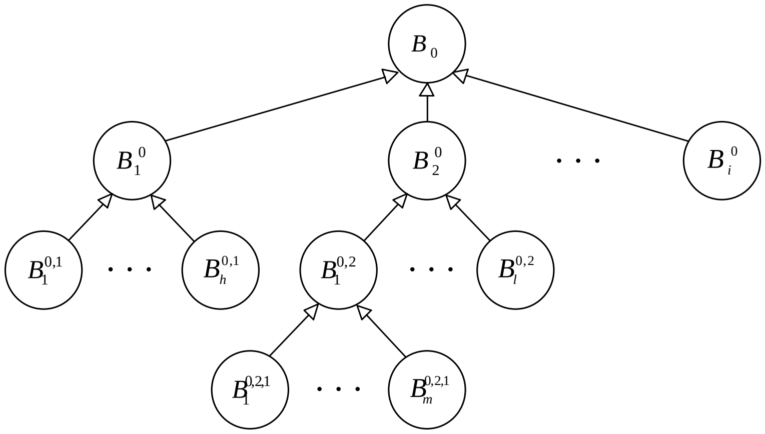


Fig. 54.8. Graph of interested parties

Interested parties ontology is widely disclosed in the work [37] and shown in Figure 54.9.

Summarized results of implementation of object-oriented and process-oriented modeling of assurance evaluation area are recorded in the Table 54.9. Results analysis shows that for each property *P*, which is necessary to evaluate are unambiguously identified: requirement *R*, from which the property is synthesized, actions *A* of property evaluation, evidences $\{E\}$, which are necessary to property evaluation, parties $\{B\}$ interested for property evaluation, and evaluation method $U(P)$.

Thus, the developed methods of object-oriented and process-oriented ontological modeling allows analyzing the assurance requirements, to which it is necessary to evaluate the examination object as being in compliance. The analysis results can be used to develop program evaluation elements and as input for the development of evaluation methodology.

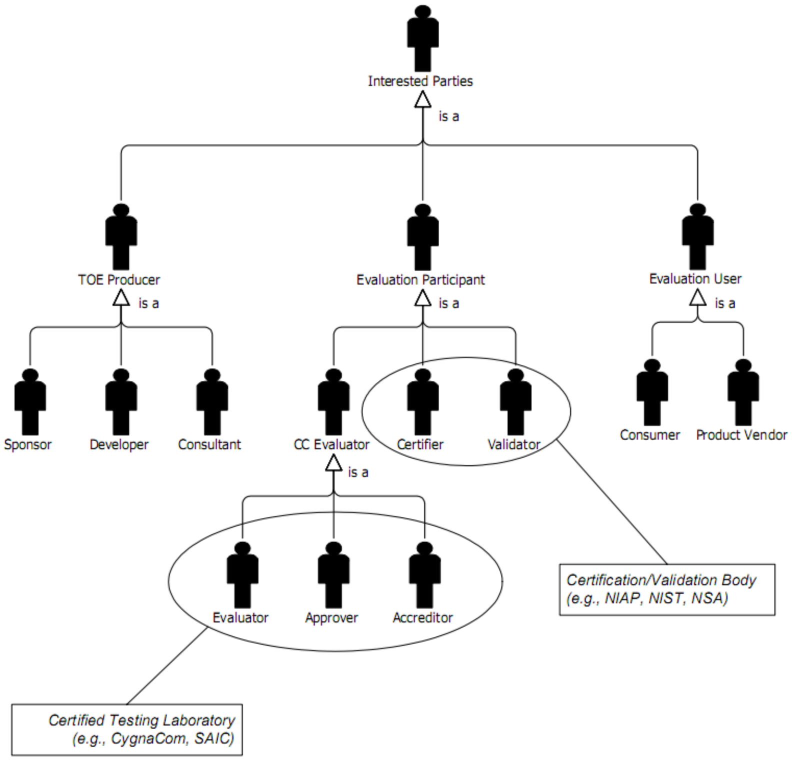


Fig. 54.9. Interested parties ontology [32]

Table 54.9. Results of object-oriented and process oriented ontological modeling

Property	Requirement	Action	Evidence	Evaluation parties	Evaluation method
$P_i^{\{j\}}$	$R_i^{\{j\}}$	$A_m^{\{n\}}$	$\{E\}^{P_i^{\{j\}}}$	$\{B\}^{P_i^{\{j\}}}$	$U(P_i^{\{j\}})$

54.4. Recommendations for the use functional modeling to describe the IT security assurance evaluation processes

Documents describing the IT security assurance evaluation processes [28, 30], are descriptive in nature and presented in verbal form. Evaluation process in the form of formal models contributes to solving of many tasks. Typically, a functional model is the basic and most used in practice. Generally accepted methodology for functional modeling is a methodology IDEF0 notation [43].

Adequacy and efficiency of IDEF0 notation, when it is used for solving assurance evaluation process modeling tasks, provided by the following concepts of IDEF0 language:

1) A model – artificial object, consisting of the system and its components. Assurance evaluation process model – set of interrelated and interacting actions (steps) on the assurance evaluation. The model describes: actions, performed during evaluation, how the evaluation management is going on, how do actions transform inputs into outputs, what evidences are used to perform an action and what results of the actions are expected.

2) Block modeling and its graphical representation. In the IDEF0 notation everything that happens in the system are called functions. In the assurance modeling every function describes a single evaluation action. Each action (function) is associated with a block. Blocks interact and/or blocks and ambient (relative to the system) interact. On the IDEF0-diagram interfaces of blocks interaction (the interaction type "block-block" or "block-ambient") are indicated by the arrows, which are material or information flows.

3) Conciseness and accuracy. Graphic language allows concisely, clearly and unambiguously show all the actions performed during the assurance evaluation, the relations and connections between them, to identify erroneous, unnecessary or redundant connections, etc.

4) The rigor and formalism. Development of IDEF0 models carried out in strict accordance with the rules of development and diagram creation (notation), described in [42]. This eliminates ambiguity when reading diagrams.

5) Iterative modeling. Model development in IDEF0 notation is a turn-based, iterative procedure. In each iteration step the developer offers a version of the model, which is subjected to discussion, peer review and editing, and then the cycle repeats.

Input data for functional modeling are results of object-oriented and process oriented ontological modeling of assurance requirements liable for evaluation. It’s reasonable to present these data as a Table 54.9. General form of functional box for assurance activity modeling is shown in Figure 54.10:

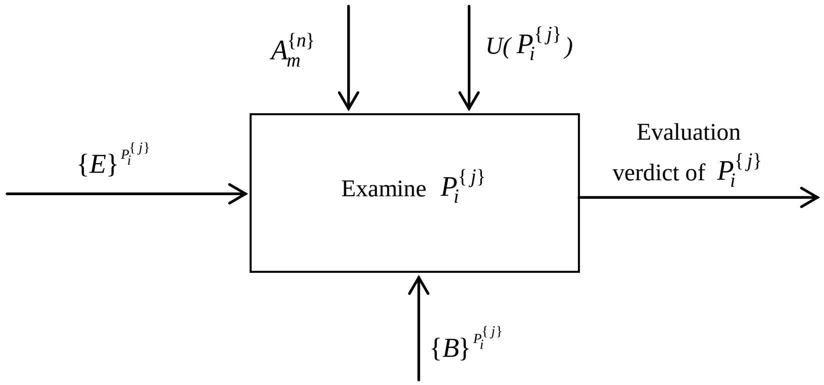


Fig. 54.10. General form of functional box

Where $P_i^{\{j\}}$ is a property liable for evaluation; $\{E\}^{P_i^{\{j\}}}$ is an evidence set necessary for property $P_i^{\{j\}}$ evaluation; $A_m^{\{n\}}$ is an action of property $P_i^{\{j\}}$ evaluation; $\{B\}^{P_i^{\{j\}}}$ is a set of interested parties participating in property $P_i^{\{j\}}$ evaluation; $U(P_i^{\{j\}})$ is a property $P_i^{\{j\}}$ evaluation method.

Thus, functional modeling allows to presenting the assurance evaluation process in the standardized and formalized form, and the results of such modeling correspond to the basic principles of models development, which include: correspondence principle of accuracy and model complexity; accuracy balance principle; principle of a sufficient variety of model elements; principle of model clarity for its developer and user; principle of block model; principle of model specialization; principle of dynamic modeling; principle of system modeling [56].

54.5. Recommendations for the use mathematical apparatus of linguistic variables to describe the IT security assurance evaluation

The use of assurance evaluation formal methods and the development of decision support tools require a formal presentation of quantitative and qualitative characteristics of the examination object. Usually, there are no problems with quantitative characteristics, and they can be expressed numerically. However, the vast majority of assurance properties can’t be quantified (for example: « Sufficiency of administrator's guide », «Completeness of the description of the identifying configuration elements method » etc.), and it can only be described qualitatively, i.e. by verbal description. For the formalization of qualitative characteristics of the examination object it is advisable to use the mathematical apparatus of linguistic variables [46,47], which is based on fuzzy sets theory [47], as values of which are allowed not only the numbers, but also the words and sentences in a natural or artificial language.

The LV means a variable, which value are fuzzy subsets, that are expressed in the form of words or sentences in a natural or artificial language . The basis of the LV concept is the term "fuzzy variable," which means a fuzzy set with some name. Let fuzzy variable [60, 61] determined as a tuple $<A, X, \tilde{A}>$, where A – fuzzy variable identifier, $X = \{x\}$ – variable range, a $\tilde{A} = \bigcup_{x \in X} \mu_x / x$ – fuzzy set on X, that puts limits on the sets of fuzzy variable values $\tilde{A}$.

In this case the LV [49,51] is defined as a tuple $<\beta, T(\beta), X, Q, W>$, where β – LV's name; $T(\beta)$ – LV's term-set (set of values), elements of which are names of fuzzy variables ($T(\beta)$ set is often called as the set of basic term of the β LV set); X – set, which is the fuzzy variable range (X set is called the universal set); Q – syntactic rule that describes the formation process of new LV values from $T(\beta)$ set (syntactic rules appointment - is the union of the primary terms, for example *full description*, with integral values, such as, *not full enough description, almost full description*); W – semantic procedure that allows each new value, formed by the Q procedure, to be presented as the fuzzy variable.

There are no special requirements to VL names and terms. To the LV term extends the ordering requirement: $T_1 < T_2 < \ldots < T_n$. Conditions to be met by the membership function of a fuzzy set of X universal set are defined in [51].

For the LV formation for the assurance properties it is necessary to adhere to the next stages :

1. Determination of the LV terms number and its ordering. Application of LV allows expanding the binary criteria system, if necessary. So, if the improved accuracy of property evaluation is important for the property, then the LV values set can be extended by intermediate values. Increasing the number of intermediate values increases the accuracy of the evaluation properties and takes into account the expert uncertainty in choosing alternatives, but process of adoption of integrated solutions is complicated (verdicts on complex properties), and increases the examination time, and, consequently, its value. Thus, a compromise between evaluation accuracy and cost-time factor should be taken by the expertise customer.

2. Determination of LV limit values. Limit values of LV term-sets of assurance properties should reflect the minimum and maximum compliance with the assurance requirements. For example, if you want to evaluate the functional examination object specification, then the LV limit values of «The description fullness of the functional specifications» should be – «no description» и «full description».

3. Determination of forming method of membership functions and carrying out the expert survey. To construct the membership functions of LV fuzzy sets of assurance properties it is convenient to use parametric methods, one of which is

the parameter assignment method [51], that allows to forming the trapezoidal and triangular membership functions. It uses the next expert information: name of LV; determination range [a, c] (universal set) of LV; the number of linguistic terms; name of each linguistic term (variable).

4. Construction of membership functions for the LV terms. Determination of membership functions of fuzzy sets of linguistic variables with non-numeric character is difficult, and it can be solved by expertise and the introduction of conventional scales. The main methods of constructing membership functions of fuzzy sets are: survey method; numerical method; quantitative paired comparison method; comparison method with the square root definition; parameter assignment method; method of forward and reverse evaluation; method of interval evaluations, and others.

Introduction of LV is accompanied by linguistic uncertainty, which is associated with using of natural language for describing the problem of decision-making. This uncertainty is related to the need to operate with a finite number of words and a limited number of phrases structure (sentences, paragraphs, texts) to describe the infinite variety of different situations, arising in the decision-making process. On the one hand, linguistic uncertainty generated by the multiplicity of language word meanings (concepts and relations) and on the other hand, by the ambiguity of the phrase meaning . There is the value selection criteria for each linguistic variable value. Criteria for alternatives selection can be developed by experts or groups of experts, and presented as verbal recommendations or standard examples of the system behavior(examination object).

54.6. Recommendations for the use fuzzy inference for assurance evaluation

According to [28] the final verdict on the assurance evaluation can be either positive or negative. Therefore, there is the problem of verdicts aggregation for individual assurance evaluation properties for the adoption of a generalized solution. When using LV, mathematical apparatus is the fuzzy inference conclusion. In this context, the fuzzy inference conclusion system is a special case of the productive fuzzy systems or systems of fuzzy production rules, where conditions of individual rules are formulated in the form of fuzzy statements about the LV certain values.

The rule of the fuzzy production is the following expression:

$$(i) : Q; P; \tilde{A} \Rightarrow \tilde{B}; S, F, N \tag{54.10}$$

where (i) – fuzzy product name; Q – the scope of application of fuzzy product; P – core applicability condition of fuzzy product; $\tilde{A} \Rightarrow \tilde{B}$ – core of fuzzy products, where $\tilde{A}$ – core condition (or antecedent); $\tilde{B}$ – core conclusion(or consequent); «$\Longrightarrow$» - sign of logical sequence ; S – a method of determination quantify value of truth degree of core conclusion; F – the certainty or confidence coefficient of fuzzy products; N – product postconditions.

The core $\tilde{A} \Rightarrow \tilde{B}$ is a central component of the fuzzy product. The core product is written in a form: «IF $\tilde{A}$, THEN $\tilde{B}$». The sequence is interpreted as a logical consequence of $\tilde{B}$ conclusion from $\tilde{A}$ condition**.** Fuzzy linguistic expressions of the form 1-3 are used as an expressions $\tilde{A}$ and $\tilde{B}$.

The simplest and most commonly used version of fuzzy products rules can be written in the form of:

RULE <#>: IF «β_1 is α», THEN «β_2 is £», (54.11)

where «β_1 is α» is a condition of the fuzzy products rule, and fuzzy statement «β_2 is £» – is a fuzzy conclusion of the rule.

More complex rules of fuzzy products may contain composite fuzzy statements. For example:

IF (PredostavRA is Predostav)
  AND (PolnotaOpFBiIntA is OpisPoln)
  AND (PolnOpisBezobSpAdm is OpisPoln)
  AND (PolnOpisPreduprFunkPriveleg is OpisPoln)
  AND (PolnOpisSobBezopDejsAdm is OpisPoln)
  AND (SoglRAsDok is Soglas)
  AND (PolnOpisTrBezopITkAdm is OpisPoln)
THEN (DostRA is Dost).

The rule base of fuzzy products is a finite set of fuzzy products rules. Most commonly the rules base is presented in the form of structured text:

RULE_1: IF «Condition_1» THEN «Conclusion_1» (F_1)
RULE _2: IF «Condition _2» THEN «Conclusion _2» (F_2)
…
RULE _n: IF «Condition _ n» THEN « Conclusion _ n» (F_n)

where F_i, $i = \overline{1,n}$ – weighting factors of rules.

The using of a recording is determined by need to follow some standard notations. For example, when modeling fuzzy inference operations using Matlab, fuzzy products base is presented in the form (2.15). Consistency of rules regarding to linguistic variables means that as conditions and conclusions of rules can be used only fuzzy linguistic expressions of the form (2.14) and the membership functions of term-set values should be defined in each of the fuzzy statements.

The main stages of fuzzy inference are [50]:

– formation of the fuzzy products rules base;
– fuzzification of input variables;
– aggregation of subconditions in fuzzy products rules;
– activation of subconclusions in fuzzy products rules;
– accumulation of conclusions of fuzzy products rules;
– defuzzification of output value.

Thus, the development of the fuzzy products rule base allows the calculation of linguistic variables values of complex properties and can be used in instrumented systems of support decision-making in the assurance evaluation process.

54.7. Recommendations for development of templates verdicts for assurance properties

The final stage of expert work is provision by him the technical evaluation report (TER). The TER structure is shown in Figure 54.11.

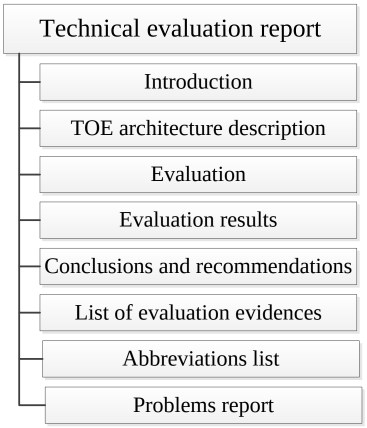

Technical evaluation report
Introduction
TOE architecture description
Evaluation
Evaluation results
Conclusions and recommendations
List of evaluation evidences
Abbreviations list
Problems report

Fig. 54.11. TER structure

The minimum required content of the information included in each TER section is determined in [38]. During the studies most attention was paid to the "Evaluation results", which should include a description of work performed on the evaluation, methods and procedures for obtaining results and expert verdicts for each action (step) of evaluation. Development of template verdicts base was proposed in the functional-linguistic approach [45]. The verdicts base must contain the verdict template for each value of the linguistic variable, i.e. to each linguistic variable β puts a set of templates $S_\beta = \{s_1, s_2, ..., s_q\}$, where s – number of linguistic variable values β. Choice of verdict option of such a base is carried out in accordance with the value of linguistic variable, which it adopted on evaluation results. Each verdict must contain the evaluation results of a single assurance property and it is necessary that the opportunity of verdict arrangement for the formation of a structured and consistent report was provided.

For example, there is a complex property P_0 , for which the term-set and dependencies within the graph of properties G^P are defined (Table 54.10).

Table 54.10. Property characteristics P_0

$P_i^{\{j\}}$	Property	Term-set	Dependencies
P_0	TOE and TOE label availability	TOE and TOE labels are available There are no TOE and TOE labels	*and ((depends-on-existentially* P_0 P_1^0*)* *(depends-on-existentially* P_0 P_2^0*))*
P_1^0	TOE availability	TOE is provided TOE is not provided	
P_2^0	TOE label availability	TOE label is available There is no TOE label	

Table 54.10 shows that P_0 property depends on the properties P_1^0 and P_2^0. I.e. the value that takes linguistic variable of P_0 property will depend on what values of linguistic variables of P_1^0 and P_2^0 properties will take. For this purpose, functional-linguistic approach provides for the use of fuzzy logic output. Thus, the base of production rules for fuzzy logic output for the property P_0 will be look as:

1) IF «*TOE is provided*»
    AND «*TOE labels are available*»
   THEN «*TOE and TOE labels are available*»;
2) IF «*TOE is provided*»
    AND «*there are no TOE labels*»
   THEN «*There are no TOE and TOE labels*»;
3) IF «*TOE is not provided*» THEN «*There are no TOE and TOE labels*».

At the same time verdict of property evaluation P_0 will consist of the verdicts arrangement of properties evaluation P_1^0 and P_2^0. Examples of these verdicts are shown in Table 54.11.

Table 54.11. Options of verdicts of properties P_1^0 and P_2^0 evaluation

Property	Term-set	Verdict template
TOE availability	TOE is provided	Checked that the *TOE is provided* for examination by developer
	TOE is not provided	Checked that the *TOE is not provided* for examination by developer
TOE label availability	TOE label is available	TOE that was provided *is marked by the TOE label*
	TOE label is not available	But TOE that was provided *is not marked by the TOE label*

With these source data the options of verdicts of properties evaluation «*TOE and TOE label availability*» will be look as:

1) Condition: IF «*TOE is provided*» AND «*TOE labels are available*» THEN «*TOE and TOE labels are available*» («*TOE and TOE labels are available*» = {«*TOE is provided*», «*TOE label is available*»})

Verdict: Checked that the *TOE is provided* for examination by developer. TOE that was provided *is marked by the TOE label*.

2) Condition: IF «*TOE is provided*» AND «*TOE labels are not available*» THEN «*TOE and TOE labels are not available*» («*TOE and TOE labels are not available* » = {«*TOE is provided*», « *TOE labels are not available* »}).

Verdict: Checked that the *TOE is provided* for examination by developer. But TOE that was provided *is not marked by the TOE label*.

3) Condition: IF «*TOE is not provided*» THEN «*TOE and TOE labels are not available*» («*TOE and TOE labels are not available* » = {« *TOE is not provided* »}).

Verdict: Checked that the *TOE is not provided* for examination by developer.

Thus, the use of template verdicts contributes to ensure compliance with the requirements of the assurance evaluation results (repeatability, reproducibility and comparability).

54.8. Recommendations for workflow modeling of assurance evaluation process

As a modeling notation of workflow modeling of assurance evaluation process is recommended to use the IDEF3 notation. IDEF3 standard – is a methodology to describe the processes that consider the sequencing of implementation and cause-and-effect relations between situations and events for the structural presentation of knowledge about the system . Modeling in IDEF3 standard is performed by using the graphical presentation of the process, material and information flows in this process, and relations between operations and objects in the process. Using IDEF3 notation, the logic of work, the sequence of their launch and completion can be described. One of the application objects of this methodology can be a process of assurance level evaluation.

IDEF notation allows describing uniquely and accurately the procedure of expert actions during examination. Standard tools allows displaying a variety of possible expert behaviors.

The main components of IDEF3 language are functional elements, elements of connections and junctions .

Each functional element displays a single action (evaluation step) of assurance level evaluation. Connection element is required for organization of relations between the diagram elements (evaluation actions) and the description of the evaluation processes dynamics. The main connection type in the modeling is the seniority connection, which expresses a temporal seniority relations between the diagram elements, and at the same time the first element (step) should be completed before it starts to run the next (next action).

Junctions are using to display the relations logic between the events set and time synchronization of IDEF3 elements activation. There are junctions for merging and branching arrows. The IDEF3 methodology uses junctions of five logical types for modeling possible sequences of actions in the process of assurance evaluation examination (Table 54.12).

Table 54.12. IDEF3 logical types

Designation	Name	The meaning in the case of arrows merging	The meaning in the case of arrows branching
&	Asynchronous AND	All previous evaluation actions should be completed	All next evaluation actions should be run
&	Synchronous AND	All previous evaluation actions are completed at the same time	All next evaluation actions are run at the same time
O	Asynchronous OR	One or more previous evaluation actions should be completed	One or more next evaluation actions should be run

O	Synchronous OR	One or more previous evaluation actions are completed at the same time	One or more next evaluation actions are run at the same time
X	XOR (Exclusive OR)	Only one previous evaluation action should be completed	Only one next evaluation action should be run

IDEF3 methodology makes it possible to present the evaluation process as a hierarchically organized set of diagrams. Every evaluation action (functional diagram element) can be decomposed, i.e. it can be divided into structural parts. Decomposed diagram describes the process in more detail. Applying the decomposition principle repeatedly, it is possible to structure the description of the evaluation process to any level of detail. Example of IDEF3 diagram is shown in Figure 54.12.

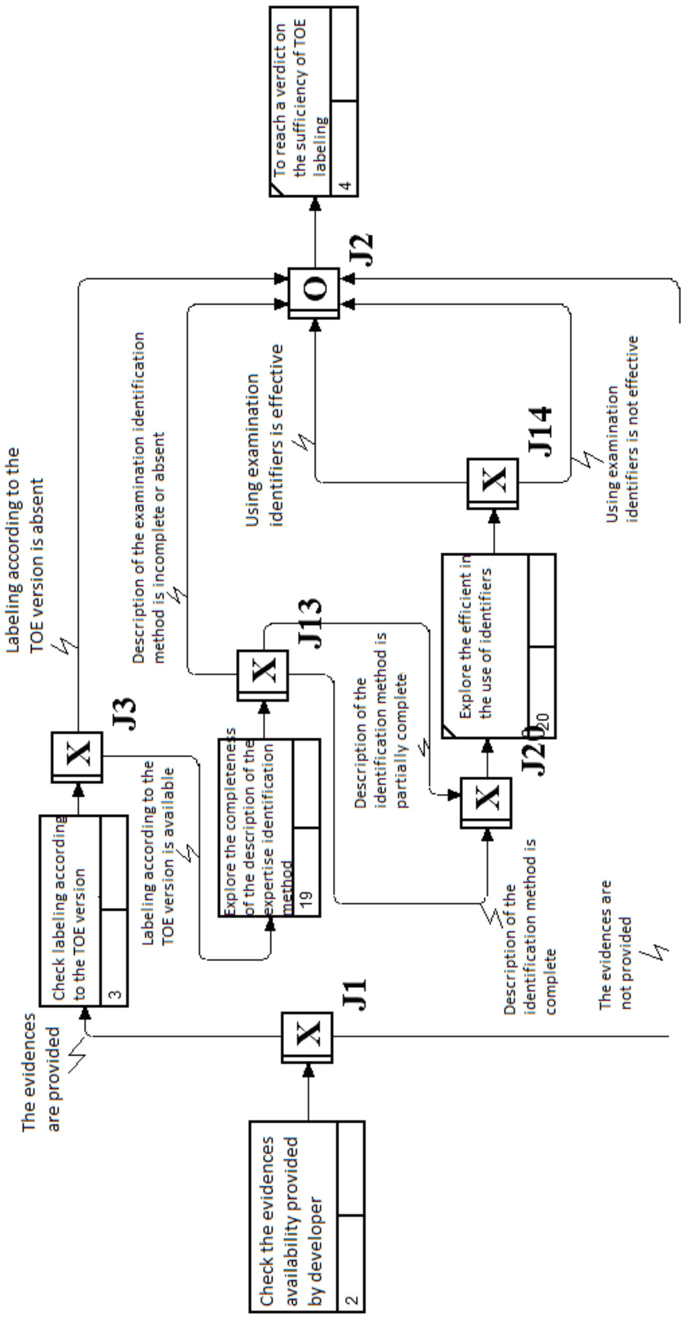

Check the evidences availability provided by developer
2
J1
The evidences are provided
The evidences are not provided
Check labeling according to the TOE version
3
J3
Labeling according to the TOE version is absent
Labeling according to the TOE version is available
Explore the completeness of the description of the expertise identification method
19
J13
Description of the examination identification method is incomplete or absent
Description of the identification method is partially complete
Description of the identification method is complete
J20
Explore the efficient in the use of identifiers
20
J14
Using examination identifiers is effective
Using examination identifiers is not effective
O
J2
To reach a verdict on the sufficiency of TOE labeling
4

Fig. 54.12. Example of IDEF3 diagram

Thus, the modeling of assurance evaluation level in IDEF3 notation makes it possible to develop unambiguous course of actions for the expert, and developed diagrams allow for cost-temporal evaluation of examination.

54.9. Conclusions according to the section

1. The method of assurance evaluation of IT security, the basic concept of which is based on the functional-linguistic approach was developed. Within the method were developed method for constructing object-oriented ontological model and method for constructing a process-oriented ontological model of assurance evaluation, and also were used the methodology of IDEF0 functional modeling, method of IDEF3 workflow modeling and mathematical apparatus of linguistic variables and fuzzy inference.
2. The task of activities formalizing of assurance evaluation was solved by processes modeling in the IDEF0 functional modeling notation and by diagramming workflows in IDEF3 notation. The task of developing criteria for decision-making, with taking into account the non-numeric character of evaluation properties, was solved by the use of mathematical apparatus of linguistic variables and fuzzy inference procedures, acting on the pre-arranged rule base of fuzzy products.

54.10. Questions and tasks for self-control

1. Give conception of functional-linguistic approach to the security assurance evaluation.
2. Give characteristic of method of the Object-Oriented Assurance Ontological Modeling.
3. Explain Graph of assurance requirements.
4. Explain Graph of assurance properties.
5. Explain graph of evidences.
6. Give characteristic of Method of the Process-Oriented Assurance Ontological Modeling.
7. Explain graph of analysis of assurance evaluation actions.
8. How to use fuzzy inference for assurance evaluation?
9. How to use mathematical apparatus of linguistic variables to describe the IT security assurance evaluation?
10. How to use functional modeling to describe the IT security assurance evaluation processes?
11. How to development of templates verdicts for assurance properties?
12. How to use workflow modeling of assurance evaluation process?
13. Explain the example of IDEF3 diagram.

References

1. ISO/IEC 15408-1. Common Criteria for Information Technology Security Evaluation. Part 1: Introduction and general model. https://www.commoncriteriaportal.org/files/ccfiles/CCPART1V3.1R4.pdf
2. ISO/IEC 15408-2. Common Criteria for Information Technology Security Evaluation. Part 2: Security functional components. https://www.commoncriteriaportal.org/files/ccfiles/CCPART2V3.1R4.pdf
3. ISO/IEC 15408-2. Common Criteria for Information Technology Security Evaluation. Part 3: Security assurance components. https://www.commoncriteriaportal.org/files/ccfiles/CCPART3V3.1R4.pdf
4. The Common Criteria ISO/IEC 15408 - The Insight, Some Thoughts, Questions and Issues. SANS Institute - http://www.sans.org/reading-room/whitepapers/standards/common-criteria-iso-iec-15408-insight-thoughts-questions-issues-545.
5. ISO/IEC TR 15446:2009 Information technology – Security techniques- Guide for the production of Protection Profiles and Security Targets.
6. ISO/IEC 18045 Common Methodology for Information Technology Security Evaluation. Evaluation methodology. https://www.commoncriteriaportal.org/files/ccfiles/CEMV3.1R4.pdf
7. A. Potij. An Ontological Modeling of Assurance Evaluation Process in Context of Functional-Linguistic Approach // A. Potij, D. Komin, I. Rebriy // Information & Security. An International Journal, Vol. 28, No. 1 – 2012. – pp. 108-120.
8. ISO/IEC 29128:2011 Information technology – Security techniques – Verification of cryptographic protocols.
9. ISO/IEC 19792:2009 Information technology – Security techniques – Security evaluation of biometrics.
10. ISO/IEC 19791:2009 Information technology – Security techniques - Information technology – Security techniques – Security assessment of operational systems/
11. ISO/IEC 19790:2012(en) Information technology — Security techniques — Security requirements for cryptographic modules.
12. ISO/IEC 24759:2014 Information technology – Security techniques – Test requirements for cryptographic modules.
13. ISO/IEC 29128:2011 Information technology -- Security techniques -- Verification of cryptographic protocols.
14. ISO/IEC 19792 Information technology -- Security techniques -- Security evaluation of biometrics.
15. Belén Fernández Saavedra Evaluation Methodologies for Security Testing of Biometric Systems beyond Technological Evaluation. Leganés, Marzo 2013. http://e-

archivo.uc3m.es/bitstream/handle/10016/17145/tesis_belen_fernandez_saavedra_2013.pdf?sequence=1

16. Piotr Cofta. Identity assurance: when four levels are not four levels. www.eurim.org.uk/

17. Identity Assurance Framework: Assurance Levels. Kantara Initiative Identity Assurance Work Group. 2009. https://kantarainitiative.org/

18. Special Publication 800-63-2. Electronic Authentication Guideline. http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-63-2.pdf

19. Charmaine Lowe. Identity Assurance Standard. 2010. - www.gov.bc.ca/

20. Commission Implementing Regulation (EU) 2015/1502 on setting out minimum technical specifications and procedures for assurance levels for electronic identification means // Official Journal of the European Union - L 235/- 09.09.2015 – pp. 8-20.

21. ISO/IEC 29115 – Information technology – Security Techniques – entity authentication assurance framework.

22. NIST Special Publication 800-63-2. - Electronic Authentication Guideline – http://dx.doi.org/10.6028/NIST.SP.800-63-2.

23. Identity Assurance Programme - National Audit Office, 2014 - www.nao.org.uk.

24. Pan-Canadian Assurance Model - Identity Management Steering Committee, Assurance, Identity and Trust Working Group, 2010.

25. Potii A.V. Formal model of information security process // Radioelectronic and computer systems. Scientific and technical journal.– 2006. – № 5(17). – pp. 128-133.

26. Archibald R. Management of high-tech programs and projects. – M.: Company IT; DMK Press, 2004. – 472 p.

27. Methodical guidelines for assessing the level of guarantees of the correctness of the implementation of functional security services in means of protection of information from unauthorized access: ND TZI 2.7-010-09. – K.: 2009. – 131 p.

28. ISO/IEC 18045:2005, Informational technology – Security techniques – Methodology for IT security evaluation.

29. Potii A.V., Komin D.S. Evaluation of information security assurance on the basis of a functional linguistic approach // Applied Radioelectronic. – 2010. – Vol 9 (№3). – pp. 421-435.

30. Criteria for assessing the security of information in computer systems from unauthorized access: ND TZI 2.5-004-99. – 1999. – 53 p.

31. Potii A.V., Komin D.S. Assessment of security assurances based on the use of linguistic variables // Information processing systems. Information and economic security. – 2010. – № 3(84). – pp. 34-37.

32. Prieto-Diaz, R. The Common Criteria Evaluation Process. Process Explanation, Shortcomings, and Research Opportunities. - Commonwealth Information Security Center Technical Report CISC-TR-2002-03, December 2002 – CISC, James Madison University, USA. 62 p.

37. Potij A. Method of Assurance Requirements Evaluation / A. Potij, D. Komin, I. Rebriy // Critical Infrastructure Safety and Security (CrISS-DESSERT'11). First International Workshop. Kirovograd, Ukraine, May 11-13, 2011. – Proceedings. Volume 1. – Kharkiv, National Aerospace University named after N.E.Zhukovsky "KhAI": 2011. – pp. 123-131.

40. Potii A.V., Komin D.S. System-ontological analysis of the subject area of evaluation of information security assurances // Radio electronic and computer systems. Scientific and Technical Journal.– 2010. – № 5(46). – pp. 50-56.

41. D. Aleksandrov, A. Kostrov. Methods and models of information management. – M.: Finance and Statistic, 2007 – 336 p.

42. Potii A.V., Komin D.S. IDEF models for assessing the level of information security assurances // Bulletin of Kharkiv National University. Series "Mathematical Modeling. Information technologies. Automated control systems ".– 2010. – №925 (vol.14) – pp. 98-105.

43. IEEE IDEF0. Standard Users Manual for the ICAM Function Modeling Method - IDEF0. IEEE draft standard P1320.1.1. 1997.

44. Potii A.V., Komin D.S. Ontological modeling of the process of evaluating guarantees in the context of a functional-linguistic approach // Radio electronics, informatics, control . – 2011. – №1(24). – pp. 64-73.

45. Potii A.V., Komin D.S. Application of a functional and linguistic approach to assess information security assurances // Special telecommunication systems and information protection. – 2010. – № 1(17). – pp. 24-31.

46. Zade L. The notion of a linguistic variable and its application to the adoption of approximate solutions – M.: Mir, 1976. – 165 p.

47. Kofman A. Introduction to the theory of fuzzy sets. – M.: Radio and Communication, 1982. – 432 p.

48. Korchenko A. G. Construction of information security systems on fuzzy sets. Theory and practical solutions – К.: «МК-Pess», 2006. – 320 p.

49. Borisov A. Processing of fuzzy information in decision-making systems – M.: Radio and Communication, 1989. – 304 p.

50. Zaychenko Y. Investigation of operations: Fuzzy optimization – К.: High education, 1991. – 191 p.

51. Borisov A. Decision-making models based on a linguistic variable – Riga: 1992. – 64 p.

O. Gordieiev, V. Kharchenko, V. Sklyar, A. Perepelitsyn, V.Kulanov,
M. Poluyanenko, O. Odarushchenko, O. Yasko, E. Babeshko, V. Kulanov,
K. Leontiev, I. Zelinko, O.Gordieiev, Ya. Chujkov , O. Illiashenko, V. Duzhyi,
O.Rusin, E. Kovalenko, D. Uzun, A. Tetskyi, A. Strielkina , O. Yevsieieva,
M. Kolisnyk, I. Piskacheva, S. Yaremchuk, O. Ivanchenko, O. Potii

SECURE AND RESILIENT COMPUTING FOR INDUSTRY AND HUMAN DOMAINS.

Techniques, tools and assurance cases for security and resilient computing

Multi-book, Volume 3

Editor Vyacheslav Kharchenko

National Aerospace University n. a. N. E. Zhukovsky
“Kharkiv Aviation Institute”
17 Chkalova street, Kharkiv, 61070, Ukraine
http://www.khai.edu