Andrzej Białas

Institute of Innovative Technologies EMAG

COMMON CRITERIA COMPLIANT IT PRODUCT DEVELOPMENT IN THE EXPERIMENTAL SECLAB LABORATORY

Introduction

The paper presents how to organize the development process of IT products (hardware, software, firmware, systems) in an experimental laboratory of the EMAG Institute – SecLab. The laboratory was established with the use of products which resulted from the CCMODE project (Common Criteria compliant Modular Open IT security Development Environment) [CCMODE].

The development process of IT products is in compliance with the ISO/IEC 15408 Common Criteria (CC) methodology [CC1-3, CCPortal]. It makes use of patterns, knowledge and tools which are the results of the CCMODE project. The paper refers to the publication [BiaFli14a] about SecLab, i.e. an experimental development environment for IT products which are to be used in high-risk applications. In the work [BiaFli14b] three basic processes of the CC methodology were presented:

- IT security development process whose objective is to prepare the Security Target (ST) document for the IT product; in the CC nomenclature the IT product is called Target of Evaluation (TOE), as it is submitted for CC-based evaluation; ST contains a black-box description of TOE security functions and the declared Evaluation Assurance Level (EAL), which decides how these functions will be implemented in the IT product during the TOE development process;
- TOE development process refers to typical activities of the developers, with special focus on the implementation of security functions in the IT product; this process is related to the elaboration of evidences which are submitted for evaluation together with the TOE;

- IT security evaluation process carried out in an independent laboratory, according to the evaluation scheme adopted by a country where the CC standard was implemented.

SecLab is a place where the two former processes can be conducted. The IT security development process was presented quite thoroughly in [BiaFli14b], while the second process will be discussed in this paper. It is important to note here that the CCMODE project resulted in the development of tools to support the third process too. However, in SecLab these tools can be treated only as self-evaluation tools because a laboratory which carries out the project cannot evaluate the results of its own works.

Before reading this paper the readers who are not familiar with the Common Criteria issues are recommended to have a look at some information sources, e.g.: [CCMODE, CC1-3, Bia11, Bia12, Hig10, Her03].

The paper presents how the TOE development process is carried out with the use of patterns and tools developed within the CCMODE project (section Implementation of TOE development process in SecLab). Additionally, the author discusses the issue of security with respect to processes conducted in SecLab and the protection of assets related to these processes (section Processes related to project security in SecLab). Finally, some conclusions are drawn.

Implementation of TOE development process in SecLab

At first glance, the Common Criteria methodology seems to be complex and full of specific terms, but in fact, it simply puts in order best practices, i.e. operations that the developers should perform while making new IT products.

CC is commonly recognized as difficult for developers, particularly those interested in very specific areas of IT and lacking a more general look at IT security. The common approach to security adopted by many IT areas (from hardware and firmware to complex software systems), along with security terms, operational methods, tools, and organizational solutions might seem unfamiliar, useless and difficult to some developers. Yet, they have to overcome these obstacles if they plan to develop an IT product which is to be evaluated and certified. In this respect they often need to use experts' services. And this is one of the major barriers against wider dissemination of the CC methodology. The products of the CCMODE project were developed with a view to facilitate the developers' work and diminish these barriers. The following were developed within the project [Bia11, Bia12]:

- evaluation evidence patterns which enable the developers to focus on the content of the IT product evidence; the developers do not have to preoccupy with the evidence structure and analyze the evidence compliance with the CC requirements;
- computer-aided supporting tool (CCMODE Tools); thanks to the tool the most difficult and labourious tasks can be delegated to the software [Rog14];
- knowledge base which enables the developers to obtain necessary knowledge any time and from the same source.

CCMODE products were developed to support all three main processes of Common Criteria. However, in this paper the author focused on the TOE development process. This process is strongly dependent on the EAL declared for the given TOE.

The preparation of evaluation evidences has an incremental and iterative character and comprises all components of the EAL package, along with extra components (i.e. augmentation) or those exchanged into higher-rigour ones (i.e. substitution). It often happens that after the evidence for the given SAR (Security Assurance Requirement [CC1-3]/part 2) component is prepared, other evidences are worked out, yet then it is necessary to return to the previous version of the evidence document and supplement it with some new data. The standard does not enforce the succession of the considered SAR components (TOE development subprocesses fulfillment) – the decision is often at the discretion of experts who support the work of developers.

There is a preferred succession of operations for SecLab in Figure 1.

This path was determined heuristically, according to the assumption that the developer goes to the subprocess for which there are most input data collected. This way it is possible to advance the work on this subprocess and on the whole project. Each subprocess is based on a suitable pattern of evaluation evidences (section 4 in [BiaFli14a]). The applied patterns are given with the names of subprocesses.

In SecLab it was assumed that the first subprocesses to run are those which are used for building a development environment (sections: *Defining the TOE life cycle* to *Establishing flaw remediation channel*). Then subprocesses used for the development of the IT product (sections: *TOE decomposition and interfaces* to *Justifying coverage of subsystems and their modules by tests*) are performed. Figure 1 shows the succession of works. In reality, the process is carried out incrementally and it is possible to return to any subprocess.

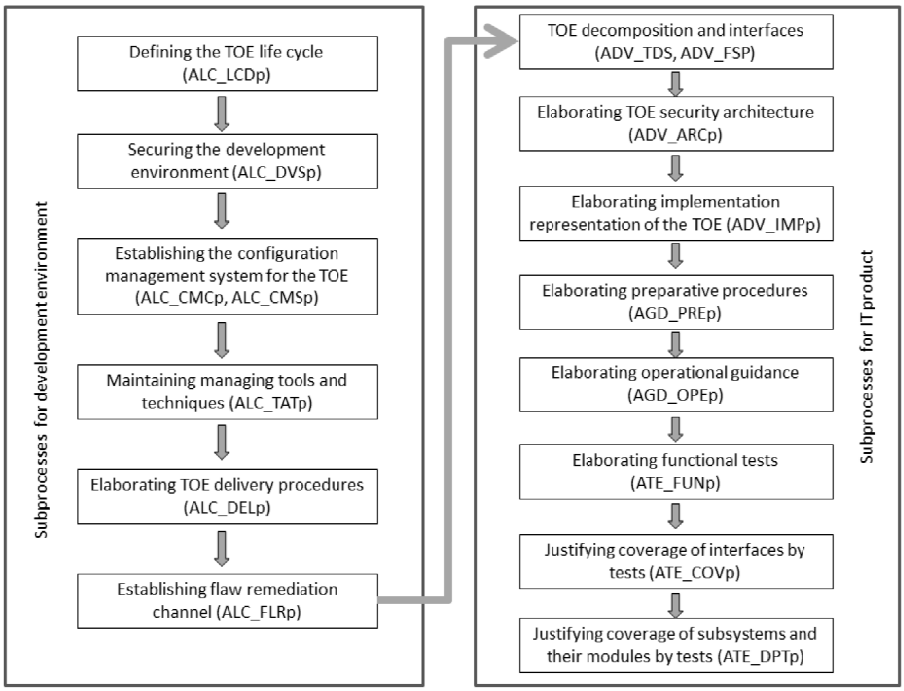

Fig. 1. TOE development process

For each subprocess there is an evaluation evidence pattern [BiaFli14a, Bia11]. The CCMODE Tools software [Rog14, BiaFli14a, Bia11] is extensively used here. The whole project is managed by means of the EMT (Environment Management Tool) application, while versioning is done by an external system – SVN (Subversion). Basic operations of the project can be carried out with the use of a popular tool Enterprise Architect® (EA) made by Sparx, or other specialized external tools. There was a special plugin developed for Enterprise Architect (EA-plugin). It makes a security model for the Security Target and is used for modelling TOE interfaces and for decomposing the TOE into subsystems and modules (Figure 2). The Redmine external subsystem is useful for bug tracing, while Testlink for tests management. The basic tool for semi-automatic generation of evaluation evidences is GenDoc, an application supported by the knowledge base.

The TOE development process will be demonstrated with the most commonly used EAL4, with one extra optional component (ALC_FLR.1 Flaw remediation), i.e. EAL4+. The developer is responsible for the preparation of evidences for all 14 components of the EAL4+ package and, obviously, for the previously elaborated Security Target which is superior to the other evidences. The Security Target is the result of the IT security development process presented in [BiaFli14b].

The section ends with the process of independent testing ATE_IND and vulnerability assessment AVA_VAN. The responsibility for their completion and preparation of evidences lies with evaluators from an independent laboratory.

For higher EALs there are more evidences and they are given with more details.

In the below sections all 14 subprocesses of the TOE development (EAL+ example) are presented. Each subprocess is based on its evidence pattern whose content the subprocesses refer to. First, the subprocesses which specify the IT product development environment are carried out, or at least are advanced enough, then the subprocesses which specify the IT product itself (see Figure 1).

Defining the TOE life cycle (ALC_LCDp)

The life-cycle model definition presents a high-level description of the TOE life-cycle and provides a framework for the entire development environment.

A life-cycle model for the given IT product is adopted based on the analysis of the project specifics and available life-cycle models. A typical model consists of four phases:

- Phase 1 – development,
- Phase 2 – production,
- Phase 3 – exploitation,
- Phase 4 – end of life.

It is necessary to determine which phases will be carried out in the development environment of the product. The subprocess will result in the life cycle model specification, comprising the following elements:

- description of successive phases,
- description of operations (processes) carried out within the phases,
- input and output data of the phases,
- relations to other phases,
- applied procedures, tools and techniques,
- roles and responsibilities,
- persons responsible for conducted operations,
- third party involvement.

Securing the development environment (ALC_DVSp)

The objective of this subprocess is to determine physical, procedural, personnel-related, and other security measures that may be used in the development environment to protect the TOE and its parts. The operations resulting from the content of the ALC_DVS.1 component can be optionally extended by the de-

ployment of the Information Security Management System (ISMS) which complies with the ISO/IEC 27001 standard. Then, thanks to the extended management processes, there is an added value obtained in the form of keeping security parameters on the declared level.

The completed ALC_DVS subprocess results in the identification of the following information and the implementation of the security system in the development environment on this basis:

- general description of the organization,
- organizational unit responsible for the TOE development,
- technical conditions of the TOE development environment,
- security policy rules, including:
  - security policy for materials,
  - policy concerning confidential information,
  - policy concerning access to information and systems of the development environment,
- personnel security,
- access control,
- transfer of protected materials,
- security management,
- justification of the obtained protection level.

Establishing the configuration management system for the TOE (ALC_CMCp, ALC_CMSp)

In the development environment it is necessary to build a subprocess for IT product configuration management (CM). The objective of this subprocess (in the case of EAL4 – according to the requirements of the ALC_CMC.4 component) is to provide tools for precise control of the version and product configuration elements so that the version delivered to the client could be in accordance with the version that was previously evaluated and certified.

The configuration management subprocess comprises the elaboration of the following elements:

- preparation of a general description of the configuration management system (CM),
- user's documentation of the configuration management system,
- technical information,
- description of identification methods for configuration elements,
- identification of applied standard tools,
- specification of extensions and adaptations of standard tools,

- other supporting measures,
- procedural information,
- roles and responsibilities,
- description of procedures,
- configuration management plan,
- description of TOE access control measures,
- description of procedures for automatic generation of the TOE,
- description of acceptance procedures,
- description of the CM system results of operation,
- output data of the configuration management system,
- configuration list of the TOE,
- sample printouts of CM working records,
- extra information.

The set of configuration elements used to make configuration lists and determined in the Configuration management scope pattern (ALC_CMSp) shows how to specify items to be included as configuration items and hence controlled by the above CM capabilities (ALC_CMCp). For each elaborated version of the IT product these configuration lists encompass different kinds of identifiable configuration items, such as: physical and logical components of the product and its documentation, software or tools used in the development process, all evidences for the product or system, reports concerning flaws removal, implementation representation (source codes, includes, electronic diagrams), etc.

The following will be worked out in the course of the ALC_CMS subprocess:

- configuration lists for the TOE,
- configuration list submitted for evaluation,
- method to identify configuration elements,
- structure and content of the configuration list.

The structure and content of the configuration list includes:

- configuration list header,
- *readme* file for the configuration list,
- ID of the TOE,
- elements of evaluation evidences,
- TOE components,
- implementation representation (schemes, source codes, etc.),
- reports from flaws remediation and their status,
- tools used for the TOE development.

In general, one can say that while the ALC_CMS subprocess defines the computer-aided system of configuration management data in the life-cycle model, the data themselves are defined in the ALC_CMS subprocess.

Maintaining managing tools and techniques (ALC_TATp)

The ALC_TAT subprocess determines tools and techniques (programming languages, documentation, implementation standards, runtime libraries, different equipment, etc.) which serve for the development, production, maintenance, and utilization of the IT product or system. It also determines the way these tools and techniques are used with special focus on options that could create ambiguity in the obtained results, e.g. during compilation or calibration.

The subprocess results in the following:

- identification of development tools,
- description of development tools,
- description of implementation options of development tools,
- implementation standards used by developers, subcontractors and suppliers,
- description of implementation for particular standards.

Elaborating TOE delivery procedures (ALC_DELp)

The objective of this subprocess is to elaborate procedures for safe delivery of the IT product from its development and production environment to the client. The procedures comprise the following elements:

- general description of procedures,
- security objectives,
- delivery procedures,
- verification of TOE integrity by the users,
- evidence of procedures efficiency assessment.

Establishing flaw remediation channel (ALC_FLRp)

The flaw remediation (ALC_FLR) subprocess implements requirements how the detected security flaws should be traced and corrected by the developer. The subprocess encompasses flaw remediation procedures for the developers:

- tracking the reported security flaws and making communication channels for the users,
- analysis of security flaws,
- removal of security flaws (correction actions, assessment of flaw removal procedures efficiency, quick reaction procedures),
- determining the status of the reported security flaws,
- methods to inform the users about the flaws:

  - information channels for users,
  - delivery of information as a part of the TOE delivery process,
  - information delivered in the flaws remediation manual,
  - information provided on the website of the developer,
  - information delivered by means of other communication channels,
- involvement of TOE users,
- extended involvement of TOE users.

Additionally, basic and extended flaw remediation procedures for the TOE users are elaborated.

TOE decomposition and interfaces (ADV_TDS, ADV_FSP)

The subprocess for working out interfaces specification on the level of TOE, its subsystems and subsystem modules (ADV_FSP), and the subprocess of TOE decomposition into these subsystems and modules (ADV_TDS) are closely connected to each other, therefore they are carried out simultaneously.

Functional specification (based on the ADV_FSPp pattern) describes the TOE security functions interfaces (TSFIs). TSFIs consist of all means for users to invoke a service from the TSF (by supplying data that are processed by the TSF) and the corresponding responses to those service invocations.

The TOE design (based on the ADV_TDSp) describes the TOE security functions (TSFs) and provides context for the description of TSFIs. The TOE decomposition is specified on different levels of detail (subsystems, modules), which is implied by the declared EAL.

The Common Criteria methodology distinguishes three categories of interfaces and structural modules (subsystems and modules):

- SFR (Security functional requirements) enforcing interfaces/subsystems/modules),
- SFR supporting interfaces/subsystems/modules),
- SFR non-interfering interfaces/subsystems/modules.

Figure 2 features a sample decomposition of the IT product into subsystems and modules, and presents their interfaces. This decomposition model in UML is elaborated with the use of the EA-plugin – one of the CCMODE Tools components. There are three subsystems here (SUB). Subsystem 2 (dependent on subsystem 1) has two modules (MOD) which have three interfaces (TSFI) in total. Subsystem 3 has one module equipped with three interfaces. Please note the relationships between classes represented by different arrows.

The degree of detail in the specification of a subsystem or module depends strongly on the above mentioned categories and on the declared EAL (starting from the description on the block level, through the description of interactions, to detailed data structures and the description of the algorithm). Similarly, the degree of detail in the specification of an interface depends on the category and EAL (interface application, application method, parameters and their descriptions, operations, error codes, and mapping into Security functional requirements).

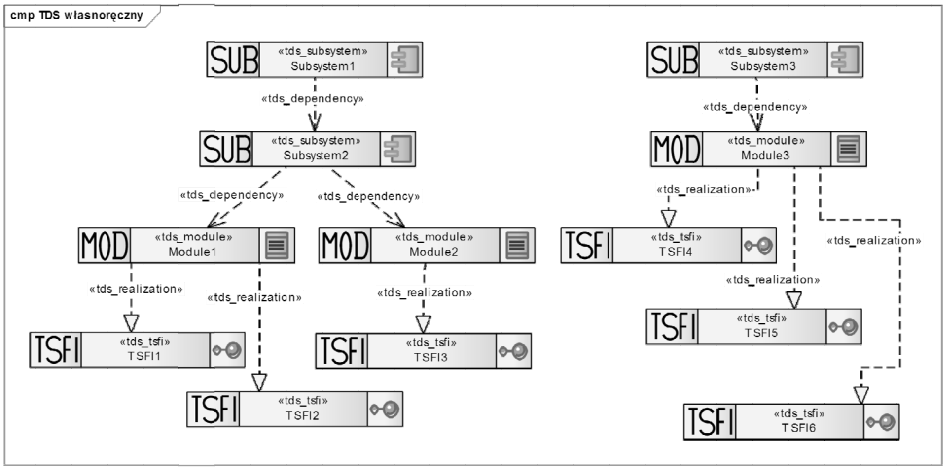

Fig. 2. Example of the TOE decomposition – subsystems, modules and their interfaces

Source: EMAG's documentation, 2014.

Elaborating TOE security architecture (ADV_ARCp)

Elaboration of the TOE Security architecture (based on the ADV_ARCp) is focused on the description of the security architecture of TOE security functions to show if they achieve desired properties.

This subprocess contains the description of self-contained or specialty implemented properties of the architecture which are used to protect such security functions as:

- distinguishing security domains,
- starting up the device after power switch-on or after a certain event occurs (Power on/Reset),
- mechanisms of protection against tampering, by-passing TOE security functions, etc.

Elaborating implementation representation of the TOE (ADV_IMPp)

Elaborating the implementation representation subprocess (based on the ADV_IMPp) which is responsible for the presentation how TSFs are implemented. They can be implemented with the use of different ways and technologies: software source codes and includes, hardware design language source codes, integrated circuits diagrams, layouts, etc.

Elaborating preparative procedures (AGD_PREp)

The subprocess is responsible for elaborating the procedures for the product/system installation and start-up. The procedures include approval for exploitation, preparation for installation, the installation itself, and start-up.

The procedures are made on the basis of the Preparative procedures pattern (AGD_PREp) which presents how the TOE has been received and installed in a secure manner as intended by the developer.

Elaborating operational guidance (AGD_OPEp)

This subprocess, based on the Operational user guidance pattern (AGD_OPEp), shows how to prepare written material intended for all types of users (roles) of the TOE in its evaluated configuration.

Elaborating functional tests (ATE_FUNp)

The objective of this subprocess is to work out the specification of functional tests, test plans and the description of the expected test results.

Justifying coverage of interfaces by tests (ATE_COVp)

The Test coverage subprocess, based on the ATE_COVp pattern, should demonstrate that all TSFIs are properly covered by tests.

Justifying coverage of subsystems and their modules by tests (ATE_DPTp)

The Test depth subprocess, based on ATE_DPTp, should demonstrate that specified TOE design elements (subsystems, modules) are properly covered by tests. Table 1 features a sample map of functional tests coverage (T-1 to T-9) of subsystems and modules (ATE_DPT). The table serves to confirm that all subsystems and modules are covered by tests.

Table 1

Sample test coverage of all subsystems and their SFR enforcing modules (EAL4)

Test name	Subsystem A			Subsystem B		Subsystem C		
	SFR enforcing modules of Subsystem A			SFR enforcing modules of Subsystem B		SFR enforcing modules of Subsystem C		
	Module A1	Module A2	Module A3	Module B1	Module B2	Module C1	Module C2	Module C3
T-1		X	X					
T-2	X		X					
T-3			X					
T-4				X	X	X		
T-5				X				
T-6						X		
T-7							X	
T-8								X
T-9								X

Source: EMAG.

First it is necessary to point the subsets of functional tests which are responsible for checking particular subsystems/modules. Then one has to justify that particular tests cover particular modules, e.g. T-4 and T-5 are enough to cover module B1.

The table with the test coverage for interfaces (ATE_COV) is constructed and justified in a similar way.

The evidences for ATE_DPT and ATE_COV are supported by EA-plugin, while tests management, including the tests definition, is carried out by an external software tool Testlink.

Independent tests (ATE_INDp)

The Independent testing subprocess (ATE_IND) has an auxiliary meaning for developers because the ATE_IND evidence is elaborated by evaluators. The evaluators repeat some tests specified by the developers and make new ones.

Vulnerability analysis subprocess (AVA_VANp)

The Vulnerability analysis subprocess has an auxiliary meaning because such evidences (similarly to the ATE_IND family) are elaborated mainly by evaluators.

Processes related to project security in SecLab

Minimal requirements concerning the development environment are determined by the components of the ALC_DVS family (for EAL4 the ALC_DVS.1 component is required). Optionally, minimal requirements can be supplemented by those described in security management standards. This way the efficiency of the security management process is improved.

SecLab is an experimental development environment for IT products whose operations (IT security development [BiaFli14b], TOE development – section 4) are secured by the implementation of an integrated information security management system with business continuity elements (ISO/IEC 27001, BS 25999, currently ISO 22301) – SecLab-SZBI/SZCD [BiaFli14a]. Both management systems are based on the Deming cycle: Plan – Do – Check – Act. The integrated management system is supported by the OSCAD tool [Bia12, OSCAD].

The basic OSCAD-supported processes in SecLab include:

- identification of assets and processes of SecLab,
- risk management – selections of security measures which are adequate to risks and costs,
- incidents management and preparation of correction actions,
- management of efficiency measures and indicators – preparation of improvement actions,
- management of documentation, tasks, emergency plans, and audits.

The need to implement the integrated information security and business continuity system in SecLab results from the necessity to protect the designing processes carried out in the laboratory, as well as the data related to these processes.

Conclusions

The paper concerns the development processes of IT products which are to be used in high-risk environments. These processes are carried out in compliance with the ISO/IEC 15408 Common Criteria methodology in the experimental SecLab laboratory organized in the EMAG Institute on the basis of the CCMODE project results.

The paper describes, with quite many details, one of the main processes of the Common Criteria methodology: the TOE development process whose result is the IT product with evidences that are submitted for evaluation together with the product. This process is the continuation of the IT security development process [BiaFli14b].

For each subprocess the author presented activities and expected results in the form of evaluation evidences or their part.

Each process runs on the basis of an evidence pattern that provides a strict framework for the subprocess and ensures compliance of the produced evidence with the CC methodology. Using such patterns, the developers can focus on the developed IT product. They do not have to edit the evidence structure based on detailed requirements which are written in a difficult language of the CC methodology. The use of patterns improves the quality and efficiency of the IT product development process.

SecLab makes use of CCMODE Tools – specialized software that supports the development process in the range of the Common Criteria methodology. CCMODE Tools is used to manage the project from the technical point of view. It provides an integrated automated development environment and knowledge necessary to carry out projects. The computer support concerns the following:

- most labourious operations (versioning, configuration management, generation of evidences),
- most difficult tasks (product modelling, security analysis, different justifications of evidence for completeness and adequacy of the applied measures).

It is possible to use the results of previously conducted projects to carry out the current one (the so called reusability). All these advantages increase the efficiency and quality of the development process. The benefits are similar to those resulting from the application of CAD/CAM/CAE systems.

Data related to the conducted projects as well as the processes of the CC methodology need special protection. That is why the use of OSCAD was proposed, i.e. an integrated computer-aided information security and business continuity management system. Reinforced protection of SecLab processes and project data, along with the compliance with the site certification concept, are still new issues in the realm of Common Criteria.

The methodology and tools of the SecLab laboratory were used in the development process of intelligent sensors for the mining industry and specialized software.

EMAG's SecLab experimental laboratory is used to carry out research and development works and trainings on IT security and to demonstrate security solutions, particularly those resulting from the CCMODE project. These activities aim to:

- support businesses which implement their own development environments or apply solutions that comply with Common Criteria,
- promote best practices in the development of IT products, either for certification or other purposes.

The objective of these operations is to disseminate the Common Criteria methodology in Poland and popularize secure IT solutions.

References

[BiaFli14b] Białas A., Flisiuk B.: IT Security Development Process in the Experimental SecLab Development Environment. In: M. Pańkowska, J. Palonka, H. Sroka (eds.): Ambient Technologies and Creativity Support Systems. Uniwersytet Ekonomiczny, Katowice 2014.

[Bia12] Białas A. (ed.): Komputerowe wspomaganie procesu rozwoju produktów informatycznych o podwyższonych wymaganiach bezpieczeństwa (Computer Support for the Development of IT Products of Enhanced Security). Wydawnictwo Instytutu Technik Innowacyjnych EMAG, financed by UE POIG 1.3.1, Katowice 2012.

[Bia11] Białas A. (ed.): Zastosowanie wzorców projektowych w konstruowaniu zabezpieczeń informatycznych zgodnych ze standardem Common Criteria (Design Patterns for the Development of IT Security in Compliance with Common Criteria), Wydawnictwo Instytutu Technik Innowacyjnych EMAG, financed by UE POIG 1.3.1, Katowice 2011.

[BiaFli14a] Białas A., Flisiuk B.: Specialized Development Environments for Security-enhanced IT Products and Systems. In: M. Pańkowska, J. Palonka, H. Sroka (eds.): Ambient Technologies and Creativity Support Systems. Uniwersytet Ekonomiczny, Katowice 2014.

[CCMODE] CCMODE (Common Criteria compliant, Modular, Open IT security Development Environment). http://www.commoncriteria.pl/.

[CC1-3] Common Criteria for IT Security Evaluation, part 1-3. v. 3.1.2009.

[CCPortal] Common Criteria Portal. http://www.commoncriteriaportal.org/.

[Her03] Hermann D.S.: Using the Common Criteria for IT Security Evaluation. CRC Press: Boca Raton, FL, USA, 2003.

[Hig10] Higaki W.H.: Successful Common Criteria Evaluation. A Practical Guide for Vendors. Copyright 2010 by Wesley Hisao Higaki, Lexington, KY 2010.

[OSCAD] OSCAD: http://www.oscad.eu.

[Rog14] Rogowski D.: Applying Computer Tool to Common Criteria Methodology. Proc. of CSS'2014, Uniwersytet Ekonomiczny, Katowice (accepted).

ROZWÓJ PRODUKTU TECHNOLOGII INFORMACJI ZGODNY Z METODOLOGIĄ COMMON CRITERIA W EKSPERYMENTALNYM LABORATORIUM SECLAB

Streszczenie

Artykuł przedstawia, jak organizować proces rozwoju produktu technologii informacji (tj. sprzęt, oprogramowanie, firmware, systemy) w eksperymentalnym laboratorium Instytutu EMAG zwanym SecLab. Laboratorium zostało utworzone wraz z zastosowaniem produktów stanowiących rezultaty projektu CCMODE (Common Criteria compliant Modular Open IT security Development Environment).

W artykule zaprezentowano, jak proces rozwoju produktu jest realizowany z użyciem wzorców i narzędzi powstałych w ramach projektu CCMODE. Dodatkowo omówiono kwestię bezpieczeństwa w odniesieniu do procesów realizowanych w utworzonym laboratorium i ochronę aktywów związanych z tymi procesami.