Theoretical and Applied Informatics
ISSN 1896-5334
Vol. 25 (2013), no. 3-4
pp. 201 – 222
DOI: 10.2478/thai-2013-0015

Validation of the Ontology Based Model of the Common Criteria Evaluation Evidences

ANDRZEJ BIAŁAS

Instytut Technik Innowacyjnych EMAG
ul. Leopolda 31, Katowice, Poland
a.bialas@emag.pl

Received 3 April 2013, Revised 27 October 2013, Accepted 2 December 2013

Abstract: The paper concerns the validation of the selected issues related to the new ontology-based approach to the elaboration and management of evidences prepared by developers for the IT security evaluation process according to the Common Criteria standard. The evidences are implied by the claimed EAL (Evaluation Assurance Level) for a developed IT product or system, called TOE (Target of Evaluation). The evidences envisage the TOE features and its development environment. The validation and use of the author's elaborated ontology are discussed, including: composing evidences for the given TOE and EAL, expressing details of evidences documents, issuing queries to get given information about model, etc. The paper also shows how the evidences are organized, developed and used. This work is aimed at the development of a prototype of a knowledge base, designed mainly for developers to allow them to compose and manage different kinds of evidences elaborated on the patterns basis. This knowledge base can be used by a software tool aiding developers who produce evaluation evidences.

Keywords: Common Criteria, IT security evaluation, knowledge engineering, modelling, ontology, assurance methods.

1. Introduction

The paper concerns the IT product or system development compliant with the ISO/IEC 15408 Common Criteria (CC) methodology [1], [2], presenting a new ontological approach to the elaboration of evidences which ought to be worked out and provided for an IT security evaluation process. The work exemplifies how to apply a knowledge engineering methodology to the security engineering domain. The paper [3] presents the Specification Means Ontology (SMO) related to the Common Criteria methodology, focusing on the evaluation evidences, discussing its background, rationale and elaboration, yet this paper shows the selected aspects of its validation and use. It continues the discussion of its predecessor [3], which the reader is kindly requested to get familiar with.

The Common Criteria methodology plays the key role in providing assurance for IT products (hardware, software, firmware) used in large businesses, industry, e-government and e-health sectors. These products, called TOEs (*Targets of Evaluation*) are rigorously elaborated, tested, analysed, documented, and their security is evaluated and certified by independent bodies.

The assurance is measurable with the special scale i.e. the EAL scale (*Evaluation Assurance Level*) in the range from EAL1 (min) to EAL7 (max). The TOE developers should provide evidences that the TOE meets the claimed EAL requirements described in Part 3 of the standard [1]. The elaboration of evidences for the given TOE should reflect the rigour derived from the declared EAL, as well as the features of the IT product, character of the development, manufacturing and operational environments.

The way from the particular EAL requirements to the structured documents presenting evidences is not easy and requires common understanding of terms, mastering many interrelated details, specialised know-how and generally – knowledge and experiences. This situation causes many problems for IT developers.

The objective of the works presented in [3] and in this paper is to improve the elaboration of evidences provided for the evaluation process, using advantages and possibilities brought by the ontological approach. The improvement generally concerns: better formalization and preciseness, possible software support, and domain knowledge management of CC compliant IT development processes. This allows to define structure and contents of evidences implied by the given EAL.

The paper includes five main sections. Section 2 contains a very short introduction to the applied here ontological approach. Section 3 characterizes concisely the domain of the discussed ontology, i.e. CC compliant IT security development- and TOE development processes. Section 4 resumes current works on the validated Specification Means Ontology. The most important is section 5, exemplifying how the evidences can be composed with the use of this ontology and how the data can be retrieved from the related knowledge base.

2. Ontological approach

The ontology, a key term of knowledge engineering, represents explicit formal specifications of a set of concepts within a domain of knowledge and the relationships between those concepts [4]. In this paper and in [3] the domain of knowledge is defined as “Common Criteria compliant IT security development and evaluation”. This domain can be considered as a subdomain of a broader “security engineering and information security” domain.

Ontology classes represent the mentioned concepts. A class can be considered as a set of similar primitives called instances. The class may have subclasses, which are more specific than the class itself (superclass). The classes may have different attributes or features called properties. The properties may have restrictions assigned.

To define and edit the ontology and the related knowledge base, the Protégé Ontology Editor and Knowledge Acquisition System developed at Stanford University [5] will be used. Please note that this editor use older equivalent terms – “individual” instead of “instance” and “slot” instead of “property”. The considered ontology is expressed by the OWL language, precisely OWL-DL (*DL – Description Logics*) which allows automatic reasoning. OWL (*Web Ontology Language*) is supported by the World Wide Web Consortium (W3C). The review of current researches related to similar ontologies, especially related to the Common Criteria methodology, was presented in the paper [3]. The conclusions is that none of these researches encompasses the entire CC-related development and evaluation processes in a holistic way and none of them considers composing and management of the required Common Criteria evidences.

3. Considered domain of knowledge

The considered domain of knowledge encompasses three CC-related processes [3]:

- IT security development process, aimed at the security target (ST) specification work out, which includes the TOE security functions (TSF);
- TOE development process, focused on the elaboration of an IT product or system and its evaluation evidences, justifying that these security functions are implemented at the claimed EAL;
- IT security evaluation and certification performed by an independent body; it is the assessment of the ST, evidences and the TOE against security assurance criteria in order to answer if EAL is met.

Currently only the first two processes are covered by the discussed here Specification Means Ontology (SMO).

The basic version of the IT security development process includes:

1. Preparing the ST introduction which contains different identifiers and informal descriptions of the TOE;
2. Security problem defining (SPD); SPD specifies threats, OSPs (organizational security policies) and assumptions;
3. Solving this problem by specifying security objectives (SO) – for the TOE and its development – and operational environments;
4. Working out the security functional requirements (SFRs) specification on the security objectives basis and a set of security assurance requirements (SARs) which are derived mainly from the declared EAL (please note: EALs are predefined packages of SARs);
5. Preparing the TOE summary specification (TSS), containing the security functions (SF) derived from the SFRs that should be implemented in the IT product or system.

The developed security functions are implemented within the TOE according to the rigour and details implied by the claimed EAL. The EAL is not only an assurance measure but also a harmonized collection of SAR components. The TOE development process encompasses the elaboration of different documents playing the role of evaluation evidences. Each evidence is implied by the assurance component of the given assurance family. The families are grouped by the assurance classes:

- ADV (*Development*) class concerns evidences dealing with the TOE architecture, functional specification, design, implementation and security policy;
- ALC (*Life cycle support*) class concerns evidences dealing with the configuration management, life cycle, product delivery, development process security, used tools, flaw remediation;

- ATE (*Tests*) class implies the tests specification, test depth and coverage evidences;
- AGD (*Guidance documents*) specifies evidences related to the product manuals and procedures;
- AVA (*Vulnerability assessment*) requires a proper vulnerability analysis of the IT product and its development environment.

In practice the evidences implied by SARs of the given EAL encompass different kinds of documents, e.g.: user's and technical documentation, tests, procedures, reports from analyses, documented behaviour, system records, etc. Some evidences concern directly the TOE while the others its development-, manufacturing- or operational environments.

4. Specification Means Ontology – short information

The Specification Means Ontology (SMO) was discussed in the paper [3], for this reason only the basic information will be provided to help the readers understand the below presented ontology validation issues.

The following important classes (subclasses of the standard ontology class `owl:Thing`) or groups of classes were defined for SMO:

- `AuxiliaryConcept` class, which usually represents enumerative subclasses whose instances are mainly used for knowledge organization and retrieving;
- `CCSecComponent` class, which expresses security requirements: assurance requirements – SAR (`SARComponent`) and functional requirements – SFR (`SFRComponent`) defined in [1] and discussed in [6-8];
- `EnhancedGeneric` class, which represents enhanced generics used as specification items for development stages other than the security requirements elaboration (i.e. items for: assets, subjects, threats, security policies, assumptions, security objectives and security functions), defined previously for the ITSDF framework [9-11] and discussed in [6-8];
- group of classes concerning evidences discussed here; `EvidenceDoc`, representing the TOE evidences as a whole and integrating their family evidences elaborated for particular assurance families (expressed by the `FamilyEvidence` class) on the patterns basis (expressed by the `EvidenceTemplate` subclasses) and with the use of guidance documents (expressed by the `EvidenceGuide` subclasses); these issues are discussed in this paper;
- group of classes which refer to particular kinds of security specifications [1] (`SecurityTarget`, `ProtectionProfile`, `LowAssST`, `LowAssPP`) and their parts (`ST_PP_Part`), e.g.: `SecProblemDef`, the `SecObjectives`, `SecRequirem`, `TSS_TOESumSpec`, defined in [12] and currently integrated with SMO.

The `AuxiliaryConcept` subclasses are varied. For example, one of its subclasses is EAL. It contains EALs definitions as instances (`EAL1`, `EAL1plus`, EAL2, … EAL6plus, EAL7). The Project subclass of AuxiliaryConcept has instances related to the particular TOE projects carried out with the use of the SMO ontology, e.g. here discussed *MyFirewall* project.

Since the evidences are implied by SARs, it is necessary to discuss them briefly. SARComponent encompasses all CC assurance classes ([1]/Part 3): ADVClass, AGDClass, ALCClass, etc. Each CC assurance class has its CC assurance families, e.g. ADVClass has ADV_ARC, ADV_FSP, ADV_IMP, etc. The hierarchy of SFRs is expressed in a similar way.

The evidence documentation for an IT product or system with respect to the declared EAL is represented by the instance of the EvidenceDoc class. This instance integrates evidences implied by particular assurance families, which are expressed by the FamilyEvidence subclasses (exactly: by their instances): ADV_ARC_EAL, ADV_FSP_EAL, ADV_IMP_EAL, …, AVA_VAN_EAL, OptEvid_ALC_FLR, OptEvid_SAR_OTHER classes. The latter two subclasses of FamilyEvidence have special meaning. The OptEvid_ALC_FLR class expresses flaw remediation requirements that can be included optionally for any EAL, while OptEvid_SAR_OTHER represents evidences added by developers for the user's defined SARs.

The family evidences are elaborated on the basis of patterns, expressed by the EvidenceTemplate subclasses, while the EvidenceGuide subclasses express guidelines how to use these patterns.

Three kinds of standard properties [4-5] are used:

- object (called also "instance-type") properties, expressing "complex properties", i.e. relationships between an individual member (instance) of the given class (the object) and other instances; e.g. when the given instance points to other instances or consists of other instances; examples: the assignedToProject property specifies a project name (Project class range) to which the given ontology item belongs (in this case the domain encompasses almost all ontology classes), the hasBasicEvidence property assigns assurance family evidences (FamilyEvidence range) to the composed set of evidences for the TOE (Evidences domain);
- data-type properties, expressing "simple properties" or "attributes", i.e. intrinsic or extrinsic properties of the instances of the most elementary classes; the data type used for this property can be any of those commonly used in modelling or programming, e.g.: integer, byte, float, time, date, enumeration, string; examples: the hasComments property, representing verbal notes (the range string) added to instances of some classes related to evidences (in this case a domain is a sum: EvidenceDoc or FamilyEvidence or EvidenceGuide), the properties: hasTitle, fileName, fileLocation (domain: EvidenceDoc or FamilyEvidence or EvidenceTemplate or EvidenceGuide and the range string) are used to reference external documents;

- annotation properties expressing the meaning of the given class, RDF-based and used to document different ontology items (classes, properties, instances); example: the `rdfs:comment` property gives more explanation of the given ontology item.

In the SMO ontology instances belong to the lowest levels of the class hierarchy and contain: functional components, assurance components, enhanced generics and the discussed here evidences. For each class, representing assurance family evidences (`ADV_ARC_EAL`, `ADV_FSP_EAL`, `ADV_IMP_EAL`, …, `AVA_VAN_EAL`), appropriate instances are created. For example, the `ADV_FSP_EAL` class has the following instances: `ADV_FSP_EAL_1` (for EAL1), `ADV_FSP_EAL_2` (for EAL2), `ADV_FSP_EAL_3` (for EAL3), `ADV_FSP_EAL_4` (for EAL4), `ADV_FSP_EAL_5` (for EAL5 and EAL6) and `ADV_FSP_EAL_7` (for EAL7).

5. Using SMO to compose evidences – selected issues concerning the ontology validation

The Security Means Ontology is complex and comprises a few subdomains. The SMO validation focused on the ST elaboration was discussed in earlier works [6-8], but this paper concerns the validation focused on evaluation evidences.

The SMO development process will be exemplified by some issues concerning the SMO ontology validation with the use of the Protégé tool based on a project about a simple firewall system (*MyFirewall* project). The *MyFirewall* project specified in Appendix E of the monograph [11] was developed on the basis of “Annex D Worked Example: Firewall PP and ST” [20].

The evidences concern the implementation of the TOE security functions at the claimed EAL, so in the first step it is demonstrated how these functions are identified and how an EAL for them is declared. In the second step, the composition of TOE evidences is discussed, i.e. how the evidences items of particular assurance families are sampled together for the TOE and a given EAL with the possibility to add and/or substitute SARs. The next step presents an idea how to elaborate a given family evidence item with the use of a template (design pattern) and guideline (methodology). Elaborated evidence items are considered knowledge base items. In the last step it is shown how these knowledge items can be retrieved with the Protégé tool.

5.1 Security assurance requirements for the TOE security functions specified within the security target

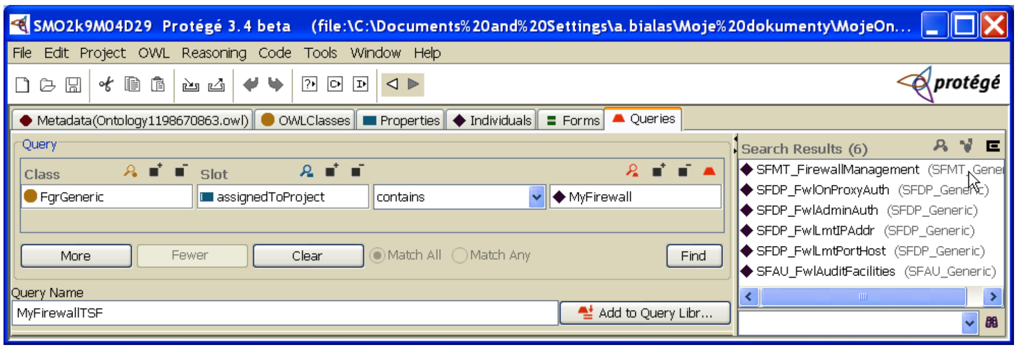
The evidences are implied by the contents of the security target specification. During the ST elaboration two issues are important: selection of proper specification means from a huge number of available generics and components included in the knowledge base, and creation of the right relationships dealing with:

- the generics parameterization, e.g. a threat generic has 2 parameters: the threatened asset substituted by an asset generic and the threat agent substituted by a subject generic;
- the generics mapping, i.e. covering problems by their solutions, e.g. assigning the right security objective to solve a given threat, assigning a given SFR to express the security objective, and specify the proper security functions implementing one or more SFRs.

Enhanced generics are used for the security problem definition and security objectives specification. The security objectives are covered by security functional components, and these, finally, by security functions (SF) expressed also by generics. It was discussed in [11], [6-7].

For the *MyFirewall* TOE the following functions are specified using enhanced generics, all together called the TOE security functionality (TSF):

- `SFDP_FwlLmtIPAddr` SF controlling IP addresses of data packages transmitted between public and protected private networks;
- `SFDP_FwlLmtPortHost` SF controlling port numbers used for data exchange between public and protected private networks;
- `SFDP_FwlOnProxyAuth` SF responsible for the user authentication on a proxy server for certain services;
- `SFDP_FwlAdminAuth` SF providing an authentication facility for the firewall administrator;
- `SFAU_FwlAuditFacilities` SF sampling and analysing information concerning auditable events;
- `SFMT_FirewallManagement` SF providing a management facility for the firewall administrator.



Class: FgrGeneric
Slot: assignedToProject
contains
MyFirewall

Query Name: MyFirewallTSF

Search Results (6)
- SFMT_FirewallManagement (SFMT_Gene
- SFDP_FwlOnProxyAuth (SFDP_Generic)
- SFDP_FwlAdminAuth (SFDP_Generic)
- SFDP_FwlLmtIPAddr (SFDP_Generic)
- SFDP_FwlLmtPortHost (SFDP_Generic)
- SFAU_FwlAuditFacilities (SFAU_Generic)

Fig. 1. The Protégé [5] query results retrieving the security functions of the *MyFirewall* system.

Please note the short names of generics – mnemonics, e.g. “`FwlLmtIPAddr`“ preceded by prefixes – grouping items according to the SFRs taxonomy [1], e.g. “`SFDP_`”, and followed by textual descriptions. Specifying these functions, i.e. the items of the TOE summary specification (TSS) of ST, completes the IT security development process, allowing to begin the TOE development while the evidences are elaborated. Fig. 1 presents the results of the `MyFirewallTSF` query submitted to the

SMO knowledge base, finding the above mentioned functions as instances (individuals) in Protégé, marked as rhombuses.

The query exemplifies one of the Protégé tool features allowing to find different ontology items, including evidence items. The query `MyFirewallTSF` means: "Find all TOE security functions, i.e. "`FgrGenerics` class instances of the `MyFirewall` project". As the result, 6 above mentioned functions were displayed.

Identification of these functions constitutes a good starting point to the followed considerations on evidences during the TOE development process.

These security functions can be implemented (and later evaluated) with different rigour and details which the developer expresses declaring an EAL for the TOE (i.e. *MyFirewall*). The given EAL refers to the EAL package which contains a set of relevant SARs belonging to different assurance families. Each family, if included in the EAL package, is represented by a right component of the family hierarchy. The elaborated evidences should demonstrate that the TOE meets the requirements included in the applied components of assurance families.

5.2 Catching the TOE evidences as a whole

As it was presented in [3], the evidences focused on particular families are represented by the `FamilyEvidence` subclasses but the `EvidenceDoc` class is responsible for integrating these families evidences into the evidences for the TOE, with the ability to add optional evidences or change evidences implied directly by an EAL. Sometimes there is a need to evaluate an IT product or system against an EAL with some added or raised requirements, i.e. against the EALn+ (marked "plus" in the Protégé environment). For example the EAL4 package can be modified in a special way by adding or substituting some SARs. In the SMO ontology, the relations expressing the structure of evidences are expressed by three properties, having the same domain (the `EvidenceDoc` class) and range (the `FamilyEvidence` class):

- `hasBasicEvidence` – integrating evidences implied by particular SARs of assurance families included in the standard EAL package;
- `hasEvidFromAddedSARs` – adding evidences implied by the SAR components intentionally added by the developer to the EAL package,
- `hasEvidFromSubstSARs` – adding evidences implied by the SAR components of higher rigour replacing some standard components of the given EAL package.

To sum up, the number and kinds of instances expressing the assurance family evidences depend on the EAL and on intentionally added/substituted SARs. All together they are expressed by the `EvidenceDoc` class instance that is shown in the next example.

<u>Example 1.</u> The ontological representation of evidences for a given IT product or system (TOE).

Let us consider the *MyFirewall* project used for the ontology validation. It was assumed that this TOE will be evaluated on EAL4+. The *ALC_FLR.2* component was added to the standard EAL4 package, but the *ALC_TAT.1* was replaced by

ALC_TAT.2 of higher rigour. The *MyFirewall* evidences for EAL4+ are presented in Tab. 1.

Component	**Required evidence**
ADV_ARC.1	Security architecture facilities: supporting self-protection, domain separation, non-bypassability and secure initialization.
ADV_FSP.4	Functional specification describing interfaces of TSF (TOE security function) subsystems and their security enforcing modules.
ADV_IMP.1	Implementation representation of TSF (software/firmware/hardware design language source code, hardware/IC diagrams, layouts).
ADV_TDS.3	TOE Design, specifying TSF subsystems and their security enforcing modules.
AGD_OPE.1	Operational user guidance for any role engaged in the development, manufacturing, maintenance and usage of the TOE.
AGD_PRE.1	Preparative procedures for the TOE, e.g.: acceptance, installation, calibration – preparation of the operational environment in accordance with the security objectives.
ALC_CMC.4	Configuration management capabilities, incl. production support, acceptance procedures and automation.
ALC_CMS.4	Configuration management scope: TOE and its parts, evidences, implementation representation; security flaws and their resolution status.
ALC_DEL.1	Delivery procedures to the consumer.
ALC_DVS.1	Specifying physical, procedural, personnel, and other security measures that are necessary to protect the confidentiality and integrity of the TOE design and implementation in its development environment.
ALC_FLR.2 (adding)	Flaw reporting procedures.
ALC_LCD.1	Life-cycle model definition to be used in the development and maintenance of the TOE.
ALC_TAT.2 (substitution)	Specifying the development tools and implementation standards, e.g. concerning the coding applied for the TOE.
ATE_COV.2	Analysis of test coverage demonstrating that all TSF interfaces specified in the functional specification have been tested.
ATE_DPT.2	Analysis of test depth demonstrating that all TSF subsystems and their security enforcing modules specified in the TOE design specification have been tested.
ATE_FUN.1	Functional testing – tests procedure and test results description.
ATE_IND.2	Independent testing – results of tests performed by evaluators.
AVA_VAN.3	Results of the vulnerability analysis performed by evaluators.

Tab. 1. *MyFirewall* evidence documentation summary for EAL4+ (*ALC_FLR.2* added, *ALC_TAT.1* substituted by *ALC_TAT.2*)

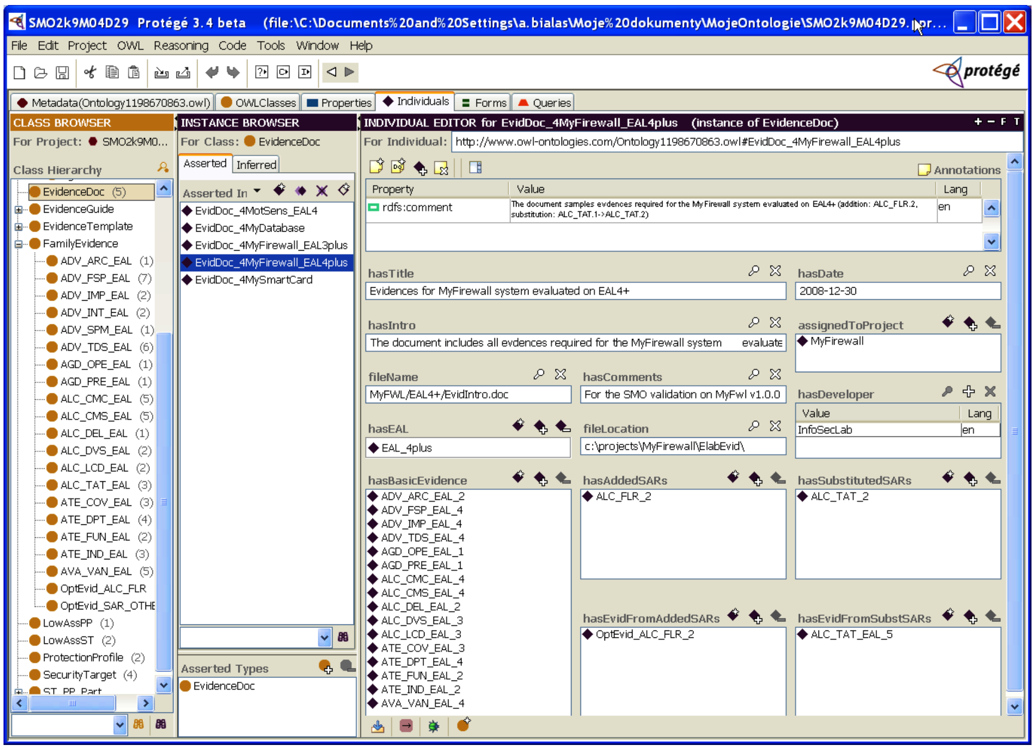
For particular components a right evidence document should be developed. It is not easy and requires patterns and knowledge which will be provided by the developed SMO ontology.

The Protégé "Class Browser" (Fig. 2) shows classes representing evidences. Please note the main `EvidenceDoc` class and `FamilyEvidence` subclasses. For the highlighted `EvidenceDoc` class its instances for some TOEs are shown in the "Instance Browser" in the middle of Fig. 2. One of them is the `EvidDoc_4MyFirewall_EAL4plus` instance, concerning the discussed firewall project. The Protégé "Individual Editor" presents details of this instance, expressing the all evidences for this target of evaluation.

With respect to the standard EAL4 package, the following evidences implied by assurance families are included using the `hasBasicEvidence` property:

`ADV_ARC_EAL_2`, `ADV_FSP_EAL_4`, `ADV_IMP_EAL_4`,
`ADV_TDS_EAL_4`, `AGD_OPE_EAL_1`, `AGD_PRE_EAL_1`,
`ALC_CMC_EAL_4`, `ALC_CMS_EAL_4`, `ALC_DEL_EAL_2`,
`ALC_DVS_EAL_3`, `ALC_LCD_EAL_3`, ~~`ALC_TAT_EAL_4`~~,
(replaced, it will be explained later)
`ATE_COV_EAL_3`, `ATE_DPT_EAL_4`, `ATE_FUN_EAL_2`,
`ATE_IND_EAL_2`, `AVA_VAN_EAL_4`.

For the *MyFirewall* project the optional "*ALC_FLR.2 Flaw reporting procedure*" component was added (the `hasAddedSARs` property), represented by the `ALC_FLR_2` instance. It implies adding appropriate assurance family evidences (the `OptEvid_ALC_FLR_2` instance) with the use of the `hasEvidFromAddedSARs` property.


SMO2k9M04D29 Protégé 3.4 beta (file:\C:\Documents%20and%20Settings\a.bialas\Moje%20dokumenty\MojeOntologie\SMO2k9M04D29.ppr...

File Edit Project OWL Reasoning Code Tools Window Help
protégé
Metadata(Ontology1198670863.owl)
OWLClasses
Properties
Individuals
Forms
Queries

CLASS BROWSER
For Project: SMO2k9M0...
Class Hierarchy
- EvidenceDoc (5)
- EvidenceGuide
- EvidenceTemplate
- FamilyEvidence
  - ADV_ARC_EAL (1)
  - ADV_FSP_EAL (7)
  - ADV_IMP_EAL (2)
  - ADV_INT_EAL (2)
  - ADV_SPM_EAL (1)
  - ADV_TDS_EAL (6)
  - AGD_OPE_EAL (1)
  - AGD_PRE_EAL (1)
  - ALC_CMC_EAL (5)
  - ALC_CMS_EAL (5)
  - ALC_DEL_EAL (1)
  - ALC_DVS_EAL (2)
  - ALC_LCD_EAL (2)
  - ALC_TAT_EAL (3)
  - ATE_COV_EAL (3)
  - ATE_DPT_EAL (4)
  - ATE_FUN_EAL (2)
  - ATE_IND_EAL (3)
  - AVA_VAN_EAL (5)
  - OptEvid_ALC_FLR
  - OptEvid_SAR_OTHE
- LowAssPP (1)
- LowAssST (2)
- ProtectionProfile (2)
- SecurityTarget (4)
- ST_PP_Part

INSTANCE BROWSER
For Class: EvidenceDoc
Asserted Inferred
Asserted In
- EvidDoc_4MotSens_EAL4
- EvidDoc_4MyDatabase
- EvidDoc_4MyFirewall_EAL3plus
- EvidDoc_4MyFirewall_EAL4plus
- EvidDoc_4MySmartCard

Asserted Types
- EvidenceDoc

INDIVIDUAL EDITOR for EvidDoc_4MyFirewall_EAL4plus (instance of EvidenceDoc)
For Individual: http://www.owl-ontologies.com/Ontology1198670863.owl#EvidDoc_4MyFirewall_EAL4plus

Annotations

Property	Value	Lang
rdfs:comment	The document samples evdences required for the MyFirewall system evaluated on EAL4+ (addition: ALC_FLR.2, substitution: ALC_TAT.1->ALC_TAT.2)	en

hasTitle: Evidences for MyFirewall system evaluated on EAL4+
hasDate: 2008-12-30
hasIntro: The document includes all evdences required for the MyFirewall system evaluate
assignedToProject: MyFirewall
fileName: MyFWL/EAL4+/EvidIntro.doc
hasComments: For the SMO validation on MyFwl v1.0.0
hasDeveloper:

Value	Lang
InfoSecLab	en

hasEAL: EAL_4plus
fileLocation: c:\projects\MyFirewall\ElabEvid\

hasBasicEvidence:
- ADV_ARC_EAL_2
- ADV_FSP_EAL_4
- ADV_IMP_EAL_4
- ADV_TDS_EAL_4
- AGD_OPE_EAL_1
- AGD_PRE_EAL_1
- ALC_CMC_EAL_4
- ALC_CMS_EAL_4
- ALC_DEL_EAL_2
- ALC_DVS_EAL_3
- ALC_LCD_EAL_3
- ATE_COV_EAL_3
- ATE_DPT_EAL_4
- ATE_FUN_EAL_2
- ATE_IND_EAL_2
- AVA_VAN_EAL_4

hasAddedSARs: ALC_FLR_2
hasSubstitutedSARs: ALC_TAT_2
hasEvidFromAddedSARs: OptEvid_ALC_FLR_2
hasEvidFromSubstSARs: ALC_TAT_EAL_5

Fig. 2. Composing EAL4+ evidences for the *MyFirewall* project using the Protégé environment [5].

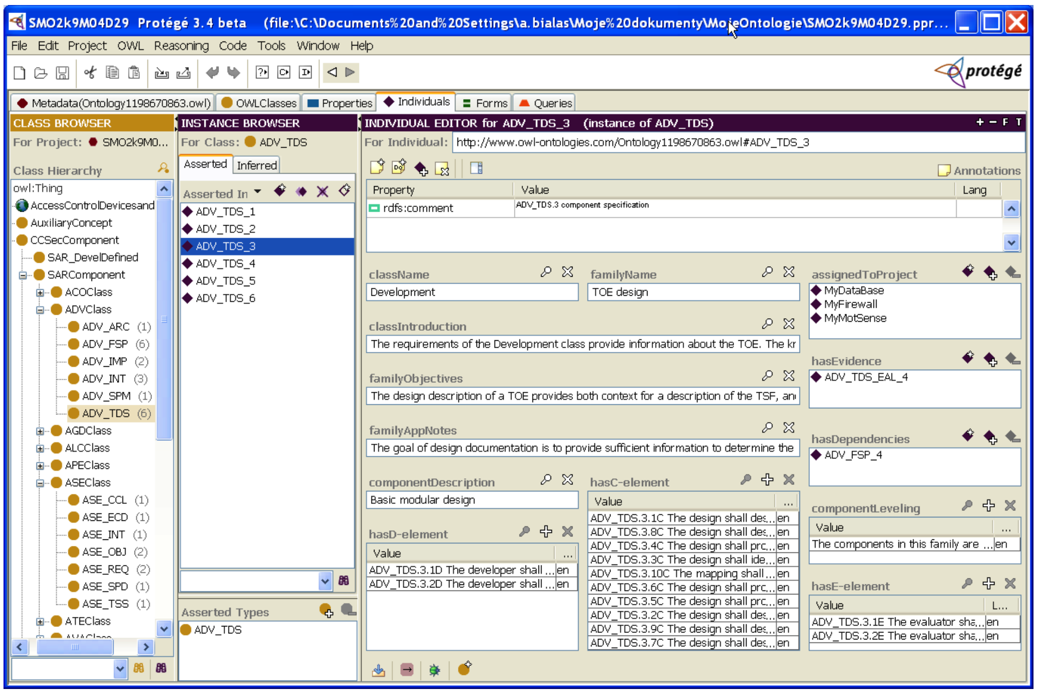
One of the *MyFirewall* project requirements was that the clients expect confirmation that the implementation standards were properly applied. For this reason the standard component of the EAL4, i.e. "*ALC_TAT.1 Well-defined development tools*" should be replaced by its higher equivalent, i.e. by "*ALC_TAT.2 Compliance with implementation standards*", requiring to describe implementation standards by the developer, e.g. concerning the coding applied for the TOE. This is an example of the SAR substitution. Instead of `ALC_TAT_1`, the `ALC_TAT_2` instance is used (see the `hasSubstitutedSARs` property), causing to remove the assurance family evidence `ALC_TAT_EAL_4` (the one crossed out above) and adding the `ALC_TAT_EAL_5` evidence using the `hasEvidFromSubstSARs` property.
□

5.3 Elaborating evidences for particular assurance families

For each EAL package an appropriate SAR component of the given assurance family is included (Tab. 1 in ([3]), for which assurance family evidence should be elaborated. The next example shows the basis and used input information for this process. The key issue is the structure of the assurance component.

<u>Example 2.</u> The ontological representation of `ADV_TDS_3` (Fig. 3) – one of the EAL4+ assurance components.

The “*ADV_TDS.3 Basic modular design*” assurance component is represented by `ADV_TDS_3` which is an instance of the `ADV_TDS` ontology class. It is one of SARs implied by the claimed EAL4+, specified for the *MyFirewall* project.


SMO2k9M04D29 Protégé 3.4 beta (file:\C:\Documents%20and%20Settings\a.bialas\Moje%20dokumenty\MojeOntologie\SMO2k9M04D29.ppr...
File Edit Project OWL Reasoning Code Tools Window Help
protégé
Metadata(Ontology1198670863.owl)
OWLClasses
Properties
Individuals
Forms
Queries
CLASS BROWSER
INSTANCE BROWSER
Asserted
Inferred
Asserted In
Asserted Types
Annotations

For Project: SMO2k9M0...

Class Hierarchy

- owl:Thing
  - AccessControlDevicesand
  - AuxiliaryConcept
  - CCSecComponent
    - SAR_DevelDefined
    - SARComponent
      - ACOClass
      - ADVClass
        - ADV_ARC (1)
        - ADV_FSP (6)
        - ADV_IMP (2)
        - ADV_INT (3)
        - ADV_SPM (1)
        - ADV_TDS (6)
      - AGDClass
      - ALCClass
      - APEClass
      - ASEClass
        - ASE_CCL (1)
        - ASE_ECD (1)
        - ASE_INT (1)
        - ASE_OBJ (2)
        - ASE_REQ (2)
        - ASE_SPD (1)
        - ASE_TSS (1)
      - ATEClass

For Class: ADV_TDS

- ADV_TDS_1
- ADV_TDS_2
- ADV_TDS_3
- ADV_TDS_4
- ADV_TDS_5
- ADV_TDS_6

Asserted Types: ADV_TDS

INDIVIDUAL EDITOR for ADV_TDS_3 (instance of ADV_TDS)

For Individual: http://www.owl-ontologies.com/Ontology1198670863.owl#ADV_TDS_3

Property	Value	Lang
rdfs:comment	ADV_TDS.3 component specification	

className: Development

familyName: TOE design

classIntroduction: The requirements of the Development class provide information about the TOE. The kr

familyObjectives: The design description of a TOE provides both context for a description of the TSF, an

familyAppNotes: The goal of design documentation is to provide sufficient information to determine the

componentDescription: Basic modular design

assignedToProject:
- MyDataBase
- MyFirewall
- MyMotSense

hasEvidence:
- ADV_TDS_EAL_4

hasDependencies:
- ADV_FSP_4

hasD-element

Value	...
ADV_TDS.3.1D The developer shall ...	en
ADV_TDS.3.2D The developer shall ...	en

hasC-element

Value	...
ADV_TDS.3.1C The design shall des...	en
ADV_TDS.3.8C The design shall des...	en
ADV_TDS.3.4C The design shall prc...	en
ADV_TDS.3.3C The design shall ide...	en
ADV_TDS.3.10C The mapping shall ...	en
ADV_TDS.3.6C The design shall prc...	en
ADV_TDS.3.5C The design shall prc...	en
ADV_TDS.3.2C The design shall des...	en
ADV_TDS.3.9C The design shall des...	en
ADV_TDS.3.7C The design shall des...	en

componentLeveling

Value	...
The components in this family are ...	en

hasE-element

Value	L...
ADV_TDS.3.1E The evaluator sha...	en
ADV_TDS.3.2E The evaluator sha...	en

Fig. 3. The assurance component ontological representation within the Protégé environment [5] – an example concerning the ADV_TDS.3 component.

Please note that the ADV_TDS assurance family expresses implementation boundaries of particular security functions and shows how these functions implement the SFRs specified for them. Generally, this family defines the TOE decomposition into subsystems and modules and their relations with security functions. Some of these modules enforce SFRs, some are supporting only, some are non-interfering. The interfaces related to these subsystems and modules should be specified according to the `ADV_FSP_4` component referenced as dependencies.

On the left side, the Protégé “Class Browser” shows a part of the SMO ontology class hierarchy – the part of the security assurance requirements, i.e. their CC classes and families. Please note the numbers of instances within the brackets of the ADV class families. For the highlighted `ADV_TDS` assurance family (i.e. ontology class) its 6 instances are shown (see “Instance Browser” – in the middle): from `ADV_TDS_1` (for EAL2) to `ADV_TDS_6` (for EAL7). The right part of the window presents the `ADV_TDS_3` details, expressed by properties. For each SAR component a set of properties (`className`, `familyName`, `classIntroduction`, `componentDescription`, `familyAppNotes`, etc.) is defined and filled with the contents, drawn from the third part of the standard [1].

The key issue is that each SAR component consists of three kinds of elements [1]:

- D – “*Developers action element*”, expressing evidences which ought to be developed and delivered for evaluation together with the TOE,
- C – “*Contents and presentation of evidences*”, expressing the shape in which evidence ought to be provided,
- E – “*Evaluator action element*”, expressing actions allowing to evaluate if all artefacts have been delivered in the right form (this issue is extended by the Common Criteria Evaluation Methodology – not discussed here).

The ontology class representing the SAR component has the `hasD-element`, `hasC-element` and `hasE-element` properties. These properties are of the `SARComponent` domain and all have the range of the `string`. These elements are very important because they decide about the contents and shape of the evidences. Currently, they are represented by data-type (`string`) properties, however they cause some trouble in information retrieving. During further ontology development, when components become more structured, these elements can be expressed by instances, making the information retrieval easier. As it was mentioned earlier, many solutions are possible during the ontology development – the ontological representation of these component elements can be considered one of them.

Please note some other details concerning properties of `ADV_TDS_3`: the `assignedToProject` property value indicates to which projects a given ontology item is assigned, the `hasDependencies` property points to the mentioned above `ADV_FSP_4` (*ADV_FSP.4 Complete functional specification*) component, while the `hasEvidence` property points to the instance of the `FamilyEvidence` class: `ADV_TDS_EAL_4`. The *ADV_TDS.3* component is included from EAL4. `ADV_TDS_EAL_4` represents evidences sampled for this component dealing with EAL4, and implied by the D, C and E elements with respect to the given TOE (i.e. validated *MyFirewall* system).
□

5.4 SMO ontology facilities supporting the elaboration of elementary evidences items

Example 1 shows that for the given EAL evidences are composed with the assurance family evidences retrieved from the SMO knowledge base. Example 2 presents the sources of information used for these evidences elaboration. The key issue is how to elaborate these evidences. The straightforward use of information included in the components is necessary but not sufficient to elaborate the evidences. Generally, this is a time-consuming and difficult process, requiring specialized know-how, patterns and experience. Even though there are some guidelines about the subject [2], [13-14], the IT security developers have some trouble with the evidences work-out. The family evidences should consider many factors, for example: EAL, character of the developed IT products or systems, applied technologies, character of the development and manufacturing environments.

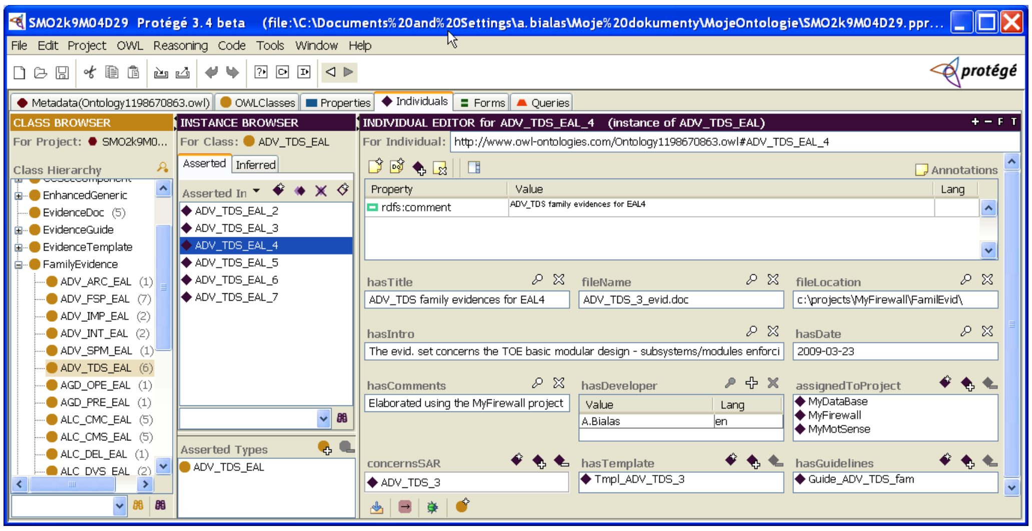
The SMO ontology provides facilities shaping and organizing the evidences as a whole, and supporting the evidences work-out. For every assurance component an appropriate evidence pattern (instance of the `EvidenceTemplate` subclass) is proposed, and for each assurance family – a guideline (instance of the

EvidenceGuide subclass), both taking into account details included in the earlier mentioned D-, C- and E elements of the SAR component. Patterns and guidelines have been elaborated with the use of IT security developers' know-how and the above mentioned guidelines, especially [14] – concerning the range: EAL1-EAL5, and have been placed into the SMO knowledge base. The patterns, guidelines and elaborated evidences can be managed on the knowledge base level. Some hints related to the methodology of the evidences elaboration are available from [15] but no patterns were included there.

During the elaboration of family evidences, i.e. evidences implied by their particular components, a right pattern and related guidelines should be used. The next example shows facilities introduced by the SMO ontology, supporting these activities.

Example 3. Elaborating family evidences on the pattern basis (Fig. 4, Fig. 5).

It was demonstrated that one of family evidences belonging to the *MyFirewall* evidences set, sampled for EAL4+, is the ADV_TDS_EAL_4 instance, which is implied by the D-, C-, E elements of the ADV_TDS_3 component.


SMO2k9M04D29 Protégé 3.4 beta (file:\C:\Documents%20and%20Settings\a.bialas\Moje%20dokumenty\MojeOntologie\SMO2k9M04D29.ppr...

File Edit Project OWL Reasoning Code Tools Window Help

Metadata(Ontology1198670863.owl) | OWLClasses | Properties | Individuals | Forms | Queries

CLASS BROWSER
For Project: SMO2k9M0...
Class Hierarchy
- EnhancedGeneric
- EvidenceDoc (5)
- EvidenceGuide
- EvidenceTemplate
- FamilyEvidence
  - ADV_ARC_EAL (1)
  - ADV_FSP_EAL (7)
  - ADV_IMP_EAL (2)
  - ADV_INT_EAL (2)
  - ADV_SPM_EAL (1)
  - ADV_TDS_EAL (6)
  - AGD_OPE_EAL (1)
  - AGD_PRE_EAL (1)
  - ALC_CMC_EAL (5)
  - ALC_CMS_EAL (5)
  - ALC_DEL_EAL (1)
  - ALC_DVS_EAL (2)

INSTANCE BROWSER
For Class: ADV_TDS_EAL
Asserted | Inferred
Asserted In
- ADV_TDS_EAL_2
- ADV_TDS_EAL_3
- ADV_TDS_EAL_4
- ADV_TDS_EAL_5
- ADV_TDS_EAL_6
- ADV_TDS_EAL_7

Asserted Types
- ADV_TDS_EAL

INDIVIDUAL EDITOR for ADV_TDS_EAL_4 (instance of ADV_TDS_EAL)
For Individual: http://www.owl-ontologies.com/Ontology1198670863.owl#ADV_TDS_EAL_4

Annotations

Property	Value	Lang
rdfs:comment	ADV_TDS family evidences for EAL4	

hasTitle: ADV_TDS family evidences for EAL4
fileName: ADV_TDS_3_evid.doc
fileLocation: c:\projects\MyFirewall\FamilEvid\
hasIntro: The evid. set concerns the TOE basic modular design - subsystems/modules enforci
hasDate: 2009-03-23
hasComments: Elaborated using the MyFirewall project

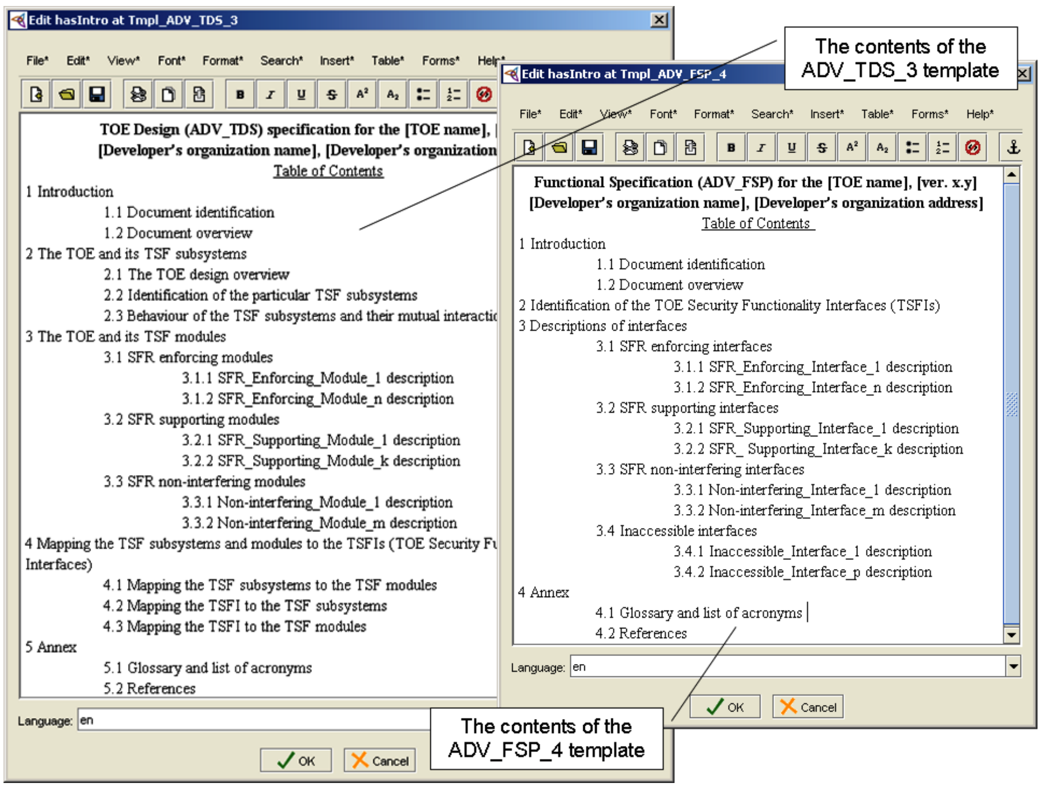
hasDeveloper

Value	Lang
A.Bialas	en

assignedToProject: MyDataBase, MyFirewall, MyMotSense
concernsSAR: ADV_TDS_3
hasTemplate: Tmpl_ADV_TDS_3
hasGuidelines: Guide_ADV_TDS_fam

Fig. 4. The elaboration of the family evidence (ADV_TDS_EAL4) with the use of the appropriate pattern and guidance within the Protégé environment [5].

Fig. 4 presents this instance in details using the Protégé "Individual Editor". Please note two properties: the first property, hasTemplate, points to the Tmpl_ADV_TDS_3 instance, containing the pattern of the "*ADV_TDS.3 Basic modular design*" component evidence; the second one, hasGuidelines, points to the Guide_ADV_TDS_fam instance, containing guidance how to elaborate evidences (here: ADV_TDS_EAL_4) for particular components of the ADV_TDS family on their patterns basis. Both instances, Tmpl_ADV_TDS_3 and Guide_ADV_TDS_fam, point to appropriate external documents, presenting respectively the pattern and guidance, assigned by properties. Their abstract (contents) versions are placed in the knowledge base. It is a temporary solution because editing large documents with the use of the current version of the Protégé environment creates some difficulties.


Edit hasIntro at Tmpl_ADV_TDS_3
File* Edit* View* Font* Format* Search* Insert* Table* Forms* Help*
Language: en
OK
Cancel
Edit hasIntro at Tmpl_ADV_FSP_4
File* Edit* View* Font* Format* Search* Insert* Table* Forms* Help*
Language: en
OK
Cancel
The contents of the ADV_TDS_3 template
The contents of the ADV_FSP_4 template

TOE Design (ADV_TDS) specification for the [TOE name], [
[Developer's organization name], [Developer's organization
Table of Contents
1 Introduction
1.1 Document identification
1.2 Document overview
2 The TOE and its TSF subsystems
2.1 The TOE design overview
2.2 Identification of the particular TSF subsystems
2.3 Behaviour of the TSF subsystems and their mutual interactic
3 The TOE and its TSF modules
3.1 SFR enforcing modules
3.1.1 SFR_Enforcing_Module_1 description
3.1.2 SFR_Enforcing_Module_n description
3.2 SFR supporting modules
3.2.1 SFR_Supporting_Module_1 description
3.2.2 SFR_Supporting_Module_k description
3.3 SFR non-interfering modules
3.3.1 Non-interfering_Module_1 description
3.3.2 Non-interfering_Module_m description
4 Mapping the TSF subsystems and modules to the TSFIs (TOE Security Fu
Interfaces)
4.1 Mapping the TSF subsystems to the TSF modules
4.2 Mapping the TSFI to the TSF subsystems
4.3 Mapping the TSFI to the TSF modules
5 Annex
5.1 Glossary and list of acronyms
5.2 References

Functional Specification (ADV_FSP) for the [TOE name], [ver. x.y]
[Developer's organization name], [Developer's organization address]
Table of Contents
1 Introduction
1.1 Document identification
1.2 Document overview
2 Identification of the TOE Security Functionality Interfaces (TSFIs)
3 Descriptions of interfaces
3.1 SFR enforcing interfaces
3.1.1 SFR_Enforcing_Interface_1 description
3.1.2 SFR_Enforcing_Interface_n description
3.2 SFR supporting interfaces
3.2.1 SFR_Supporting_Interface_1 description
3.2.2 SFR_ Supporting_Interface_k description
3.3 SFR non-interfering interfaces
3.3.1 Non-interfering_Interface_1 description
3.3.2 Non-interfering_Interface_m description
3.4 Inaccessible interfaces
3.4.1 Inaccessible_Interface_1 description
3.4.2 Inaccessible_Interface_p description
4 Annex
4.1 Glossary and list of acronyms
4.2 References

Fig. 5. Contents of templates (design patterns) used to family evidences work-out – two examples (ADV_TDS_3, ADV_FSP_4) shown in the Protégé environment [5].

Fig. 5 presents abstracts (contents) of two patterns as the examples.

□

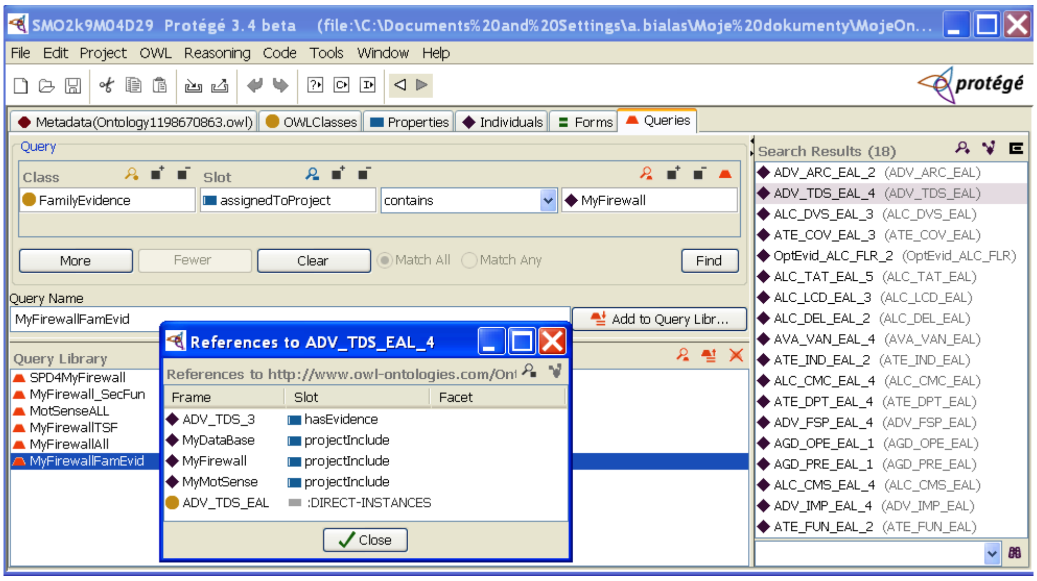
The patterns validation on real projects and the sampled experience allow to structure and formalize these patterns more deeply but reasonably and, finally, to develop an advanced knowledge base system supporting CC developers. The strategic objective of this work is to build a well-structured, multi-variant set of patterns, configurable according to the current IT product or system character, technology, development and manufacturing environments and EAL. A more advanced knowledge base means better possibility of knowledge management.

5.5 Query facilities retrieving information for development processes

The "Protégé Queries" simple facility can be used for information retrieving during the IT security development (Fig. 1) and TOE development (Fig. 6) processes. For the given class and property a query can be defined to find instances meeting the specified condition, which will be shown in the next example.

Example 4. Query operations on evidences.

The main window (Fig. 6) shows a simple query called “`MyFirewallFamEvid`” retrieving all evidence items implied by the components of EAL4+ used in the considered project, i.e. all instances of the `FamilyEvidence` class assigned to this project. On the right side, the query results, i.e. 18 items, are shown. Clicking the selected instance, e.g. `ADV_TDS_EAL_4`, the developer can perform some operations on it, e.g. he/she can see its references (shown in the small pop-up window), can display its details in “Protégé Individual Editor” (it is not shown there) or export the property value to the spreadsheet using the csv format.


SMO2k9M04D29 Protégé 3.4 beta (file:\C:\Documents%20and%20Settings\a.bialas\Moje%20dokumenty\MojeOn...
File Edit Project OWL Reasoning Code Tools Window Help
protégé
Metadata(Ontology1198670863.owl) OWLClasses Properties Individuals Forms Queries
Query
Class
Slot
More Fewer Clear Match All Match Any Find
Query Name
Add to Query Libr...
Query Library
Search Results (18)
References to ADV_TDS_EAL_4
References to http://www.owl-ontologies.com/Ont
Close

Query: Class FamilyEvidence; Slot assignedToProject; contains; MyFirewall

Query Name: MyFirewallFamEvid

Query Library:
- SPD4MyFirewall
- MyFirewall_SecFun
- MotSenseALL
- MyFirewallTSF
- MyFirewallAll
- MyFirewallFamEvid

Search Results (18):
- ADV_ARC_EAL_2 (ADV_ARC_EAL)
- ADV_TDS_EAL_4 (ADV_TDS_EAL)
- ALC_DVS_EAL_3 (ALC_DVS_EAL)
- ATE_COV_EAL_3 (ATE_COV_EAL)
- OptEvid_ALC_FLR_2 (OptEvid_ALC_FLR)
- ALC_TAT_EAL_5 (ALC_TAT_EAL)
- ALC_LCD_EAL_3 (ALC_LCD_EAL)
- ALC_DEL_EAL_2 (ALC_DEL_EAL)
- AVA_VAN_EAL_4 (AVA_VAN_EAL)
- ATE_IND_EAL_2 (ATE_IND_EAL)
- ALC_CMC_EAL_4 (ALC_CMC_EAL)
- ATE_DPT_EAL_4 (ATE_DPT_EAL)
- ADV_FSP_EAL_4 (ADV_FSP_EAL)
- AGD_OPE_EAL_1 (AGD_OPE_EAL)
- AGD_PRE_EAL_1 (AGD_PRE_EAL)
- ALC_CMS_EAL_4 (ALC_CMS_EAL)
- ADV_IMP_EAL_4 (ADV_IMP_EAL)
- ATE_FUN_EAL_2 (ATE_FUN_EAL)

Frame	Slot	Facet
ADV_TDS_3	hasEvidence	
MyDataBase	projectInclude	
MyFirewall	projectInclude	
MyMotSense	projectInclude	
ADV_TDS_EAL	:DIRECT-INSTANCES	

Fig. 6. Using the Protégé query facility [5] to retrieve family evidences for the *MyFirewall* system evaluated against EAL4+.

Single queries can be merged into more complex ones using AND/OR operations (“Match Any” / “Match All” tool options). The defined queries can be stored by the Protégé system (see the “Query Library” in the left bottom part).

□

Currently, the defined queries are rather simple. They allow to retrieve different specification means concerning the security target and evidences specification.

6. Conclusions

The paper shows how to apply the knowledge engineering methodology for the security engineering domain – more precisely – it concerns the ontological approach to the IT security development and implementation processes compliant with the Common Criteria standard.

It summarizes works on the Specification Means Ontology (SMO) development and uses it in these processes. The SMO has broader meaning, but the paper is focused on the evaluation evidences, showing how these evidences can be organized and used.

The SMO ontology elaboration, especially its part related to the evidences, was discussed in [3], but this paper extends and exemplifies those works by some issues encountered during the ontology validation on a simple firewall example:

- the specification of TOE security functions (TSF) at the claimed EAL as the output of the IT security development process, at the same time the input of the TOE development process;
- the composition of TOE evidences, i.e. how the evidences items of particular assurance families are sampled together for the TOE and a given EAL with the possibility to add and/or substitute SARs;
- the elaboration of a given assurance component evidence with the use of a template (design pattern) and guideline (methodology);
- retrieving from the knowledge base different information about ontological representation of the model.

The ontological approach to the CC evidences elaboration presented in the paper allows:

- to improve common understanding of the domain terms,
- to build a detailed ontological model of evidences and to express relationships between evidence items,
- to better structure and formalize evidences, facilitate the process of evidences issuing and management and, may be in the future, automate this process,
- to create design patterns (templates) of evidences for assurance families and EALs,
- to compose evidences from elementary items implied by SARs, following provided guidelines,
- to use the Protégé tool to compose the right evidences for a given EAL,
- to retrieve knowledge concerning evidences and other CC methodology issues.

The basic advantages and possibilities of the ontological approach were demonstrated and used to improve the CC-compliant IT development process, allowing its better formalization and preciseness, tool support, project knowledge management and reusability. Please note that for SMO all general objectives related to an ontology development [3-4] are here achieved:

1. The elaborated SMO ontology, in the same way as many other ontologies, enables to share common understanding of the structure of information among people (here: IT security engineers) and software (here: the Protégé Ontology Editor and Knowledge Acquisition System and other OWL-compatible). SMO supports common understanding of evidences and their elaboration as well as specification means: CC- defined and author's defined.
2. Ontologies facilitate the reuse of domain knowledge. In this case, the same evidence patterns and guidelines, specification means and artefacts from similar projects can be used for other projects. The SMO knowledge base can be considered a library of different kinds of predefined design patterns.

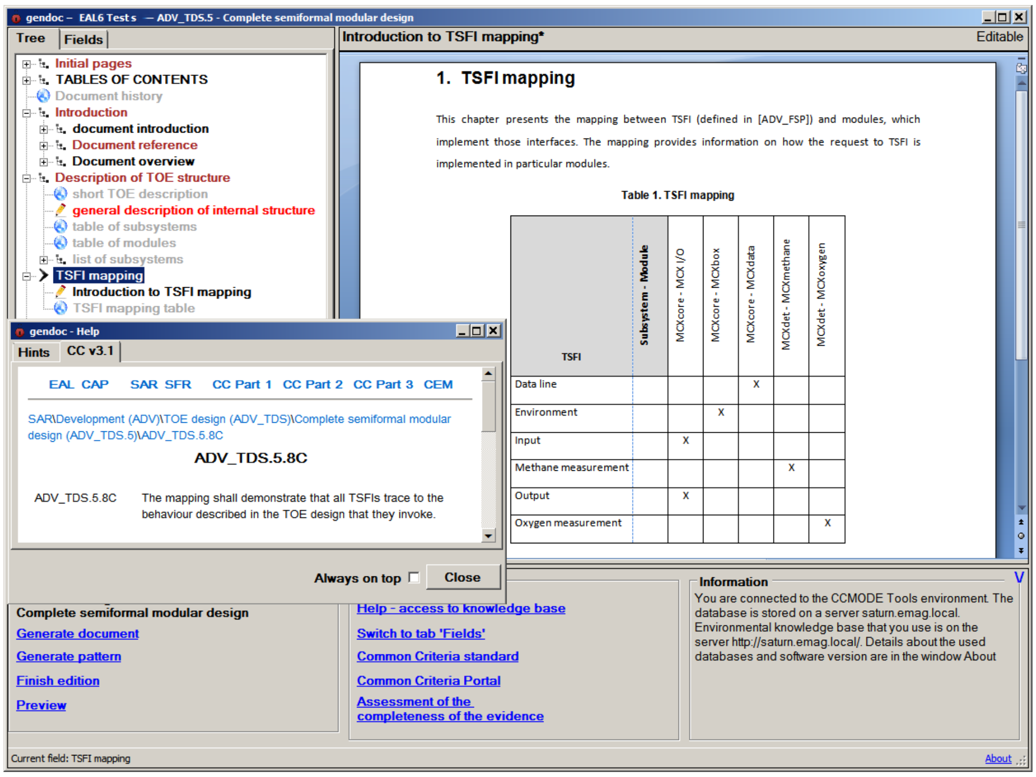
3. SMO allows to make explicit assumptions for a domain. It deals mainly with predefined relationships. For example, the family evidences are based on components, patterns and guidelines assigned for them, while the specification items have predefined mapping items.
4. SMO separates domain knowledge (items related to evidences, specification means as a whole, designed to use in many different projects) from operational knowledge (how to use this domain knowledge to compose a new security target specification or to elaborate new TOE evidences).
5. SMO, as many other ontologies, supports the analyses of the domain knowledge. For example, some variants of security solutions, evidences or relationships can be considered for use (trade-offs). A simple risk analysis can be performed and, based on its results, security objectives can be specified.

The ontology development is always an iterative and incremental process, beginning with the basic classes and properties. As more complicated classes, properties and restrictions are added, the ontology is getting more matured and is able to express sophisticated relationships and to get answers for advanced competency questions. It brings additional benefits to the ontology users. This implies future works.

During validations, SMO is iteratively extended, focusing on: deeper structurization, more design patterns, additional guidelines, and more enhanced ontology properties and restrictions. After revision and extension the discussed SMO is called now the IT Security Development Ontology (ITSDO).

From the practical point of view some problems were encountered. The Protégé is a good tool for the ontology development, editing and experimentation. It is useful to elaborate even complicated models of the data structures and relationships in the Common Criteria knowledge domain. It can be used by the ontology/knowledge base developers, but this tool is completely inconvenient for the Common Criteria methodology end-users (IT security developers, consultants, evaluators). The results of the presented works (data structurization, relationships, data forms, queries, functionality, etc.) can be very useful as conceptual models during the software tool development – tool supporting the CC-related processes. The Protégé, equipped with the elaborated knowledge base, is very useful for experimentation and the users requirement identification to create such a tool.

The results (models) of presented researches were used as the input for the CCMODE R&D Project (Common Criteria compliant, Modular, Open IT security Development Environment) carried out by the Institute of Innovative Technologies EMAG [16]. The objective of the project is to work out a methodology and tools to develop and manage development environments of IT security-enhanced products and systems for the purposes of their future certification. The SMO is a prototype of different solutions which have been elaborated in this project, e.g.: an SMO part related to enhanced generics is used in decision support related to the ST elaboration, a part related to the evidences is the basis for defining the patterns of evidences and their software implementation.



gendoc – EAL6 Tests – ADV_TDS.5 - Complete semiformal modular design

Tree | Fields

- Initial pages
- TABLES OF CONTENTS
- Document history
- Introduction
  - document introduction
  - Document reference
  - Document overview
- Description of TOE structure
  - short TOE description
  - general description of internal structure
  - table of subsystems
  - table of modules
  - list of subsystems
- TSFI mapping
  - Introduction to TSFI mapping
  - TSFI mapping table

Introduction to TSFI mapping* Editable

1. TSFI mapping

This chapter presents the mapping between TSFI (defined in [ADV_FSP]) and modules, which implement those interfaces. The mapping provides information on how the request to TSFI is implemented in particular modules.

Table 1. TSFI mapping

TSFI / Subsystem - Module	MCXcore - MCX I/O	MCXcore - MCXbox	MCXcore - MCXdata	MCXdet - MCXmethane	MCXdet - MCXoxygen
Data line			X		
Environment		X			
Input	X				
Methane measurement				X	
Output	X				
Oxygen measurement					X

gendoc - Help

Hints | CC v3.1

EAL CAP SAR SFR CC Part 1 CC Part 2 CC Part 3 CEM

SAR\Development (ADV)\TOE design (ADV_TDS)\Complete semiformal modular design (ADV_TDS.5)\ADV_TDS.5.8C

ADV_TDS.5.8C

ADV_TDS.5.8C The mapping shall demonstrate that all TSFIs trace to the behaviour described in the TOE design that they invoke.

Always on top ☐ Close

Complete semiformal modular design

Generate document

Generate pattern

Finish edition

Preview

Help - access to knowledge base

Switch to tab 'Fields'

Common Criteria standard

Common Criteria Portal

Assessment of the completeness of the evidence

Information

You are connected to the CCMODE Tools environment. The database is stored on a server saturn.emag.local. Environmental knowledge base that you use is on the server http://saturn.emag.local/. Details about the used databases and software version are in the window About

Current field: TSFI mapping About

Fig. 7. Evidence example elaborated on the SMO basis in the CCMODE project.

Fig. 7 presents the "*ADV_TDS.5 Complete semiformal modular design*" evidence (for EAL6) in the GenDoc application, which is one of the software components of the CCMODE Tools. The Gendoc, based on MS Word, is equipped with the specialized functionality and integrated with other software components and the knowledge base [17]. In the CCMODE Tools several ontologies were applied (not discussed here), but one of them, related to the evaluation evidences, is based directly on the R&D works presented in this paper (SMO). Referring to section 5.4, this ontology considers more details of evidences (in comparison with the `EvidenceTemplate` subclass), and instead of the common guide for the given assurance family (the `EvidenceGuide` subclass) it offers the enhanced, well structurized context help based on the Common Criteria details and providing hints and templates for fields (text-, tables-, lists patterns, etc.).

In the upper left corner of Fig. 7 the structure of the *ADV_TDS.5* evidence document is shown as the tree of fields. Some fields are filled in by the TOE project knowledge base data automatically, others are filled by the TOE developer with the help system use. The example of a field is shown in the upper right part of Fig. 7. It presents mapping of the TOE security functions interfaces (TSFIs) to the modules of the TOE (here: intelligent sensor MCX used in mines).

Acknowledgments

This work was conducted using the Protégé resource, which is supported by the grant LM007885 from the United States National Library of Medicine.

The results of the paper were used in the project CCMODE co-financed from EU Resources within European Regional Development Fund (UDA POIG 01.03.01.156/08).

References

[1] Common Criteria for IT security evaluation, part 1-3. v. 3.1. 2009.

[2] Common Criteria Portal. http://www.commoncriteriaportal.org/ Accessed Feb 2013.

[3] Bialas A.: *Ontology Based Model of the Common Criteria Evaluation Evidences*. Theoretical and Applied Informatics, ISSN 1896-5334, Vol. 25 (2013), no. 2, pp.69-91.

[4] Noy, N.F., McGuiness, D.L.: *Ontology Development 101: A Guide to Creating Your First Ontology*. In: Stanford Knowledge Syst. Lab. Tech. Rep. KSL-01-05 and Stanford Medical Informatics Tech. Rep. SMI-2001-0880. Stanford University, CA, 2001. http://www-ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness-abstract.html Accessed March 2013.

[5] Protégé Ontology Editor and Knowledge Acquisition System, v.3.4. Stanford University. http://protege.stanford.edu/ (2008). Accessed May 2008, March 2013.

[6] Bialas, A.: *Ontology-based Security Problem Definition and Solution for the Common Criteria Compliant Development Process*. In: Proceedings of 2009 Fourth International Conference on Dependability of Computer Systems (DepCoS-RELCOMEX 2009), pp. 3-10. ISBN 978-0-7695-3674-3, IEEE Computer Society, Los Alamitos; Washington; Tokyo.

[7] Bialas, A.: *Validation of the Specification Means Ontology on the Simple Firewall Case*. In: Arabnia H., Daimi K. (Eds.), Proc. of the 2009 Int. Conf. on Security and Management, (WORLDCOMP'09 – The 2009 World Congress in Computer Science, Computer Engineering, and Applied Computing), Vol. I, pp. 278-284. ISBN: 1-60132-124-4, 1-60132-125-2 (1-60132-126-0, CSREA Press, Las Vegas.

[8] Białas A., *Ontological approach to the motion sensor security development*. Electrical Review (Przegląd Elektrotechniczny), ISSN 0033-2097, vol. 85 (R.85), Number 11/2009, 36-44. Sigma-NOT, Warsaw.

[9] Bialas, A.: *Semiformal framework for ICT security development*. Presentation on the 8th International Common Criteria Conference, Rome, 25-27 September 2007. http://www.8iccc.com/index.php Accessed April 2009.

[10] Bialas, A.: *Semiformal Approach to the IT Security Development*. In: W. Zamojski, J. Mazurkiewicz, J. Sugier, T. Walkowiak (Eds.) Proc. of the Int. Conf. on Dependability of Computer Systems DepCoS-RELCOMEX 2007, pp. 3-11. ISBN 0-7695-2850-3, IEEE Computer Society, Los Alamitos; Washington; Tokyo 2007.

[11] Bialas, A.: *Semiformal Common Criteria Compliant IT Security Development Framework*. Studia Informatica vol. 29, Number 2B(77), Silesian University of Technology Press Gliwice 2008. http://www.znsi.aei.polsl.pl/ Accessed March 2013.

[12] Bialas, A.: *Ontology-based Approach to the Common Criteria Compliant IT Security Development*. In: H. Arabnia, S., Aissi, M., Bedworth (eds.) Proc. of the 2008 Int. Conf. on Security and Management, pp. 586-592. CSREA Press, Las Vegas.

[13] ISO/IEC TR 15446. Guide for the production of protection profiles and security targets, 2009.

[14] BSI Guidelines for Developer Documentation according to Common Criteria Version 3.1. Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2007.

[15] Higaki W.H.: *Successful Common Criteria Evaluation. A Practical Guide for Vendors*. Copyright 2010 by Wesley Hisao Higaki, Lexington, KY 2011.

[16] CCMODE (Common Criteria compliant, Modular, Open IT security Development Environment). Project co-financed from EU Resources within European Regional Development Fund (UDA POIG 01.03.01.156/08 http://www.commoncriteria.pl/. Accessed 20 March 2013.

[17] Białas A. (pod red.): *Komputerowe wspomaganie procesu rozwoju produktów informatycznych o podwyższonych wymaganiach bezpieczeństwa*, Wydawnictwo Instytutu Technik Innowacyjnych EMAG, sfinansowano ze środków UE POIG 1.3.1, Katowice 2012; (Bialas A. (Ed): *Computer-aided development of the high integrity IT products*, Institute of Innovative Technologies EMAG, Katowice, 2012, monograph in Polish).

Walidacja modelu materiału dowodowego do oceny zabezpieczeń według metodyki *Wspólne Kryteria* bazującego na ontologii

Streszczenie

Artykuł przedstawia wybrane zagadnienia dotyczące walidacji modelu materiału dowodowego służącego do oceny zabezpieczeń informatycznych, opracowanego w oparciu o metody inżynierii wiedzy i metodykę "Wspólne Kryteria" (ISO/IEC 15408 Common Criteria). Artykuł stanowi kontynuację artykułu [3], nakreślającego tło tematyki i wprowadzającego ontologię środków specyfikacji (*SMO – Specification Means Ontology*). Niniejszy artykuł poświęcono walidacji modelu danych opartego na SMO i związanego z metodyką Common Criteria.

Na wstępie artykuł zawiera krótkie informacje wprowadzające dotyczące: podejścia ontologicznego, rozważanej dziedziny wiedzy oraz pojęć wprowadzonych przez SMO. Główną część artykułu (Rozdział 5) poświęcono wybranym zagadnieniom dotyczącym wykorzystania ontologii SMO i związanej z nią bazy wiedzy – zagadnieniom napotkanym podczas walidacji rozwiązań na przykładzie projektu systemu zaporowego.

W sekcji 5.1 wskazano punkt startowy tworzenia materiału dowodowego dla produktu informatycznego, czyli przedmiotu oceny (*TOE – Target of Evaluation*). Punktem tym jest specyfikacja funkcji zabezpieczających zawarta w zadaniu zabezpieczeń (*ST – Security Target*). Funkcje te należy zaimplementować w produkcie informatycznym na zadanym poziomie uzasadnionego zaufania EAL (*Evaluation Assurance Level*), czyli opracować na odpowiednim poziomie szczegółowości materiał dowodowy. Jego wykaz dla systemu zaporowego *MyFirewall*/EAL4+ zawiera Tab.1.

Sekcja 5.2 ilustruje model materiału dowodowego implikowanego przez komponenty danego pakietu EAL – pokazuje organizację materiału jako całości z uwzględnieniem wzbogacenia i rozszerzenia pakietu EAL4 do EAL4+.

W sekcji 5.3 pokazano w szczegółach jak modelowany jest przykładowy komponent uzasadniający zaufanie (tu: *ADV_TDS.3*) wraz ze swoimi elementami D, C, E i jak to wpływa na odpowiadającym jemu materiał dowodowy (tu: `ADV_TDS_EAL4`). Kontynuując tę dyskusję, w sekcji 5.4, pokazano organizację materiału dowodowego. Został on zawarty w pliku `ADV_TDS_3_evid.doc`, opracowanym na podstawie szablonu `Tmpl_ADV_TDS_3` (Fig.5) według wytycznych zawartych w `Guide_ADV_TDS_fam`. W ten sposób wskazano związki miedzy wymaganiami każdego komponentu a implikowanym przez niego materiałem dowodowym. Dla każdego z nich podano wzorzec i wytyczne (wspólne dla rodziny komponentów), wskazując, że dalsza strukturyzacja danych na potrzeby tworzenia aplikacji użytkowych jest możliwa. W sekcji 5.5 pokazano prosty sposób wyszukiwania danych o modelu. W toku tworzenia ontologii i jej walidacji posługiwano się narzędziem Protégé.

Rozwiązano problem wzorców materiału dowodowego, tworząc dla nich szablony opisujące wymaganą zawartość i instrukcje ich stosowania, opisujące jak te szablony wypełnić treścią dotyczącą konstruowanego produktu informatycznego. Działania te mają na celu wyposażenie konstruktorów produktów informatycznych we wzorce, by oni sami, bez pomocy kosztownych konsultantów mogli tworzyć materiał dowodowy dla swoich produktów informatycznych.

Ontologię SMO wykorzystano jako model materiału dowodowego, implikowanego przez komponenty SAR należące dla poszczególnych pakietów EAL. Model ten, po poddaniu walidacji i rewizji stał się podstawą dla modelu danych komputerowego systemu wspomagania tworzenia materiału dowodowego opracowanego w ramach projektu [16]. W podsumowaniu pokazano docelową prezentację materiału dowodowego w oknie aplikacji GenDoc (Fig.7), powstałej ramach tego projektu i będącej jednym z narzędzi CCMODE Tools [17].