Ontological Mapping of Common Criteria's Security Assurance Requirements

Andreas Ekelhart, Stefan Fenz, Gernot Goluch and Edgar Weippl
Security Research – Secure Business Austria
Favoritenstrasse 16, 1040 Vienna
{aekelhart, sfenz, ggoluch}@securityresearch.at,
WWW home page: http://www.securityresearch.at

Abstract. The Common Criteria (CC) for Information Technology Security Evaluation provides comprehensive guidelines for the evaluation and certification of IT security regarding data security and data privacy. Due to the very complex and time-consuming certification process a lot of companies abstain from a CC certification. We created the CC Ontology tool, which is based on an ontological representation of the CC catalog, to support the evaluator at the certification process. Tasks such as the planning of an evaluation process, the review of relevant documents or the creating of reports are supported by the CC Ontology tool. With the development of this tool we reduce the time and costs needed to complete a certification.

1 Introduction

The Common Criteria for Information Technology Security Evaluation (CC) describes an international standard regarding the criteria for the evaluation and certification of IT products and systems pertaining to data security and data privacy. Requirements for the security functions of IT products and systems as well as requirements for assurance measures applied to the security functions during a security evaluation are provided by the CC [1]. The security functions of such products and systems and the applied assurance measures have to meet defined requirements to obtain a certain level of confidence during the evaluation process. With the results of the evaluation process the costumer should be able to estimate the actual security risks regarding the evaluated IT product or system. One of its major strengths is that the CC process offers a standardized approach for product and system evaluation. Raskin [17] is one of the first to introduce ontological semantic approaches to information security. He implies that one of the ultimate goals is the inclusion of natural language data sources to facilitate the specification and

evaluation of information security certification by organizing and unifying the terminology.

Despite being a standard, the CC offer the flexibility to certify only requirements that are important to the customer. *Protection Profiles* state the desired requirements of a particular community in almost any combination desired [12]. Other standards such as the Orange Book follow an "all-or-nothing" approach. If a product or system misses even a single and for the customer irrelevant requirement, it cannot be certified at the desired level. The CC's flexibility make it harder both for the developer and evaluator to keep track of what is required for a certain level and which security functions are to be included.

The drawback of such a comprehensive standard is the fact that it is very time-consuming and expensive to evaluate a certain product or system against a specific CC evaluation assurance level. Little commercial interest is driving the CC market; most evaluations and certifications result from government regulation or government purchase [8].

Katzke suggested several ways to deal with CC's problems and shortcomings [9]. The suggestions include better administration and management processes, long-term planning and budget processes, and accountability for meeting goals, milestones, and deliverables. We concentrate on the first of Katzke's points because sophisticated support tools for management and administration of CC processes are still not available; both the evaluator and on the developer would certainly benefit from such a tool. Furthermore the CC include rather abstract verbalizations: e.g. "ALC_DVS.1.1C: The development security documentation shall describe all the 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" [3]. Such abstract definitions do not provide sufficient information on the concrete measures a company has to fulfill and therefore often leads to conflicts during the evaluation process. To counter the abstract verbalizations, we align the CC ontology with the Security Ontology [4,5,6], and thus are able to offer concrete threat and countermeasure terminology for demanded security requirements.

We eliminated the aforementioned flaws by creating a CC ontology which comprises the entire CC domain [1][2][3]. In comparison to the available PDF or paper version of the CC, the ontology is easily browseable with any standard RDF [16] – or OWL [13] (ontology) visualization tool and thus easier to handle, especially pertaining to relationships. Furthermore, due to the OWL representation the CC domain is now available in a machine readable format and can be utilized in computer programs. Another important advantage of our approach is the option to query the data structure in an efficient way, taking advantage of the well known RDF- or OWL-based query languages such as SPARQL [18]. Due to the complexity of these languages it is nonetheless necessary to create an intermediate layer, which translates the user input into a valid query and thus the ontology has to be designed in a way that easy query transformations are feasible.

Following this, based on the CC ontology and the SPARQL query language stated above, we developed a tool to support the CC evaluation process in several ways.

Our contribution is:

- The *CC Ontology* covers the entire domain of the CC. It can be used to query the data structure in an efficient way using SPARQL.
- The *CC Ontology Tool* takes the CC ontology as input and supports the evaluation process in several novel and useful ways such as tagging and linking.

The previous pages already mentioned "ontology" as a central term in this paper. Even though ontologies are widely used in research of semantic systems, this subsection provides some definitions and defines the scope of an ontology in the context of this paper. The term ontology has its origin in the philosophical discipline, where it means a systematic explanation of being. One of the first ontology definitions regarding to the computer sciences sector, was published 1991 by Neches:

> 'An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations to define extensions to the vocabulary.' [11]

This definition shows already the basic elements of an ontology in the sense of computer sciences: a set of defined terms, their interrelations, rules for combining terms and the scope of the ontology. The main components of an ontology are: (1) Classes: represent concepts (e.g. trees, animals, …) (2) Relations: represent an association between concepts (3) Functions: special case of relations in which the n-th element of the relation is unique for the n-1 preceding elements [7] (4) Formal axioms: model sentences that are always true (5) Instances: represent elements or individuals in an ontology.

2 Common Criteria Ontology

A machine-readable ontology representing the CC is required for two reasons. First, users can easily navigate the ontology with a standardized tool and have a better overview of the entire process. Second, the ontology is the knowledge base upon which our CC ontology tools builds. This tool automatically configures the list of required certification documents and customizes the checklists to fit the specific needs of the certification process.

2.1 Common Criteria Terms and Definitions

The following list of CC terms and definitions explains the main terms used in the following chapters. For a complete list please refer to [1], Chapter 3.

- The *evaluation assurance level* is a set consisting of assurance components from CC security assurance requirements (compare [3]) representing a level on the predefined assurance scale.

- A *class* is a package of families, sharing a common focus. *Families* are sets of *components*, which are the smallest selectable group of elements, sharing security objectives.

- *Developer action elements* are activities which should be performed by the developer. *Content and presentation of evidence elements* encompass several required aspects of the evidence: (1) what the evidence should demonstrate; (2) what information the evidence should convey; (3) when the evidence is considered appropriate and (4) specific characteristics of the evidence that either the TOE or this assurance must possess [3]. *Evaluator action elements* are activities that should be performed by the evaluator, which explicitly include confirmation that the requirements prescribed in the content and presentation of evidence elements have been met.

2.2 The Common Criteria Security Assurance Ontology

The evaluator's view on the CC focuses on the security assurance requirements. We therefore concentrated on modeling them since they are used by the evaluator as a mandatory statement of evaluation criteria when determining the assurance of the TOE and when evaluating protection profiles and security targets. This information is clearly also of vital value for developers as a reference when interpreting statements of assurance requirements and determining assurance approaches for the TOE. We used Protégé [14] to maintain, visualize and navigate the ontology.

Table 1 shows all concept relations including their range and special characteristics. Due to the size of the CC security assurance ontology it is not possible to show the entire knowledge base. Following this we extracted relevant parts, which we are going to discuss in this chapter.

Table 1. Concept Relations

Domain	Relation	Range
Activity	evaluates_component	Component
Class	has_family	Family
Component	has_content_and_presentation_of_evidence_element	Content_and_Presentation_of_Evidence_Element
Component	has_developer_action_element	Developer_Action_Element
Component	has_input	Evidence_Element
Component	has_evaluator_action_element	Evaluator_Action_Element
Component	has_dependency	Component
Content_and_Presentation_of_Evidence_Element	has_workunit	Workunits
Developer_Action_Element	has_workunit	Workunits
EAL_Evaluation	has_activity	Activity
Evaluator_Action_Element	has_content_and_presentation_of_evidence_element	Content_and_Presentation_of_Evidence_Element

Evaluator_Action_Element	has_developer_action_element	Developer_Action_Element
Family	has_component	Component

With the ontology it is possible to reconstruct the complete CC security assurance evaluation process taking into account the used *evaluation assurance level* by the following relations (see Fig 1 for EAL4 and configuration management activity): (1) *has_activity* describes which activities are necessary for the needed evaluation assurance level (e.g. configuration management activity) and (2) *evaluates_component* defines which specific component has to be evaluated to comply to the evaluation activity (e.g. ACM_CAP.4 for configuration management activity on EAL4).

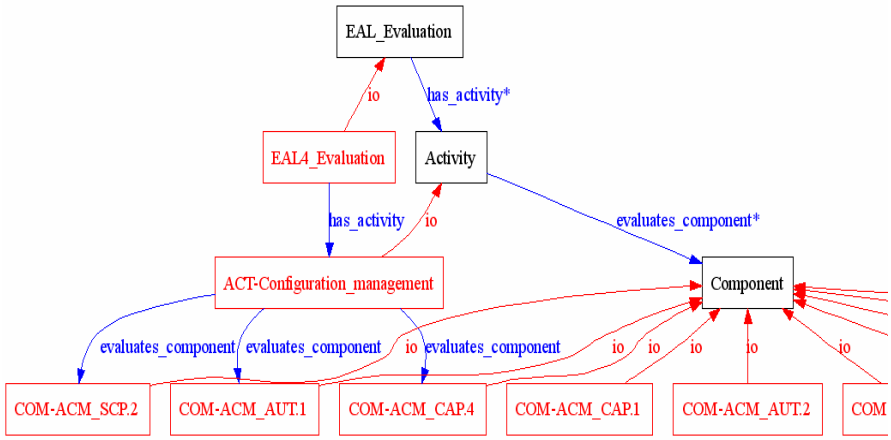

Fig 1. Evaluation Activities

Furthermore the dependencies and relationships between CC *classes*, *families* and *components*, including cross references of assurance component dependencies, are shown by the (1) *has_family* and (2) *has_component* relations (see Fig 2 for Class ACM, Family CAP and Component 4).

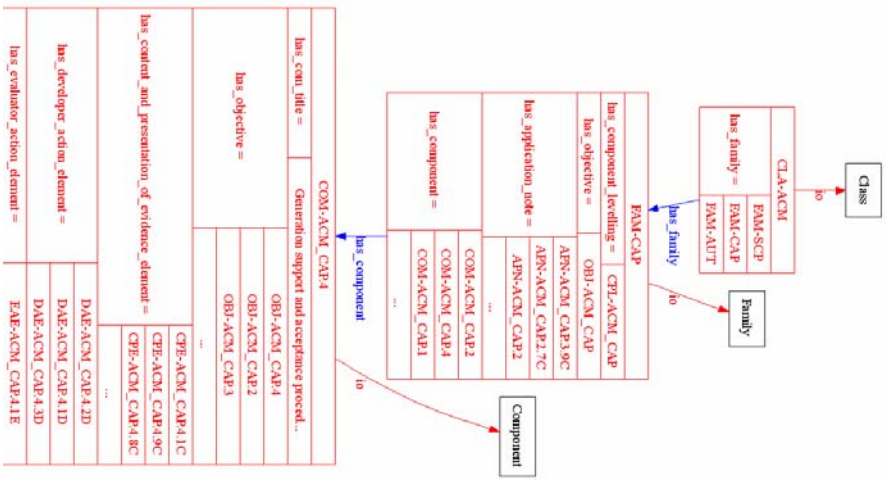

Fig 2. Class, Family, Component

Additionally every component is refined with the following relations and their corresponding items (1) *has_content_and_presentation_of_evidence_element* links to the corresponding *content and presentation of evidence elements* (2) *has_developer_action_element* links to the corresponding *developer action elements* and (3) *has_evaluator_action_element* links to the corresponding *evaluator action elements* (e.g. Component 4 in Family CAP from Class ACM [abbr.: ACM_CAP.4] has content and presentation of evidence Element 5C, developer action element 2D and evaluator action element 1E). *Workunit* elements, which optionally refine the elements stated above, are linked to content and presentation of evidence elements, developer action elements and evaluator action elements (e.g. Workunit ACM_CAP.4-8 is linked to content and presentation of evidence Element ACM_CAP.4.5C).

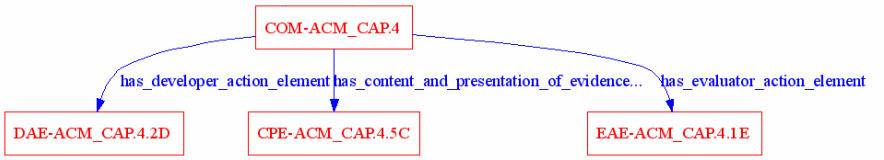

Fig 3. Class, Family, Component

Specific *evidence elements* are linked to their corresponding components through a *has_input* relationship (e.g. Component ACM_CAP.4 needs evidence input configuration management documentation and the TOE suitable for testing).

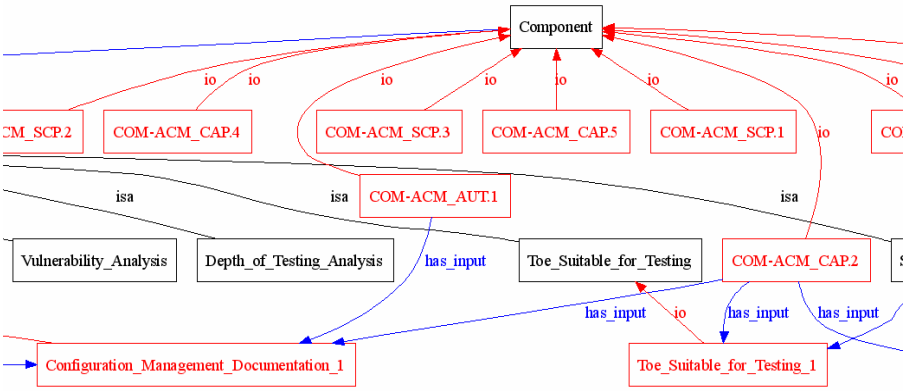

Fig 4. Evidence Elements

Using our ontology, the following knowledge about the aforementioned elements can be derived: E.g. evaluation assurance level 4 needs several security assurance activities [3]. Among them is the activity "configuration management". This activity evaluates specific components, such as Component 4 (generation support and acceptance procedures) in Family CAP (configuration management capabilities) from Class ACM (configuration management). Further drilling down shows that, this component has specific dependencies (e.g. to ALC_DVS.1: Identification of security measures), developer action elements (e.g. "The developer shall provide CM documentation."[3]), content and presentation of evidence elements (e.g. "The CM documentation shall include a configuration list, a CM plan, and an acceptance plan**"** [3]) and evaluator action elements (e.g. "The evaluator shall confirm that the information provided meets all requirements for content and presentation of evidence" [3]). These elements can be refined optionally by specific workunits (e.g. "The evaluator *shall examine* the configuration list to determine that it identifies the configuration items that comprise the TOE" [3]) for the content and presentation of evidence element stated above.

3 Common Criteria Ontology Tool

Based on the Common Criteria Ontology, introduced in Chapter 2, we created an evaluation tool to support the CC certification process. Figure 5 shows the main user interface. It enables the user to augment each certification sub-process with comments and the progress status. The tool is also useful to mitigate the "composition problem". These conflicts may arise when combining different protection profiles or security targets [10].

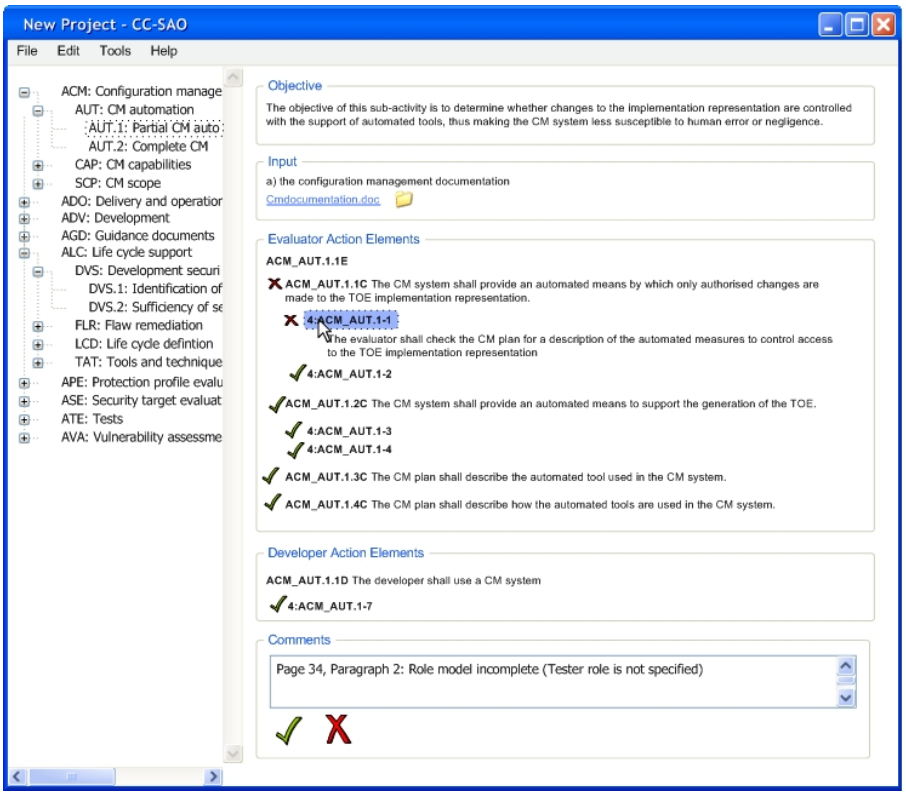

Fig 5. Common Criteria Ontology Tool – main user interface

3.1 Document Preparation and Linking

Besides the simple listing of all CC classes, including their families and concepts, the tool is able to filter only those components which are necessary for a certification against a specific EAL level. Due to the hiding of unnecessary components this feature eases the work for the evaluator and enhances the quality and efficiency of the certification process.

The bulk of preparatory work for a typical Common Criteria certification consists of checking documents against a certain target state required by the certification level; we thus implemented a feature to link relevant documents with certain components to enhance the clarity for the evaluator. Vetterling et al. [19] identified the increased need for documentation and the interdependencies between the documents as one of the major causes for the additional costs of a certification. Keeping all documents current and maintaining consistency can be achieved easier by using the previously described approach: (1) hiding of currently not relevant documents, and (2) linking of relevant documents.

The status of each developer and evaluator action element is indicated by a status symbol; the tool provides the evaluator with the option to augment component instances with comments to document the evaluation process. Using comments and linking relevant documents enables the user to generate reports including a history of the evaluation process. By storing the entire evaluation history it is possible to generate various report types, specifically tailored to different target groups. The tool creates a concise executive summary of the evaluation results and, in addition, a detailed comparison (including comments and progress status) of different evaluation process states. Such reports are invaluable in review cycles both for the evaluator and the developer.

So far we described a tool that supports the work for the evaluator by preparing, storing and organizing the evaluation data in a single repository. The "Tagging" approach (Subsection 3.1) enriches every component with relevant keywords to enhance the evaluation process.

3.2 The "Tagging" approach

By linking documents with the corresponding Common Criteria components and tagging each component with specific keywords we established the basis to support the evaluator in the document review.

COM-ACM_AUT.1	
has_com_title =	~#en Partial CM automation
has_objective =	OBJ-ACM_AUT.1
has_content_and_presentation_of_evidence_element =	CPE-ACM_AUT.1.2C
	CPE-ACM_AUT.1.3C
	CPE-ACM_AUT.1.1C
	CPE-ACM_AUT.1.4C
has_dependency =	COM-ACM_CAP.3
has_developer_action_element =	DAE-ACM_AUT.1.1D
	DAE-ACM_AUT.1.2D
has_input =	Configuration_Management_Documentation_1
has_evaluator_action_element =	EAE-ACM_AUT.1.1E
has_keyword =	~#en CVS
	~#en SVN
	~#en SourceSafe

Fig 6. Component instance with keyword tags

Based on our experience we believe that certain components and the corresponding documents often contain similar keywords and concepts. Figure 6 visualizes a typical component instance with its keywords and the input document reference. The keywords represent concepts which can be evidence for the compliance with the connected action elements. For example the keyword "CVS" in Figure 6, appearing in the Configuration Management Documentation, may be an indication that a configuration management system is used. Moreover, the text parts are visually

highlighted in the input document, so the evaluator can check the corresponding information.

Obviously the keywords have to be entered into the ontology; this process can be done either manually by the evaluator or automatically supported by the following approach: In our previous research work on security ontologies [4],[5],[6] we presented an ontology which comprises infrastructure components as well as security related concepts (threats and countermeasures). The goal of this work was to run a risk analysis and threat simulations against corporate assets. Our security ontology (i.e. a knowledge base) can be used to extract keywords for the Common Criteria Ontology by querying it with SPARQL [18].

In the following this approach is described in detail in connection with the Life Cycle Support Activity - Evaluation of Development security (ALC_DVS.1). One point of Action ALC_DVS.1.1E states that the evaluator shall search for security measures: “physical, for example physical access controls used to prevent unauthorized access to the TOE development environment (during normal working hours and at other times)”. Figure 7 shows an abridgement of biometric access control systems (Category sec:PhysicalThreatPrevention), taken from the Security Ontology.

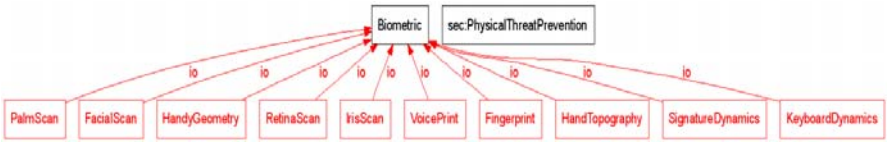

Fig 7. Biometric access control systems

To fill the Common Criteria automatically with keywords we have to query the ontology - in the following listing a SPARQL query is shown that lists all biometric system instances (the keywords we need):

```
SELECT ?biometricKeyword
WHERE { ?biometricKeyword rdf:type sec:Biometric}
```

An example will clarify our approach: if the evaluator wants to examine the Action Element “Action ALC_DVS.1.1E”, the system automatically searches for the keywords listed in Figure 7 (and corresponding notation variants) to present the evaluator evidence for biometric access control systems. Combining various keywords increases the hit possibility of the demanded section. On the other hand, if no keywords are found, it is not very likely that a biometric access control is in place.

The tagging approach has similar goals as the method proposed by Razzazi et. al [15], i.e. to speed up the entire process. In contrast, however, we do not propose to decompose the entire CC process into smaller subtasks as this tighter framework will impede experienced developers and evaluators.

Conclusion

To conquer the very time-consuming and expensive common criteria evaluation process for a specific CC evaluation assurance level, we first presented a CC ontology, comprising the entire CC domain with special focus on security assurance requirements relevant for the evaluation.

The ontology is easily browseable with any standard RDF or OWL visualization tool – unlike the already available PDF or paper version of the CC standard. Second, our approach provides the possibility to query the data structure in an efficient way using SPARQL.

Our third contribution is the CC Ontology Tool; this tool takes the CC ontology as input and supports the evaluation process in several novel ways such as tagging and linking.

Additionally several CC certifications showed us that certain components and the corresponding documents often contain similar keywords and concepts, hence we introduced the "Tagging" approach in our CC ontology, which supports the evaluator in the document review by reusing information produced in earlier CC evaluation certification processes.

Acknowledgment

This work was performed at the Research Center Secure Business Austria funded by the Federal Ministry of Economics and Labor of the Republic of Austria (BMWA) and the federal province of Vienna.

References

[1] Common Criteria for Information Technology Security Evaluation. Part 1: Introduction and general model, Version 2.3 (2005).
[2] Common Criteria for Information Technology Security Evaluation. Part 2: Security functional requirements, Version 2.3 (2005).
[3] Common Criteria for Information Technology Security Evaluation. Part 3: Security assurance requirements, Version 2.3 (2005).
[4] A. Ekelhart, S. Fenz, M. Klemen, and E. Weippl, Security Ontologies: Improving Quantitative Risk Analysis, in Proceedings HICCS (2007).
[5] A. Ekelhart, S. Fenz, M. Klemen, a. Tjoa, and E. Weippl, Ontology-based Business Knowledge for Simulating Threats to Corporate Assets, in Proceedings of the International Conference on Practical Aspects of Knowledge Management PAKM (2006), Springer Lecture Notes in Computer Science.
[6] S. Fenz, and E. Weippl, Ontology based IT-security planning, in IEEE Proceedings on IEEE International Symposium Pacific Rim Dependable Computing PRDC (2006).

[7] A. Gómez-Pérez, M. Fernández-López, and O. Corcho, *Ontological Engineering.* (Springer, London, first edition, 2004.)
[8] J. Hearn, Does the common criteria paradigm have a future? Security & Privacy Magazine, IEEE, 2:64–65 (2004)
[9] S. Katzke, The Common Criteria Years (1993–1998): Looking Back and Ahead. Presentation, 4th International Common Criteria Conference (2003).
[10] Keblawi, F.; Sullivan, D., "Applying the common criteria in systems engineering," Security & Privacy Magazine, IEEE , vol.4, no.2pp. 50- 55, March-April 2006
[11] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. Swartout, Enabling technology for knowledge sharing. AI Magazin, 12(3):36–56 (1991).
[12] Olthoff, K. G. 2000. A cursory examination of market forces driving the use of protection profiles. In Proceedings of the 1999 Workshop on New Security Paradigms (Caledon Hills, Ontario, Canada, September 22 - 24, 1999). NSPW '99. ACM Press, New York, NY, 61-66. DOI= http://doi.acm.org/10.1145/335169.335195
[13] Owl web ontology language. http://www.w3.org/TR/owl-features/ (2004).
[14] The protege ontology editor and knowledge acquisition system. http://protege.stanford.edu/ (2005).
[15] Razzazi, M.; Jafari, M.; Moradi, S.; Sharifipanah, H.; Damanafshan, M.; Fayazbakhsh, K.; Nickabadi, A., "Common Criteria Security Evaluation: A Time and Cost Effective Approach," Information and Communication Technologies, 2006. ICTTA '06. 2nd , vol.2, no.pp. 3287- 3292, 24-28 April 2006
[16] Resource description framework (rdf). www.w3.org/RDF/ (2006).
[17] Raskin, V., Hempelmann, C. F., Triezenberg, K. E., and Nirenburg, S. 2001. Ontology in information security: a useful theoretical foundation and methodological tool. In *Proceedings of the 2001 Workshop on New Security Paradigms* (Cloudcroft, New Mexico, September 10 - 13, 2001). NSPW '01. ACM Press, New York, NY, 53-59. DOI= http://doi.acm.org/10.1145/508171.508180
[18] Sparql query language for rdf. http://www.w3.org/TR/rdf-sparql-query/ (2006).
[19] Vetterling, M., Wimmel, G., and Wisspeintner, A. 2002. Secure systems development based on the common criteria: the PalME project. In Proceedings of the 10th ACM SIGSOFT Symposium on Foundations of Software Engineering (Charleston, South Carolina, USA, November 18 - 22, 2002). SIGSOFT '02/FSE-10. ACM Press, New York, NY, 129-138. DOI= http://doi.acm.org/10.1145/587051.587071