The CORAS Approach for Model-based Risk Management applied to e-Commerce Domain

Dimitris Raptis, Theo Dimitrakos, Bjørn Axel Gran and Ketil Stølen
INTRACOM, Greece, drap@intracom.gr: CLRC Rutherford Appleton Laboratory, UK, t.dimitrakos@rl.ac.uk: Institute for Energy Technology, Norway, bjornag@hrp.no: Sintef Telecom & Informatics, Norway, kst@sintef.no

Abstract: The CORAS project develops a practical framework for model-based risk management of security critical systems by exploiting the synthesis of risk analysis methods with semiformal specification methods, supported by an adaptable tool-integration platform. The framework is also accompanied by the CORAS process, which is a systems development process based on the integration of RUP and a standardised security risk management process, and it is supported by an XML-based tool-integration platform. The CORAS framework and process are being validated in extensive user trials in the areas of e-commerce and telemedicine. This paper presents an overview of the CORAS framework, emphasising on the modelling approach followed in the first of the user trials (concerning the authentication mechanism of an e-commerce platform) and it provides some examples of the risk analyses employed in this context.

Key words: Security, Risk Analysis, Modelling, e-Commerce.

1. INTRODUCTION

The emerging electronic services (e-services) in the areas of e-commerce, e-health, telemedicine and e-government impose new and increasingly demanding requirements to the underlying infrastructure. A proper understanding of the limitations of the existing infrastructures is an important prerequisite for designing new services with a satisfying degree of security. An improved methodology for risk management is a necessary first step towards verifying and/or improving the security of such systems.

The issues that risk management needs to address are the adequacy of the deployed security mechanisms to meet the application specific security requirements. One of the goals of CORAS is to incorporate suitable risk assessment techniques that address the security requirements of a developed system into appropriate phases of object-oriented software development processes. To this end, systematic model-based risk analysis can help development in two ways:

– Validate that no security aspects are overlooked in a systems' design.
– Provide feedback to refine the security requirements or improve the security mechanisms.

The CORAS project aims to develop an integrated framework for model-based risk analysis of security critical systems. Extensive trials are performed to validate the applicability and effectiveness of the framework in the e-commerce and telemedicine domains. The first trial, using the initial version of CORAS framework, was based on the user authentication mechanism of an e-commerce platform.

CORAS is a European R&D project partially funded by the 5th Framework Programme (FP5) on Information Society Technologies (IST). The CORAS consortium consists of 11 partners from industry, research and academia, from four European countries: three industrial partners, Telenor (NO), Intracom (GR) and Solinet (D), seven research institutes IFE (NO), NR (NO), SINTEF (NO), NCT (NO), RAL (UK), CTI (GR) and FORTH (GR), and an academic partner: QMW college, U. of London (UK).

Section 2 presents an overview of the CORAS model-based risk analysis framework for security critical systems. Section 3 presents some results and experiences of applying the framework on the user authentication mechanism of an e-commerce platform. Conclusions are presented in Section 4.

2. THE CORAS FRAMEWORK

2.1 Objectives

The overall objective for the CORAS project is to provide an integrated methodology to aid the design of secure systems by:

a) Developing a practical framework for a precise, unambiguous and efficient risk analysis, by exploiting the synthesis of risk analysis methods with semiformal specification methods (in particular, methods for object oriented modelling) and computerised tools, in order to improve the risk analysis of security critical systems. The framework will be supported by an adaptable integration platform based on data and information exchange

between risk analysis and modelling tools. The emphasis of this synthesis is on

- adapting an appropriate combination of risk analysis methods to the security critical systems;
- using (semi-)formal modelling techniques in order to isolate the important aspects and obtain an suitable overview of complex security critical systems;

b) Assessing the applicability, usability and efficiency of this framework by extensive experimentation in the fields of e-commerce and telemedicine;

Although the scope of CORAS is security-critical systems in general, it places particular emphasis on information security defined broadly by[1]:

- Confidentiality: ensuring that only appropriate access is allowed to data, both from inside or outside the organisation;
- Integrity: ensuring that no unauthorised changes are made to data – either in storage or transmission;
- Availability: ensuring that data is accessible as required;
- Accountability: ensuring that users are accountable for their security-relevant actions.

2.2 Overview

The main result of CORAS is a tool-supported framework for model-based risk assessment. It is model-based in the sense that it gives detailed recommendations for the use of UML modelling in conjunction with risk assessment. In fact, it employs modelling technology for three main purposes:

1. To describe the target of assessment at the right level of abstraction.
2. As a medium for communication and interaction between different groups of stakeholders involved in risk assessment.
3. To document risk assessment results and the assumptions on which these results depend.

As illustrated *Figure 1*, the CORAS framework has four main anchor-points.

[1] Non-repudiation is also a security concern of cryptographic mechanisms.

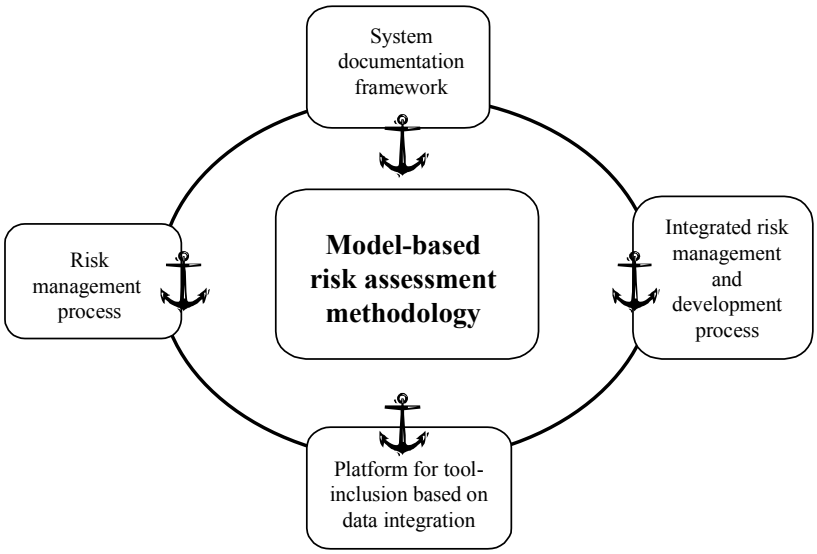

Figure 1. The CORAS framework for model-based risk analysis

The CORAS system documentation framework is based on the ISO/IEC 10746 standard "Basic Reference Model for Open Distributed Processing" (RM-ODP) [8] using UML [12] as modelling notation. RM-ODP defines a reference model for distributed systems architecture, based on object-oriented techniques. The CORAS documentation framework is further elaborated in [4].

The CORAS risk management process is based on ISO 17799 standard "Code of Practise for Information Security Management" [10] as it is complemented by ISO 13335 standard "Guidelines for the Management of IT Security" [9] and Australian standard AS/NZS 4360 "Risk Management" [1].

The CORAS integrated risk management and development process is based on an adaptation of RUP [12] integrating the AS/NZS 4360 risk management process [1] and supporting RM-ODP [8] inspired viewpoint oriented modelling.

The CORAS platform for tool integration will be based on data integration implemented in terms of XML technology. The platform will be built around an internal data representation formalised in XML [15] and in particular XMI [13] for UML models. Standard XML tools will provide much of the basic functionality.

2.3 Risk Assessment

The CORAS risk assessment methodology is build on HazOp analysis [14], Fault Tree Analysis (FTA) [11], Failure Mode and Effect Criticality Analysis (FMECA) [5], as well as CRAMM [2]. Below is a brief description of these methods:

A Hazard and Operability (HazOp) analysis is a systematic study of how deviations from the design specifications in a system can arise, and whether these deviations can result in hazards. The analysis is performed using a set of guidewords and attributes. In general terms, a HazOp analysis is performed as a kind of "brain storming" activity. An analysis team is gathered, consisting of different experts, and headed by a HazOp leader. In its nature, HazOp can make use of almost any kind of information types.

Failure Mode Effect (and Criticality) Analysis (FMEA or FMECA) is an analysis that concentrates on the potential failure modes of individual components. The basis of the FMECA is functional description of the system to be analysed in terms of its components. For each of the components in the system, the aim is to identify all possible or potential modes of failure and classify them according to their criticality. Each potential failure is ranked by the criticality of its effect in order that appropriate corrective actions may be taken to eliminate or control high-risk items. FMECA is usually carried out progressively in two parts. The first part identifies failure modes and their effects. The second part ranks failure modes according to the combination of criticality and the probability of that failure mode occurring. FMECA is a "bottom-up" approach, especially suited to examine all conceivable failure modes and to determine their consequences.

A Fault Tree is a logical diagram, which shows the relation between system failure, i.e. specific undesirable events in the system, and causes that lead to these events. Fault Tree Analysis (FTA) is a method based on deductive logic. First, an undesirable event is defined, and then causal relationships of the failures leading to that event are identified. Fault tree analysis is the most important and most frequently used of the methods available to quantify system performance. It provides not only a mean for system quantification but also a diagram representation of the causes of system failure, which is ideal for communicating the failure relationships.

The British Government's Central Computer and Tele-communications Agency (CCTA) Risk Analysis and Management Methodology (CRAMM), aims to provide a structured and consistent approach to computer security management for all ICT systems. The UK government considers CRAMM to be the standard for the risk analysis of information systems. CRAMM consists of three stages: asset valuation, assessment of threats and

vulnerabilities and considering suggested counter-measures. The final tailoring of the security package should include a balance of physical, personnel, procedural and technical security countermeasures. All these decisions should be recorded in the CRAMM software for later review. Where some countermeasures are not implemented this implies that some elements of perceived risk are not covered. The software also allows a series of "what if" questions to be answered with respect to planned changes. In contrast to CORAS however, CRAMM has been angled to support structured systems analysis and design, whereas CORAS aims for object-oriented analysis and design processes. The role of risk analysis for security concerns in systems development is reviewed extensively in [3].

3. TRIALS

The application of CORAS framework aims at object-oriented systems that are at developing stage but it is equally applicable for developed or maintained systems, like the e-commerce platform.

The e-commerce platform was developed in the context of the R&D project EP-27046-ACTIVE, co-funded by the European Commission under the ESPRIT programme. ACTIVE introduced a generic global Electronic Commerce platform [6] based on Java and the Internet, that supports integrated retail services, providing an intelligent interface upon which the involved players (retailers, suppliers and consumers) can interact. The e-commerce platform used in the CORAS trials constitutes a core part of the ACTIVE system.

In the e-commerce platform, users need be authenticated in order to access the personalised interface or preferences, like shopping lists. Technically, this is not a trivial issue as various alternative approaches have different trade-offs and several implementation pitfalls [7]. In the first trial, the user authentication mechanism used by the e-commerce platform was analysed.

In this section, the modelling approach used to express in UML the user authentication mechanism deployed by the e-commerce platform is presented, and then, examples of the risk analyses performed on the modelled functionality following the CORAS approach are described.

3.1 Modelling

The specification of a system's behaviour can be expressed using UML diagrams like State and Sequence diagrams. In particular, the overall behaviour of a Web application like the e-Commerce platform can be

described as a statechart where each state corresponds to a specific HTML page. The exact data presented in the page are abstracted away. Events correspond to links to other HTML pages in the application. The submission of HTML forms and links that pass parameters to the server correspond to events that carry parameters (the form fields, or the link's parameters). Using the same conventions for events (activation of HTML links), specific interactions of users with the application are expressed as sequence diagrams.

In this section the behaviour of the e-Commerce platform with respect to the user-authentication mechanism will be used to present two examples demonstrating this modelling approach.

3.1.1 Dynamic Behaviour Example

Using the modelling approach presented above, the state machine in *Figure 2* provides a high level description of the E-Commerce platform behaviour with respect to the user authentication and identification.

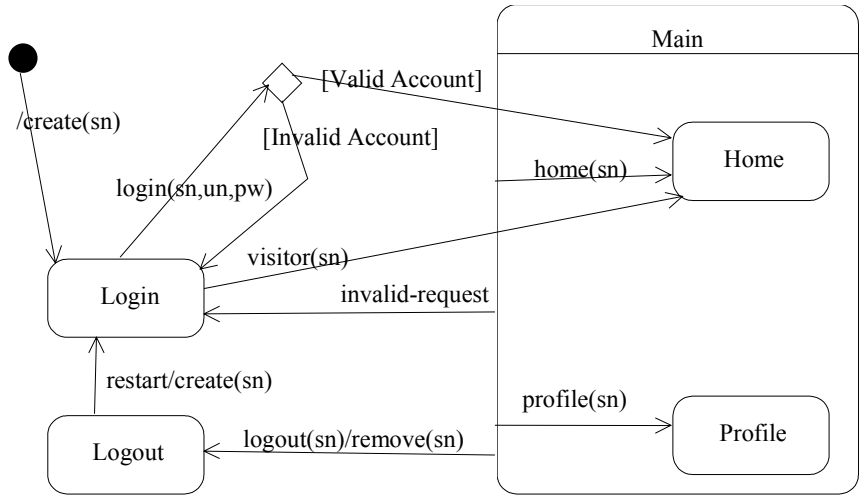

Figure 2. e-Commerce User Authentication behaviour

All HTML pages of the E-Commerce platform (like most web applications) are created using a specific HTML template. The only pages were the template is not applied are the "Login" and "Logout" pages that appear before and after the login and logout. This template contains links to major functions of the application. The superstate "Main" includes the states that can be reached directly from the template. These states will normally include more states corresponding to pages reached from internal links.

The actual behaviour of the E-Commerce platform template has much more states but, for the shake of brevity, only those pertinent to login and registration are shown here:

- The superstate "Main" contains one state for each link in the template of the HTML pages, although only two of them are shown here.
- The state "Profile" is actually a superstate with one state for each HTML page corresponding to categories of user data (i.e., “Registration”, “Preferences”, “Interests”, etc.) but it is included here as a simple state for simplicity because the users can change their password in the initial page.

According to the above diagram, when a user accesses the Login page, the server creates a unique *session ID* to identify the specific client. The session ID is used to associate each user’s client with the user’s data stored on the server. This session ID is sent to the user’s client in all subsequent HTML pages: All HTML links contain the session ID as a parameter. In the state machine above the session ID is denoted as the parameter “(sn)”. The login even caries also the username and password as a parameter. Users can also access the platform as visitors without authentication but there are not able to use all functionality like shopping lists.

The server stores the session numbers of all users, both registered (that have been authenticated) and visitors, and associates the registered ones with their corresponding data. A session terminates when a user logs out. In that case, the session number used to identify the session is removed from the database and the user is presented with the “Logout” page.

Invalid requests are those where the parameters were modified with invalid values. In these cases, the server responds with the Login page. Following some time without interaction with the server, a session times-out and the corresponding user is logged off (this holds for visitors as well). The use of time-outs prevents the use of bookmarks for accessing specific pages in the platform.

The use of session numbers for client identification has some repercussions in the behaviour of the application. For example, a session does *not* terminate immediately when a client disconnects. Users can bookmark a specific application page, exit their browser, then restart the browser and go to the specific page via the bookmark before the timeout period expires.

3.1.2 Scenario Description Example

The use of session numbers for client identification can have undesirable consequences if a malicious actor captures a client’s session ID. This actor can use the session ID to login in the platform using a second account with

an offensive profile (e.g., vulgar names) using the legitimate user's session ID. From that point on, all interactions of the legitimate user with the platform will have the profile of the second account.

This behaviour can be expressed as the sequence diagram in *Figure 3*.

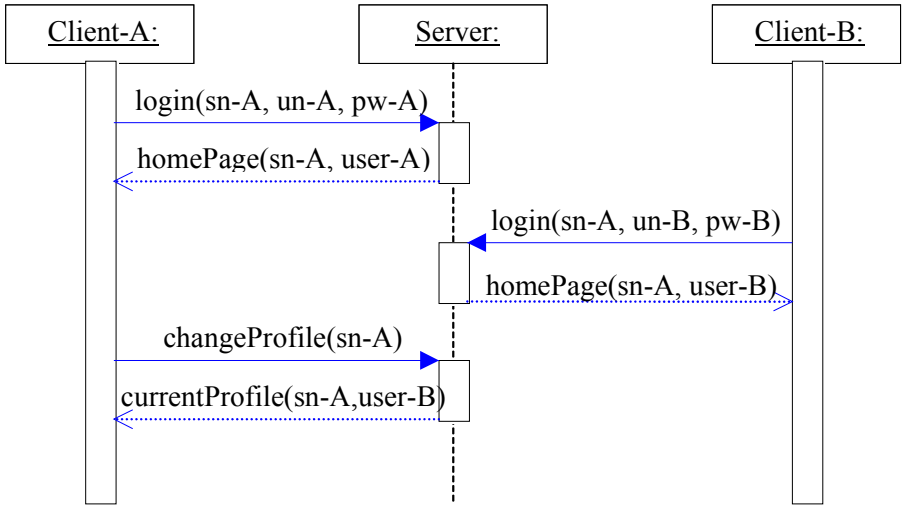

Figure 3. Session hijacking scenario

A legitimate user, Client-A, logs in the platform with the username/password `un-A/pw-A`, using session ID `sn-A`, and receives back a page personalised to Client-A `user-A`. After a malicious user, Client-B, logs in with the session ID of Client-A, the profile that Client-A accesses is the profile of Client-B (e.g., the Client-B's name and shopping lists).

This example is not the result of implementation defects, but a consequence of the design decision to use session IDs for client identification. It should be noted that the use of cookies is subject to similar deficiencies [7].

3.2 Risk Analysis

The risk analysis of the user authentication mechanism deployed by the e-commerce platform was based on models of its behaviour like those presented above. Initially CRAMM was applied in order to provide an identification of assets, which in turn provide a basis and justification for the security requirements that the mechanism need to meet. Then HazOp, FMEA and FTA was performed, examples or which are presented below.

The objective of HazOp, is to identify possible deviations from expected behaviour, as well as their causes and consequences. The expected behaviour was described as a statechart, like the one presented in *Figure 2*. Using this model, each interaction with the system (event) was systematically analysed. As an example, an excerpt of HazOp applied on a user's request to access the Login page ("/create(sn)" event) is presented in *Figure 4* and described below.

No.	Entity	Description	Security attribute	Deviation	Causes	Consequences	Actions	Remarks
1	/create (sn)	A user requests to access the Login Page. Server creates a new session number (SN)						
1.1			Disclosure					
1.1.1				User request captured	Openess of Internet	Not exploitable	N/A	No confidential information transmitted
1.1.2				Server response captured	Openness of Internet	SN revealed to capturer	No encryption justified	Deliberate session hijacking is possible
1.2			Manipulation					
1.2.1				A browser or proxy responds with a cached page	Browser or proxy (mis)configuration	User gets a page with invalid SN	N/A	The Login page will returned in the following client request
1.2.2						User gets a SN used by another user	Use large numbers for SN	Inadvertent session hijacking
1.3			Denial / Delay					
1.3.1				User request is blocked by proxy server	Proxy configuration	Server is not accessed	N/A	The server is not accessed
1.3.2				Server response is too slow				Generic deviation
1.4			Unaccountability					
1.4.1				Artificially large number of requests are generated	Deliberate server attack	(1) Creation of too many SNs (2) Server performance degradation	Block access based on client's IP address	Sensitive issue for SN-based user identification

Figure 4. The HazOp table for accessing the Login page

In the HazOp table above, the first column, Entities, correspond to the events of the system behaviour followed by a brief informal description. The Security attributes correspond to possible breach of security requirements (as expressed in Section 2.1) of Confidentiality, Integrity Availability and Accountability. The deviations column presents deviations from normal or expected behaviour, like undesirable (accidental or malicious) interactions with the system. The next columns presents possible causes that enable or cause the deviations, and the consequences of these deviations. The Actions

column presents some steps that can be taken to avoid or mitigate the risk of the deviation to occur. Some Remarks are presented in the last column. It should be noted that, in CORAS, there is no standard template for presenting the HazOp results. In particular, the Security attributes, used as guidewords to aid the identification of security violations, may also be customised depending on the aspects of the application that are assessed.

As can be seen from the above table, items 1.1.2 and 1.2.2 can contribute to the generation of the session hijacking scenario presented in section 3.1.2.

FMEA is used to identify possible *failure modes* of individual components. For software systems, like the e-commerce platform, these failures can be wrong results, non-termination, and exceptions or error values returned by function calls to software components. During the trials, the e-Commerce platform was modelled using UML component diagrams. Then each component was analysed, identifying the specific failure modes of the component. Due to the large size of modern software systems the FMEA table may become very large and time consuming to produce. The CORAS trial therefore focused only on small parts of high-level components, like the Web, Application and Database servers of the e-commerce platform. However significant parts of the results were generic and therefore they can be reusable.

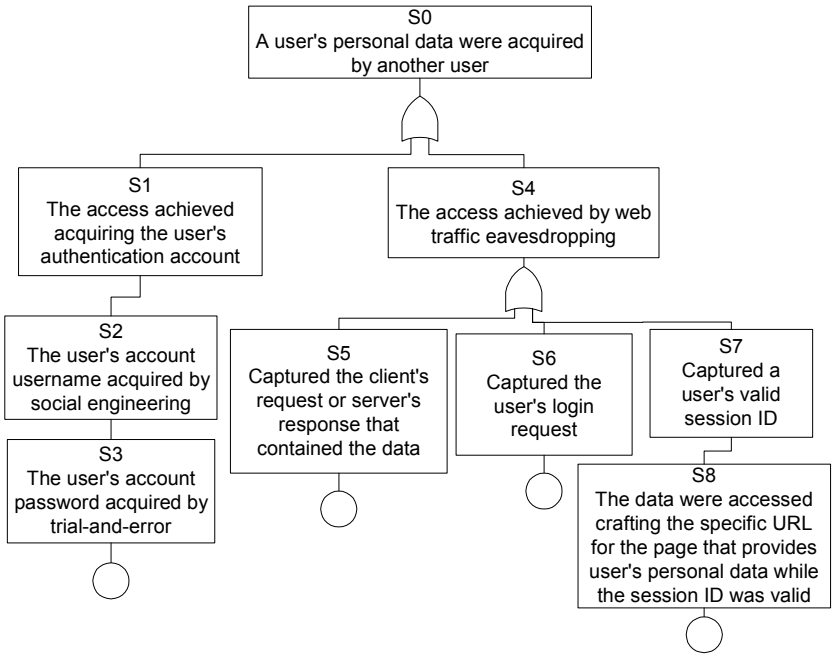

Figure 5. A Fault Tree example for capturing confidential data

The objective of Fault Tree Analysis is to document in a structured way the possible routes that can lead to the violations of security requirements identified by HazOp or failures identified by FMEA. As an example, an excerpt of a Fault Tree demonstrating some possible routes that lead to breach confidentiality by accessing a user's personal data in the e-commerce platform is presented in *Figure 5*.

The nodes of a fault tree are called *event blocks*, and the root node called *top-event*. The *OR-gates* join alternative means that can lead to their parent nodes. The round circles indicate that the parent events are *basic events* that are not analysed further. In this example, the tree is a branch of larger tree that covers all range of violations of security requirements. There are also more situations resulting in state S0, like circumventing the web server or internal fraud, but these are not presented here.

There is a close relation between the deviations identified by HazOp analysis and the possible component failures identified by FMEA with the fault tree constructed in that these deviations or failures appear as nodes ("event blocks") in the fault tree. For example, item 1.1.2 of HazOp table in *Figure 4* identified that a capture of a server response leads to the disclosure of Session ID. This is reflected in FTA tree in *Figure 5* where state S4 can be achieved by means of reaching state S7. The tighter integration between the complementary risk analysis methods deployed is the following step in the development of the CORAS framework for model-based risk analysis of security critical applications.

4. CONCLUSION

The CORAS framework for model-based risk analysis of security critical systems offers a structured and systematic approach to identify and assess security issues of ICT systems. The framework is based on modern object-oriented approaches to develop, express and document the systems structure and behaviour.

Extensive trials in the e-commerce and telemedicine domain are performed in order to assess the applicability and effectiveness of the framework and provide insight and feedback for further improvements.

The user authentication mechanism of an e-commerce platform was selected and used for the first trial. Despite the simplicity of the functionality and the wealth of available information on the issue, considerable issues were identified. These issues need be further addressed in the cases where the authentication of users over the Web is critical.

The work presented here is still in progress. Following further developments of the CORAS framework, the secure payment mechanism

and the use of agents for automatic transactions with the e-commerce platform will be analysed.

ACKNOWLEDGEMENTS

The authors are grateful to Rune Fredriksen and Eva Skipenes for their valuable comments and corrections. The CORAS project is partially funded by the European Commission under the FP5 Information Society Technologies Programme (IST) by Contract no. IST-2000-25031.

REFERENCES

[1] Australian/New Zealand Standard AS/NZS 4360:1999: *Risk Management*.
[2] Barber, B., Davey, J. *The use of the CCTA risk analysis and management methodology CRAMM*. Proc. MEDINFO92, North Holland, 1589 –1593, 1992.
[3] R. Baskerville, *Information Systems Security Design Methods: Implications for Information Systems Development*, ACM Computing Surveys, Vol. 25, No 4, Dec. 1993, pp. 375-414.
[4] den Braber, F., Dimitrakos, T., Gran, B.A., Stølen K., Aagedal, J.Ø. *Model-based Risk Management using UML and RUP*, Issues and Trends of Information Technology Management in Contemporary Organizations 2002, Information Resources Management Association International Conference, May 2002. (To appear).
[5] Bouti, A., Ait Kadi, D. *A state-of-the-art review of FMEA/FMECA*. International Journal of Reliability, Quality and Safety Engineering 1:515-543, 1994.
[6] EP-27046-ACTIVE, *Final Prototype and User Manual*, D4.2.2, Ver. 2.0, 2001-02-22.
[7] K. Fu, E. Sit, K. Smith and N. Feamster, *Dos and Don't of Client Authentication on the Web*, MIT Technical Report 818, MIT Laboratory for Computer Science, 2001. http://cookies.lcs.mit.edu/webauth:tr.pdf
[8] ISO/IEC 10746 series: 1995 Basic reference model for open distributed processing.
[9] ISO/IEC TR 13335-1:2001: Information technology – Guidelines for the management of IT Security – Part 1: Concepts and models for IT Security.
[10] ISO/IEC 17799: 2000 Information technology – Code of practise for information security management.
[11] IEC 1025: 1990 Fault tree analysis (FTA).
[12] Krutchten, P. *The Rational unified process, an introduction*. Addison-Wesley, 1999.
[12] OMG, *Unified Modeling Language (UML) Specification*, Ver. 1.3, Mar. 2000.
[13] OMG, *XML Metadata Interchange (XMI) Specification*, Ver. 1.1, Nov. 2000.
[14] Redmill, F., Chudleigh, M., Catmur, J. *Hazop and Software Hazop*. Wiley, 1999.
[15] World Wide Web Consortium, *Extensible Markup Language (XML) v1.0*, W3C Recommendation, Second Edition, 6 Oct. 2000.