Model based Security Risk Analysis for Web Applications: The CORAS approach

T. Dimitrakos, B. Ritchie
Central Laboratory of the Research Councils (CLRC-RAL), UK.
t.dimitrakos@rl.ac.uk, b.ritchie@rl.ac.uk

D. Raptis
Intracom S.A, Greece.
drap@intracom.gr

K. Stølen
SINTEF Group, Norway.
ketil.stoelen@sintef.no

Security evaluation and security assurance are important aspects of trust in e-business. CORAS is a European project which is developing a tool-supported framework for precise, unambiguous, and efficient risk assessment of security critical systems. The framework is obtained through adapting, refining, extending, and combining methods for risk analysis of critical systems and semiformal modelling methods. In this paper we provide an overview of the CORAS framework for model-based risk assessment, emphasising its application on Web-enabled B2C e-commerce services and the meta-data based deployment model underpinning the CORAS extensible platform for tool inclusion.

1. INTRODUCTION

The increasing complexity of today's systems urges the improvement of existing methods of analysing systems and their specification in order to reduce the likelihood that important threats remain unidentified. Such an improvement can be achieved by combining different risk analysis methodologies with respect to the system architecture. For example, qualitative methodologies for analysing risk lack the ability to account for the dependencies between events, but are effective in identifying potential hazards and failures within the system, whereas tree-based techniques take into consideration the dependencies between each event. We are not aware of an already developed integrated approach to system modelling and risk analysis, where the architecture expressed in the information system model is used to guide the combined application of risk analysis techniques. This need is being addressed in the European project CORAS 0 for the area of security risk assessment. In this paper we provide an overview of the results obtained in the ongoing CORAS project emphasising the pursued integration of risk management and semiformal modelling throughout the evolution of an iterative system development process. We conclude by summarising the achievements of CORAS and indicating how the project results can be used in order to improve trust and confidence in information systems for e-business and e-government.

CORAS provides a tool supported framework for precise, unambiguous, and efficient risk assessment of security-critical systems. CORAS focuses on the tight integration of viewpoint-oriented UML modelling in the risk management process. In this context, CORAS emphasises the practical use of UML and UP 0 in the context of security and risk assessment, and provides support for integrity, availability, accountability, authenticity, and reliability of IT systems.

The main output of the CORAS project is a framework for model-based risk assessment having four anchor points:

1 A risk management process based on the AS/NZS 4360 [1] standard.

2 A risk documentation framework based on the ISO standard RM-ODP [16].

3 An integrated risk management and development process based on UP [20].

4 A platform for tool-inclusion based on XML [27].

The CORAS consortium consists of three commercial companies: Intracom (Greece), Solinet (Germany) and Telenor (Norway); seven research institutes: CLRC/RAL (UK), CTI (Greece), FORTH (Greece), IFE (Norway), NCT (Norway), NR (Norway), and SINTEF (Norway); as well as one university college: Queen Mary University of London (UK). CORAS runs from January 2001 to July 2003.

2. MODEL-BASED RISK ASSESSMENT

We use the term "risk assessment" in order to refer to the combination of the systematic processes for risk identification and determination of their consequences, and for how to deal with these risks. Many risk assessment methodologies exist, focussing on different types of risks or different areas of concern. The CORAS risk assessment methodology builds on:

- HAZard and OPerability study (HAZOP) [23];
- Fault Tree Analysis (FTA) [19];
- Failure Mode and Effect Criticality Analysis (FMECA) [5];
- Markov analysis methods (Markov) [21];
- CCTA Risk Analysis and Management Methodology (CRAMM) [3].

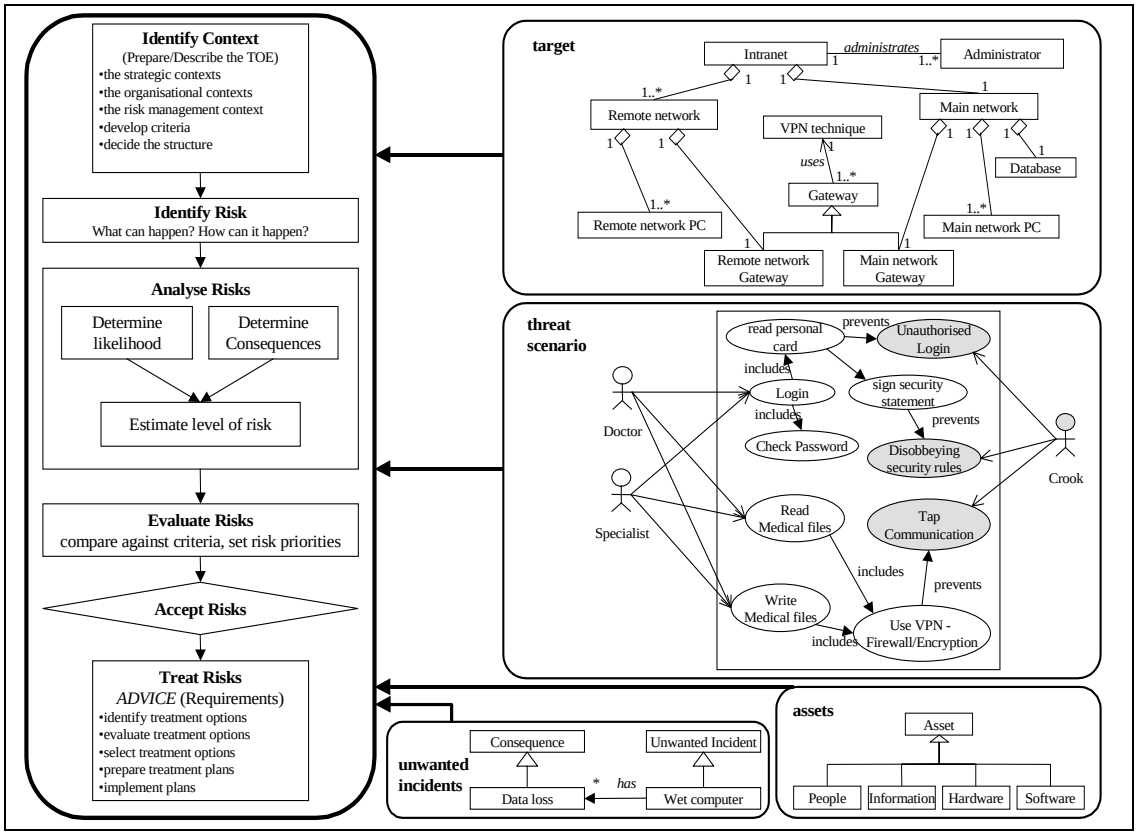

FIGURE 1: The role of UML in the CORAS risk management process

These methods are to a large extent complementary. They also cover all phases in the system development and maintenance process.

The CORAS risk assessment methodology incorporates a documentation framework, a number of integrated risk analysis techniques and a risk management process based upon widely accepted standards. It gives detailed recommendations for the use of UML-oriented modelling in conjunction with risk assessment.

The CORAS risk management process is based on the following standards: AS/NZS 4360:1999 "Risk Management" [1] and ISO/IEC 17799-1:1999 "Code of Practice for Information Security Management" [18].

AS/NZS 4360 provides a sequencing of the risk management process into sub-processes for context identification, risks identification, risks analysis, risks evaluation, and risks treatment (as illustrated in Figure 1). The CORAS methodology provides guidelines about which models are best suited for each sub-process, and how they should be expressed. We use this process to position models within risk assessment. AS/NZS 4360 also provides two parallel processes aiming at the communication and consultation, and at the monitoring and review of each of the sequential sub-processes, hence giving rise to an iterative risk management process. The correlation between the risk management iterations and the system design and development process is elaborated in the following section.

The CORAS system documentation framework is a specialisation of the Reference Model for Open Distributed Processing (RM-ODP). Whilst incorporating RM-ODP as a whole, the CORAS system documentation framework refines only those parts of RM-ODP that are directly relevant for risk assessment of security critical systems. This is achieved by introducing two new classes of terminology - concepts for risk assessment and security - and an additional structuring to the viewpoints – dividing the RM-ODP viewpoint structure into cross-viewpoint concerns targeting model-based security risk assessment.

These concerns may be understood as more specialised cross-viewpoint perspectives linking together related information within the five viewpoints. The concerns are further decomposed into models. A model provides the content of a concern with respect to a particular viewpoint. For each model there are guidelines for its development, including recommendations of which modelling languages to use. A number of identified concerns have also been related to the five sequential sub-processes of the CORAS risk assessment process (Figure 2).

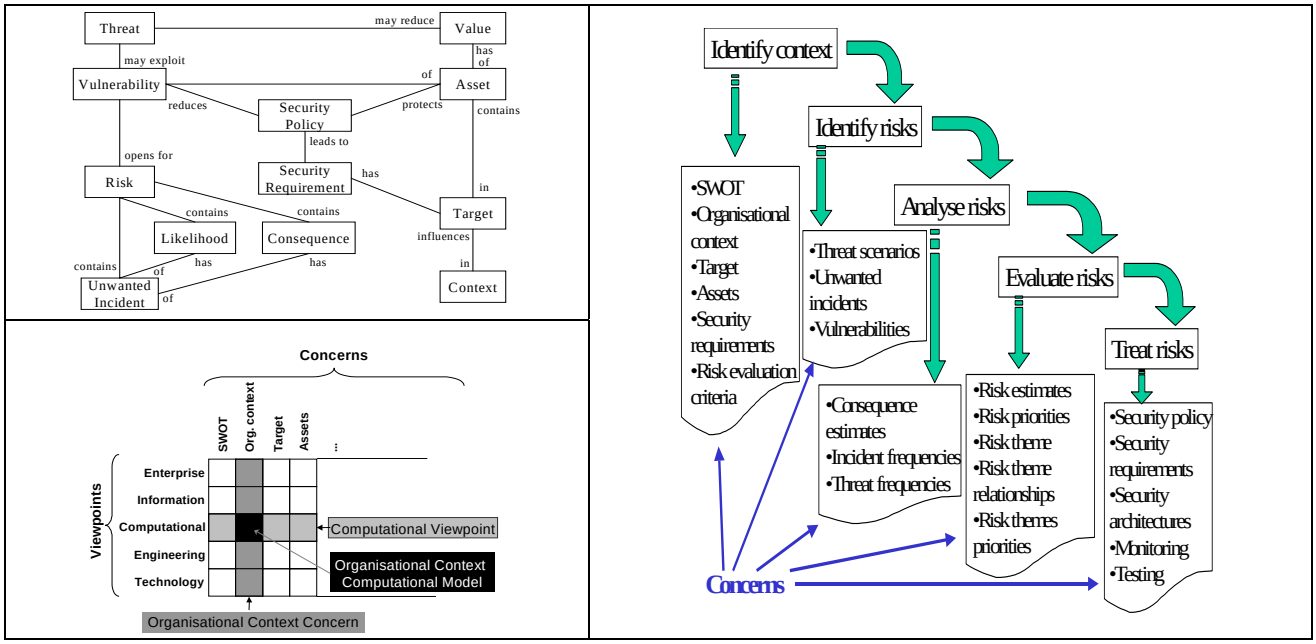

FIGURE 2: Anticlockwise: Basic concepts for Security Risk Assessment; Concerns, viewpoints and models; Concerns and risk management sub-processes.

Figure 1 provides an example of relating modelling concepts to the risk management process. The four diagrams to the right of Figure 1 illustrate:

- specification of the target of evaluation with the help of a UML class diagram (aspect of the *target concern* depicted in Figure 2);
- specification of a threat scenario with the help of a misuse case diagram 0 (example element of the *threat scenarios concern* listed In Figure 2);
- specification of the assets to be protected with the help of a UML class diagram (aspect of the *assets concern* listed in Figure 2);

- specification of an unwanted incident with the help of a UML class diagram (example element of the *unwanted incidents concern* in Figure 2).

The CORAS risk management process follows the main iterations made in the CORAS system development process, as indicated by Figure 3. Each main iteration adds more detail to the target and the context of the assessment and previous results may need to be re-evaluated.

3. THE INTEGRATED RISK MANAGEMENT AND SYSTEM DEVELOPMENT PROCESS

The CORAS integrated risk management and system development process is based on an integration of the risk management process in UP supported by UML-based viewpoint-oriented modelling.

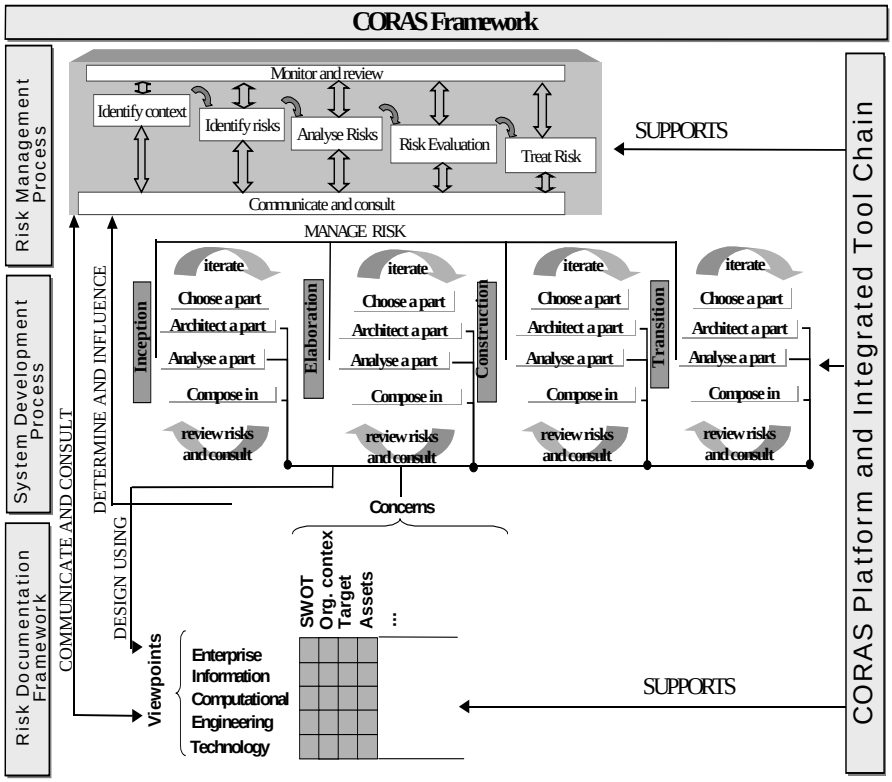

FIGURE 3: The integrated risk management and system development process

In the following paragraphs we highlight the defining characteristics of the CORAS process that are summarised in the visualisation seen in Figure 3.

An incremental iterative process. In analogy to RUP, the CORAS process is *both* stepwise *incremental and iterative*. In each phase of the system lifecycle, sufficiently refined versions of the system (or its model) are constructed through subsequent iterations before progressing to the next phase. In analogy to the RM-ODP viewpoints, the viewpoints of the CORAS framework are not layered; they are different abstractions of the same system focusing on different areas of concern. Therefore, information in all viewpoints may be relevant to all phases of the lifecycle. The CORAS risk management process follows the iterations made in the CORAS process. As more detail is added to target and the context of the analysis In each iteration, previous results may need to be re-evaluated.

Multiple risk management instances. The description of the CORAS risk management process (section 2) defines an abstract pattern that is instantiated using different risk analysis methods in order to analyse different

parts of the system in different phases of the development lifecycle. The choice of method depends on the target and the context of the analysis.

Propagation of risk management instances. A single component of the system model, in one level of abstraction, may correspond to a whole subsystem, in another.[1] This combination of refinement and decomposition inevitably results in the propagation of the risk analysis activity from the abstraction of the composite to its actual components.

Interactions between risk assessment and system design. As more information is incorporated in the context of the analysis there is always the potential that new vulnerabilities are identified (because of the choice of implementation or refinement) and this must be taken into account to the risk analysis of the composite entity. Consequently the risk management process (which includes risk analysis specific sub-processes) follows the iterations of the development lifecycle. While modelling sets the context for risk analysis and enforces revisions by refining or changing this context, security risk analysis contributes to selection of what needs to be refined and when. Security risk management guides refinement (in conjunction with cost management, deployment deadlines and implementation concerns).

The close integration between security risk management and modelling underpinning the CORAS development process gives rise to iterations of system design that are instigated by the risk assessment activities and vice versa. In addition to the usual iterations prescribed by the employed risk management and system development processes, we distinguish the following types of iterations that are due to interactions between risk management and systems design:

1 The iteration of the whole risk management process instigated by the change of development phase.
2 The iterations of the RA tasks instigated by refinement or decomposition of the system models. The behavioural or architectural structuring detail that is introduced by refinement or decomposition may allow the identification of new vulnerabilities or necessitate an update of the consequences of an already identified risk or lead to a revision of the risk evaluation results.
3 The iterations of the system design and development steps (within the same development phase) instigated by the need of the risk analyst for more information about a system aspect, which may result in a refinement or decomposition of some system models.
4 The iterations of the system design and development steps addressing design or implementation revisions that may be necessitated by the assessment of unacceptably high risks, due to bad design choices.

Iterations of type 2 take place for example when the description of a system component is refined to a subsystem. The RA results on the component that offers an abstract view of the subsystem both constrain the RA results of the components in the subsystem while the RA results on the later may give rise to design revisions therefore instigating a design iteration within the same development phase (iteration of type 4). Also when type 3 iteration happens, it instigates a refinement of the design at hand which may necessitate a further evaluation and perhaps making of design decisions. As such type 3 may bring about type 2 resulting in a propagation of parallel risk assessment sessions.

Another consideration with respect to improving systems security that is being addressed by the CORAS approach is to assess security policies and recommend how a system should be used in order to minimise loss or damage in the presence of identified vulnerabilities. We expect that the CORAS framework can support incorporating changes to the way the system is used or operates into the system documentation and control guidelines, targeting at the improvement of the system's security. This expectation is supported by the fact that RM-ODP allows for modelling policies and control entities in the system as well as describing policies and stakeholder roles in the enterprise context. Having established a correspondence between enterprise policies and the operational security management of the system, one may use model-based risk analysis, not only to assess the effectiveness of a security policy but also to decide the way in which a policy is enforced. (Policies can be enforced in a preventive, reactive or corrective manner, depending on the security risks they address weigh against implementation cost of their enforcement, the impact on the operational efficiency of the system.)

4. THE PLATFORM FOR TOOL INCLUSION

The CORAS platform for tool inclusion is built around internal data representations expressed in XML [27]. The CORAS platform consists of three interfaces for XML based data exchange:

[1] This is formalised using the ODP foundation rules for abstraction and composition.

– Interface based on IDMEF (Intrusion Detection Exchange Format) [12]. IDMEF is an XML DTD targeting tools for intrusion detection and has been developed by the Intrusion Detection Working Group.
– Interface based on XMI (XML Metadata Interchange) standardised by the Object Management Group and targeting tools for UML modelling.
– Interface targeting risk assessment tools.

The CORAS platform (Figure 4) contains a repository divided into two parts, which follow the organisation into viewpoints and concerns outlined in Figure 2:

(1) The assessment repository storing the concrete results from already completed assessments and assessments in progress.
(2) The reusable modelling elements repository storing reusable models, patterns and templates from pre-defined or already completed risk assessments.

At present the deployment model depicted in Figure 4 is under construction.

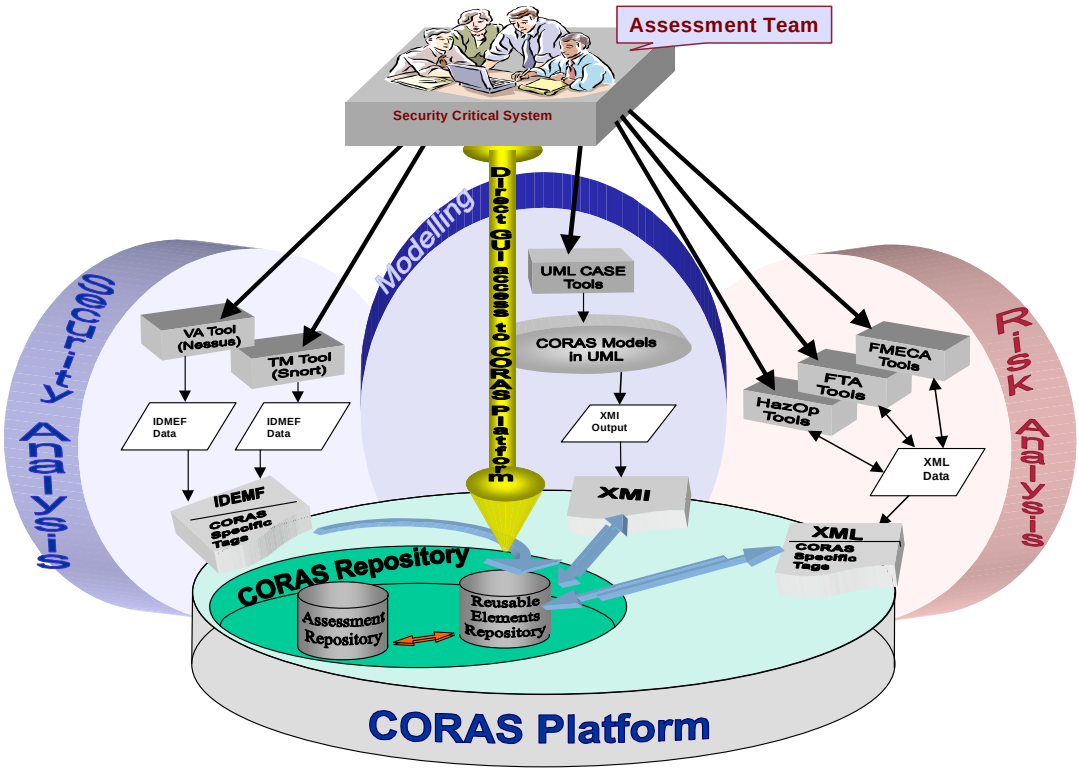

FIGURE 4: Overview of the CORAS tool inclusion architecture

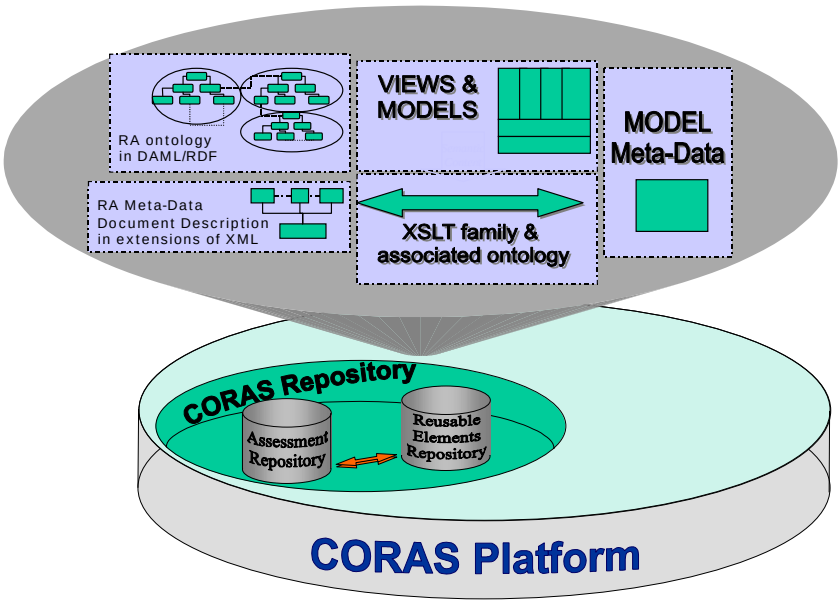

FIGURE 5: Overview of an extension of the CORAS tool inclusion framework achieving data-oriented tool integration

Following the completion of the CORAS project, we plan to extend this tool inclusion platform in order to achieve in-depth data-oriented tool integration. Building on top of XSLT [6], we would like to support mapping relevant aspects of the internal data representation of one class of tools to the internal data representations of other tools. This extension of the current deployment model is summarised in Figure 5. This will effectively allow communicating relevant data between sophisticated case-tools targeting system development, risk assessment tools and tools for vulnerability and threat management. At present, in the context of CORAS, we are developing XML-schemata defining information models for the different risk analysis methods used in CORAS.

5. A CASE STUDY

The CORAS framework and process are being validated in extensive user trials in the areas of e-commerce and telemedicine. In this section we summarise the e-commerce trials and provide some indicative examples of the modelling approach and the risk analyses employed in this context.

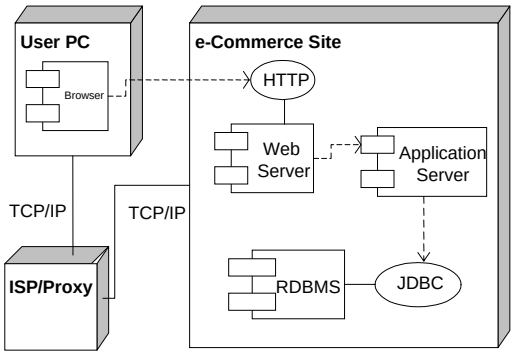

FIGURE 6: An overview of the Web enabled B2C commerce platform that is used in the CORAS e-commerce trials.

The target platform of the e-commerce trials was developed in the context of the R&D project EP-27046-ACTIVE, partly funded by the European Commission under the ESPRIT programme. ACTIVE introduced a generic global e-commerce platform 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. The deployment architecture of the e-commerce platform is summarised in Figure 6. This is essentially a three-tier architecture, where the tree main nodes represent a client (e.g. user with browser), the service enabling infrastructure (e.g. intermediaries of Web traffic over TCP/IP) and the server node providing the core B2C functionality and data.

The e-commerce trials focus on the security aspects of three separate functionalities of the target platform (presented in increasing complexity):

- The user authentication mechanism of the e-commerce platform was analysed in the first e-commerce trial and this case study is based on the results of this trial.
- The Secure Payment mechanism, which is based on the use of digital certificates is analysed in the second e-commerce trial, which is currently in progress.
- The use of autonomous agents for purchases is going to be analysed in the third e-commerce trial, which will take place in the first quarter of 2003.

For public versions of the deliverables referring to results from the CORAS trials, please refer to the CORAS project web-site at http://www.nr.no/coras

5.1 Context identification

The behavioural specification of system can be expressed using UML diagrams such as State or Activity diagrams and Sequence diagrams. In particular, the overall behaviour of a Web application like the e-Commerce platform can be described as a UML state machine where each state corresponds to a specific HTML page. The state machine in Figure 7 provides a high level description of the E-Commerce platform behaviour with respect to the user authentication and identification.

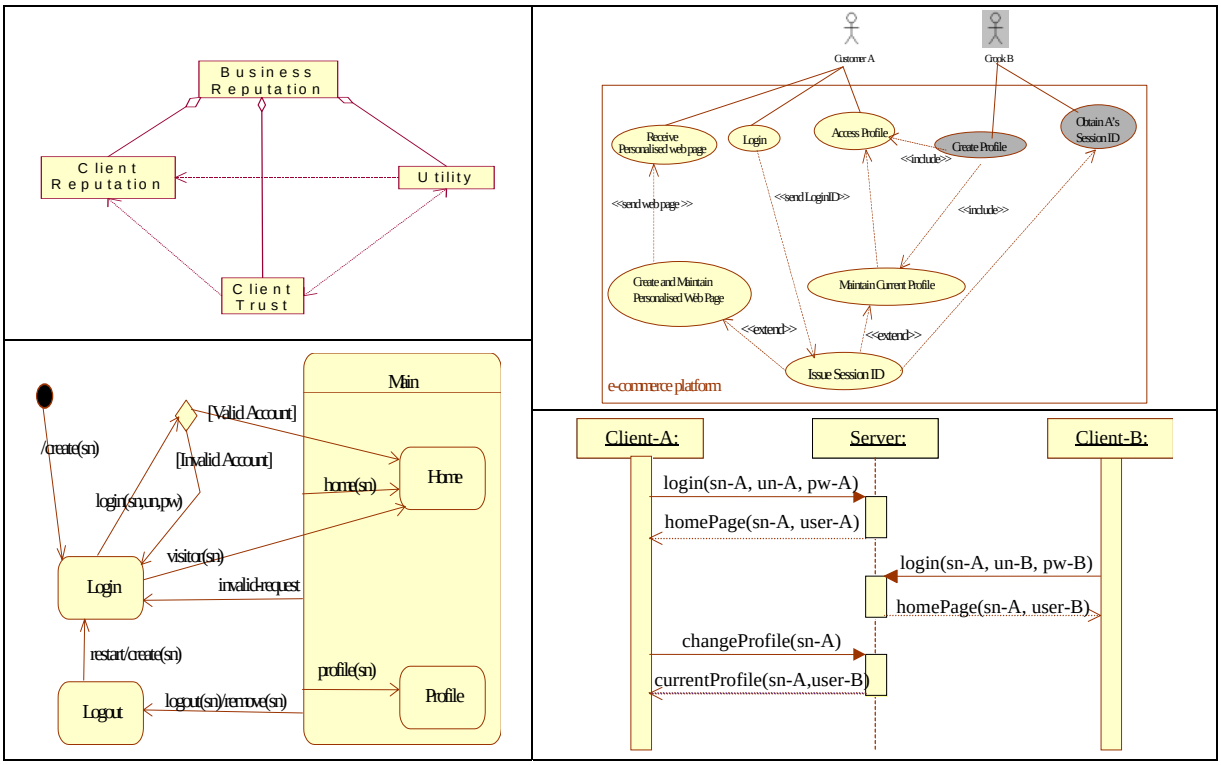

FIGURE 7: Anti-clockwise from top left: part of asset specification, part of user authentication behaviour, part of scenario specification, part of scenario description via a sequence diagram.

According to the diagrams in, 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 carries the username and password as parameters. Users can also access the platform as visitors without authentication but they are not able to use all functionality like shopping lists.

In addition to system behaviour and configuration, identification of assets is crucial in order to be able to perform risk assessment. For example, business reputation (of the e-Commerce platform provider) can be modelled as an aggregate asset built up from the utility of the business, the trust that its clients have in the business, the reputation of each client, etc. (Note that we look at a client as an entity having its own reputation).

5.2 An Illustrative Scenario

The use of session numbers for client identification can have undesirable consequences if a malicious actor captures a client's session ID. This actor is able to embarrass the legitimate user exploiting the captured session ID by logging into the platform using a new (second) account with an offensive profile (e.g., having vulgar names) but using the captured session ID instead of a new one. From that point on, all interactions of the legitimate user with the platform will have the profile of the second account. The corresponding behaviour of the system is as the sequence diagram in Figure 7, which can be perceived as part of a specialisation of the scenario description presented via the misuse case:

A legitimate user, Client-A, presented with the login page, logs into the platform with the username/password un-A/pw-A, using session ID sn-A, (event login(Sn-A, un-A, pw-A)). As a response she receives back a page personalised to Client-A (event homePage(sn-A, user-A)). Then, a malicious user, Client-B, that captured the session ID of Client-A, logs in with this session ID (sn-A of Client-A) and a new account with username/password un-B/pw-B (event login(sn-A, un-B, pw-B)). As a response he receives back a page personalised to Client-B (event homePage(sn-A, user-B)). However, from now on the session ID sn-A of Client-A is used by the system to refer to the profile of Client-B. Therefore the profile that Client-A accesses (event changeProfile(sn-A)) is actually the profile of Client-B (response currentProfile(sn-A, user-B)) i.e, 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. Notably the use of cookies is subject to similar deficiencies 0. This scenario (focusing on the documentation of threats and vulnerabilities) is presented as a (Mis-)Use Case in a misuse case diagram. This diagram includes the regular use cases in normal colour and the use cases describing abuses of the system by a malicious actor as shaded.

To → ↓From	HazOp	FTA	FMECA
HazOp	HazOp identifies incidents at different levels of abstraction.	The incidents identified by HazOp are inserted in fault trees based on abstraction level and the relationship between the incidents.	Incidents identified by HazOp may be understood as failure modes and thereby can be considered as starting points for FMECA.
FTA	A basic event (a leaf node in the fault tree representing an incident) may correspond to a sub-system/service on which HazOp may be applied.	A fault tree may be part of another fault tree, i.e., the top incident of one fault tree may be a causing incident in another fault tree.	Basic events (leaf nodes in the fault tree representing incidents) may be understood as failure modes and thereby can be considered as starting points for FMECA.
FMECA	From a basic incident (failure mode) one can associate a sub-system/service for applying HazOp on.	The analysis of a basic incident (failure mode) may identify a scenario leading to an unwanted incident. This may be represented as a path in the fault tree.	Basic incidents (or failure modes) may lead to incidents that are basic incidents (failure modes) in another FMECA.

FIGURE 8: A summary of the relationship between different RA methods recommended by CORAS for threat identification **.**

5.3 Risks assessment

In order to identify risks, risk analysis is performed. The risk analysis of the user authentication mechanism deployed by the e-commerce platform was based on models of its behaviour like those presented in the previous

subsection. 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 were performed, and some examples of this are presented below.

The objective of HazOp is to identify possible unwanted incidents, as well as their causes and consequences. Starting with the system's behaviour as expressed by the state machine in Figure 7, all events are independently 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 9. The first column, Entity, corresponds to the events of the system behaviour followed by a brief informal description. The Security attributes correspond to possible breach of security requirements 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.

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 9: HazOp table for the Login page.

FMEA was used to identify possible failure modes of individual components. For software systems, like the e-commerce platform, these failures can be wrong results and exceptions or error values returned by function calls to software components. As the complexity of the target system increases, the FMEA table becomes correspondingly large and time consuming to produce. The CORAS guidelines [11] (see also Figure 8)

recommend the use of FMEA on critical parts of the system identified by the HazOp analysis. Consequently, the CORAS trials focused FMEA only on small parts of the Web, Application and Database servers of the e-commerce platform. 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 9. 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, the possible failures identified by FMEA and the fault tree constructed in that these deviations appear as nodes ("event blocks") in the fault tree. For example, item 1.1.2 of HazOp table identified that a capture of a server response leads to the disclosure of Session ID. This is reflected in FTA tree, where state S4 can be achieved by means of reaching state S7. When the risks are identified, analysis of impact and likelihood are performed. The results of this analysis typically annotate existing models or one creates tables listing the risks and assigning likelihood and consequences to them. Similarly, the risk evaluation activity produces levels of risk and risk priorities that further annotate existing specifications.

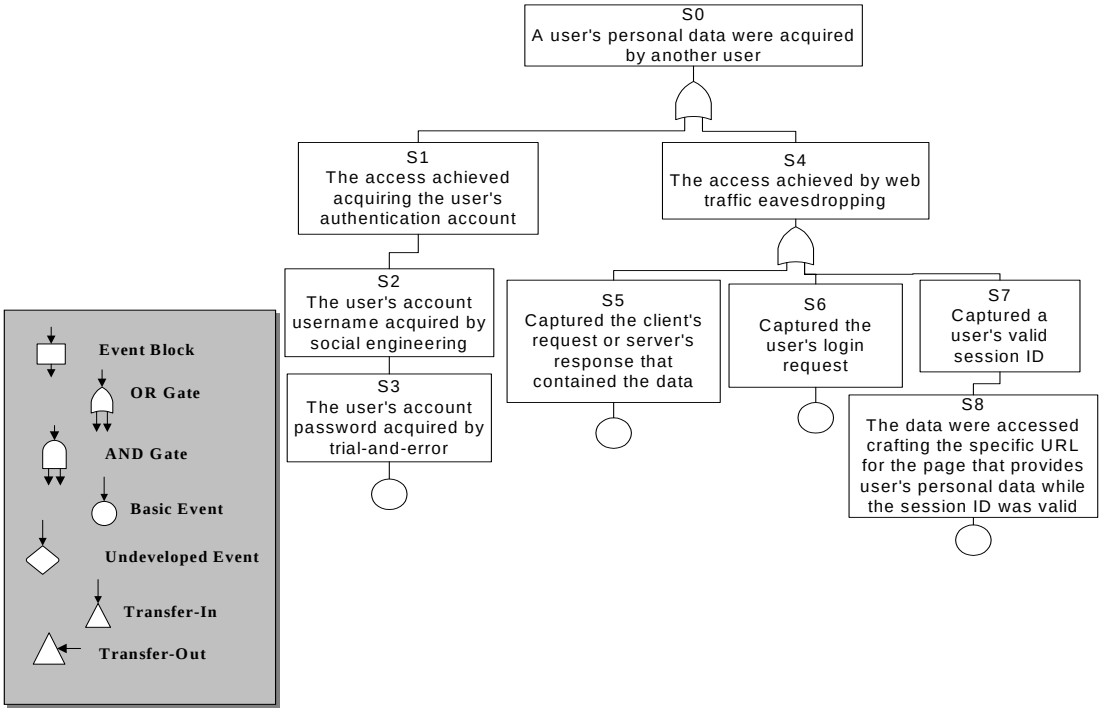

FIGURE 10:. Fault tree example related to the HazOp table of Figure 9.

6. RELATED WORK

Since 1990, work has been going on to align and develop existing national and international schemes in one, mutually accepted framework for testing IT security functionality. The Common Criteria [8] (CC) represents the outcome of this work. Increasingly it is replacing national and regional criteria with a worldwide set accepted by the International Standards Organisation (ISO15408) [17]. The CC and CORAS are orthogonal approaches. The CC provides a common set of requirements for the security functions of IT products and systems, and a common set of requirements for assurance measures applied to the IT functions of IT products and systems during a security evaluation. CORAS provides concrete specification technology that specifically addressing risk analysis.

Surety Analysis (SA) [24], is a methodology based on the creation of an explicit model that covers several aspects of the system's behaviour. OPRA, the tool that has been implemented to support SA, has a "tight" integration of

existing commercial software packages with tools developed in Sandia Labs especially for this project. The CORAS platform will facilitate a "loose" integration platform based on widely deployed interchange standards that will allow for different users to adapt the CORAS platform to their own needs.

RSDS [22] is a tool-supported methodology developed by King's College London and B-Core UK, ltd. CORAS focuses on security risk analysis whereas current work on RSDS focuses on safety and reliability analysis.

COBIT [9] focuses on control objectives defined in a process-oriented manner following the principles of business re-engineering. COBIT and CORAS are orthogonal.

CCTA Risk Analysis and Management Methodology (CRAMM) [3] was developed by the British Government's Central Computer and Telecommunications Agency (CCTA) with the aim of providing a structured and consistent approach to computer security management for all systems. The UK National Health Service considers CRAMM to be the standard for the risk analysis of information systems within healthcare establishments. CRAMM is an important source of inspiration for CORAS, and aspects of CRAMM have been incorporated in CORAS. Contrary to CRAMM, CORAS provides a risk analysis process in which modelling is tightly integrated, and CORAS complies with state-of-the-art international standards for risk management, documentation, modelling and development of systems. CCTA has extended CRAMM into an overall system development process by developing an interface between CRAMM and SSADM (Structured Systems Analysis and Design Method). This corresponds to the CORAS integrated risk management and system development process based on AS/NZS 4360 and UP.

7. CONCLUSION

Information and communication technologies (ICT) are becoming a dominant part of people's everyday life in Europe. The demand for trusted and cost-effective security solutions is increasing as the use of decentralised distributed systems built on open networks proliferates. Such systems are in many cases vital for the enterprise-wide functions. The CORAS approach provides one way of increasing trust and confidence in information and communication systems based on the integration of security risk management and graphical semiformal modelling for the efficient and unambiguous security assessment of such systems. In particular we anticipate that the integration of risk analysis and semiformal modelling, which underpins the CORAS framework, will provide several benefits, including:

- An improvement to the current state-of-the-art of methods for risk analysis;
- The ability to handle more complex systems, analysed at different levels of abstraction;
- The ability to adapt the same framework for analysing legacy systems and systems under development, depending on the user needs;
- The ability to re-use a part of the risk analysis results during system maintenance, for example when new subsystems or components are introduced in an already analysed system
- The ability to use the system model architecture in order to guide the co-use of complementary risk analysis methods in a way that increases the effectiveness of the overall result.

Obvious future extensions include improving the CORAS tool inclusion platform by implementing a deeper data-integration as indicated in section 5. Other future extensions may involve extending the integrated risk analysis and systems modelling framework of CORAS with a model of trust (such as the model outlined in [14] and elaborated in [13]) in order to support analysing trust in e-services, as well as applying the CORAS experience to the assessment of executable electronic contract descriptions (such as Service Level Agreements orchestrating a number of component web service in order to provide end-to-end application services).

REFERENCES

[1] Australian/New Zealand Standard AS/NZS 4360:1999: Risk Management.
[2] Atkinson, C., Bayer, J., Bunse, C., Kamsties, E., Laitenberger, O., Laqua, R., Muthig, D., Paech, B. Wüst, J., Zettel, J. Component-based product line engineering with UML. Addison-Wesley, 2002.
[3] Barber, B., Davey, J. The use of the CCTA risk analysis and management methodology CRAMM. Proc. MEDINFO92, North Holland, 1589 –1593, 1992.

[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.
[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] Clark, J. XSL transformations (XSLT) 1.0, World Wide Web Consortium recommendation REC-xslt, November 1999.
[7] Cockburn, A. Structuring use cases with goals. Journal of object-oriented programming, Sep/Oct: 35-40, Nov/Dec: 56-62, 1997.
[8] Common Criteria Organisation, “Common Criteria for Information Technology Security Evaluation”, accessed: 2002.
[9] Control Objectives for Information and related Technology, “COBIT”, http://www.isaca.org/ct_denld.htm
[10] CORAS project web-site. http://www.nr.no/coras
[11] CORAS consortium. Guidelines and templates for model-based risk assessment. Internal Deliverable IST-2000-25031. August 2002.
[12] Curry, D., Debar Merrill Lynch, H. Intrusion detection message exchange format (IDMEF). Working draft, December 28, 2001.
[13] Dimitrakos Th., Bicarregui J.C. " Towards A Framework for Managing Trust in e-Services". In Proceedings of the 4th International Conference on Electronic Commerce Research, ATSMA, IFIP, November 2001. ISBN 0-9716253-0-1.
[14] Dimitrakos Th. "System Models, e-Risk and e-Trust. Towards bridging the gap?" in Towards the E-Society: E-Business, E-Commerce, and E-Government, eds. Schmid B., Stanoevska-Slabeva K., Tschammer V., Kluwer Academic Publishers, 2001. ISBN-0-7923-75297
[15] 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
[16] ISO/IEC 10746 series: 1995 Basic reference model for open distributed processing.
[17] ISO/IEC TR 13335-1:2001: Information technology–Guidelines for the management of IT Security, Part 1.
[18] ISO/IEC 17799: 2000 Information technology – Code of practise for information security management.
[19] IEC 1025: 1990 Fault tree analysis (FTA).
[20] Jacobson, I., Rumbaugh, J., Booch, G. The unified software development process. Addison-Wesley, 1999.
[21] Littlewood, B. A reliability model for systems with Markov structure. Appl. Stat. 24:172-177, 1975.
[22] Reactive System Design Support, “RSDS”, http://www.dcs.kcl.ac.uk.
[23] Redmill, F., Chudleigh, M., Catmur, J. Hazop and Software Hazop. Wiley, 1999.
[24] Sandia National Laboratories, “Surety Analysis”, http://www.sandia.gov, 2002.
[25] Schneider, G., Winters, J. P. Applying use cases: a practical guide. Addison-Wesley, 1998.
[26] Sindre, G., Opdahl, A. L. Eliciting security requirements by misuse cases. In Proc. TOOLS_PACIFIC 2000. IEEE Comp. Soc. Press.
[27] World Wide Web Consortium, Extensible Markup Language (XML) v1.0, W3C Recommendation, Second Edition, 6 Oct. 2000.