THE CORAS METHODOLOGY: MODEL-BASED RISK ASSESSMENT USING UML AND UP

Folker den Braber[1], Theo Dimitrakos[2], Bjørn Axel Gran[3], Mass Soldal Lund[1], Ketil Stølen[1], Jan Øyvind Aagedal[1]

[1] Sintef Telecom and Informatics, P.O.Box 83, N-0314 Oslo, Norway
{folker.den.braber, mass.s.lund, ketil.stoelen, jan.aagedal}@informatics.sintef.no
[2] CLRC Rutherford Appleton Laboratory, Oxfordshire, OX11 0QX, UK
t.dimitrakos@rl.ac.uk
[3] Institute for Energy Technology, P.O. Box 173, N-1751 Halden, Norway
bjorn.axel.gran@hrp.no

THE CORAS METHODOLOGY: MODEL-BASED RISK ASSESSMENT USING UML AND UP

Abstract

This chapter introduces the CORAS methodology in which UML and UP are combined to support a model-based risk assessment on security critical systems. The hypothesis is that modelling techniques like UML contribute to increased understanding for the different stakeholders involved during a risk assessment. In the CORAS methodology a traditional risk management process is integrated with UP, a well accepted system development process. CORAS tries to show how UML can contribute to better understanding, documentation and communication during the different phases of the risk management process. CORAS addresses both systems under development and systems already in use.

INTRODUCTION

After the development of information technology in the last part of the previous century, it has become impossible to imagine a world without IT-systems. The impact of this development has been enormous and has opened for a lot of new possibilities and challenges. One of these challenges regards risks. To make use of these new techniques in a dependable way, it is of vital importance to get an overview and understanding of the different risks connected to the use of IT-systems. This chapter addresses model-based risk assessment, a methodology developed in the CORAS project. CORAS (2000) is funded by the European Union and develops a tool-supported framework for precise, unambiguous, and efficient risk assessment of security critical systems. CORAS aims at a methodology for risk assessment that is easy to understand and functions as a natural part of both the IT-system development and the maintenance life cycle. To achieve this CORAS leans on the knowledge gained from the use of models in graphical semi-formal languages like the Unified Modeling Language (UML) (OMG, 2001b). Main focus in this chapter lies on the part of the CORAS project that addresses the integration of risk management and system development.

The remainder of this chapter is divided into five sections. First some background is presented. The section thereafter addresses model-based risk assessment and some of the problems connected to this. The main part of the chapter is contained in the section on the risk management process and the integrated risk management and system development process. After a section on related work, a brief conclusion is given.

BACKGROUND

The CORAS approach focuses on the tight integration of viewpoint-oriented UML modelling in the risk management process. An important aspect of the CORAS project, is the practical use of UML and the Unified Process (UP) (Kruchten, 1999) in the context of security and risk assessment. This chapter concentrates on the integration of UML and UP in the risk assessment process.

CORAS addresses security critical systems in general, emphasising IT security. IT security includes all aspects related to defining, achieving, and maintaining confidentiality, integrity, availability, non-repudiation, accountability, authenticity, and reliability of IT systems (ISO/IEC TR 13335:2001). An IT system in the sense of CORAS is not just technology, but also the humans interacting with the technology and all relevant aspects of the surrounding enterprise context.

The Rationale

Model-based risk assessment 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.

Model-based risk assessment is motivated by several factors:

- Risk assessment requires correct descriptions of the target system, its context and all security relevant features. The modelling technology improves the precision of

such descriptions. Improved precision is a prerequisite for improving the quality of risk assessment results.

- The graphical style of UML aims at improving the communication and interaction between stakeholders involved in a risk assessment. This is expected to improve the quality of results, and also speed up the risk assessment process since the danger of wasting time and resources on misconceptions is reduced.
- The modelling technology facilitates a more precise documentation of risk assessment results and the assumptions on which their validity depends. This is expected to reduce maintenance costs by increasing the possibilities for reuse.
- The modelling technology provides a solid basis for the integration of assessment methods that should improve the effectiveness of the assessment process.
- The modelling technology is supported by a rich set of tools from which the risk management may benefit. This may improve quality (as in the case of the two first bullets) and reduce costs (as in the case of the second bullet). It also increases productivity and maintenance.
- The modelling technology provides a basis for tighter integration of risk management and assessment in the system development process. This may considerably reduce development costs and help ensure that the specified security level is achieved.

The main CORAS result is the CORAS framework for model-based risk assessment. As illustrated by Figure 1, the CORAS framework has four main anchor points.

1. A risk management process.
2. A risk documentation framework.
3. An integrated risk management and development process.
4. A platform for tool-inclusion based on data integration.

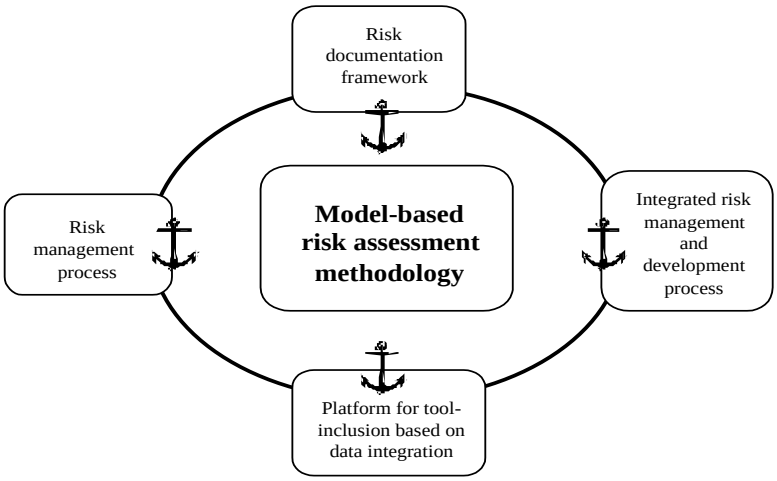

Figure 1 The CORAS framework for model-based risk assessment

The risk management process provides the core for the CORAS process from a traditional risk analysis background. Combined with the risk documentation framework this provides the basis for the development of the integrated risk management and development process. The fourth anchor point represents the CORAS platform, which is a tool that is interoperable with different other tools from both the risk analysis field and the modelling world, providing a model-based risk

assessment product that can be used on either existing systems or systems under development.

Project Background

For a project like CORAS it is impossible to develop everything from scratch. A lot of the work is based on already existing standards and techniques. To avoid mystifying this chapter by mentioning every used standard or method when it occurs, we summarise the main references here. In addition we will also say something about the parts of the project that are not addressed in this chapter.

The CORAS risk management process is based on the following standards: AS/NZS 4360:1999 "Risk Management" (AS/NZS 4360, 1999) and ISO/IEC 17799: "Code of Practice for Information Security Management" (ISO/IEC 17799:2000), which are both well established in the field of risk assessment. It is complemented by ISO/IEC 13335: "Guidelines for the management of IT-Security" (ISO/IEC 13335:2001) and IEC 61508: "Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems" (IEC 61508:2000).

The CORAS risk documentation framework is a specialisation of the Reference Model for Open Distributed Processing (RM-ODP) (ISO/IEC 10746, 1995).

In order to make the CORAS methodology available, the CORAS platform is developed. The CORAS platform is built around an internal data representation formalised in XML (W3C, October 6, 2000). Based on XSLT (Clark, November 1999), relevant aspects of the internal data representation may be mapped to the internal data representations of other tools (and the other way around). This allows the integration of sophisticated case-tools targeting system development as well as risk assessment tools and tools for vulnerability and threat management.

The CORAS platform consists of three interfaces for XML based data exchange:

- Interface based on IDMEF (Intrusion Detection Exchange Format) (Curry, December 28, 2001). 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) (OMG, 1999) standardised by the Object Management Group and targeting tools for UML modelling.
- Interface targeting risk assessment tools.

The CORAS platform contains a repository divided into two parts:

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

The CORAS framework and process are being validated in extensive user trials in the areas of e-commerce and tele-medicine, it started in January 2001 and runs until July 2003.

The Consortium

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).

MODEL-BASED RISK ASSESSMENT

The CORAS risk assessment methodology incorporates a documentation framework and a number of closely integrated risk assessment 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 in the form of guidelines and specified diagrams. Risk assessment requires a firm, but nevertheless easily understandable, basis for communication between different groups of stakeholders. Graphical object-oriented modelling techniques have proven well suited in this respect for requirements capture and analysis. It is fair to assume that they are equally well suited as part of a language for communication in the case of risk assessment. Class diagrams, use case diagrams, sequence diagrams, activity diagrams, dataflow diagrams and state diagrams represent mature paradigms used daily in the IT industry throughout the world. They are supported by a wide set of sophisticated case-tools, they are to a large extent complementary and, together, they support all stages in a system development.

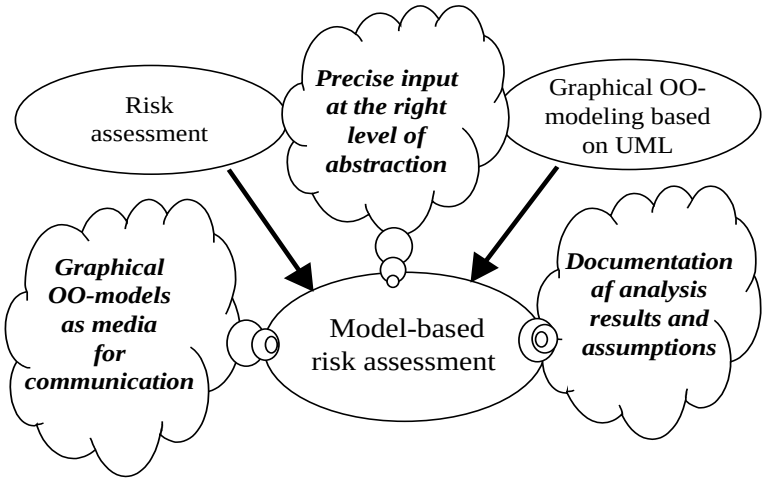

Figure 2 Model-based risk assessment

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, focusing on different types of risks or different areas of concern.
The CORAS risk assessment methodology is built on:

- HAZard and OPerability study (HazOp) (Redmill, 1999),
- Fault Tree Analysis (FTA) (IEC 1025, 1990),
- Failure Mode and Effect Criticality Analysis (FMECA) (Bouti, 1994),
- Markov Analysis (Littlewood, 1975).

These methods are to a large extent complementary. All types of risks associated with the target system can potentially be revealed and dealt with using these methodologies. They also cover all phases in the system development and maintenance process. However, they are traditionally applied as separate methods, each providing their own results. To combine their output and integrate them with a modelling language like UML is what makes CORAS unique.
Another aspect of the CORAS risk assessment methodology is the process part. The CORAS process is based on the UP. As a well understood and largely accepted process in software development, the UP is a suitable candidate to serve as a basis in developing our methodology for integrating risk management in the development process. Of course, the Unified Process needs some adjustment here and there in order to be able to profit from the risk management carried out in parallel.
For example, two aspects of risk assessment that effect the development process are system requirements and security policies.
A set of agreed system requirements is one important outcome of the inception and elaboration phases. As one cannot expect that all security requirements are present from start, they have to be elicited. We anticipate that (appropriately adapted) model-based security risk assessment can help with eliciting security requirements. However, risk assessment methods are traditionally designed to cope with unwanted incidents arising from design errors rather than specification problems related to missing requirements. For security risk assessment to play a significant role in the elaboration phase, the CORAS risk assessment methods are being adapted to address requirement elicitation properly. For this purpose we are currently developing tailored templates to extend a scenario-driven analysis with security risk assessment throughout the development life cycle. Important sources of inspiration include the work of Cockburn (1997) and Schneider/Winters (1998).
Another consideration with respect to improving the system's 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 the identified security vulnerabilities. We expect that the CORAS framework can support incorporation of system documentation and control guidelines (about changes to the way the system is used or operated) targeting the improvement of the system's security.

THE RISK MANAGEMENT PROCESS AND THE INTEGRATED RISK MANAGEMENT AND DEVELOPMENT PROCESS

Following the overview of the CORAS approach to model-based risk assessment provided in the previous sections, we now focus on the integration of the CORAS risk management methodology into a system development and maintenance process.
The left part of Figure 3 shows the risk management process divided into sub-processes for context identification, risk identification, risk analysis, risk evaluation, and risk treatment. For each of these stages, the CORAS methodology gives detailed advice with respect to which models should be constructed, and what they should express.

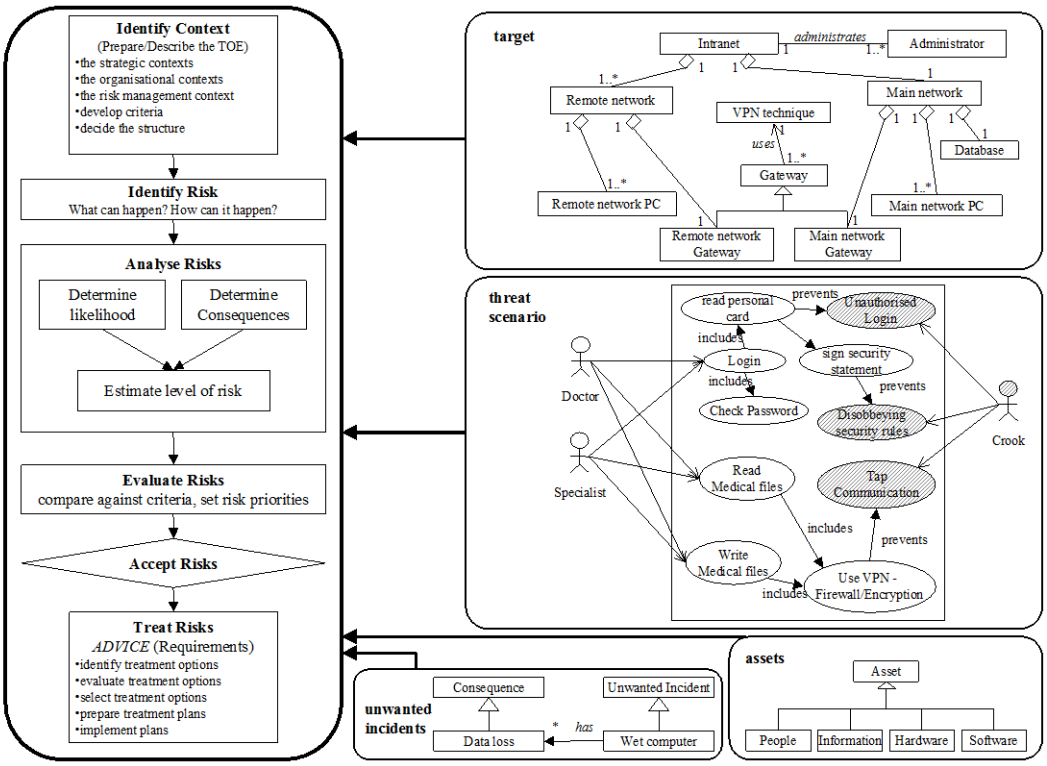

Figure 3: The CORAS risk management process and the role of UML

The CORAS risk management process is divided into five subprocesses. These subprocesses are again divided into activities and each activity has one or more concerns connected to it.

Each concern is divided into the five RM-ODP viewpoints. One or several of its viewpoints may be empty depending on the concern in question as well as the target and rigour of evaluation. However if not empty, each concern-viewpoint will contain a set of element instances. An element may be a model, a risk assessment table, a tree, natural language text, etc. Different instances of the same element may occur within different viewpoints of the same concern. Elements may be classified into:

- Elements containing non-CORAS specific documentation, which refers to elements that are not prepared as a part of the CORAS risk management process. Since CORAS should be applicable to a wide scope of systems, including already existing systems, this kind of elements is unconstrained.
- Modelling elements (constructed as part of the risk management process) expressed in UML.
- Logs from intrusion detection tools and computerised vulnerability assessment represent.
- Risk assessment tables and trees.

Using RM-ODP as the basis for our documentation framework, every concern can be observed from five different viewpoints. CORAS defines 22 concerns shown in Figure 4. This Figure shows the complete CORAS documentation framework with the concerns and viewpoint divided over the different activities under their subprocesses.

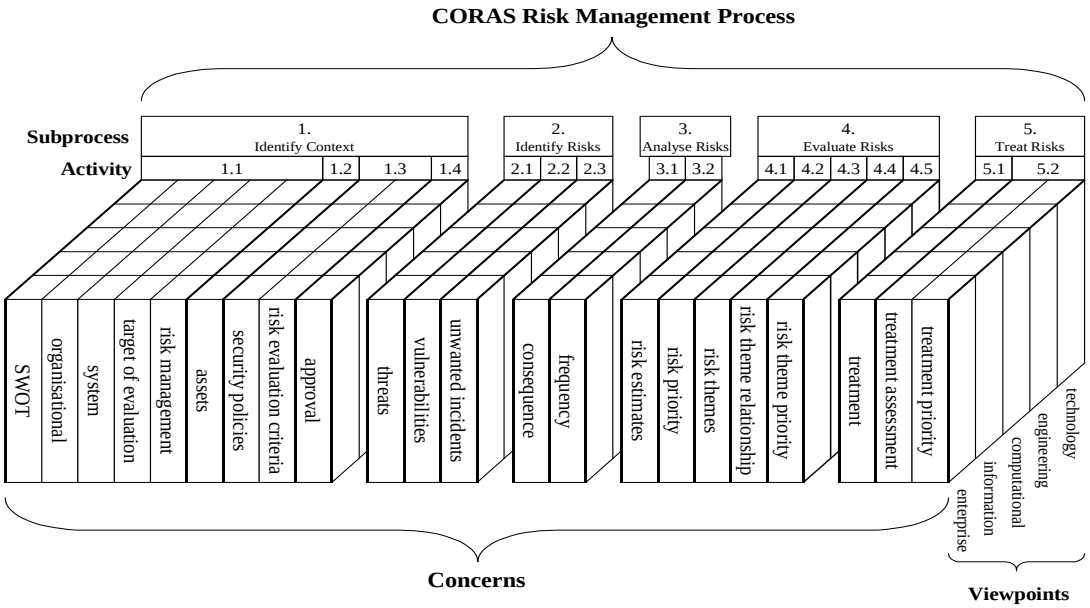

Figure 4 The CORAS Documentation Framework

The CORAS Ontology

Carrying out a risk assessment requires a clear understanding of all the areas of the system to be assessed. Typically, many different stakeholders are involved in a risk assessment and it is therefore crucial that all parties involved have the same understanding of the language and concepts used during the assessment. Figure 5 provides an overview of the elements of the CORAS ontology. The concepts are drawn as classes in a simplified class diagram. The guiding words give some extra explanation about the specified relations.

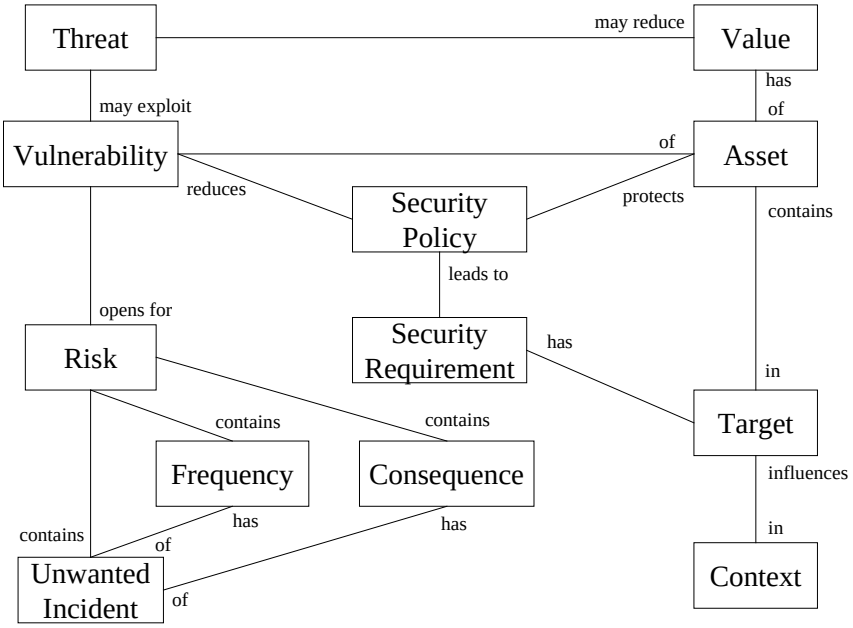

Figure 5 The CORAS ontology

Starting from the bottom right corner we see that the *Context* influences the *Target* that contains *Assets* and has its *Security requirements*. *Security requirements* lead to *Security policies*, which protects *Assets* by reducing their *Vulnerabilities*. Continuing, a *Threat* may exploit the *Vulnerability* of an *Asset* thereby reducing the *Value* of the *Asset*. A *Risk* contains an *Unwanted incident* having a certain *Consequence* and *Frequency* of occurring.
Figure 6 provides a pictorial overview of the integration of the model-based risk management methodology into the system development and maintenance process. In analogy to UP, the system development 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. Then the system lifecycle moves from one phase into another.

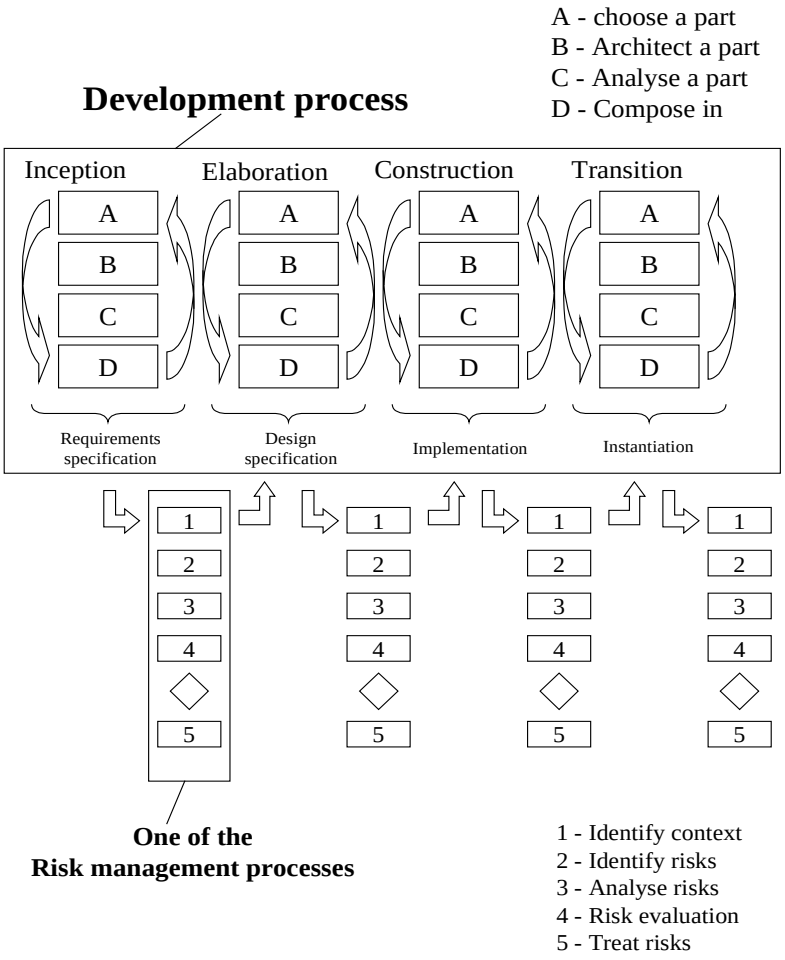

Figure 6 The integrated risk management and development process

After every phase of the development process a risk assessment is carried out with its five complete sub processes. It should be clear that the character of the specific risk management process is different depending on the actual phase of development. Identifying, analysing and treating risks will be done after all four phases but, for example, treating a risk indentified after the inception phase would be different from treating a risk after the construction phase.

Subprocess 1: Identify Context

In this subprocess of the CORAS process the context of the system is identified. Context identification establishes the strategic, organisational and risk management context.
This means identifying everything that has to do with the system, including the system itself but also the people using it, maintaining it and configuring it.
The identify context subprocess is divided into four activities:

- Identify areas of relevance;
- Identify and value assets;
- Identify policies and evaluation criteria;
- Approval.

Activity 1.1: Identify areas of relevance

The results from this activity are documented by five concerns:

- SWOT (Strengths, Weaknesses, Opportunities, Threats);
- Organisational;
- System;
- Target of evaluation;
- Risk management.

The SWOT concern

The result of a SWOT-analysis may be displayed in a SWOT-diagram. SWOT-diagrams are specialised UML class diagrams in the *enterprise viewpoint*, where strengths, weaknesses, opportunities and enterprise threats are represented by easy-to-understand symbols. These are clear examples of how modelling in UML can contribute to easy understanding and communication.
The "SWOT-elements" are related to key stakeholders and key assets. This may provide additional direction to the assessment.
An example of a SWOT-diagram is shown in Figure 7.

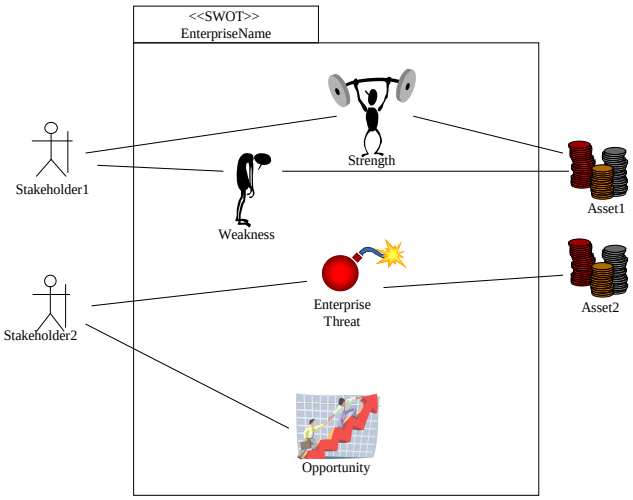

Figure 7: SWOT diagram

The organisational concern

The organisational context concern documents the relevant aspects of the organisation within which the assessment will be conducted.

The system concern

The system documentation concern has six elements:

- System documentation in any form.
- A specification of the main components and the system boundaries using Dataflow diagrams or UML component/deployment diagrams.
- A specification of the stakeholders and main services of the system using UML use-case and sequence diagrams.
- A specification of the main information structure using UML class diagrams.
- A specification of the logical infrastructure including the main interfaces of components and their connections based on UML state diagrams.
- A specification of the physical infrastructure and the distribution of components in this infrastructure using UML deployment diagrams.

The target of evaluation concern

The target of evaluation concern specifies the part of the system to be assessed with particular emphasis on characterising the borderline between the part that should be considered and the rest.

The risk management concern

The risk management concern documents the risk management context for the assessment. The risk management concern is based on four tables. One describing the team that will perform the risk management process. One table describing detailed plans for the meeting. One shows the applied methods, and one contains additional informal descriptions.

Activity 1.2: Identify and value assets

This activity is only connected to one concern, the asset concern.

The asset concern

The assets concern documents the results from the assets identification and validation. Assets play a key role in risk assessment, talking about risks means basically talking about assets. For every system the assets are the carriers of the value of the system. This can be anything from knowledge to physical hardware, information etc. A risk is nothing else than the change of an asset decreasing its value. This is why the identification of assets is central in connection with risk assessment.
The assignment of assets to asset themes and stakeholders may be illustrated in a UML class diagram, in the *enterprise viewpoint*, as the one in Figure 8. This kind of diagram may also document important relations between the assets.

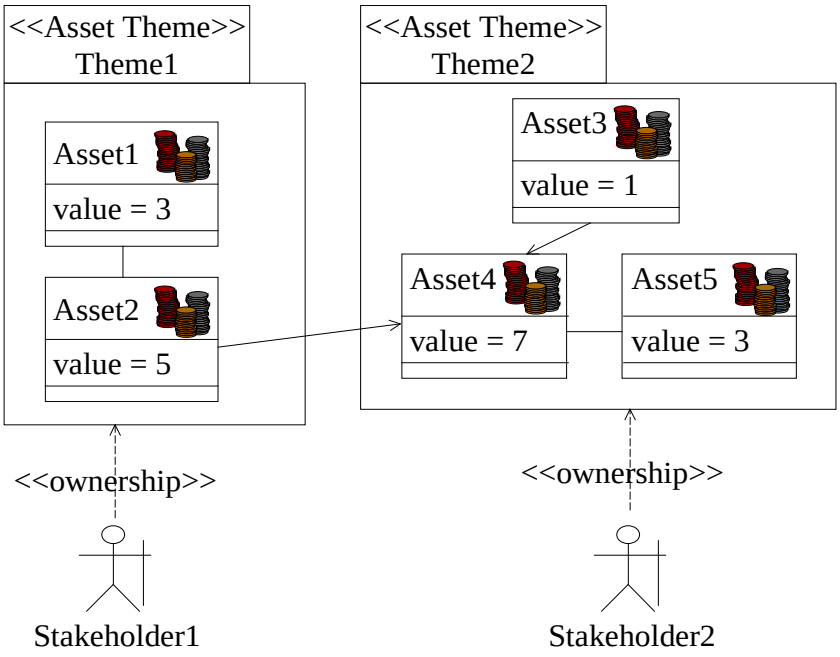

Figure 8: Asset relationship diagram

Activity 1.3: Identify policies and evaluation criteria

Two concerns are used to document the results from this activity:

- Security policies;
- Risk evaluation criteria.

The security policy concern

The security policy concern documents the security policies of relevance. The security policies concern has one element being the security policy documentation. No specific requirements are given about the format of this documentation since the CORAS framework should be applicable to already existing systems for which security policies probably already exist in a format that most likely is not CORAS specific.

The risk evaluation concern

The risk evaluation concern documents the risk evaluation criteria. They specify the acceptable loss from the perspective of the assessment customer. The risk evaluation concern has one element in the form of a table.

Activity 1.4: Approval

The approval activity is covered by the approval concern.

The approval concern

The approval concern documents the results from a “formal walk through” through all relevant documents at the end of the “Identify context” sub-process. The approval concern contains one table listing the agreed changes identified by the different stakeholders.

Sub-process 2: Identify risks

The risk identification sub-process consists of three activities of which the first two are complementary and may be carried out in any order. The third is performed first after the two others have been completed.
Activities 2.1 and 2.2 address the risk identification from two different angles. Activity 2.1 focuses on identifying threat scenarios that may lead to loss in asset value. Activity 2.2 focuses on identifying the vulnerabilities of assets (or the associated system) that may be exploited by threats. Table 1 illustrates the complementary nature of Activities 2.1 and 2.2.

Table 1: Threat, vulnerability and unwanted incident relationships

Threat	Vulnerability	Unwanted Incident
TRUE	TRUE	Unwanted incident: There exists a threat that may exploit an existing vulnerability.
FALSE	TRUE	Potentially unwanted incident: There exists a vulnerability, but (so far) no threat that may exploit this vulnerability.
TRUE	FALSE	"Blocked" unwanted incident: A threat exists, but has no known vulnerabilities to exploit.
FALSE	FALSE	Ideal/normal behaviour.

Note that although Activities 2.1 and 2.2 have threats and vulnerabilities as their main focus, where the analyst will try to identify the associated unwanted incidents each time a new threat or vulnerability is found. Hence, the combined effort of Activities 2.1 and 2.2, addressing both threats and vulnerabilities, results in the identification of unwanted incidents. The objective of Activity 2.3 is to integrate, refine, organise and carefully document the results from the two previous activities.

Activity 2.1: Identify threats to assets

The threat concern documents results from activity 2.1. The risk analysis methods provided for activity 2.1 are HazOp, FMECA and FTA. These three methods provide different approaches on how to identify threats and their unwanted incidents. They work at three different levels of details. HazOp is best suited for identifying general threats to a system, where FMECA deals with each component in the system separately. FTA for this activity is used to identify threats related to identified high-level unwanted incidents, such as for example unwanted incidents identified through SWOT.

The threat concern

Threat diagrams are inspired by the misuse cases proposed by Sindre and Opdahl (2000). Threat diagrams are specialised use case diagrams.
Figure 9 provides an example of a Threat diagram. Threats are specialised use cases that model possible scenarios that must be considered threats. As with use cases, threats are specified by textual descriptions. Sequence diagrams and Activity diagrams may also be used to specify threats if that is desirable. A threat is related to the asset it is a threatening, and may also be related to a User Role or a Mis-user Role if the threat involves actions carried out by humans.
In a Threat diagram, assets are defined by their vulnerabilities as well as by their attributes and operations. Vulnerabilities are the negative or undesired properties of an asset, and are modelled by specialised attributes and operations. These are attributes that an asset preferably should not have, and operations that preferably should not be possible. Vulnerability operations may be specified further by, e.g., State diagrams.

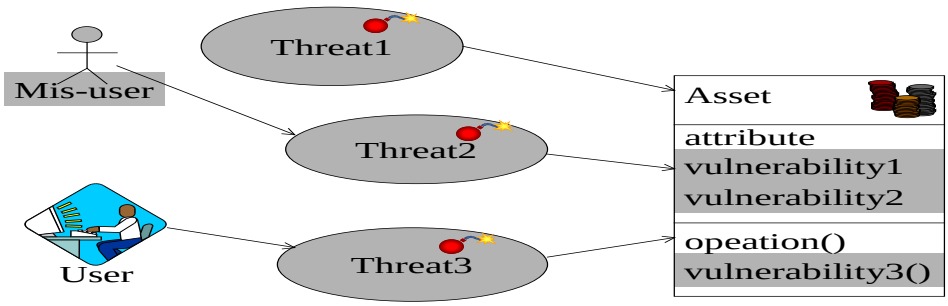

Figure 9: Threat diagram

Models supporting HazOp

HazOp analysis is basically structured brainstorming. Input models to HazOp must communicate important system issues and help guiding the brainstorming process. UML sequence diagrams and Activity diagrams can be used to provide this information. One important feature of sequence diagrams is that they are able to capture how the functionality in a system is provided at any abstraction level. The abstraction level provided by the sequence diagram need to reflect the knowledge of the members in the analysis group and the level of abstraction wanted for the results provided.
In Figure 10 an example of a HazOp table is given showing how a message originating from a sequence diagram can be analysed through HazOp guidewords.

Item	Attribute	Guideword	Unwanted Incident
Message: Advice from doctor to patient	Data content	No	Patient does not get medical advice
		Parts missing	Patient gets incorrect advice
		Faulty	Patient gets incorrect advice
		Too much	Patient may get misleading advice
	Control flow	Wrong sender	Patient may get advice concerning another patient
		No receiver	Patient does not get medical advice
		Wrong receiver	Patient does not get medical advice
			Unauthorised persons get information about patient
		Too many receivers	Unauthorised persons get information about patient
	Timing	Too early	Probably no problem
		Too late	Patient may not get medical advice
	Presentation	Difficult to read	Patient may get misleading advice
		Wrong colours	
etc.	etc.	etc.	etc.

Figure 10: Example of a HazOp diagram

Models supporting FMECA

The identified assets are based on the target of evaluation models. All relevant components should then be identified and refined through these two activities. A list of assets supported by a sequence diagram illustrating the functionality in the system covering the role of the assets identified, whether it is a physical or an information

asset, should be provided as input to FMECA. Again, the abstraction level of the sequence diagram needs to reflect the abstraction level of the assets identified. Figure 11 shows an example of a FMECA table that assesses the transmission of data in a tele-medicine application.

Ref No.	Item	Failure modes	Failure cause	Failure mechanism	Local failure effect	System failure effect	Consequences	Failure detection	Frequency	Remarks
1	Transmission of patient data from parent to doctor	Disclosure	Transmis-sion is being read by other actor (not parent or doctor)	Transmis-sion can be read by other actor (not parent or doctor)	Patient data are exposed	Confidentiality of system is compromised because patient information is revealed	Critical	None	High	Encryption of transmission is a possible solution.
2		Manipulation	Transmis-sion is written by other actor (not parent)	Transmis-sion can be written by other actor (not parent)	Patient data are not correct	Integrity of system is compromised because other actors may produce, alter or send information	Catastrophic	None	Medium	A digital signature on the transmitted data is a possible solution
			Transmis-sion is altered by other actor (not parent)	Transmis-sion can be altered by other actor (not parent)						
			Transmis-sion is sent by other actor (not parent)	Transmis-sion can be sent by other actor (not parent)						
3		Denial	Transmis-sion is not possible.	Transmis-sion can not be sent	Patient data are not received	Availability of system is compromised because system is not available	Marginal	System is not available	High	

Figure 11: Part of FMECA diagram.

Models supporting FTA

FTA used during threat identification is used to relate threats to identified unwanted incidents, either through SWOT analysis or with help of HazOp and FMECA. The top event would be the unwanted incident identified. The input model for the levels above would then be a sequence diagram illustrating all involved assets and their relationship. This sequence diagram would need to reflect the relationships in the asset graph and would for this system include human, information, software and physical assets. To be able to identify the threats leading to the unwanted incident one could consider information assets first, then software assets and finally physical assets. However, this depends on the required abstraction level.

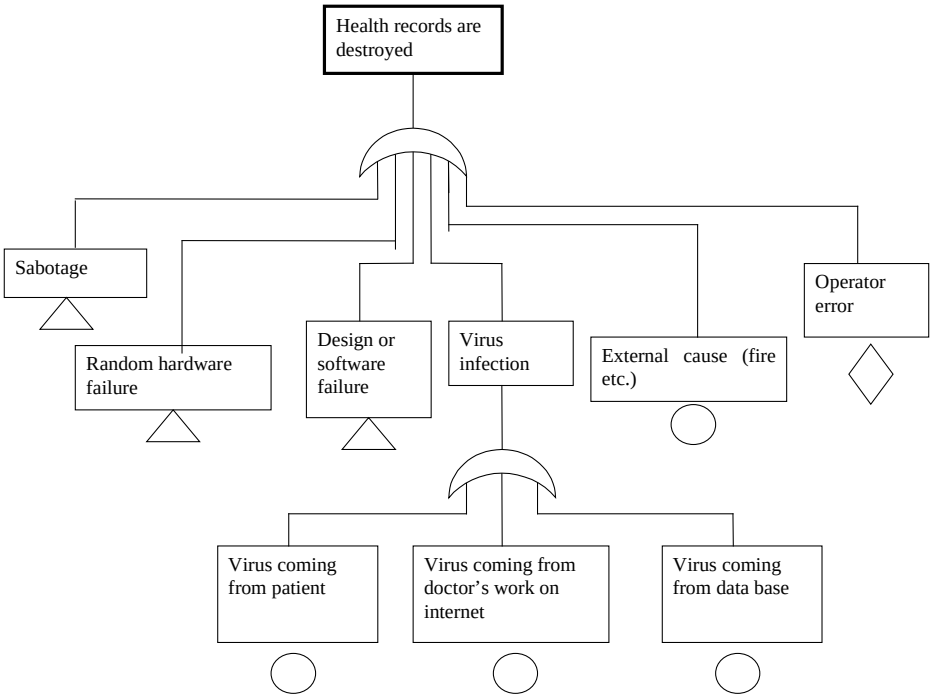

Figure 12 FTA example

Activity 2.2: Identify vulnerabilities of assets

Earlier we have seen that vulnerability is related to both to threats and assets. It is the (negative) glue between them. A threat is only a harmful threat to an asset if there exists a vulnerability that makes the asset vulnerable to the threat. It looks like a vulnerability is a negative quality of an asset.

The vulnerability concern

The vulnerabilities are documented in the threat scenario diagrams (Figure 9).

Activity 2.2: Document unwanted incidents

Unwanted incidents are generated from the results from activity 2.1 and 2.2 where threats and vulnerabilities have been identified. Unwanted incidents specify the events of what happens if a threat becomes reality. Events are typically specified in UML through state diagrams.

The unwanted incident concern

Unwanted incidents are documented in tables, however they appear also in state diagrams, as the one given in Figure 15.

Sub-process 3: Analyse risks

Risk analysis is the systematic use of available information to determine how often specified unwanted incidents might occur and the magnitude of their consequences. Figure 13 shows the frequency and consequence values in one matrix. Note that this information is available before any risk is identified. Knowing the consequence and frequency of a certain risks makes it possible to determine the total risk value through the risk matrix.

Frequency / Consequence	Rare	Unlikely	Possible	Likely	Almost certain
Insignificant	No	No	Low	Low	Moderate
Minor	No	Low	Low	Moderate	Moderate
Moderate	Low	Low	Moderate	Moderate	High
Major	Low	Moderate	Moderate	High	High
Catastrophic	Moderate	Moderate	High	High	**Extreme**

Figure 13 Risk Matrix

Activity 3.1: Consequence evaluation

Consequence concern

State diagrams can be used to specify consequences, they can describe all possible behaviour in the system, both “good” and “bad” (Houmb, 2002). The “good” behaviour in the system is reflected through the normal behaviour states. These states illustrate normal and authorised behaviour within the system. The “bad” behaviour is reflected through the unwanted incident states identified in sub process 2, identifying risks.

An example giving such a state diagram is given in Figure 14.

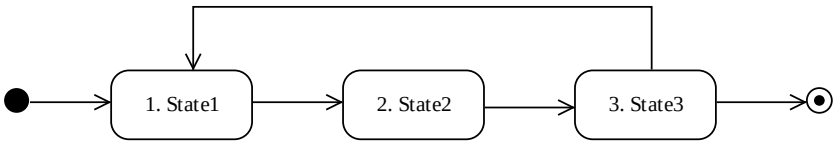

Figure 14 Normal behaviour

If we now add the “bad” behaviour to the state diagram we see that these are represented by ‘thick’ states. The interesting thing is that we find the already unwanted incidents back on the transitions from good states to bad states.

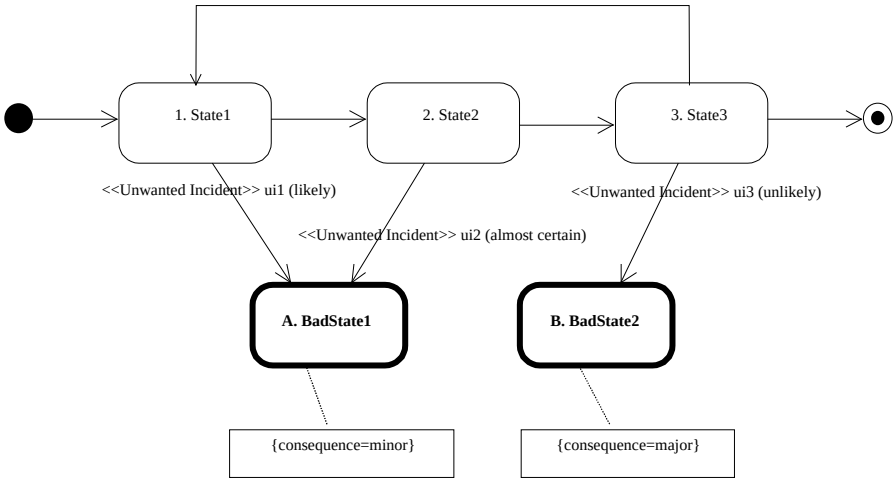

Figure 15 Bad states, their unwanted incidents and their consequences

Activity 3.2: Frequency evaluation

Frequency concern

To be able to calculate or simulate likelihood of occurrence for the identified consequences ("bad" states) we need to extend the diagram with a state transition probability matrix. The state transition probability matrix stores probabilities related to all possible transition within the state diagram, the probability for going from state i to state j. These probabilities will typically be obtained through empirical data or through subjective expert judgement or both. Figure 16 provides an example of such a transition probability matrix. A transition probability matrix will be based on a state diagram as the one in Figure 15.

		To State						
From State		**Begin**	**1**	**2**	**3**	**End**	**A**	**B**
	Begin	0	P_{Begin1}	0	0	0	0	0
	1	0	0	P_{12}	0	0	P_{1A}	0
	2	0	0	0	P_{23}	0	P_{2A}	0
	3	0	P_{31}	0	0	P_{3End}	0	P_{3B}
	End	0	0	0	0	0	0	0
	A	0	0	0	0	0	0	0
	B	0	0	0	0	0	0	0

Figure 16: Transition probability matrix

Sub-process 4: Risk evaluation

Risk evaluation is the process to determine risk management priorities by comparing the level of risk against predetermined standards, target risk levels or other criteria. The Risk evaluation subprocess is divided into five activities of which each has its own concern connected to it.

- **Activity 4.1: Determine level of risk - *Risk estimates concern*.** The risk estimates concern documents the results from the risk level determination.

- **Activity 4.2: Prioritise risks - *Risk priority concern*.** The risk priority concern documents the results from risk prioritisation.

- **Activity 4.3: Categorise risks - *Risk themes concern*.** The risk themes concern documents the results from risk categorisation.

- **Activity 4.4: Determine interrelationships among risk themes - *Risk theme relationship concern*.** The risk themes relationship concern documents the results from identification of interrelationships among risk themes.

- **Activity 4.5: Prioritise the resulting risk themes and risks - *Risk theme priority concern*.** The risk themes priority concern documents the results from the risk theme prioritisation.

Sub-process 5: Risk treatment

Risk treatment is the selection and implementation of appropriate options for dealing with risks. In this phase of the process decisions need to be taken about which treatments are affordable and which are not. These are important decisions that require a thorough understanding of the issues. These choices are often made by decision-makers that often have a position further away from the system and its characteristics. This is one of the main reasons that the results of the assessment need to be presented in a way that is also readable for these stakeholders.

Activity 5.1: Identify treatment options

Treatments concern

The treatment concern documents the results from the treatment identification. In connection with the treatments concern two tables are used. One is the Risk treatment table and the other is the Risk *theme* treatment table.
In the *enterprise viewpoint*, treatment options may be described as use cases preventing threats and protecting assets. This is illustrated in Figure 17. The viewpoint and diagrams used for describing the treatment options in more detail will depend on the type of the treatments.

- Security policies: Ponder (Damianou, 2000); class diagrams with OCL (OMG, 2001a) constraints; sequence, activity and state diagrams describing procedures (*various viewpoints*).
- Security requirements: Use case diagrams; class diagrams; sequence diagrams (*enterprise viewpoint*).
- Security architecture: OCL; class diagrams; state diagrams; object diagrams; collaboration diagrams (*engineering viewpoint*).
- Monitoring: State diagrams (*computational viewpoint*); deployment diagrams (*engineering viewpoint*).
- Testing: Sequence diagrams and TTCN test suites to describe the tests.

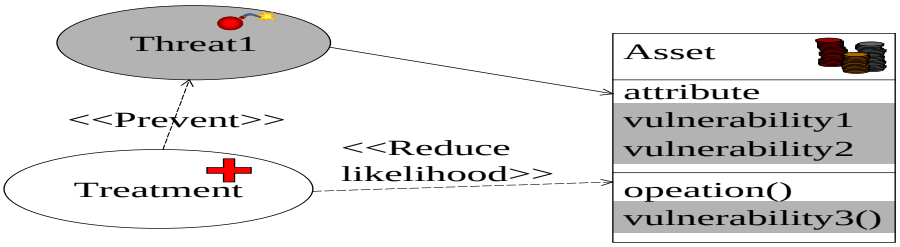

Figure 17: Treatment options

In the context of system development, the different types of treatments will be of different importance in the different phases of UP:

- Security requirements are the dominating treatment in the risk assessment carried out after the inception phase.
- Security architecture and testing are the dominating treatments in the risk assessment after the elaboration phase.
- Testing and security policies are the dominating treatments in the risk assessment after the construction phase.
- Security policies and monitoring are the dominating treatments in the risk assessment after the transition phase.

In addition possible treatments may be:

- New iterations on specific parts before a new phase is entered.
- Guidelines on what target of evaluation should contain in the risk assessment after the next phase.

One way of specifying security architecture is to use object or collaboration diagrams with tags (Wimmel, 2002). In Table 2 some of these tags are proposed. Figure 18 shows an example of the use the tag <critical>. Here the server and the transmission channel are critical, while the client is non-critical.

Figure 18: Example of use of <critical>

Table 2: Tags to be applied for objects or collaboration diagrams

Tag	Description
critical	This part of the system contains data or information that should be protected against unauthorised operations. May be used on both components and data channels.
noncritical	Does not contain data or information that need to be protected. All parts of the system that are not tagged are noncritical by default.
private	A private channel is a dedicated connection between two components, and may not be accessed by intruders. A channel without tag is private by default.
public	A public channel may be accessed by intruders and must be considered to have an interface that an intruder may exploit.
replace	Is used on replaceable components. A component is replaceable if an intruder may replace it by another component and fool the other components

	to communicate with it.
node	Is used on components that only contain non-replaceable components and private channels.
secret	A secret channel may not be read by an intruder, but the intruder may write to it.
auth	Is used on authentic channels. An authentic channel may not be written to by an intruder, but the intruder may read the channel.

Figure 19 shows an example of the use of <public>. When a channel is public, this is interpreted as it has an interface an intruder may exploit. This is illustrated in Figure 20.

Figure 19 Example of use of <public>

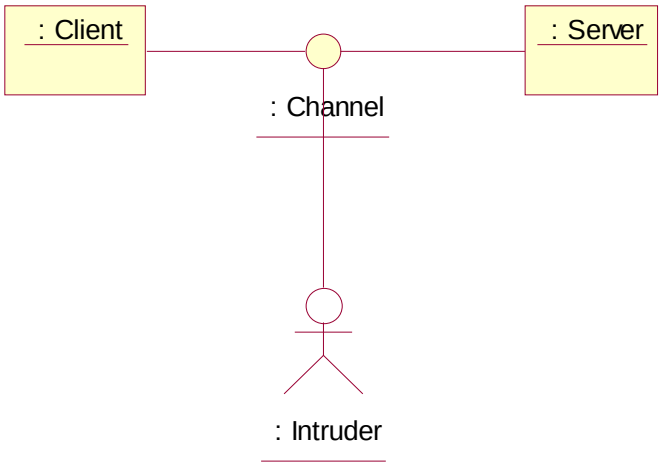

Figure 20 Interpretation of <public>

Figure 21 shows an example of the use of the tag <replace>. Here there is a possibility that the server may be replaced by a false server, and that the client is fooled to communicate with this one instead on the real server.

Figure 21 Example of use of <replace>

Activity 5.2: Assess alternative treatment approaches

Treatment assessment concern

The treatment assessment concern documents the results from the treatment assessment. For each risk and risk theme the evaluation of the treatment option and approach is documented in two tables. One is the Risk treatment evaluation table and the other is the Risk *theme* treatment evaluation table.

Treatment priority concern

The treatment priority concern documents the results from the treatment prioritisation. The risk treatment priority concern makes again use of two tables. The Risk treatment priority table and the Risk *theme* treatment priority table.

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 (CC) (CCO, 2002) represents the outcome of this work. The Common Criteria project harmonises the European "Information Technology Security Evaluation Criteria (ITSEC)" (Communications-Electronics Security Group, 2002), the Canadian "Canadian Trusted Computer Product Evaluation Criteria (CTCPEC)" and the American "Trusted Computer System Evaluation Criteria (TCSEC) and the Federal Criteria (FC)". Increasingly it is replacing national and regional criteria with a worldwide set accepted by the International Standards Organisation (ISO15408) (ISO/IEC, 1999). 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 provides a common set of requirements for assurance measures applied to the IT functions of IT products and systems during a security evaluation. CORAS addresses and develops concrete specification technology addressing specifically risk assessment.

Surety Analysis (SA) (Sandia National Laboratories, 2002), developed in Sandia National Laboratories, is a methodology based on the creation of an explicit model that covers several aspects of the system's behaviour. The modelling framework in SA is proprietary whereas CORAS uses the standardised RM-ODP as a common basis. SA supports modelling by means of basic techniques such as interaction and state and dataflow diagrams. CORAS aims to use the descriptive power of UML/OCL (Object Constraint Language) and to investigate its enhancement with aspects of other modelling paradigms specific to security modelling.

RSDS (Reactive System Design Support) is a tool-supported methodology developed by King's College London and B-Core UK, ltd. The methodology has been applied in the specification and risk analysis of reactive systems in automated manufacturing and chemical process control. Both RSDS and CORAS aim to integrate object-oriented modelling and risk analysis for critical systems. However, CORAS focuses on security risk assessment whereas current work on RSDS focuses on safety and reliability analysis.

The Control Objectives for Information and related Technology (COBIT) (Control Objectives for Information and related Technology) addresses the management of IT. COBIT and CORAS are orthogonal approaches. COBIT focuses on control objectives defined in a process-oriented manner following the principle of business re-engineering. The IT process of assessing risks satisfies the business requirement of supporting management decisions through achieving IT objectives and responding to threats by reducing complexity, increasing objectivity and identifying important decision factors. It is enabled by the organisation engaging itself in IT risk-identification and impact analysis. CORAS provides a tight integration of viewpoint-oriented modelling in the whole risk management process, including the sub-processes of risk identification and risk analysis.

CCTA Risk Analysis and Management Methodology (CRAMM) (Barber, 1992) 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.

CONCLUSIONS

Security and dealing with risks is often not getting the attention it deserves and is maybe even forgotten now and then. Probably apart from parachutists and basejumpers, most people will let security live its own life and just hope for the best. This natural attitude is probably necessary in our everyday life, but when it comes to IT-systems, and especially security critical IT-systems, such an attitude could be fatal. We therefore need to break this habit of living on hope and turn it into informed, calculated and constructive acting. To be able to do this, methods and tools are needed to guide us on our security analyses.

In this chapter we have given an insight into a methodology and a platform being developed in the CORAS project that provides guidelines on how risk assessment can become a natural part of the system development process. The length constraints imposed on this chapter limits us from presenting the full details but it is clear that the use of UML in a risk assessment context contributes to understanding and communication during the process. The combination of risk assessment with UP makes it possible to make security related adjustments during all phases of the development process. However, this does not keep us from applying risk treatments after the system has been finished. This means that the method is also applicable during maintenance of existing systems.

We focus in this chapter on the process part of CORAS, another important part of the methodology is the implementation through the CORAS platform. Making the methodology executable in the form of a computerised tool is critical to its practical use.

Based on the current developments in the security field and the great interest in the CORAS project, it looks like businesses are getting more aware of the importance of securing their IT-systems. There is a long road ahead in the field of IT-security, but with its methodology for model-based risk assessment of security critical systems, CORAS provides a solid step in the right direction.

REFERENCES

Australian/New Zealand Standard AS/NZS 4360 (1999). Risk management.

Barber, B., Davey, J. (1992). The use of the CCTA risk analysis and management methodology CRAMM. Proc. MEDINFO92, North Holland, 1589 –1593.

Bouti, A., Ait Kadi, D. (1994). A state-of-the-art review of FMEA/FMECA. International Journal of Reliability, Quality and Safety Engineering 1:515-543.

Clark, J. (November 1999). XSL transformations (XSLT) 1.0, World Wide Web Consortium recommendation REC-xslt.

Cockburn, A. (1997). Structuring use cases with goals. Journal of object-oriented programming, Sep/Oct: 35-40, Nov/Dec: 56-62.

Common Criteria Organisation (2002). Common Criteria for Information Technology Security Evaluation [On-line]. Available: http://www.commoncriteria.org

Communications-Electronics Security Group (2002). Information Security Evaluation Criteria [On-line]. Available: http://www.cesg.gov.uk/assurance/iacs/itsec/index.htm

Control Objectives for Information and related Technology. COBIT [On-line]. Available: http://www.isaca.org/ct_denld.htm

CORAS (2000). A platform for risk analysis of security critical systems. IST-2000-25031. (http://www.nr.no/coras/)
Curry, D., Debar Merrill Lynch, H. (December 28, 2001). Intrusion detection message exchange format (IDMEF). Working draft.
Damianou, N., Dulay, N., Lupu, E., & Sloman, M. (2000). Ponder: A Language for Specifying Security and Management Policies for Distributed Systems. The Language Specification - Version 2.2. Research Report DoC 2000/1, Department of Computing, Imperial College, London.
Houmb, S. H. (2002). Stochastic Models and Mobile E-Commerce: Are stochastic models usable in the analysis of risk in mobile c-commerce? Unpublished master's thesis, Østfold University College, Faculty of Computer Sciences, Halden, Norway.
IEC 1025: 1990 Fault tree analysis (FTA).
IEC 61508: 2000 Functional safety of electrical/electronic/programmable safety related systems.
ISO/IEC 10746, 1995: Basic reference model of Open Distributed Processing.
ISO/IEC TR 13335:2001: Information technology – Guidelines for the management of IT Security.
ISO/IEC (1999). Information Technology -- Security techniques -- Evaluation Criteria for IT Security ISO/IEC, 15408-1.
ISO/IEC 17799: 2000 Information technology – Code of practise for information security management.
Kruchten, P. (1999). The Rational unified process, an introduction. Addison-Wesley.
Littlewood, B. (1975). A reliability model for systems with Markov structure. Appl. Stat. 24:172-177.
OMG (2001a). Object Constraint Language Specification. Part of the UML specification.
OMG (2001b). Unified Modeling Language Specification. Version 1.4.
OMG (1999). XML Metadata Interchange (XMI).
Reactive System Design Support, RSDS [On-line]. Available: http://www.dcs.kcl.ac.uk
Redmill, F., Chudleigh, M., Catmur, J. (1999). Hazop and Software Hazop. Wiley.
Sandia National Laboratories. Surety Analysis [On-line]. Available: http://www.sandia.gov, accessed: 2002
Schneider, G., Winters, J. P. (1998). Applying use cases: a practical guide. Addison-Wesley.
Sindre, G., Opdahl, A. L. (2000). Eliciting security requirements by misuse cases. In Proc. TOOLS_PACIFIC 2000. IEEE Computer Society Press, 120-131.
Wimmel, G., and Wisspeintner, A. (2001). Extended description techniques for security engineering, IFIP/SEC 2001 – 16th International Conference on Information Security, Kluwer.
World Wide Web Consortium (6 October, 2000). Extensible Markup Language (XML) v1.0, W3C Recommendation, Second Edition.