STFY40 A03062 – Unrestricted

REPORT

An Experience Repository Supporting Security Risk Analysis

Folker den Braber, Chingwoei Gan, Mass Soldal Lund,
Fredrik Seehusen, Ketil Stølen, Fredrik Vraalsen

SINTEF Telecom and Informatics

November 2003

SINTEF

SINTEF Telecom and Informatics

Address: NO-7465 Trondheim
NORWAY
Location Trondheim:
S.P. Andersens v 15
Location Oslo:
Forskningsveien 1
Telephone: +47 73 59 30 00
Fax: +47 73 59 43 02

Enterprise No.: NO 948 007 029 MVA

SINTEF REPORT

TITLE

An Experience Repository Supporting Security Risk Analysis

AUTHOR(S)

Folker den Braber, Chingwoei Gan, Mass Soldal Lund, Fredrik Seehusen, Ketil Stølen, and Fredrik Vraalsen

CLIENT(S)

REPORT NO.	CLASSIFICATION	CLIENTS REF.	
STF40 A03062	Unrestricted		
CLASS. THIS PAGE	ISBN	PROJECT NO.	NO. OF PAGES/APPENDICES
Unrestricted	82-14-03107-9	40332800	19
ELECTRONIC FILE CODE		PROJECT MANAGER (NAME, SIGN.)	CHECKED BY (NAME, SIGN.)
rapport_forside		Ketil Stølen	Jan Øyvind Aagedal
FILE CODE	DATE	APPROVED BY (NAME, POSITION, SIGN.)	
	2003-11-10	Bjørn Skjellaug, Research director	

ABSTRACT

This paper presents an experience repository aiming to support the reuse of generic aspects of security risk analyses documentation. The generic aspects address three main purposes. They are used to (1) characterize and document the target of evaluation, its context and the assumptions on which the analysis is based, (2) specify undesirable behavior, impacts and scenarios resulting from security breaches and threats, (3) document security risk analysis results. The generic aspects are documented in so-called experience packages. An experience package is either constructive or supportive. A constructive experience package may be instantiated, specialized, extended or adjusted into concrete risk analysis documentation. A supportive experience package, supports the process of documenting by capturing methodological aspects in the form of checklists, patterns, and manual or automated procedures. Each experience package is decomposed into 22 concerns, each of which addressing a specific aspect in the overall security risk management process. Concerns consist of elements, examples of which are UML diagrams, table formats, domains of risk values, or sets of guidewords for structured brainstorming. The repository currently contains 139 elements. The elements have been extracted from major security analyses within e-commerce and telemedicine.

KEYWORDS	ENGLISH	NORWEGIAN
GROUP 1	Information technology	Informasjonsteknologi
GROUP 2	Risk analysis, experience	Risikoanalyse, erfaring
SELECTED BY AUTHOR	Security assessment, reuse	Sikkerhetsanalyse, gjenbruk
	Modelling	Modellering
	Repository	Arkiv

An Experience Repository Supporting Security Risk Analysis

Folker den Braber[1], Chingwoei Gan[2], Mass Soldal Lund[1], Fredrik Seehusen[1], Ketil Stølen[1], and Fredrik Vraalsen[1]

[1] SINTEF Telecom and Informatics
Postbox 124, Blindern, N-0314 Oslo, Norway
{fbr,msl,fse,kst,fvr}@sintef.no
[2] Department of Electronic Engineering, Queen Mary University of London
E1 4NS, United Kingdom
chingwoei.gan@elec.qmul.ac.uk

Abstract. This paper presents an experience repository aiming to support the reuse of generic aspects of security risk analyses documentation. The generic aspects address three main purposes. They are used to (1) characterize and document the target of evaluation, its context and the assumptions on which the analysis is based, (2) specify undesirable behavior, impacts and scenarios resulting from security breaches and threats, (3) document security risk analysis results. The generic aspects are documented in so-called experience packages. An experience package is either constructive or supportive. A constructive experience package may be instantiated, specialized, extended or adjusted into concrete risk analysis documentation. A supportive experience package, supports the process of documenting by capturing methodological aspects in the form of checklists, patterns, and manual or automated procedures. Each experience package is decomposed into 22 concerns, each of which addressing a specific aspect in the overall security risk management process. Concerns consists of elements, examples of which are UML diagrams, table formats, domains of risk values, or sets of guidewords for structured brainstorming. The repository currently contains 139 elements. The elements have been extracted from major security analyses within e-commerce and telemedicine.

1 Introduction

The use of UML has in recent years gone beyond the traditional domains of business modeling and software development. The recently concluded CORAS project [1] has developed a tool-supported UML methodology for model-based security risk analysis (in the sequel referred to as security analysis). CORAS addresses security-critical systems in general, but places particular emphasis on 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 [12]. An IT system in our approach

is not just technology, but also the humans interacting with the technology, and all relevant aspects of the surrounding organization and society.

An important aspect of the CORAS project has been the practical use of UML to support security management in general, and security analysis in particular. The CORAS security analysis methodology makes use of UML models for three different purposes:

- *To describe the target of evaluation at the right level of abstraction.* To properly analyze security, technical system documentation is not sufficient; a clear understanding of system usage and its role in the surrounding organization or enterprise is just as important. UML allows these various aspects to be documented in a uniform manner.
- *To facilitate communication and interaction between different groups of stakeholders involved in a security analysis.* One major challenge when performing a security analysis is to establish a common understanding of the target of evaluation, threats, vulnerabilities and security risks among the stakeholders participating in the analysis. CORAS has developed a UML profile aiming to facilitate improved communication during security analysis, by making the UML diagrams easier to understand for non-experts, and at the same time preserving the well-definedness of UML. The profile has been integrated as part of the proposed UML Profile for Modeling Quality of Service and Fault Tolerance submitted to OMG in August this year [21].
- *To document security analysis results and the assumptions on which these results depend to support reuse and maintenance.* Security analyses are costly and time consuming and should not be initiated from scratch each time we analyze a new or modified system. Documenting analyses using UML supports reuse of analysis documentation, both for systems that undergo maintenance and for new systems, if similar systems have been analyzed earlier.

A security analysis mirrors in many respects a requirements analysis. Users, system experts, decision makers and analysts are required to communicate in order to capture and analyze the functionality and characteristics of the system in question. Whereas the desired behavior of the system is the focus during a requirements analysis, a security analysis focuses on the *undesirable* behavior. Together these views complement each other by documenting both the 'good' and the 'bad' side of the analyzed system. This duality is illustrated by Fig. 1.

In the case of requirements analysis we employ use-case diagrams to capture desired behavior; in the case of security analysis we employ mis-use-case diagrams [18] for the same purpose. Of course there is more to security analysis than describing threat scenarios, but to assign risk values to risks we need to understand under which circumstances risks may appear and which consequences they may have on system assets. Specification languages such as UML has an important role to play here.

A significant part of the results of a security analysis carried out on an IT-system will typically have a certain general character. To avoid starting from scratch for every new analysis, it is important to gather these general aspects.

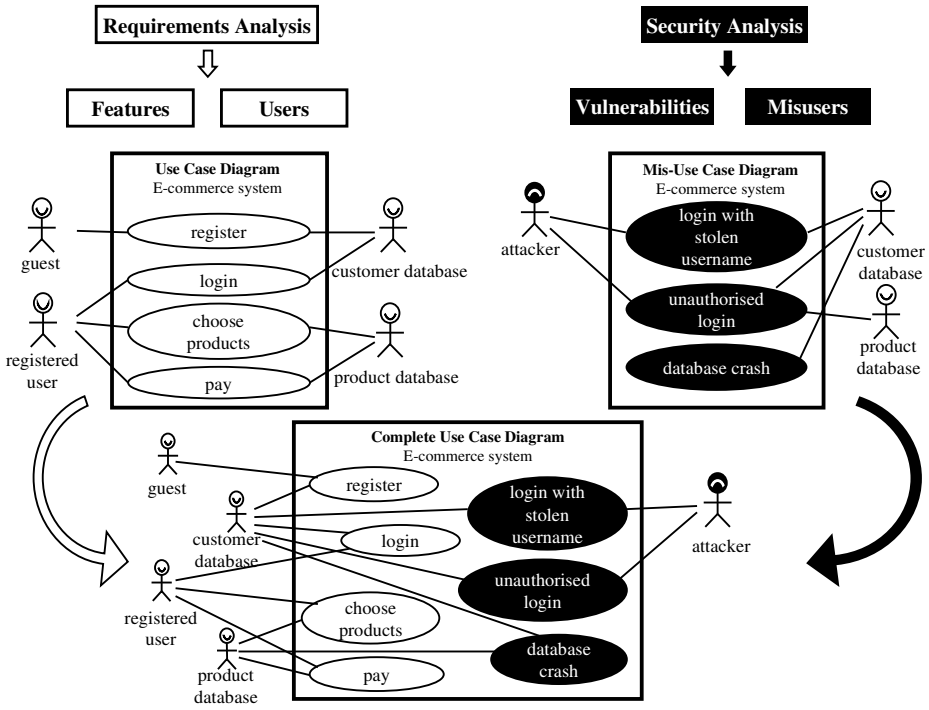

Fig. 1. Composed use-case diagram

Such generic aspects are captured in so-called reusable elements which may take the form of UML-diagrams, tables and natural language. These elements are often related and this motivates the grouping of related elements into experience packages.

The process of risk analysis may produce a large volume of information, and dealing with this information is a non-trivial task. Furthermore, the security analysis process is highly elaborate. This motivates the need for a computerized repository which effectively manages the extraction, reuse and maintenance of experience packages. Such a repository has been development within the CORAS project. The objective of this paper is to present the underlying structure and design of this repository.

The remainder of the paper is structured into five sections. Section 2 starts by defining a structure for experience packages. It also explains the difference between constructive and supportive packages. Section 3 describes the existing library of experience packages and examples of how the elements were extracted. Section 4 presents a number of usage scenarios. Section 5 provides an overview of the computerized experience repository and its role in the CORAS platform. Section 6 provides a brief summary and relates the proposed approach to known approaches from the literature.

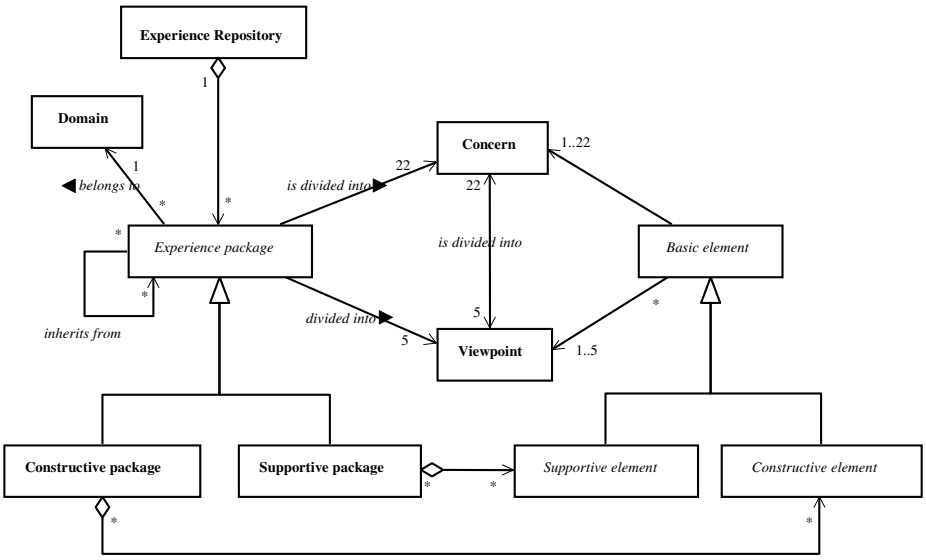

Fig. 2. High-level specification of repository and experience packages

2 Structuring Experience Packages

Reusable elements which logically belong together are grouped into experience packages. Each element may be applicable in several settings and can therefore be included in several experience packages. The experience packages belong to a specific domain, e.g., e-commerce or telemedicine. The elements in each experience package are divided into 22 concerns and 5 viewpoints. The concerns correspond to the various activities in the CORAS model-based security analysis methodology [3], and the viewpoints are taken from the ISO standard RM-ODP (Reference Model for Open Distributed Processing) [11].

One example of a concern is the *assets* concern. This concern contains elements that may be used to support and assist asset identification. Asset identification is the activity of identifying anything that stakeholders find valuable and which must be protected. An example of a reusable element in the asset concern is presented in Sect. 4.2. Each element belongs to one or more concerns and one or more viewpoints. The elements of the assets concern typically belong to the enterprise viewpoint which is one of the five viewpoint defined by RM-ODP.

Fig. 2 shows a UML specification of the repository structure. Experience packages are classified as either constructive or supportive packages, which contain constructive and supportive elements, respectively. This classification is based on whether or not the element can be used directly as part of a security analysis result. Constructive elements become an actual part of the generated results during a security analysis, while supportive elements provide support in generating these results by capturing methodological aspects and patterns.

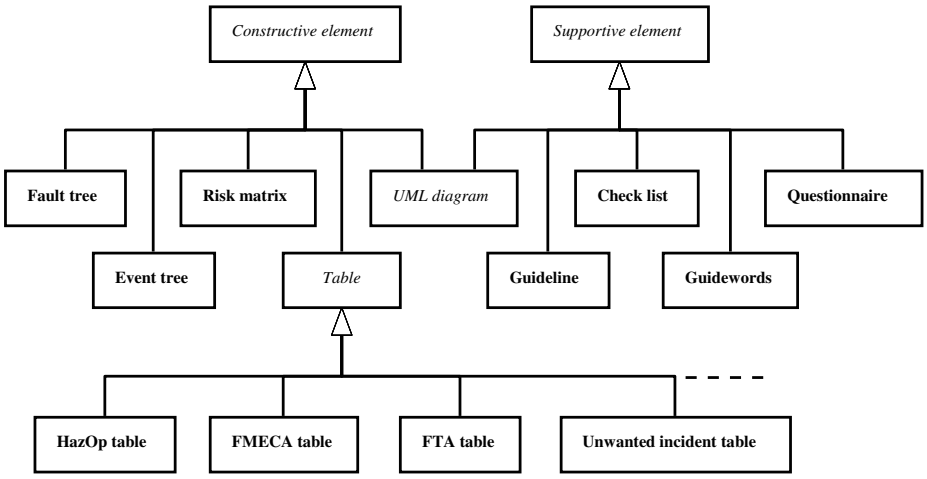

Fig. 3. Element types

Constructive elements basically provide starting points when performing the security analysis. They often have the form of a diagram, list or table where specific open fields need to be filled in. In other words, constructive elements provide outlines to make the generation of security analysis results as easy as possible. Supportive elements on the other hand have a more guiding character. They provide a recipe, or helping hand, during either the instantiation of constructive reusable elements or the making of new reusable elements.

Notice that constructive and supportive elements can also be combined. A constructive element providing a skeleton of an element containing security analysis results, could be easier to fill in with the help of a supportive element. Fig. 3 shows the different types of constructive and supportive elements that are used in the experience packages.

3 Library of Experience Packages

The computerized repository developed under the CORAS project currently contains 139 reusable elements. In the following we describe in what way, and from which sources they were extracted, in addition some statistics are provided on the kind of elements that were found.

The 139 reusable elements were basically extracted from three sources: the security analysis documentation resulting from two major security analysis trials carried out during the CORAS project, in addition to the experiences from the development of the CORAS methodology itself.

One trial addressed an e-commerce platform, the other trial analyzed a telemedicine application. The e-commerce platform has been developed in the context of the R&D project EP-27046-ACTIVE, co-funded by the European Commission under the ESPRIT program. The e-commerce platform [8] is based

on Java and Internet technology and supports integrated retail services. The platform addressed in the CORAS trial constitutes a core part of the ACTIVE system. See [16] for more details, experiences and results.

The other trial addressed two telemedicine applications deployed within the greek regional health network on Crete that links hospitals and public health centers [19, 20].

3.1 Trial generated elements

Results from the trials come in different forms; UML-diagrams, tables, check lists and natural language, to mention the most important ones. Table 1 provides a classification of the 139 identified elements. The table distinguishes between constructive (con) and supportive (supp) elements, in addition to the general, e-commerce and telemedicine domains.

38 of these elements are UML-diagrams, 58 are tables and check lists, and 43 are expressed in natural language. The UML diagrams are either expressed in standard UML [15] or in the CORAS UML profile for security analysis [9, 21].

In both trials UML-diagrams were used to describe the target of evaluation. The constructive tables under the general category are standard table formats strongly related to the CORAS methodology. The questionnaires, procedures and guideline elements, helping the analysts during the analysis, are typically supportive elements.

During the trials, analysis documentation is produced in the different forms described above. Most of the documentation will have a specific character. Some aspects though, will have a certain generic flavor, making them potentially reusable for later analyses. These aspects are the source for generating the reusable elements, like the 139 elements mentioned above. Giving some insight into this process, Fig. 4 illustrates the extraction of four reusable elements from a threat diagram.

The core diagram illustrates how four attackers may interact with three assets via two mis-use-cases. From this diagram four reusable elements are extracted. Three of these are supportive, in the sense that they provide check lists. Generalization is used to classify the different types of information, threats and attackers. The constructive element provides a general pattern for the construction of threat diagrams.

4 Usage Scenarios

As mentioned previously, the CORAS risk management process is supported by 22 concerns. In the following we demonstrate how elements may be reused in a security analysis of a contrived Internet application named Internet Music Shop (IMS) with respect to two of these concerns.

The example also demonstrates the CORAS UML profile for security analysis [9, 21]. This profile makes extensive use of stereotypes with easy-to-understand

Table 1. Element Classification

	General		E-commerce		Telemedicine		Total
	Con	Sup	Con	Sup	Con	Sup	
Class diagram			8		1		**9**
Collaboration / Deployment			2		1		**3**
Use-case diagram			5		4		**9**
Threat diagram	1		1		2		**4**
Sequence			5				**5**
Activity diagram			5				**5**
SWOT diagram			1		2		**3**
Table	33						**33**
Checklist		7		8		10	**25**
Questionnaire		3					**3**
General guideline	4	8					**12**
Procedures		11					**11**
Definition	6			3	8		**17**
	44	**29**	**27**	**11**	**18**	**10**	**139**

icons for representing various concepts of relevance for security analysis documentation. The motivation for the use of such stereotypes are twofold: (1) UML diagrams are used as medium for the communication between participants in a security analysis, which also include people that are not necessarily experts on modeling or security analysis but rather domain experts or experts on the target of evaluation. The use of stereotypes with icons support this communication. The domain experts are not themselves required to draw these diagrams, but they need to understand them. (2) A security analysis usually involves a good deal of swapping between diagrams and tables, and because of this we sometimes want to extract data from a diagram and put them in a table. This is made simpler by marking the data in diagrams with stereotypes.

4.1 Internet Music Shop

The Internet Music Shop (IMS) is an internet portal where users may register and buy single songs. The two main functions of IMS are (1) *register*, which enables users to register an account and get a customer number and a password, and (2) *login* which enables registered users to log on to the system and buy requested songs.

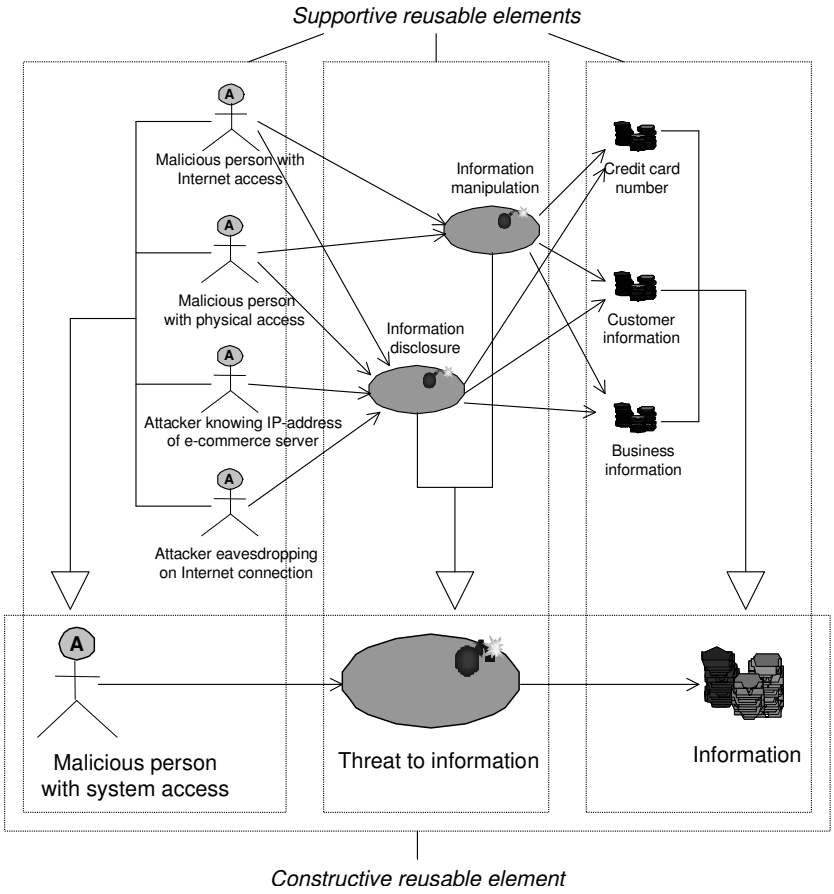

Fig. 4. Resulting elements leading to reusable elements

The UML sequence diagram in Fig. 5 shows a normal interaction between a customer and IMS. After the customer has registered with a name and a credit card number, the IMS provides the customer with a customer number and a password. The customer then logs in whereupon IMS returns a list of songs and a session number which is used by throughout the rest of the session. The box labeled *loop* in the figure indicates that a customer may request and pay for songs several times during a session. The box labeled *alt* specifies that after a song has been requested either (1) an insufficient credit message is returned *or* (2) a confirm messages is returned if the customer has sufficient credit to pay for the requested song.

The state diagram in Fig. 5 shows the behavior of IMS after a customer has logged in.

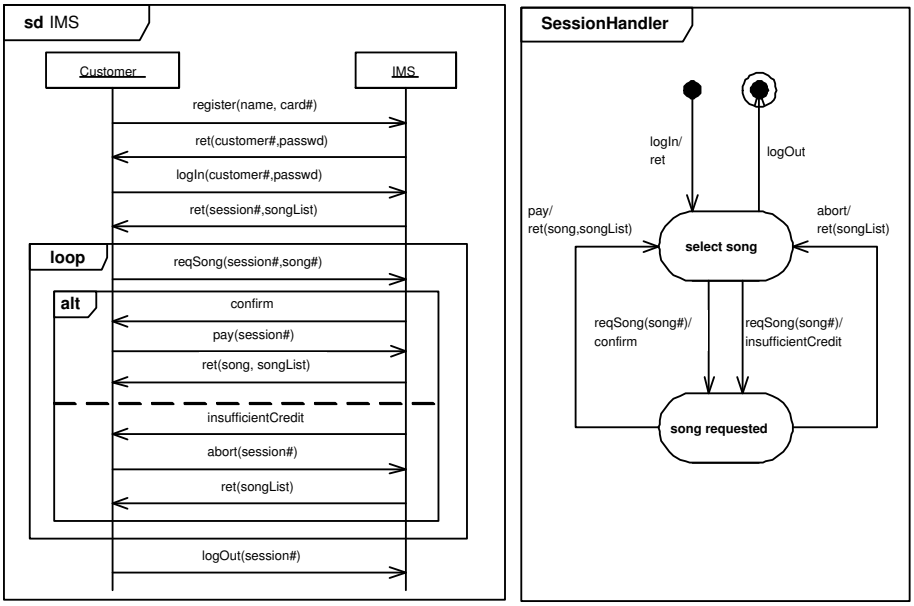

Fig. 5. Sequence diagram and state diagram

4.2 Assets Concern

An important part of a security analysis is to identify the assets of relevance for the target of evaluation; if there is nothing of value, there is really nothing to analyze. Because of similarities in classes of security-critical systems, like Internet services that include payments, generic assets are easily identified. Such generic assets are organized in specialization trees and these trees will in the asset identification play the role as check lists or guidelines. An example is shown in Fig. 6.

Generic assets may be included in the documentation of asset identification, alongside with more specific assets. The owner and value of assets are not necessarily generic and all generic assets will not always be relevant, so the asset trees are mainly help and inspiration for the team carrying out the security analysis and should not necessarily be included in the security analysis documentation as a whole.

Looking at Fig. 6 again, we can identify 5 assets that are relevant with respect to IMS. These are *customer personal data*, *credit card number*, *order history*, *customer list* and *sales statistics*.

Identified assets are documented in asset diagrams where also the stakeholders to whom the assets belong are represented. The assets are assigned values in specific asset table. An example of an asset diagram related to IMS is illustrated in Fig. 7. Here all relevant information assets that were identified in the reusable element shown in Fig. 6 are included in addition to some new assets.

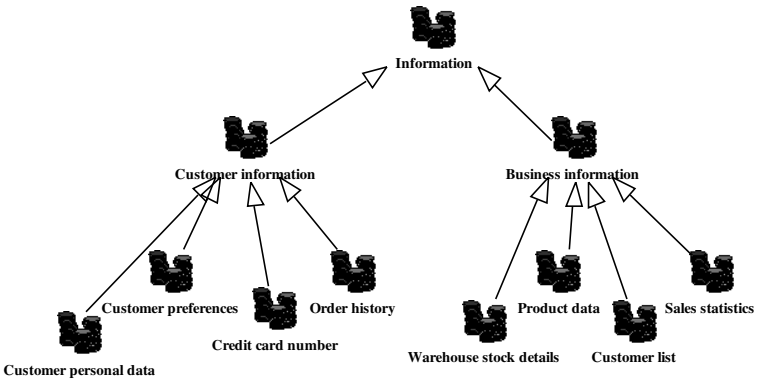

Fig. 6. Hierarchy of generic information assets from the e-commerce domain

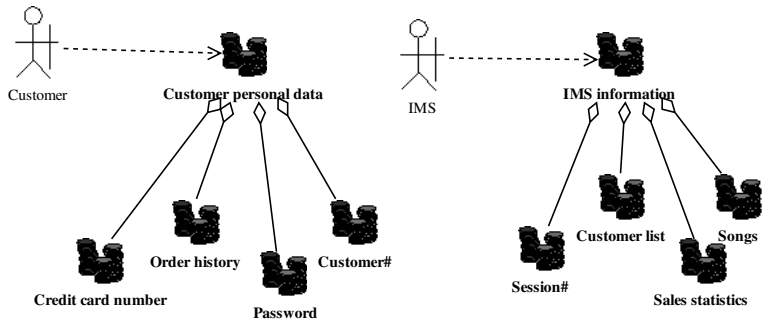

Fig. 7. Diagram showing identified information assets for IMS

4.3 Threats Concern

Much of a security analysis is about identifying threats to assets. One of the techniques applied for threat identification in our methodology is Hazard and Operability analysis (HazOp) [17]. HazOp is in general a brainstorming activity structured by the use of guidewords. In addition to providing structure, the purpose of the guidewords is to help the participants in a HazOp session – usually a mixture of HazOp experts, security experts and experts on the target of evaluation – to identify as many of the existing threats as possible. The outcome of a HazOp analysis is a table, filled in during the session, that organizes the identified threats.

The experience repository provides a number of predefined sets of guidewords as supportive elements. To put even more structure to a HazOp and to increase

```
For each input use case diagram
  For each use case
    For each sequence diagram specifying the use case: Transfer
        information from the sequence diagram to the HazOp table.
      For each asset in the sequence diagram (represented either as
        message or as object): Insert asset in the asset column together
        with an ID in the ID column
        CASE: Asset represented as message (comment: an asset can be
          represented as several messages with the same name)
          ...
        CASE: Asset represented as object
          For each input or output event on its lifeline
            For each guideword in the guideword list
              For each attribute in the attribute list: Identify threats to event
                For each identified threat: Identify unwanted incident
                  For each unwanted incident: Describe consequence if possible
```

Fig. 8. Procedure for applying use case and sequence diagrams in HazOp

Table 2. HazOP table

ID	**Stake-holder**	**Asset**	**Item**	**Guide-word**	**Threat-agent**	**Threat-scenario**	**Unwanted incident**
...	...	...	...	...	...	...	...
1.3.13	Customer	Password	ret	dis-closure	*Eaves-dropper*	*Password stolen*	*Misuse of account*
...	...	...	...	...	...	...	...
2.2.20	IMS	IMS money	abort	other msg	*Hacker*	*Unfair exchange*	*Cust. gets song for free*
...	...	...	...	...	...	...	...

the understanding of the analyzed system, the security analysis team may use parts of system documentation, for example UML sequence diagrams, as input to a HazOp.

In order to identify threats based on the diagram in Fig. 5, a security analysis team would combine this with a set of guidewords, for example {*disclosure, other msg*}, in correspondence with the procedure of Fig. 8. The result will clearly be a very long table, but not all the rows will be interesting. The team will go through the table and identify the rows that might represent a threat, and in case, what kind of threat.

Table 2 shows a small sample of the rows in the resulting table. The italic entries are filled in by the team, while the rest is the result of applying the procedure together with the sequence diagram and the set of guidewords. Application of the procedure can be automated, saving the team for a considerable amount of work.

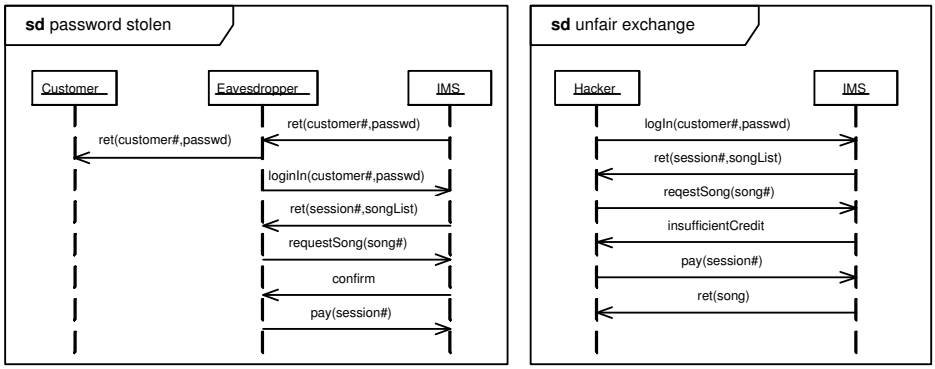

Fig. 9. Threat scenario sequence diagrams

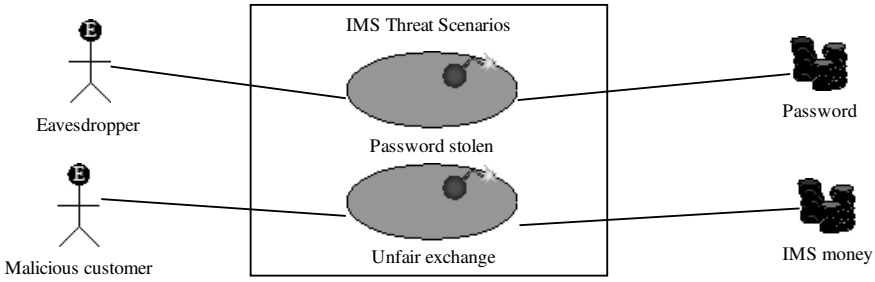

Fig. 10. Threat scenario overview

The two threat scenarios identified in the HazOp table are documented as sequence diagrams in Fig. 9. The left most diagram illustrates what may happen if the *ret* message is disclosed by an eavesdropper. The eavesdropper may then use the disclosed information to log in and buy songs on IMS. The right most diagram specifies what happens if a malicious customer sends a *pay* message instead of an *abort* message after the insufficient credit message has been received. The customer would then get a song for free. Please consult the state diagram in Fig. 5 to see that this situation can occur. This state diagram is, of course, poorly designed and it would have to be modified. An overview of the threat scenarios is visualized in the threat scenario diagram in Fig. 10.

4.4 Summary

Considering the above examples, it is clear that there are several kinds of reusable elements and different ways of applying them. In this section we have seen examples of some usage scenarios:

– Generic assets from a check list (supportive reusable element) were used to inspire the construction of an asset diagram. This can be generalized to other generic types: stakeholders, threats, treatments, etc.
– A HazOp table (constructive reusable element) was filled in and added to the security analysis documentation. This can clearly be generalized to all kind of tables.
– The table (constructive reusable element) was partially filled in by use of a procedure (supportive reusable element) applying a set of guidewords (supportive reusable element) and a UML sequence diagram. Similar procedures have been defined for other kind of security analysis methodology and kinds of UML diagrams. For example do asset diagrams (as in Fig. 7) provide the basis for filling in much of an asset table automatically.
– A threat was modeled in a threat diagram using a threat scenario template (constructive reusable element) that was specialized to represent an identified threat.

5 Computerized Repository

As should be clear by now, the security analysis process is highly elaborate and involves documentation of many different kinds. An experience repository to support this process must deal with large volumes of information. Computerized support for managing the extraction, maintenance and reuse of experience packages is highly required.

A computerized repository based on the principles outlined above has been made publicly available as open source [5]. In the following we describe the overall structure and design of this repository, and how it is integrated in the overall CORAS platform provided for security analysis.

During a security analysis, a lot of valuable information is generated. Future access to the information needs to be guaranteed, and the information needs to be shared between various modeling and security analysis tools. In addition, it is important that information stored in different places is consistent.

Though the security management process is model-driven, a UML repository for security analysis data does not deal exclusively with models, but must also be able to include tables and other constructs such as event trees [6] and fault trees [10]. To satisfy this requirement, the repository is based on standardized, XML-based data formats, such as XMI for the interchange of UML models.

Our UML repository is implemented on top of the open-source XML database eXist [14]. An XML database provides many of the features found in conventional databases: storage (XML documents), schemas (XML schema, DTDs), query languages (XQuery, XPath, XQL etc.), programming interfaces (SAX, DOM, JDOM), and so on.

Fig. 11 shows an overview of the CORAS platform. The platform is used to store the results from ongoing and completed security analyses, as well as the reusable elements and experience packages. These are stored in two separate repositories, the Assessment repository for the analysis results, and the Experience repository for the reusable elements.

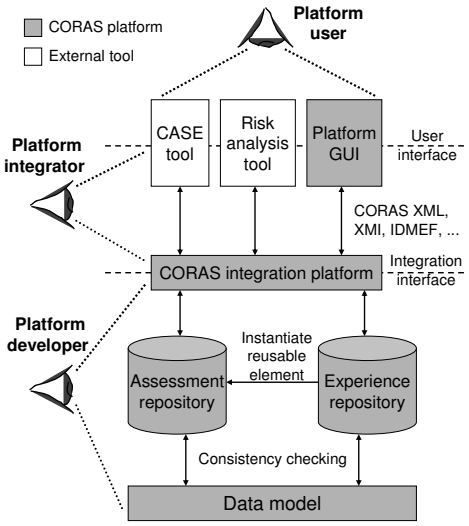

Fig. 11. Overview of CORAS platform

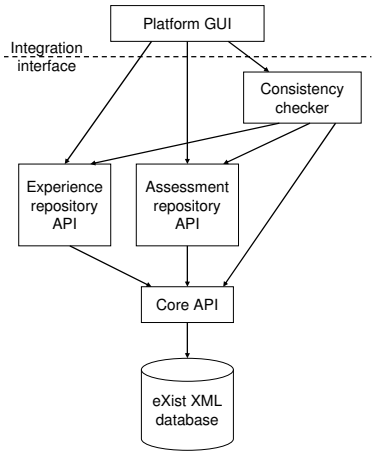

Fig. 12. Integration layer and platform

The platform GUI provides the end-user with administrative functionality, such as creating new security analysis projects and managing the reusable elements and experience packages.

Many entities of a security analysis, e.g. assets, stakeholders and threats, appear in several different places in the analysis results. It is therefore important to ensure that the various security analysis results are mutually consistent with regards to these entities. To do this, we designate certain results from a security analysis as *authorative* sources of information regarding the various data entities, e.g. the stakeholder table is the authorative source of information for stakeholder data. This authorative information is extracted from these results and represented in an internal data model, and is then later used for consistency checking, e.g. to ensure that the various security analysis results are consistent with the set of stakeholders defined by the stakeholder table. A similar approach is followed to ensure consistency between reusable elements in the experience packages.

A wide variety of UML modeling tools and risk analysis tools exist and are in use by security analysts and system engineers today. It is therefore important for our platform to provide flexible support for integration with external tools. To this end, the platform provides an integration layer with a defined API which tools can use to integrate with the platform, utilizing standardized XML formats for data integration. Fig. 12 shows the components of this integration layer and the platform in more detail. Logically, the platform contains the two repositories as well as the internal data model shown in Fig. 11.

Fig. 11 defines three different viewpoints or roles. The platform user represents the end-user, e.g. a security analyst who uses the CORAS platform together

with various modeling and security analysis tools to support the security analysis process. The platform integrator is responsible for integrating the various external tools with the CORAS platform, using the integration interface. The platform developer is responsible for implementing the CORAS platform itself, making sure that it provides the functionality specified by the integration interface.

Many modern UML modeling tools and security analysis tools are able to readily generate textual notation of a table or model in XML. Semantic interoperability between UML models and security analysis documents is made possible by exploiting a plethora of XML technologies and tools. For instance, consistency checking between different elements in the repository is implemented using XSLT. The clear separation of content from style, the powerful search mechanism, the enhanced portability and the interoperability of XML have proven to be very helpful for our purposes. Such an XML-based UML repository can manage both textual - risk tables, procedures etc. - and non-textual documents - fault tree diagrams, UML class diagram etc.

6 Conclusions

We have presented the overall design and architecture of a computerized experience repository aiming to support security analysis. In particular, we have specified the storage structure and explained the difference between constructive and supportive elements. We have also presented statistics with respect to the library of reusable elements generated so far, and described from what context and how they were extracted. Furthermore, we have illustrated their intended role in a number of usage scenarios. Finally, we have described the overall structure and architecture of the computerized repository, and outlined its integration in the CORAS platform.

For details on the overall CORAS approach we refer to [1, 3, 7]. The CORAS UML profile is further described in [9, 21]. Experiences from the use of CORAS in major trials within e-commerce and telemedicine are documented in [16] and [19], respectively.

There are other approaches to model-based risk assessment; see for instance CRAMM [2], ATAM [4], SA [22] and RSDS [13]. The particular angle of the CORAS approach with its emphasis on security analysis tightly integrated in a UML and RM-ODP setting is however new. In particular, the issue of maintenance and reuse of analysis results has received very little attention in the literature.

Acknowledgements

The research on which this paper reports has partly been funded by the Research Council of Norway projects COBRA (152209/431), SECURIS (152839/220) and partly by the 5th Framework EU project CORAS (IST-2000-25031). The CORAS consortium consists of eleven partners from four countries: CTI (Greece), FORTH

(Greece), IFE (Norway), Intracom (Greece), NCT (Norway), NR (Norway), QMUL (UK), RAL (UK), SINTEF (Norway), Solinet (Germany) and Telenor (Norway). The results reported in this paper have benefited from the joint efforts of the CORAS consortium.

References

1. Aagedal, J. Ø., den Braber, F., Dimitrakos, T., Gran, B. A., Raptis, D., Stølen, K.: Model-based risk assessment to improve enterprise security. In: Proc. EDOC2002. IEEE Computer Society (2002) 51–62
2. Barber, B., Davey, J.: The use of the CCTA risk analysis and management methodology CRAMM. In: Proc. MEDINF. North Holland (1992) 1589–1593
3. den Braber, F., Dimitrakos, T., Gran, B.A., Lund, M.S., Stølen, K., Aagedal, J.Ø.: The CORAS methodology: model-based risk management using UML and UP. Chapter in book titled UML and the Unified Process. IRM Press (2003)
4. Clements, P., Kazman, R., Klein, M.: Evaluating software architectures: methods and case studies. Addison-Wesley (2002)
5. The CORAS project, http://coras.sourceforge.net, 2003.
6. Cross, A.: Fault trees and event trees. In: A.E. Green, editor, High Risk Safety Technology. John Wiley & Sons, New York (1982) 49–65
7. Dimitrakos, T., Ritchie, B., Raptis, D., Aagedal, J.Ø., den Braber, F., Stølen, K., Houmb, S.-H.: Integrating model-based security risk management into eBusiness systems development: The CORAS approach. In: Proc. I3E2002. Kluwer (2002) 159–175
8. EP-27046-ACTIVE: Final Prototype and User Manual. D4.2.2, Ver. 2.0, (2001-02-22)
9. Houmb, S.-H., den Braber, F., Lund, M.S., and Stølen, K.: Towards a uml profile for model-based risk assessment. In: Proc. UML'2002 Satellite Workshop on Critical Systems Development with UML. Munich University of Technology (2002), 79–91
10. IEC 1025: 1990 Fault tree analysis (FTA).
11. ISO/IEC 10746: 1995: Basic reference model for open distributed processing (1995)
12. ISO/IEC TR 13335-1:2001: Information technology Guidelines for the management of IT Security Part 1: Concepts and models for IT Security (2001)
13. Lano, K., Androutsopoulos, K., Clark, D.: Structuring and design of reactive systems using RSDS and B. In: Proc. FASE 2000, LNCS 1783. Springer (2000) 97–111
14. Meier, W.: eXist: An Open Source Native XML Database. In: Akmal B. Chaudri, Mario Jeckle, Erhard Rahm, Rainer Unland (Eds.): Web, Web-Services, and Database Systems. NODe 2002 Web- and Database-Related Workshops, Erfurt, Germany, October 2002. Springer LNCS Series, 2593 (2002)
15. OMG-UML. Unified Modeling Language Specification, version 1.4. (2001)
16. Raptis, D., Dimitrakos, T., Gran, B.A., Stølen, K.: The CORAS approach for model-based risk analysis applied to the e-commerce domain. In: Proc. Communication and Multimedia Security (CMS-2002). Kluwer (2002) 169–181
17. Redmill, F., Chudleigh, M.F. and Catmur, J.R.: Principles underlying a guideline for applying HAZOP to programmable electronic systems. Reliability Engineering and System Safety (1997)
18. Sindre, G., Opdahl, A. L.: Eliciting security requirements by misuse cases. In: Proc. TOOLS-PACIFIC 2000. IEEE Computer Society Press (2000) 120–131

19. Stamatiou, Y.C., Henriksen, E., Lund, M.S., Mantzouranis, E., Psarros, M., Skipenes, E., Stathiakos, N., Stølen, K.: Experiences from using model-based risk assessment to evaluate the security of a telemedicine application. In: Proc. Telemedicine in Care Delivery (TICD 2002). (2002) 115–119
20. Stamatiou, Y.C., Skipenes, E., Henriksen, E., Stathiakis, N., Sikianakis, A., Charalambous, E., Antonakis, N., Stølen, K., den Braber, F., Lund, M.S., Papadaki, K., Valvis, G.: The CORAS approach for model-based risk management applied to a telemedicine service. To appear in: Proc. Medical Informatics Europe (MIE'2003). (2003)
21. UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms, Revised submission. OMG Document: realtime/2003-08-06, 2003.
22. Wyss, G. D., Craft, R. L., Funkhouser, D. R.: The use of object-oriented analysis methods in surety analysis. SAND Report 99–1242. Sandia National Laboratories (1999)