Building an Experience Factory for a Model-based Risk Analysis Framework

Chingwoei Gan, Eric Scharf

Department of Electronic Engineering,
Queen Mary University of London
E1 4NS, United Kingdom
{chingwoei.gan, e.m.scharf}@elec.qmul.ac.uk

Abstract: This paper describes the integration of an experience factory in a model-based risk analysis framework called CORAS[1]. CORAS aims at developing a new model-based risk analysis framework for security critical application. The framework's cornerstone of combining methods for risk analysis of critical systems and semiformal modelling methods in a tool-supported environment targeting openness and interoperability has not been tried before. We adapted the experience factory concept to fit our needs in documenting and exploiting the risk analysis results, as well as to support the structured process of risk management. Our internet-enabled experience factory takes advantage of XML as the vehicle for data exchange within an environment that involves heterogeneous tools. The effectiveness of the framework will be reviewed and measured against the objectives of the project during planned future trials.

1 Introduction

The increased reliance of modern businesses and corporations on IT-related services imposes new and increasingly demanding requirements on the underlying infrastructure. An unambiguous understanding of the limitations of the existing infrastructure – through risk analysis - has become an important prerequisite for designing new services with a satisfactory degree of security. Risk analysis is now considered a useful and critical means for abstracting information from reality into a more formal description. Its importance has been recognized in the process industry, finance and business areas where methods for risk management have been developed [Di02].

Traditionally, risk analysis of security critical systems is performed on the basis of informal descriptions of the target of evaluation. Such an approach is prone to misunderstandings [St02]. Further, the growing complexity of today's systems urges the improvement of existing methods of analyzing systems and their security specification in order to increase the likelihood that all possible security threats are taken into consideration. Consequently, the demand for a more orderly and formal treatment of risks is getting greater.

[1] CORAS is a research and development project under the European Information Society Technologies Fifth Framework Programme (IST-2000-25031), to be completed by July 2003; website: http://www.nr.no/coras

More recently we have witnessed the emergence of an improved methodology for security risk assessment in CORAS [Aa02, St02, Gu02, Iv02] and other similarly-motivated projects such as Reactive System Design Support (RSDS) [LAC00, La00, LAK00] and Surety Analysis [WCF99]. These initiatives share one thing in common – the integration of state-of-the-art Unified Modelling Language (UML) based methodology and conventional risk analysis techniques including Fault Tree Analysis (FTA), Failure Mode and Effect Criticality Analysis (FMECA) and Hazard and Operability study (HAZOP).

CORAS, on which the main subject of this paper is built, aims to produce an improved methodology for precise, unambiguous, and efficient risk analysis of security critical systems. The focus of the CORAS project is on the tight integration of viewpoint-oriented visual modelling in the risk assessment process. An important aspect of the CORAS project is the practical use of UML in the context of security and risk assessment. A further overview of CORAS and its methodology can be found in Section 2.

During risk assessment, knowledge and experiences are gained, other computerized tools are queried and a lot of documents of different types are produced by the risk analysis team. In the context of CORAS, this includes UML diagrams, textual report, tree diagrams, tables, guidelines and many others. It is a non-trivial task when dealing with information of such volume and variety in a heterogeneous operating environment. Further, the risk management process is an elaborate process, involving both humans and computerized tools, and as such, is prone to delays and errors. A system is needed which effectively manages, queries and reuses different types of risk analysis results according to a given assessment scenario. We have implemented such a system, known as the *CORAS platform*, designed to integrate the various components, technologies and results - risk analysis methods, semiformal methods, object-oriented modelling methods, and tools - into an overall tool-supported CORAS framework/process.

A framework that attempts to operate within an organization and to realize the goal of organizational learning will require some measure of Knowledge Management (KM). We have chosen to incorporate the concept of Experience Factory (short: EF [BCR94]) into the CORAS platform. EF is an example of knowledge management approach initially designed for software organizations. In our work to carry out model-based risk assessment, we have found that a tailored version of the experience factory approach is beneficial for creating a learning organization even though the main subject of our project does not fall into the conventional category of software development. The CORAS experience factory is the first known application of EF in the field of model-based risk analysis.

In this paper we introduce the CORAS framework (Section 2) and how this framework is supported by our adapted experience factory (Section 3) - including certain design issues, taxonomy of our experience package and the experience repository - in order to document and exploit the risk analysis results effectively. We also look at the various issues faced during development, and the rationale behind the choice of using XML as the building blocks of our CORAS platform and the decision to adopt a web-based approach. Finally, we summarize our experiences and conclude with future directions in Section 4.

2 CORAS Framework

2.1 Overview of the CORAS Framework

The CORAS framework has four main anchor-points – a risk management process based on AS/NZS 4360[2] and ISO/IEC 17799[3], a risk documentation framework based on Reference Model for Open Distributed Processing (RM-ODP) viewpoints architecture, an integrated risk management & development process based on (Rational) Unified Process and a XML-based tool independent platform.

This framework is also characterised by a careful integration of aspects from partly complementary risk assessment methods like HAZOP, FTA, FMECA, Markov analysis and CRAMM[4]. A more in-depth discussion into each of these anchor-points is covered in [Aa02, St02, Gu02, Iv02]. For the purpose of this paper, we focus only on the CORAS risk documentation framework and the CORAS platform, for which an EF has been set up.

2.2 CORAS Risk Documentation Framework

The CORAS risk documentation framework divides the RM-ODP viewpoint structure into 22 concerns targeting security in general and model-based risk assessment in particular. Implicit within the risk documentation framework is the CORAS risk management process (Figure 2.1), which is to be performed in a sequential fashion.

Each cross-viewpoint concern is tied to a particular sub-process within the risk management process, and may contain concrete *elements* which are relevant for the purpose of risk assessment. 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. Different instances of the same element may occur within different viewpoints of the same concern. Elements can 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.
- Modelling elements expressed in UML.
- Reports and logs from intrusion detection systems, as well as computerised vulnerability & threat management tools.
- Risk assessment tables and FTA tree diagrams.

For simplicity, we refer to these elements as *experience* from hereon.

---

[2] Australian/New Zealand Standard AS/NZS 4360:1999: Risk Management. 1999.
[3] ISO/IEC 10746 series: Basic reference model for open distributed processing. 1995.
[4] Central Computer & Tele-communications Agency's Risk Analysis and Management Methodology.

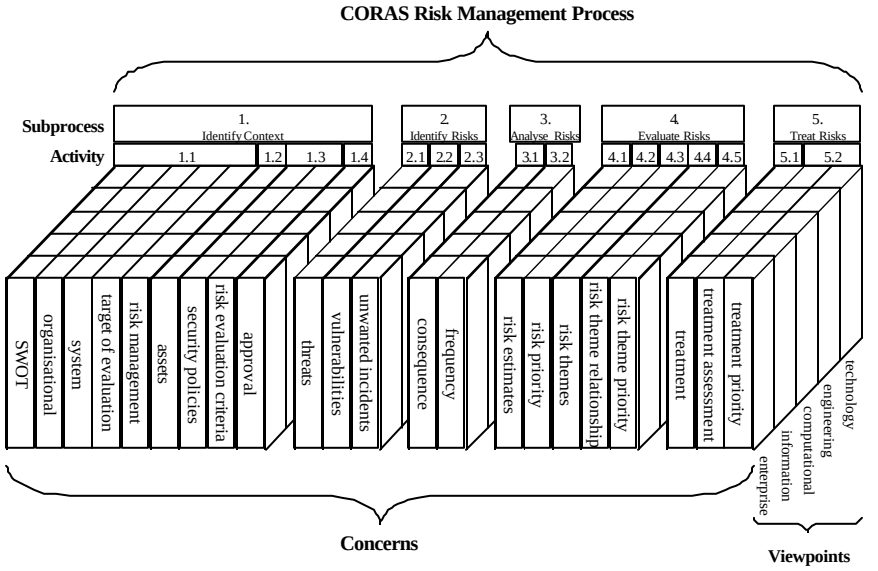

Figure 2.1 CORAS Risk Documentation Framework

2.3 CORAS Platform

From a conceptual view, the CORAS platform, or better understood as the computerized part of the CORAS framework, integrates experience from three broad categories of toolkits or applications. The resulting experience will be sorted and stored in the platform according to the CORAS risk documentation framework. The motivation behind such a high level of data integration and tools interoperability is to ensure broad applicability of CORAS. The pre-requisite to achieving this goal is therefore to have a common data exchange format that is capable of open sharing, in which case the universal XML format has been chosen. Besides, given the ubiquity of XML-related tools, we can reduce the cost and time of deployment significantly.

The three categories of toolkits that can conceivably communicate with the CORAS platform are: UML Comp uter Aided Software Engineering (CASE) tools, risk assessment tools and intrusion detection tools (including vulnerability & threat management tools). Each toolkit category generates experience in a specific XML data model that can be transported and exchanged within the confine of the platform. Some of the supported XML data models are:

- Data based on XML Metadata Interchange (XMI) standardised by the Object Management Group and targeting tools for UML modelling.

- XML-compliant data format targeting risk assessment tools.
- Data based on Intrusion Detection Messaging Exchange Format (IDMEF). IDMEF is an XML data model designed to represent the information exported by the intrusion-detection systems.
- XML-compliant data format targeting vulnerability assessment tools.

Figure 2.2 illustrates the concept behind the CORAS integration platform.

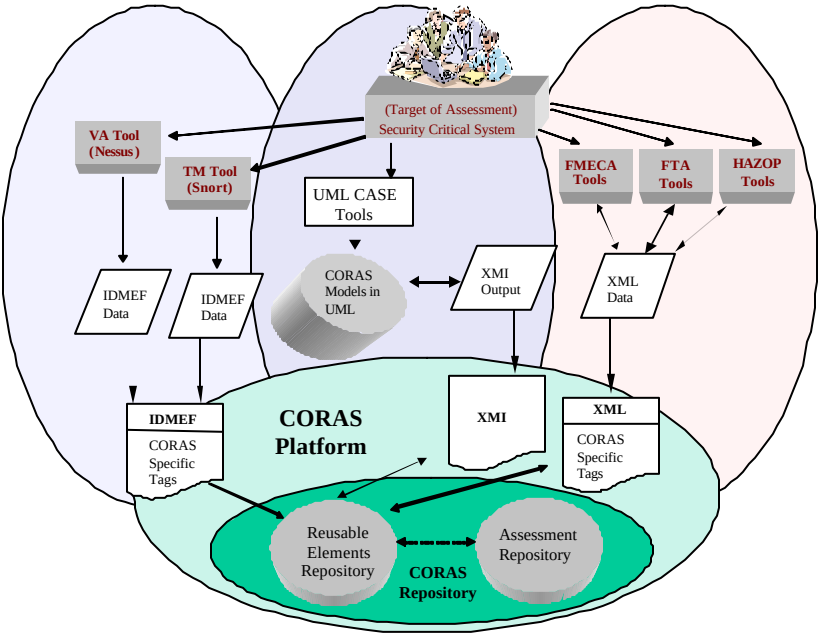

Figure 2.2 CORAS Integration Platform

Given our conceptual view of the CORAS platform, we discovered a list of high-level requirements that the platform should satisfy. The requirements are as below:

- The CORAS platform shall support geographically distributed organizations allowing them to share and manage experience packages remotely.
- The repository shall be robust, reliable and portable to standard computer platforms.
- The GUI shall be platform independent, and allow for visual information navigation.
- The data model shall be simple but powerful enough to model diverse classes of experience packages. The CORAS platform will adapt to the current practices, processes, and products or different organizations, and not vice-versa.
- To allow for effective and efficient use of the repository, the CORAS platform shall be furnished with easy-to-use GUIs and search mechanisms enabling the users to "plug into" the platform, thereby manipulate experience they deem relevant.

- The platform shall increase the reusability of CORAS results and to promote the sharing of such results, as well as to improve the effectiveness and efficiency of model-based risk assessment.

Armed with these requirements, we embarked to discover other strategies that are needed in building an EF for the CORAS framework.

3 Experience Factory in CORAS

3.1 Applying the EF approach to CORAS

As part of a concerted effort for improving risk assessment, in the project we are concerned with establishing an experience management system at both logical and physical organizational level that supports the unique model-based risk assessment approach of CORAS. We need a system that can analyze and synthesize our assessment results and experience, acting as a repository for such experience, and supplying that experience to future assessments on demand.

The EF approach aims to establish an organizational infrastructure to facilitate systematic and continuous organizational learning through the sharing and reuse of experiences in software engineering [BCR94]. It is clear from the beginning that the EF concept was motivated by the lack of understanding of the nature of software and software development in software business. To some extent, software is different from most products. First of all, software is developed in the creative, intellectual sense, rather than produced in the manufacturing sense. Each software system is developed rather than manufactured. Likewise in the context of CORAS, risk management is an evolving process developed in an iterative and interactive manner, instead of being carried out in a simple straight-line process. Secondly, there is a non-visible nature to software. Unlike an automobile or a television set, it is hard to see the structure or the function of software. Similarly for CORAS, the outcome of the risk management process is not immediately apparent. The CORAS risk management process requires understanding, continuous improvement, and the packaging of experience for reuse.

Hence, the EF approach seems like a good fit for CORAS. However, we have not, at least at the present time, incorporated the entire EF concept within our framework, as has been accomplished by a number of software development organizations including SEL-NASA [Ba02, Ba92], FC-MD [Ba01], PERFECT [Pe96] and Daimler Benz AG [HSW98]. One reason is that the EF/QIP approach remains a theoretical, abstract framework, which lacks explicit and easily applicable guidelines on implementation (also pointed out in [HSW98, KJL00]). The time aspect is also a critical point, for instance the EF at NASA evolved over 15 years whereas our EF is supposed to be running within 6 months! Thus, our objective is to build a scale-down but functional EF, tailored to our specific organizational need within the shortest possible time-frame, without the overhead – both time and resources – that is normally required of an organization wishing to tap into the many benefits of EF.

3.2 Building the EF – Top-down Approach

There are two approaches to starting an EF – top-down or bottom-up. The top-down approach compares and organization's process with some generally accepted standard process. Five key steps characterize the top-down approach [KJL00]: (1) Obtain commitment (2) Establish structure (3) Establish processes (4) Produce baseline (5) Identify potential changes. The bottom-up approach assumes that process change must be driven by an organization's goals, characteristics, product attributes, and experiences. In CORAS there are four existing standardization initiatives: (1) AS/NZS 4360 (2) ISO/IEC 17799 (3) UP (4) RM-ODP. These initiatives were already glued together in a comprehensive risk analysis framework. The goal is to provide the facility to support a smooth operation of such a framework. It was obvious that the top-down approach is most suitable for our purpose, so any change will be driven and guided, *not* by experience, but by a set of best practices and the inherent organizational requirements.

EF is based on the Quality Improvement Paradigm (QIP). The QIP focuses on the notion that improving the software process requires the continual accumulation of evaluated experiences (learning) in a form that can be effectively understood and modified (experience models) into a repository of integrated experience models (experience base) that can be accessed and modified to meet the needs of the current project (reuse) [BCR94]. This paradigm implies the logical separation of project development (performed by the Project Organization) from the systematic learning and packaging of reusable experience (performed by the Experience Factory) [ABT00]. In particular, the EF analyses and synthesizes all kinds of experiences drawn from these projects, acts as a repository for such experiences by documenting, storing, qualifying, and updating them using a so-called experience base (EB; Figure 3.1), and supplies those experiences back to projects on demand. The task of the EF, besides support during the risk assessment process, is to package experience by building informal, formal or schematized various risk management processes, components and other forms of knowledge.

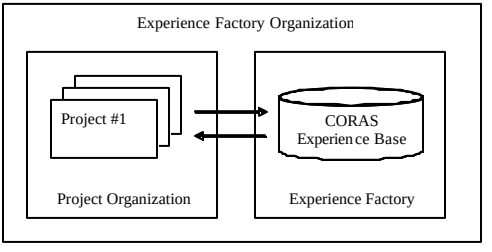

Figure 3.1 EF Organization

The interacting PO and EF also realize two feedback loops, a project feedback loop that takes place in the usage phase and an organizational feedback loop that takes place after a project is completed. The second feedback loop changes or improves the organization's understanding of risk assessment by packaging and reusing experience and making it

accessible to future projects. New methods and techniques can be defined or old ones refined.

3.3 Information Structure for the EF

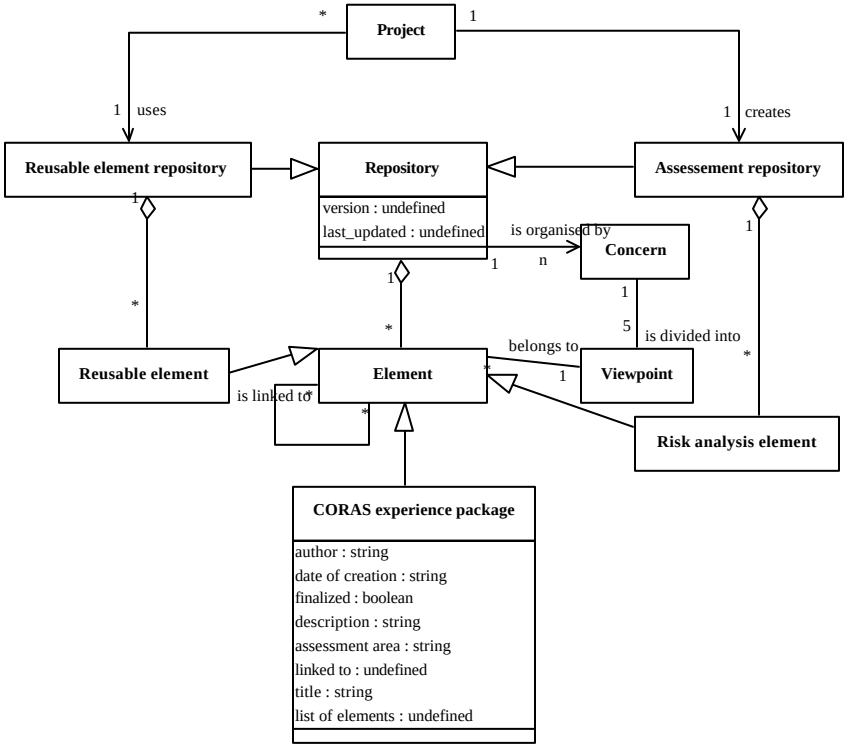

Figure 3.2 Information Structures for the CORAS Repository and CEP

In practice, the result of the PO is the Assessment Repository while the EF materializes in the form of a Reusable Element Repository (Figure 3.2). From the information structure, it is clearly seen that the content of a concern with respect to a particular viewpoint is found in the *element* class itself. Elements in this respect are all kind of experience stored either in AR or RER. The RER focuses on storing reusable UML models, table patterns and formats, as well as supporting the process of experience internalization in an effort to improve the risk management practice. The AR focuses on storing and delivering concrete experience from already completed assessments and/or assessments in progress. Although the roles of the AR and RER are separate, they interact to support each other. Further, though not shown, the reusable element class and risk analysis element class inherit from the XML data models (or types).

During the design stage, a proposal to incorporate the CEP as *part* of the elements was also considered, but eventually abandoned. The reasons being that – firstly, to impose package type at the lowest level, i.e. by integrating any specific information inside the UML model itself will be troublesome as there are many different types of UML diagram, which prohibits a clear-cut approach (if only there is just one type of diagram - class diagram!); secondly, one could encode any additional CORAS-specific tags in the generated XML output but this approach by means of post-rendering complicates the process of information exchange with the repository and may result in a non-canonical XML document.

The internal data structure of both sub-repositories mirror the decomposition model that links together CORAS concerns and viewpoints as outlined previously in Figure 2.1. The sub-repositories are also organized into groups of elements which we regard as CORAS Experience Packages – the main product of our EF. Such packaging scheme functions by giving a structure on how to cluster together various experience in support of the operation of the CORAS platform.

3.4 CORAS Experience Package

As a means to support efficient manipulation of the repository, a unique CORAS Experience Package (CEP) class has been introduced into the information structure. One or more element(s) in the repository can be linked to a CEP class, which is also an element per se. A CEP class is described by three parts: (i) a characterization part used to classify packages in the experience base; (ii) a relationship part used to establish relations between packages; and (iii) a body part that contains the content of the experience package itself.

Each CEP is associated with a package type that defines the type of attributes that will be used in its characterization part; the type of links that will be used in its relationship part, and the type of entities that will compose its body.

Package attributes contains well-defined naming and typing. These attributes build classification taxonomy for each CEP. The attributes effectively define facets that are filled in by the author of an experience package to characterize the package being submitted to the repository. These attribute values are then used to search the repository for packages of interest. Package attributes are typed as “title”, “author”, “date of creation”, “description”, “finalized” and “assessment area”.

Package links also have well-defined naming and typing. As they are used to establish relationships between packages, a package link can be typed as a pointer to a CEP or a list of them. The CORAS platform is built atop of a web-based native XML database that supports URL addresses as pointers to internal CEPs. For this reason, a link can be typed as a URL address or a list of them.

Package entities are typed as an element or as a list of elements. There is no constraint on the type pf elements that can be stored within a package. From a practical point of view,

the repository may store of wide variety of elements that capture experience, including annotation and feedback. An example of important constituents is guidelines and methodology for the use of UML to support the risk assessment methodology. Package entities can also employ similar URL referencing mechanism as package links.

Figure 3.3 illustrates the properties of a CEP, along with its attributes, links and entities in the top left dashed box of the diagram. In the bottom is an instance of a CEP with all the parameters instantiated. The arrows show the association between linked entities and their original representation (in the forms of tables, UML diagrams, fault tree etc.).

The CEP class is an important component, without which the EF concept would not have been possible to be put to practise. The usefulness of the CEP is summarized as follows:
- All CEP attributes are useful for searching.
- CEP links are useful for associating present CEP with other similarly motivated CEP.
- CEP entities contain useful experience for reuse.

Generally, CEP allows experience to be packaged in a systematic and structured manner thereby fulfilling our initial goals. Similar to the JCI EMS [Li02], the CORAS experience management system consists of only one package type, making it a specialized experience base, exclusively for risk analysis elements. It is also possible to have different types of CEPs, for instance a package of type "Project" that may be useful in cataloguing groups of smaller packages.

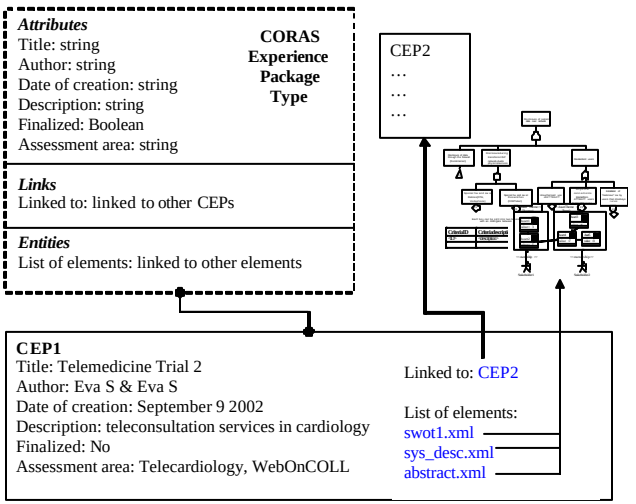

Figure 3.3 CEP and its instantiation

3.5 Technologies for Deployment

One of the aspects in which experience management differs, or rather expands from knowledge management is that the former also looks at the methods and technologies that

are suitable in facilitating the necessary process. Here, it is important to note that EF, as an important ingredient of experience management system, can benefit from the use of tools. Thus, one of the crucial decisions to be made when building an EF infrastructure is the choice between using open-source technologies or proprietary tools such as Lotus Notes - a workflow application, SpotFire - a Visual Query Interface (VQI) tool [Ma01] or Nvivo [Pe96]. We examined the potential of applying workflow and VQI technologies to the specific problem in CORAS, but decided they had inadequate support for retrieval. Tools like Nvivo can be used to automate the searching and coding, as well as facilitate searching for trends and packaging the results. However the tool is not portable to the Web, which is one of our project requirements. The fact that none of these tools are open-source tools is superfluous. Consequently, we decided to implement our system utilizing a suite of open-source tools, built upon a native XML database called eXist[5]. eXist is tightly integrated with Apache's Cocoon[6] publishing framework. Cocoon offers a powerful mechanism called XSP to write XML-based dynamic web pages which fits well with eXist's search engine that has been designed to provide fast XPath queries, using indexes for all elements, text and attribute nodes. eXist is lightweight and well suited for the deployment of our repositories. The server is accessible through easy to use HTTP and XML-RPC interfaces and supports the XMLDB API for Java programming. Furthermore, the latest version of eXist also supports Web-based Distributed Authoring and Versioning (WebDAV) a powerful HTTP extension that allows users to collaboratively edit and manage files on remote web servers.

Some of the advantages of adopting a web-based approach are:
-The system can be easily deployed across different platform and OS, making it a highly distributed and portable solution.
-The system can be extended to interoperate with future web services to provide additional value-added functionalities.

3.6 Initial Results and Trials

Six trials have been planned for the CORAS project; three within e-commerce (one of which has already been completed) and three within telemedicine (two of which has been completed) to measure the effectiveness of the framework. The completed trials were conducted prior to the completion of the prototype CORAS platform, so no substantial findings on the use of the platform are currently available. However, initial informal evaluation of the platform, which focused mainly on its usability and linked interfaces, has found the system to be acceptable. Users are able to navigate and query the repository, as well as to manipulate results via the CEPs. Future trials will evaluate the platform and test its effectiveness in supporting the CORAS methodology.

[5] Open-source XML database, largely developed by Wolfgang Meier (http://exist.sourceforge.net)
[6] Open-source XML publishing framework (http://xml.apache.org/cocoon)

4 Conclusion and Future Work

In this paper we described an ongoing work that aims to build a generalized Experience Factory approach into CORAS, which is model-based risk analysis framework for security critical applications. The planned usage of the platform is geared towards consistent use and integration into the CORAS risk management process. The purpose is to promote the reuse and sharing of risk analysis results, as well as to improve the effectiveness and efficiency of model-based risk assessment. We looked at the properties of the prototype EF-driven CORAS platform including the information structure of the repository and its experience package. The prototype and its web approach, takes advantage of modern technologies like XML for distributed storage, access and dissemination of relevant knowledge. The prototype encompassed only some of the automated features that are required of a system aimed at supporting an EF. Much technical and organizational work are pending. The ultimate goal of an EF-driven system is to turn an organization into one which is shaped by constant learning. Therefore, having obtained and stored the explicit knowledge/experience in the experience base, the next step is to ensure that such experience is internalized as active knowledge and practical skills. This is still a largely semi-automated process in CORAS which indicates room for further research and improvement.

Immediate plans have been made to extend the functionality of the CORAS platform. This includes adding: (1) automatic consistency and semantic translation for the stored experiences (for instance between a RA table and UML model) (2) content management features: authentication and backup (3) an XML data model targeting model-based risk analysis (currently absent in the XML industry).

Acknowledgement

The authors wish to thank the European Commission for supporting the CORAS project, as well as all the partners from the project consortium - CTI (Greece), FORTH (Greece), IFE (Norway), Intracom (Greece), NCT (Norway), NR (Norway), RAL (UK), Sintef (Norway), Solinet (Germany) and Telenor (Norway) – for their excellent contribution and cooperation.

Bibliography

[Aa02] Aagedal, J. Ø., Braber, F. den, Dimitrakos, T., Gran, B. A., Raptis, D., Stølen, K.: Model-based Risk Assessment to Improve Enterprise Security, 6th IEEE International Enterprise Distributed Object Computing Conference, September, 2002.

[ABT00] Althoff, K.-D., Bomarius, F. & Tautz, C.: Knowledge Management for Building Learning Software Organizations In Information System Frontiers Journal, Vol. 2, Nr. 3/4, 349-367, 2000.

[Ba92] Basili, V.; Caldiera, G.; McGarry, F.; Pajerski, R.; Page, G.; Waligora, S.: The Software Engineering Laboratory-an Operational Software Experience Factory, International Conference on Software Engineering, 1992.
[Ba01] Basili, V.; Costa, P.; Lindvall, M.; Mendonca, M.; Seaman, C.; Tesoriero, R.; Zelkowitz, M.: An experience management system for a software engineering research organization, Proceedings of the 26th Annual NASA Goddard Software Engineering Workshop, 2001.
[Ba02] Basili, V.R., McGarry, F.E., Pajerski, R., Zelkowitz, M.V.: Lessons learned from 25 years of process improvement: The Rise and Fall of the NASA Software Engineering Laboratory, Proceedings of the 24th International Conference on Software Engineering, 2002. ICSE 2002. May 2002.
[BCR94] Basili, V. R., Caldiera, G., and Rombach, D. H.: The Experience Factory, Encyclopaedia of Software Engineering -2 Volume Set, pp. 469-476, 1994.
[Di02] Dimitrakos, T. et al.: Integrating Model-based Security Risk Management into e-Business Systems Development. 2nd IFIP Conference on e-commerce, e-business, e-government. Lisbon, 7-9 October, 2002.
[Gu02] Gustavsen, T.S., Houmb, S-H., Gran, B. A. Stølen, K.: Security Risk Analysis in e-Commerce, 7th Nordic Workshop on Secure IT Systems 2002.
[HSW98] Houdek, F., Schneider, K. and Wieser, E.: Establishing Experience Factories at Daimler Benz: An Experience Report, Proc. 20th Int'l Conf. on Software Engineering, Kyoto, May 1998.
[Iv02] Djordevic, I., Gan, C., Scharf, E., Mondragon, R. et al.: Model-based risk management of security critical systems. In Proc. Risk Analysis III, series: Management Information Systems, Volume 5, WIT Press, 2002.
[KJL00] Koennecker, A., Jeffery, R., Low, G.: Implementing an experience factory based on existing organisational knowledge, Proceedings of the 2000 Australian Software Engineering Conference, 2000.
[La00] Lano, K., Clark, D., Androutsopoulos, K. and Kan, P.: Invariant-based Synthesis of Fault-tolerant Systems. TRTFT 2000, Pune, India, 2000.
[LAC00] Lano, K., Androutsopoulos, K. and Clark, D.: Structuring and Design of Reactive Systems using RSDS and B. FASE 2000, LNCS, Springer-Verlag, 2000.
[LAK00] Lano K., Androutsopoulos, K. and Kan, P.: Structuring Reactive Systems in B AMN. In ICFEM 2000, IEEE Computer Society Press, 2000.
[Li02] Lindvall, M., Frey, M., Costa, P., Tesoriero, R.: Lessons Learned about Structuring and Describing Experience for Three Experience Bases, Lecture Notes in Computer Science, 2001.
[Ma01] Manoel Gomes de Mendonça Neto, Carolyn B. Seaman, et al.: A Prototype Experience Management System for a Software Consulting Organization. Proceedings of the International Conference on Software Engineering and Knowledge Engineering (SEKE01), Buenos Aires, Argentina, June 2001.
[Pe96] PERFECT Consortium: PIA Experience Factory, The PEF Model, ESPRIT Project 9090, D-BL-PEF-2-PERFECT9090, 1996.
[St02] Stølen, K. et al.: Improving security through model-based risk assessment, Business CBSE Chapter 11, Kluwer Academic Publishers, 2002.
[WCF99] Wyss G., Craft, R., and Funkhouser, D.: The Use of Object-Oriented Analysis Methods in Surety Analysis. Sandia National Laboratories Report, 1999.