The CORAS Tool for Security Risk Analysis

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

SINTEF, Norway,
{fvr,fbr,msl,kst}@sintef.no

Abstract. The CORAS Tool for model-based security risk analysis supports documentation and reuse of risk analysis results through integration of different risk analysis and software development techniques and tools. Built-in consistency checking facilitates the maintenance of the results as the target of analysis and risk analysis results evolve.

1 Introduction

The CORAS framework for UML-based security risk analysis, in the following referred to as security analysis, consists of among other things a methodology, a language, and a tool. The CORAS methodology integrates aspects from partly complementary risk analysis techniques, like HazOp [1], FMEA [2], and FTA [3], with state-of-the-art system modeling methodology based on UML 2.0 [4]. A graphical UML-based language has been developed to support documentation and communication of security analysis results [5].

The integration of different risk analysis approaches facilitates the analysis of different aspects of a system or organisation, e.g. security, legal issues and business processes, resulting in a combined picture of the relevant risks. For example, the notion of trust is tightly interwoven with notions like security and usability [6, 7]. Furthermore, it is difficult to separate trust from the expectation of a legal framework that offers protection in the cases where the trust relationship fails [8]. An analysis of trust should therefore encompass a number of issues including technological, legal, sociological and psychological aspects.

This paper presents the CORAS Tool which has been developed to provide computerized support for performing security analysis in accordance with the CORAS methodology using the graphical CORAS language. Section 2 describes the CORAS Tool itself, while Sect. 3 outlines plans for future work.

2 The CORAS Tool

Security analysis is a highly elaborate and prolific process which involves many different types of documentation, such as UML models, tables with analysis data, and natural language descriptions of the target of evaluation. Access to this information and its change history needs to be provided. The information also needs to be shared between the various modelling and analysis tools used

P. Herrmann et al. (Eds.): iTrust 2005, LNCS 3477, pp. 1–4, 2005.

by the analyst. In addition, it is important to maintain consistency between all the different pieces of information. Computerised support for documentation, maintenance and reuse of analysis results is thus of high importance.

The CORAS Tool has been developed based on the principles and requirements outlined above and is publicly available as open source [9]. The tool follows a client-server model and is developed entirely in Java. The security analyst uses the CORAS client application to create new analysis projects, document and edit security analysis results, generate analysis reports, and manage and reuse experiences from previous analyses. Help is provided to the user in the form of integrated electronic versions of the CORAS methodology as well as user guides. A screenshot of the CORAS client application is shown in Fig. 1.

Analysis data is stored in a centralised database on the server, enabling multiple users to collaborate on the same analysis project. The server is based on standard Enterprise Java Beans (EJB) technology and runs on top of the open source JBoss application server. Figure 2 shows an overview of the structure of the CORAS tool.

There are two databases in the CORAS Tool, called repositories. The assessment repository stores results from the actual security analyses, while the experience repository contains reusable results from previous analyses in the form of UML-models, checklists, procedures and more. By facilitating reuse, the tool helps the user avoid starting from scratch for each new analysis. The repositories are versioned, so that all changes are maintained and the user can go back and look at the state of a security analysis at a previous point in time. The repositories are implemented on top of the open-source XML database eXist [10].

A wide variety of UML modelling tools and security analysis tools exist and are in use by security analysts and system engineers today. It is therefore important for our tool to provide flexible support for integration with external tools. Though the security analysis process is model-driven, the tool does not

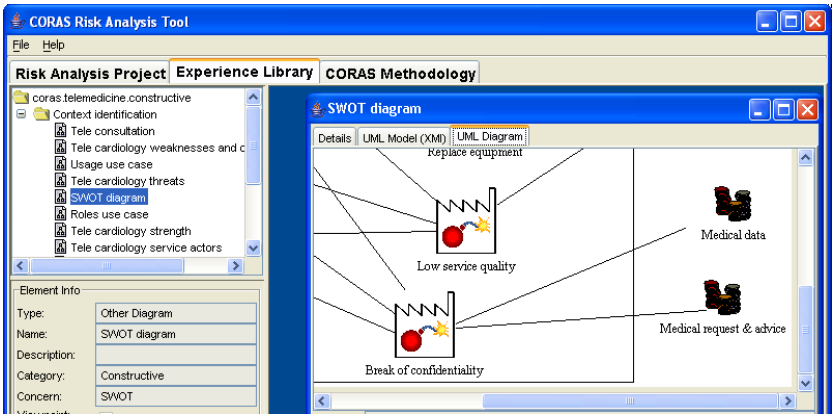

Fig. 1. Screenshot of the CORAS Tool

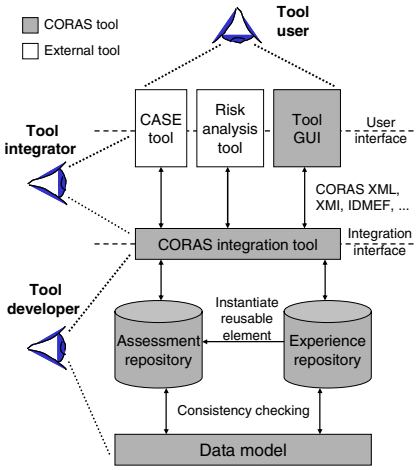

Fig. 2. Overview of the CORAS Tool

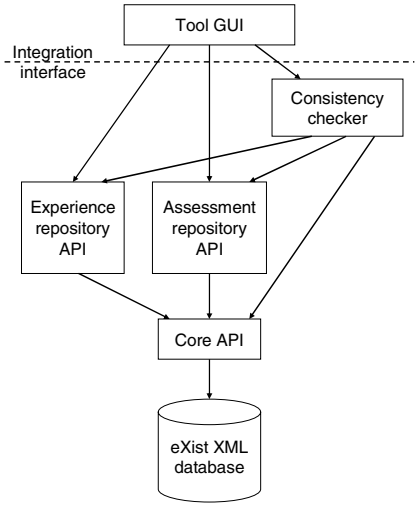

Fig. 3. Integration interface

deal exclusively with UML models, but must also be able to include tables and other constructs such as fault tree diagrams, intrusion detection logs and natural language descriptions. To satisfy these requirements, the tool provides an integration layer with a defined API which can be used by other tools to integrate with the CORAS tool. Standardised XML formats are utilised for data integration, such as XMI for the interchange of UML models.

Three different viewpoints or roles are defined in Fig. 2. The *tool user* represents the end-user, e.g. a security analyst who uses the CORAS tool together with various modelling and security analysis tools The *tool integrator*, on the other hand, is responsible for integrating the various external tools with the CORAS tool, using the integration interface. Finally, the *tool developer* will implement the functionality of the CORAS tool itself.

Many entities of a security analysis, e.g. assets, stakeholders and threats, appear multiple times throughout the analysis results. It is therefore important to ensure that the various analysis results are mutually consistent with each other. We have defined an internal risk analysis data model based on the concepts and relationships defined in the CORAS language [5]. This model extracts information from the analysis data and uses this as a basis for the consistency checking. Consistency rules are defined using the XML Stylesheet Transformation Language (XSLT). Figure 3 shows how the main components of the tool interact.

3 Future Work

Although the CORAS project was completed in 2003, development of the CORAS tool and methodology is continuing in a number of other projects. Support for

creating and editing UML diagrams directly in the CORAS application is under development, as well as improvements to the security analysis methodology and language to facilitate e.g. legal risk analysis [11]. Integration of process and workflow techniques to facilitate a more methodology-driven approach is being investigated. We are also working on upgrading the consistency mechanisms, e.g. to support consistency languages such as CLiX [12] and to provide consistency repair functionality.

Acknowledgements

The work on which this paper reports has partly been carried out within the context of the EU-projects TrustCoM (IST-2003-01945) and CORAS (IST-2000-25031) as well as the SECURIS (152839/220) project funded by the Research Council of Norway.

References

1. Redmill, F., Chudleigh, M., Catmur, J.: HazOp and software HazOp. Wiley (1999)
2. Bouti, A., Kadi, D.A.: A state-of-the-art review of FMEA/FMECA. International journal of reliability, quality and safety engineering **1** (1994) 515–543
3. IEC 1025: Fault Tree Analysis (FTA). (1990)
4. OMG: UML 2.0 Superstructure Specification. (2004) OMG Document: ptc/2004-10-02.
5. Lund, M.S., Hogganvik, I., Seehusen, F., Stølen, K.: UML profile for security assessment. Technical Report STF40 A03066, SINTEF Telecom and informatics (2003)
6. Jøsang, A., Ismail, R., Boyd, C.: A Survey of Trust and Reputation Systems for Online Service Provision. Decision Support Systems (to appear) http://security.dstc.edu.au/papers/JIB2005-DSS.pdf.
7. Egger, F.N.: Towards a model of trust for e-commerce system design. In: CHI 2000: Workshop Designing Interactive Systems for 1-to-1 E-commerce. (2000) `http://www.zurich.ibm.com/~mrs/chi2000/contributions/egger.html`.
8. Jones, S., Wilikens, M., Morris, P., Masera, M.: Trust requirements in e-business. Communications of the ACM **43** (2000) 81–87
9. CORAS: The CORAS project (2005) http://coras.sourceforge.net/ (visited February 2005).
10. Meier, W.: eXist: An Open Source Native XML Database. In Chaudri, A.B., Jeckle, M., Rahm, E., Unland, R., eds.: Web, Web-Services, and Database Systems. Volume 2593 of LNCS., Erfurt, Germany, NODe 2002 Web- and Database-Related Workshops, Springer (2002)
11. Vraalsen, F., Lund, M.S., Mahler, T., Parent, X., Stølen, K.: Specifying Legal Risk Scenarios Using the CORAS Threat Modelling Language - Experiences and the Way Forward. In: Proceedings of 3rd International Conference on Trust Management (iTrust 2005). LNCS, Roquencourt, France, Springer (To appear 2005)
12. Systemwire: Clix: Constraint language in xml (2003) http://www.clixml.org/ (visited February 2005).