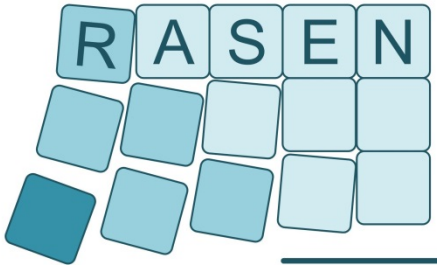

Compositional Risk Assessment and Security Testing of Networked Systems

Deliverable D3.3.2 and D4.3.2

Tools for Compositional Security Risk Assessment and Security Testing v.2

Brief Description, Documentation and Installation Guide

Project title:	RASEN
Project number:	316853
Call identifier:	FP7-ICT-2011-8
Objective:	ICT-8-1.4 Trustworthy ICT
Funding scheme:	STREP – Small or medium scale focused research project

Work package:	WP3 and WP4
Deliverable number:	D3.3.2 and D4.3.2
Nature of deliverable:	Prototype & Report
Dissemination level:	PU
Internal version number:	1.0
Contractual delivery date:	2014-09-30
Actual delivery date:	2014-09-30
Responsible partner:	Fraunhofer

Contributors

Editor(s)	Johannes Viehmann (Fraunhofer)
Contributor(s)	Fabien Peureux (Smartesting), Julien Botella (Smartesting), Johannes Viehmann (Fraunhofer), Fredrik Seehusen (SINTEF), Bjørnar Solhaug (SINTEF), Frank Werner (SAG)
Quality assuror(s)	Fredrik Seehusen (SINTEF)

Version history

Version	Date	Description
0.1	14-09-14	ToC and contents overview
0.2	14-09-18	Contents provided to all sections
0.3	14-09-28	Complete version finalized for internal review
1.0	14-09-30	Final version

Abstract

This report provides some basic information about the tools of the WP3 and WP4 prototype deliverable month M24. The delivered tools are CORAS from SINTEF, RACOMAT from Fraunhofer FOKUS, CertifyIt from Smartesting and ARIS Business Architect from Software AG.

Keywords

Security, security risk assessment, security testing, test-based risk assessment, tool support, prototype

Executive Summary

The overall objective of RASEN WP3 and WP4 is to develop tools and techniques to support test-based and compositional security risk assessment. Deliverable D3.2.2 presents the WP3 techniques that were developed during the first two years of the project and deliverable D4.2.2 shows the WP4 techniques that were developed during the first two years of the project, while D5.3.1 presents the methodologies that the WP3 techniques should support. This report accompanies the D3.3.2 prototype deliverable, and gives an overview and introduction to the four tools of that deliverable.

The set of tools are as follows:

- The CORAS tool from SINTEF, supporting model-driven risk analysis
- The RACOMAT tool from Fraunhofer FOKUS, which is a tool for the bidirectional combination of low level risk assessment with security testing that could also be used as an integration platform for other tools in such a process
- The CertifyIt tool from Smartesting for model-based security testing
- The ARIS Business Architect from Software AG, supporting security assessment and risk modeling of software and IT systems

Table of contents

1 Introduction

This report is a brief documentation of and introduction to the RASEN WP3 and WP4 prototype tools. The prototypes are developed to provide support for the methods and techniques that are also developed in this work package. The reader is referred to deliverables D3.2.2 and D4.2.2 for an overview of the WP3 and WP4 research results after the first two years of the RASEN project.

The prototypes presented in this report serve as a part of the RASEN tool-box for security risk assessment and security testing that is developed in the context of WP5. WP5 defines the common RASEN data meta-model that will be used for integrating the tools by facilitating the communication between them. An important part of the RASEN prototype development is therefore to provide support for exporting and importing data to and from the common RASEN data model.

The tools that are presented in the next two sections are RACOMAT, Smartesting CertifyIt, ARIS Business Architect, and CORAS. In Section 2 we give a brief overview of the main characteristics of the tools, and in Section 3 we present the tools in some more details, give installations and user guidelines, and explain the planned and current status for the development of tool features relevant in WP3 and WP4. Finally, in Section 4 we conclude.

2 Characteristics of the Identified WP3 and WP4 Tools

In this section we give an overview of the RASEN tools relevant for WP3, providing some basic general information and technical information, as well as other additional information that may be useful.

2.1 RACOMAT Tool

General information	
Name	RACOMAT (Risk Analysis COMbined with Automated Testing) Tool
Provider	Fraunhofer FOKUS
Topic addressed	RACOMAT Method, the entire combined test-based risk assessment (TBRA) and risk-based security testing (RBST) process.
Description	The RACOMAT tool automatically generates initial fault trees or CORAS risk graphs with linked system models for programs, libraries, components and web interfaces. These editable initial risk analysis artefacts already contain data from existing risk related libraries like Mitre CAPEC and CWE. The RACOMAT tool supports compositional risk analysis with simple drag and drop for existing artefacts and it calculates likelihoods for dependent incidents, e.g. by performing Monte Carlo simulations. Security test patterns are automatically associated with risk analysis artefacts as well as system model components (e.g. input and output ports) and their priority is calculated. If no appropriate test patterns exist in the library, the tool allows its users to create new test patterns within the tool and to upload them to the library for sharing. Given an appropriate test pattern, test generation, execution and result aggregation are at least semi-automated. Indeed, there is no need for additional manual work at all for testing many common issues like overflows or SQL injections. Security testing metrics suggested by the test patterns can be used to analyze and evaluate test results. New security testing metrics can be created and edited in the RACOMAT tool. With appropriate security testing metrics, it is possible to update the risk graphs automatically with more precise likelihood estimates or new faults based on the test results. Besides using the RACOMAT tool as a stand-alone tool, it is possible to use the RACOMAT tool as an integration platform and to utilize other eventually more specialized tools for some steps in the combined TBRA and RBST process.
License	RASEN project partner can obtain a license for the project duration. The final license model is not yet decided.
Website	N/A, planned for 2015
Technical information	
Download site	N/A, planned for 2015
OS	Windows
Technology environment	.Net 4.0, WPF
Other dependencies	none
Additional information	
Known issues/risks	N/A
Additional useful information	N/A

2.2 Smartesting CertifyIt

General information	
Name	Smartesting CertifyIt
Provider	Smartesting
Topic addressed	Model-Based Testing solution
Description	The tool is composed by a Rational Software Architect plugin for modelling activities, and a standalone Java application for the test generation and test case management
License	RASEN project partner can obtain a license for the project duration.
Website	www.smartesting.com
Technical information	
Download site	www.smartesting.com
OS	Win or Linux
Technology environment	Rational Software Architect v8.0.x and v8.5.x, EMF compliant JAVA 1.6, 1.7 compliant
Other dependencies	N/A
Additional information	
Known issues/risks	N/A
Additional useful information	N/A

2.3 ARIS Business Architect with RASEN Plugin

General information	
Name	ARIS Business Architect + RASEN Plugin
Provider	Software AG
Topic addressed	Security Assessment and Risk Modeling of Software and IT Systems
Description	The model extension is based on the ARIS Business Process Analysis Platform (http://www.softwareag.com/corporate/products/aris/bpa/default.asp), a proprietary solution of Software AG and provides an interface to model risk assessment of IT security systems.
License	Proprietary
Website	http://www.softwareag.com/corporate/products/aris/bpa/architect_design/overview/default.asp
Technical information	
Download site	Download site of the RASEN Model Extension https://project.sintef.no/eRoomReq/Files/ikt2/RASEN/0_33361/SAG%20Prototype.zip
OS	Windows, Linux, (Any OS supported by the ARIS Business Architect)
Technology environment	None
Other dependencies	None
Additional information	
Known issues/risks	None
Additional useful information	None

2.4 CORAS

General information	
Name	CORAS tool
Provider	SINTEF
Topic addressed	Model-based risk assessment
Description	The tool is based on Eclipse and the GMF/EMF framework, but it is distributed as a stand-alone tool
License	Eclipse Public License v1.0 (http://www.eclipse.org/legal/epl-v10.html)
Website	http://coras.sourceforge.net
Technical information	
Download site	http://coras.sourceforge.net/downloads.html
OS	Tested on Windows; a non-tested distribution is also available for Linux
Technology environment	The tool needs Java
Other dependencies	None
Additional information	
Known issues/risks	None
Additional useful information	None

3 Description of the Prototype Tools

3.1 RACOMAT Tool

The RACOMAT Tool prototype is developed to cover the entire process of combined test-based risk assessment (TBRA) and risk-based security testing (RBST) within a single standalone tool. It will allow us to apply and evaluate the RACOMAT Method without troubling about interoperation and interaction between different tools. Therefore, we can get quicker results than with a large tool chain.

However, more specialized tools might be more powerful than the RACOMAT Tool for some tasks. Eventually, use case partners do already use some existing tools for some steps of the risk assessment and security testing process. Therefore, we try to build bridges from the RACOMAT tool to other tools, especially to those developed or used by our project partners. The idea is to make the RACOMAT Tool the central integration platform which can be used to control the overall combined TBRA and RBST process while for some tasks within that process, other tools are used. Since the RACOMAT Tool itself implements the entire process, there will be no gaps, for sure, even if for some steps within the process there were no other tools than the RACOMAT Tool available.

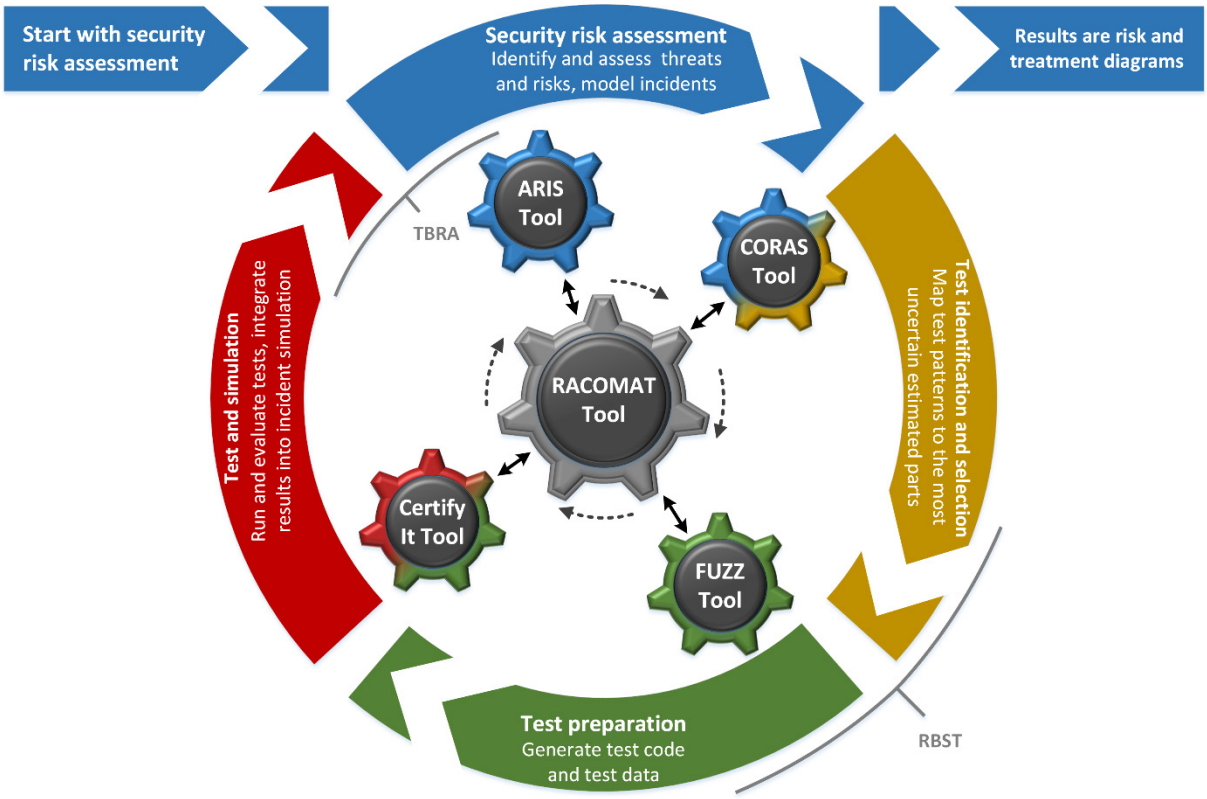

Figure 1 – The process of the RACOMAT Method with the RACOMAT tool as a platform integrating various other tools

3.1.1 Features within the RASEN project

Current Status

- System analysis and risk assessment
  - Automatically creates interface models for programs, APIs, components, Web-Pages or Web-Services
  - Generates semi automatically initial fault trees or CORAS risk graphs
  - Uses risk catalogues (Mitre CWE / CAPEC, BSI IT-Grundschutz …)

    - Edit and compose per Drag and Drop
    - Calculates likelihoods for dependent incidents automatically
        - Supports timing issues – likelihoods might change over time
    - Identifies and prioritizes elements worth further investigation
- Security Test Pattern instantiation
    - Suggests test patterns for identified threat scenarios
    - Assisted association with risk artefacts and system components by drag and drop
        - Indicates where to stimulate and what to observe
- Execution of tests, observation
    - Once a test pattern is instantiated, generating, executing and evaluating tests woks at least semi automatically
        - Often no manual work is required at all, e. g. for overflows or (SQL-) Injections
    - Automatic observation and basic aggregation of raw test results
- Updates the risk picture based upon the test results semi automatically
    - Makes suggestions using the metrics suggested by the security test pattern
        - More precise likelihood values
        - Allows to add unexpected observations as new faults or unwanted incidents by dragging them to the risk graph
- Create and edit security test patterns
- Create and edit security testing metrics
- Import from ARIS Business Architect (Software AG)

Planned Features

The relevant upcoming features expected within the RASEN project are the following:

- Calculate how much test effort should be spend for which tests
- Import from CORAS Tool, CertifyIT …
- Export to ARIS Business Architect, CORAS Tool, CertifyIT …
- Open Security Test Pattern Library STPL and open Security Testing Metrics Library STML
    - User can contribute feedback and they can suggest extensions for the open libraries
    - Quality management with ratings / comments of the users
- Assistant for Cloud Security Risk Assessment and Cloud Security Testing

3.1.2 Documentation and Installation

For installation, RACOMAT requires Microsoft .NET 4.5 or a later / compatible version. The ZIP file RACOMAT.zip in the tool deliverable does not contain the .NET runtime, which can be obtained from here: http://www.microsoft.com/de-de/download/details.aspx?id=30653. After installing .NET, unpack RACOMAT.zip and simply start Setup.exe to install RACOMAT locally.

RACOMAT is still an early prototype. The documentation is currently under construction. However, there are already some tool tips in the program. Furthermore, a short Video Demo / Tutorial showing the basic workflow is included in the tool deliverable ZIP file.

3.2 Smartesting CertifyIt

This section introduces the tool CertifyIt developed and provided by the company Smartesting to generate security test cases from vulnerability risk assessment, and to produce test results, which can actively contribute to identify, estimate and finally treat the risk of the tested application

It should be noted that a more detailed presentation of the tool including tutorial, and tips to use it efficiently, are provided with the tool (modeling guide, UML/OCL reference guide and software documentation).

3.2.1 Presentation

Smartesting CertifyIt is a tool suite that automatically generates test cases based on a model of system requirements. Manual test design is labor intensive and error prone; this manual work can be avoided for complex applications by modeling the key concepts (abstraction) and allowing Smartesting CertifyIt to automate your test design work. Since the model is more expressive and simpler than the system-under-test, it can more readily be reviewed for correctness and coherency, as well as be updated more easily. Smartesting CertifyIt supports UML/OCL models as the specification modeling language, and generates test cases to cover security test patterns used as test objectives. Finally, the generated test cases can be exported in UML sequence diagrams and can be fully integrated into the process promoted by the RASEN project.

Smartesting CertifyIt enables to:

1. Implement your testing strategy by importing security test patterns.
2. Generate your tests from several criteria, based on risk assessment, and manage traceability.
3. Publish abstract test cases for documentation or manual exploitation.
4. Publish test scripts into a testing environment for automated execution.
5. Easily manage the evolution of the specification: update your model and Smartesting CertifyIt will generate the new test cases.

3.2.2 Installation Guidelines

Smartesting test generation solution works with models edited by an Eclipse-based UML modeler. Smartesting provides a plug-in that checks whether the model fits the Smartesting CertifyIt restrictions on UML, and exports a model to be used by the Smartesting CertifyIt tool to generate the test cases. To use Smartesting test generation solution, both the Eclipse-based modeler plug-in for Papyrus and the Smartesting CertifyIt software are needed.

IBM RSA Modeler Plug-in installation

1. This plug-in must be installed with the Eclipse update site tool. The following steps describe the installation of a new release.
2. Start RSA modeling tool.
3. From the tool menu, select Help->Software Updates->Find and Install...
4. A dialog opens. Select "Search for new features to install", and click "Next". Select "New Archived Site ...".
5. Specify the path to a .zip file available in the 'install' folder of the Smartesting CertifyIt installation directory. Once the path is selected, the following window should appear. You can use the 'Name' field in order to define your own Local Site name.
6. Click the "Finish" button of the "Install" window to launch the installation.
7. Check the Smartesting CertifyIt plug-in, and click « Next ».

At this stage, you may have warnings indicating that some features cannot be installed (due to unavailable dependencies). In that case, check the “System Requirements and Supported Software”, and upgrade RSAmodeler if required.

8. Accept terms of the license agreement, and click « Next ».
9. Click the « Finish » button to install the plug-in.

If verification messages appear, just accept those in order to complete the installation. It should be noted that write permissions for the install location are needed. Afterwards, restart RSA modeling workbench. The Smartesting CertifyIt plug-in should be successfully installed at this point.

Smartesting CertifyIt software installation

To install Smartesting CertifyIt software, simply run the setup program and follow the installation instructions (this implies accepting the terms of the displayed license agreement).

Before using Smartesting CertifyIt, a license needs to be defined. Prerequisites are: Smartesting CertifyIt must be installed and the RSA Eclipse-based modeler plug-in must be installed. There are two types of licenses: floating license server or capacity-based license.

3.2.3 User Guide

A model is an abstraction of the reality to achieve an objective. The objective of Smartesting solution is to support the validation engineer in its testing effort. Hence, the activity of creating UML models must always be evaluated against a testing goal: “Will this provide the relevant tests required for my system?”. In this perspective, the validation engineer will make a number of choices while modeling its system:

- What are the boundaries of the system?
- What are the inputs and outputs relevant for testing the system?
- What is the amount of information needed to model the behavior of the system?

The objective of the validation engineer is not to model all the software system in its most accurate detail, but, most of the time, to model just enough and rightly focused system behavior to leverage Smartesting CertifyIt capabilities to generate relevant test cases.

This is why the use of Smartesting CertifyIt, and the use of models, must always be considered as part of a larger perspective on how the target system must be tested. As a result, a test-oriented model should not be considered as design model. It only describes the system as a black box and only models parts of the system relevant for Smartesting CertifyIt. Consequently some UML design concepts such as interfaces, ports, and others are not useful.

What are the required steps to specify a test-oriented model?

1. Select a functionality that needs to be tested (a requirement).
2. Define the available control points and observation points for this functionality.
3. Model the control and observation points on a class diagram.
   a. Defining the required level of abstraction.
   b. Using classes, operations and data types to model those points.
4. Model the expected system behavior of this functionality with OCL and state machines.
5. Model an initial state for the system as object instances and links.

3.2.4 Features within RASEN project

During the RASEN project, several Smartesting CertifyIt extensions have to be developed.

1. Test Purpose language extension, enabling sets creation using OCL code evaluated on the initial state of the system
2. Keyword creation to be used by Test Purposes. This mechanism enables to create generic Test Purposes, and to help for maintenance and reuse.
3. Capability to link a Test Purpose to a requirement identifier to ensure the traceability through the all test generation process.
4. Test Purpose catalogue import/export to reuse and apply Test Purposes on several systems under test.
5. Generation and Animation standalone Java API to enable fuzzed test sequences validation regarding the model.
6. A DSL designed to assist the validation engineer during the UML model creation. This DSL specifically addresses web-based applications.
7. A RASEN dedicated Wizard the assist the user during the model creation, and pattern selection regarding the Risk analysis.

Those features are presented in a more detailed way in the D4.2.2 RASEN deliverable.

3.3 ARIS Business Architect – RASEN Plugin

3.3.1 Presentation

The ARIS Business Architect (ABA) is proprietary software from Software AG. A screenshot of the ABA with the different modeling aspects is depicted in Figure 2. It shows on the left hand side a list of all possible Common Weakness Enumerations (CWEs)[1] which can directly be aligned to a software component. On the right hand side (cf. Symbol Box) a selection of other components can be selected and drawn into the modeling pane in the center of the screen.

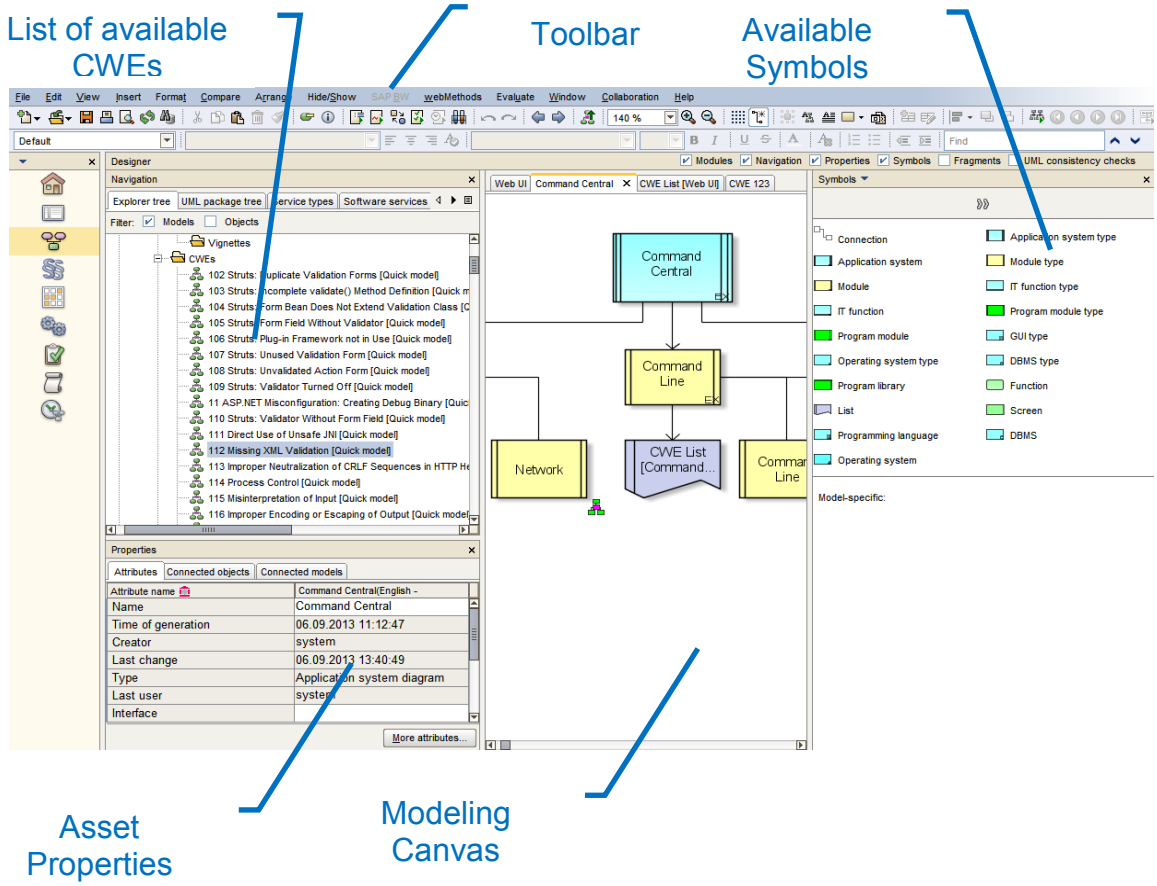

Figure 2 – Screenshot of the Security Risk Assessment in RASEN model

3.3.2 Installation Guidelines

An individual installation program can be started using the provided Setup.exe which is guiding the used through the provided setup-routine. If system files are changed during installation, you are prompted to reboot your computer after installation. In addition to that there is a detailed installation guide of how to install the ARIS Business Architect on most spread operating systems.

On top of the base installation of the ARIS Business Architect, the RASEN methodology is added by importing the RASEN artefacts from a ZIP file which contains the base package, consisting of the reports, the definition of necessary modeling elements, the macro, a preliminary set of already defined CWEs, and a predefined set of generic component types previously generated.

[1] http://cwe.mitre.org

3.3.3 User Guide

The Product Model builds the base of the proposed modeling approach. The respective product is modeled as the root entry of the model; the corresponding components are modeled as child entries (cf. Figure 5 An exemplary RASEN model of the Comment Central Component is shown in the figure below, providing a full detailed description of the Software AG's product, the CCE Vignette, the component tree and the final CWSS scoring.

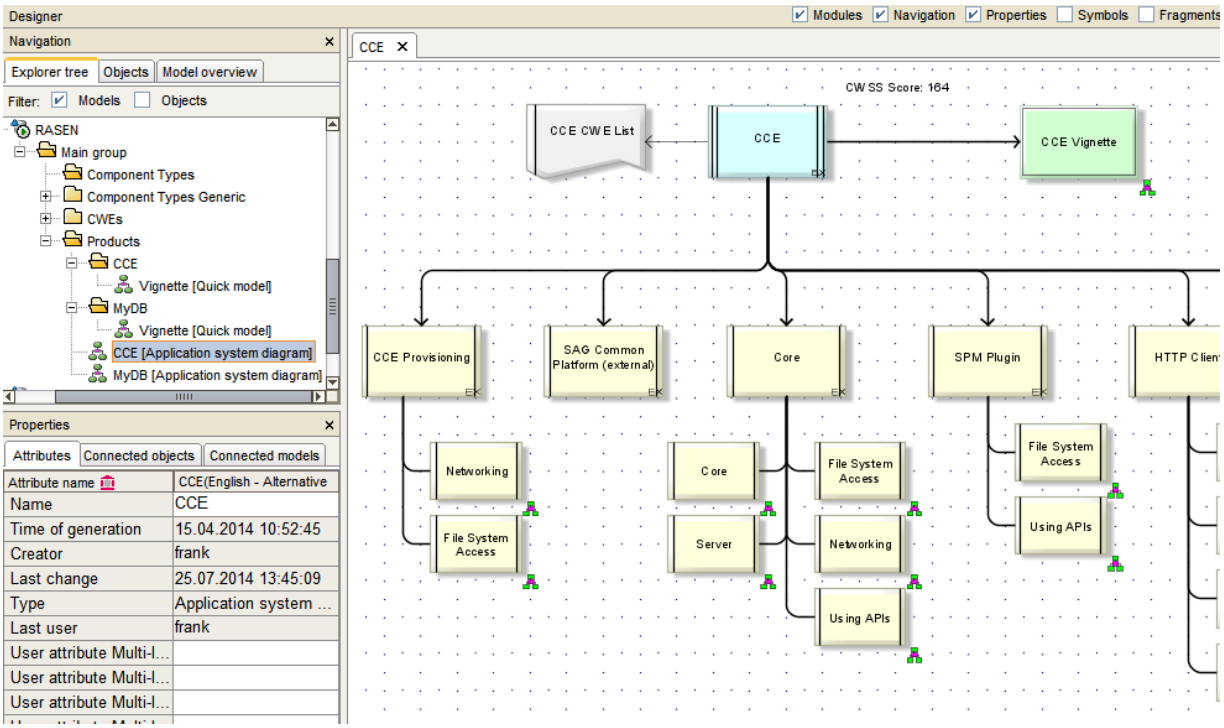

Figure 5 – The final product model with the resulting CWSS Score (CCE Score 164)

At least one so called Component Template which specifies a special type of component is assigned to each component. Additionally a list of CWEs is assigned to the component. This list represents the union of all relevant CWE for this component which are automatically derived from the Component Template. This list can be modified in sense of deleting irrelevant CWEs or extending them according to the needs of the security expert doing the risk assessment.

The present model type represents a program under test as a set of components with their hierarchical relation. This relation is not restricted to one level but as several layers of sub-components building are feasible (subcomponents, sub-subcomponents, etc.), this can also be reflected within the model. Each component has a list of CWEs defined by the Component Template as denoted earlier. The initial content of these lists is defined by the connected Component Types provided as a Risk Template.

The product is now tested for all CWEs in its components lists. Afterwards the test results are imported into the tool and consequently all irrelevant CWEs where deleted from the property list. A CWE Model contains all relevant information as the ID, name and technical impacts about one CWE from the CWE database. The technical impacts where mapped to 8 technical impacts to fit the vignette schema and later on used for score computations. A clipping of the vignette is shown in the figure below.

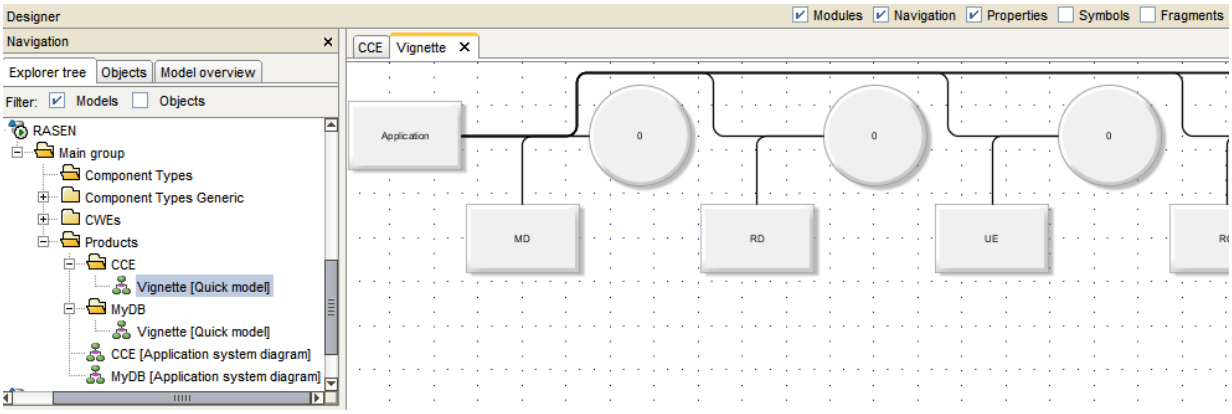

Figure 3 – The Product's Vignette containing information about the deployment scenario

The most frequently used components are defined as generic components which can be used as a kind of library to quickly instantiate a new product with components and accelerate the modelling effort. As shown in the figure below, each Generic Component Type consists of a set of assigned CWEs and the assignment follows best practices.

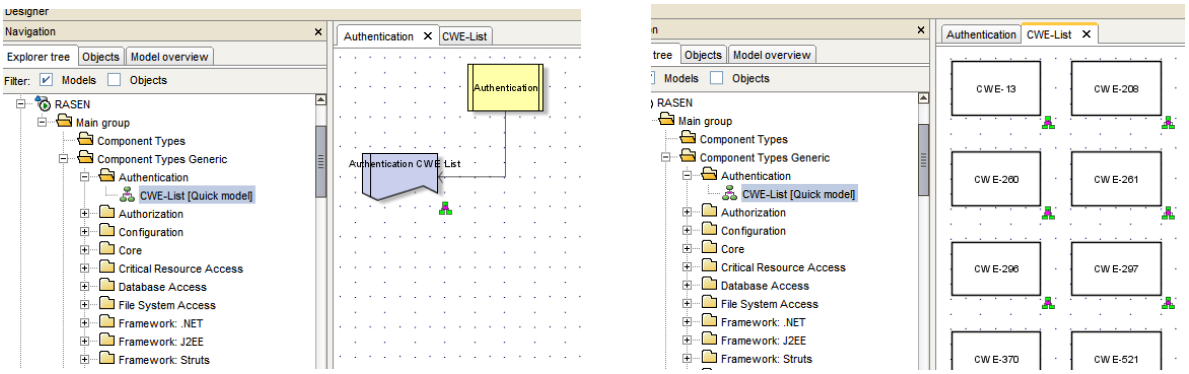

Figure 4 – ARIS Security Risk Assessment: the left the associated CWE list for the generic "Authentication" type; right: Clipping of CWEs linked to the "Authentication" type

The following functionality is implemented as either Reports (run on the ARIS server) or as Marcos (executed in the ARIS client):

- New Product – A wizard which supports the creation all necessary templates and directory for the security risk assessment of a new product
- New Component Type – A Wizard to support the creation of defining user-specific component types
- Import CWE Database --.Import and update the ARIS RASEN CWE database with inputs obtained from the MITRE CWE database
- Import Component Types – Import the set of generic component types from an external file or database
- Union Components – Compute the aggregation of all components
- Evaluate: The functionality which computes from the lower level aggregated and the vignette the CWSS score of the modeled product

An exemplary RASEN model of the Comment Central Component is shown in the figure below, providing a full detailed description of the Software AG's product, the CCE Vignette, the component tree and the final CWSS scoring.

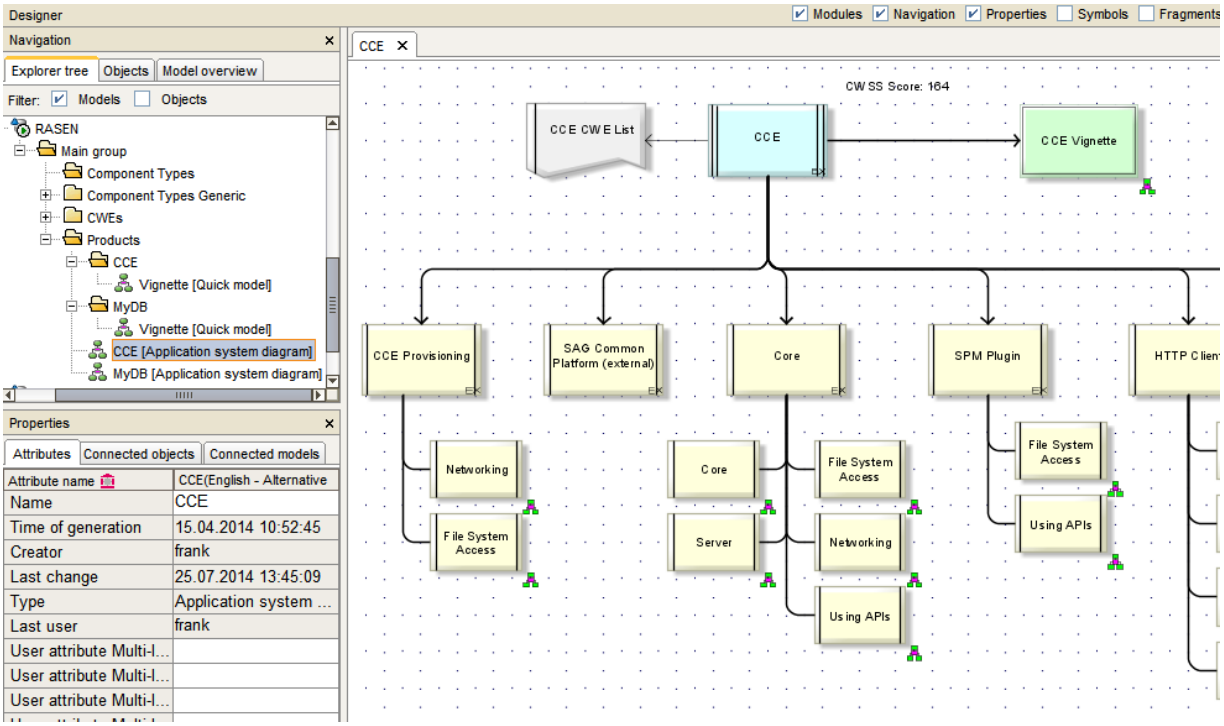

Figure 5 – The final product model with the resulting CWSS Score (CCE Score 164)

3.4 CORAS

In this section we describe how the CORAS tool for model based risk assessment is being extended in the RASEN project. We also describe a stand-alone tool called Capec2Coras which has been developed in RASEN.

3.4.1 Presentation

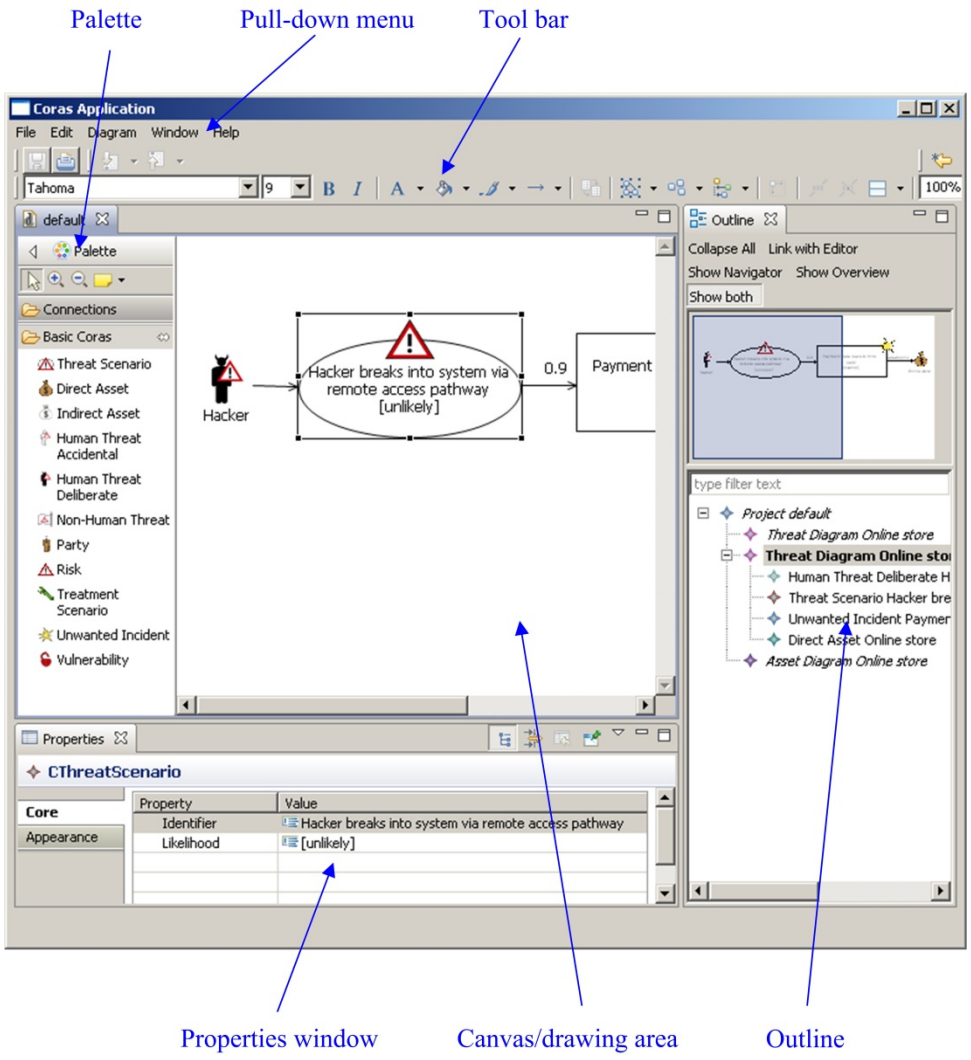

Figure 6 – Screenshot of the CORAS tool

The CORAS tool is an open source diagram editor that supports the CORAS risk analysis language. The CORAS language is a graphical language whose constructs correspond to notions that are relevant during a risk analysis, e.g. threats, vulnerabilities, risks, unwanted incidents, threat scenarios and assets. The CORAS tool is intended to be used intensively during workshops where information is gathered through structured brainstorming. The tool is also intended to be used to document a risk analysis and to present the risk analysis results.

The CORAS tool is designed to support on-the-fly modeling using all five kinds of basic CORAS diagrams, thus facilitating the entire CORAS risk analysis process. A screenshot of the CORAS diagram editor is given in Figure 6. As indicated in the figure, the editor has six main parts:

- A pull-down menu that offers standard functions such as open, save, copy, cut, paste, undo and print.

- A tool-bar that offers easy access to the standard functions of the pull-down menu.
- A palette that contains the model elements and relations for drawing the diagrams.
- A drawing area or canvas for drawing the diagrams.
- A properties window that lists the properties of a selected element, and that can be used to edit the values of the properties.
- An outline that presents a project and its diagrams as a tree.

Except for the pull-down menu and the tool bar, all parts of the tool can be closed or hidden.

In the tool, a project is a collection of diagrams, and each diagram must belong to a project. A project must therefore be created before any diagrams are created.

The outline contains a tree representation of the project. The diagrams of the project are listed at the first level, and under each diagram all the diagram elements are listed. When a new element is created in the drawing area, it is automatically added to the tree under the correct diagram.

The drawing area is the part of the tool where the diagrams are made by inserting, editing, annotating and deleting elements. This is also where likelihoods and consequences are inserted to diagrams as part of the risk estimation, and it is also where risk levels are inserted as part of the risk evaluation.

3.4.2 Installation Guidelines

Extended CORAS tool:

- Download the zip-file containing the current version of the tool (SINTEF-RASEN-CORAS_and_Capec2Coras_Tool_Deliverable.zip).
- Extract the zip-file to any folder of your choosing.
- Double click the file "Coras.exe" located in the Coras folder of the distribution.

Capec2Coras

- Download the zip-file containing the current version of the tool (SINTEF-RASEN-CORAS_and_Capec2Coras_Tool_Deliverable.zip)..
- Extract the zip-file to any folder of your choosing.
- Double click the file "Capec2Coras.exe" located in the Capec2Coras folder of the distribution.

3.4.3 User Guide: CORAS

Creating a new project

Before the CORAS tool can be properly used, a new project must be created. To create a new project:

- Select File -> New -> Coras Project in the File menu located near the top left corner of the window.
- Enter the desired file name and folder of the new project.
- Check “High Level CORAS”, “Dependent CORAS”, or “Legal CORAS” if you want to use these extensions of the basic CORAS language.
- Press “Finish” when done.

Creating a new diagram

The first time a new project is created, a new threat diagram called unnamed is automatically created within the project as shown in the outline view of the right hand side of the Figure 6. To create an additional diagram:

- Right-click the project entry in the tree outline located on the right hand side of the CORAS tool window.

- In the pull-down menu, select “new X diagram” (where X is the type of diagram that can be created e.g. threat, asset).
- The new diagram should appear in the tree outline. In order to open the new diagram, double click the diagram in the tree outline.

The difference between the diagrams is the kind of risk model elements that can be created within them. Note that the diagram type "Treatment" can contain (almost) all risk model elements; all other diagrams are subsets of this diagram (except the asset diagram).

Editing diagrams

Once a project and an appropriate diagram have been created, risk model elements can be added to the diagram by selection the appropriate element in the palette on the left hand side of the window, and left-clicking the diagram canvas. Relations can be created by selecting them from the pallet, or by holding the mouse over a source or target element until two small boxes appear on the border of the element. Holding down the left-mouse button on one of these small boxes will allow you to create a new relation.

3.4.4 User guide: Capec2Coras

Capec2Coras is program that enables the user to transform an XML file containing CAPEC attack patterns into a CORAS file which can be read by the CORAS tool. The tool allow the CAPEC-file to be viewed and edited before being transformed into CORAS. Parameters which are needed by the transformation can also be configured.

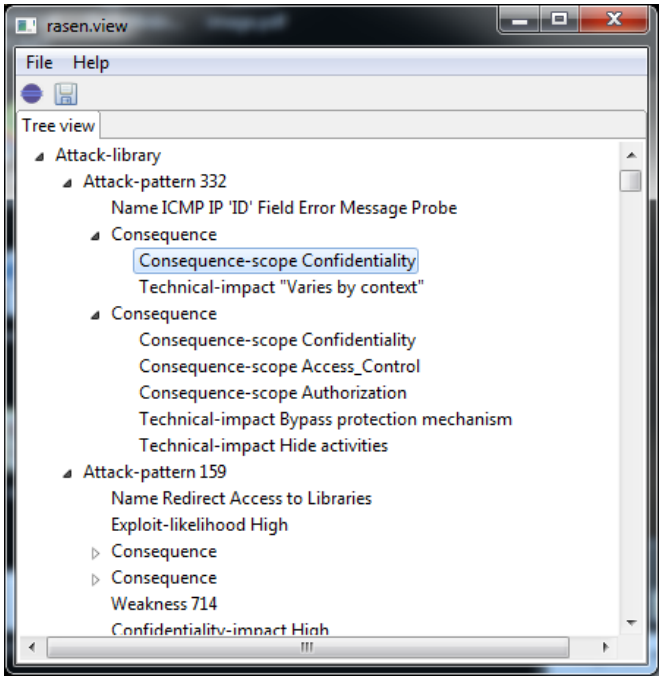

Figure 7 - Screenshot of Capec2Coras tool

After starting the Capec2Coras tool, the user is presented with a tree view as shown in Figure 7. This is where the CAPEC attack patterns and the parameters of the transformation can be edited. The tool has two main functions:

- Open: This function allows two kinds of files to be opened and displayed in the tree view of the Capec2Coras tool. The first kind of file is an XML-file containing a CAPEC library of attack patterns. The second kind is a file (not XML) which contains a previously edited version of the imported CAPEC library into the Capec2Coras tool.
- Save as: This function allows two kinds of saves (depending on the extension of the file being saved). The first kind transforms and saves the file into a CORAS-file (if the file extension

".coras_project" is used). The second kind saves the CAPEC library into a file so that he edits made to the file are not lost (if the file extension is not ".coras_project").

After a CAPEC library has been opened in the Capec2Coras tool, a tree view containing two top nodes is displayed as shown in Figure 8. These are "Attack-library" and "Default-likelihoods". The former contains all the attack-patterns which are defined in the CAPEC library, the second contains values which are used as parameters by the transformation. Specifically, default values which are produced by the transformation to CORAS can be configured as shown in Figure 8.

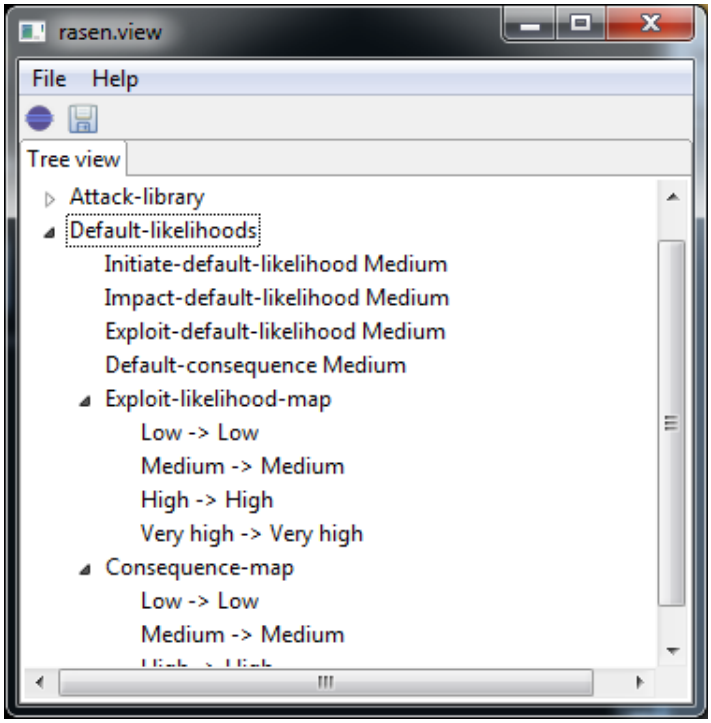

Figure 8 - Screenshot showing transformation parameters

3.4.5 Features within RASEN project

Planned features

The following extensions to the CORAS tool are have been planned within the RASEN project

- Develop import/export from CORAS to RASEN generic format. The export should include test procedures generated from the CORAS model. The import should include measurements which have been generated on the basis for security test results.
- Develop support for CORAS risk model generation based on CAPEC and predefined vulnerabilities.
- Add support for identifying, prioritizing and selection test procedures based on the risk model. This feature is broken down into the following features/tasks:
    - Implement an algorithm for calculating test procedure priority.
    - Implement support for specifying estimates needed for test procedure prioritization that is not part of the standard CORAS risk model. Specifically, estimates for *effort* and *uncertainty* need to be supported.
    - Implement support for specifying likelihood, consequence, and risk types. The calculation of the test procedure priority relies on the likelihood and consequence estimate in the risk model. However, for this calculation to be correct, the user needs

to specify what kind of likelihood and consequence estimates are in the model, e.g. if the likelihood is probability, or a frequency, etc.

Current status

The main changes/additions to the CORAS tool since the previous deliverable are:

- The export function from CORAS to RASEN generic format has been updated.
- The prioritization algorithm for generating test procedures has been updated and annotations which are needed by the algorithm no longer need to be hard-coded, but can be specified in the tool.
- Support for importing RASEN data models and showing measurements which can be used to update the estimates of the risk model.
- Predefined vulnerabilities/weaknesses are now imported into the tool when loading CORAS-projects which are generated from a CAPEC library (see Capec2Coras user guide).

The second version of an export from the CORAS model to the current version of the RASEN data meta-model has been implemented. This version addresses the new version of the RASEN meta model which is a simplification of the old version (see WP5 integration deliverable). The function takes a CORAS project as input and produced an XML-file representing the RASEN data model.

The algorithm for test procedure prioritization and selection has been updated in the current version the CORAS tool. The user interface has only changed minimally: The user can now annotation transitions with uncertainty and effort values in addition to likelihood values. The main updated is the algorithm for calculation the priority values which is more accurate now in the current version than the previous version of the tool.

A new button labeled "Import" has been added to the GUI interface as shown in Figure 9. This function enables the user to import a RASEN-file containing risk assessment measurements and to display the measurement results in the text area at the button of the screen. These measurements can then be used as a basis for updating the estimates of the risk model.

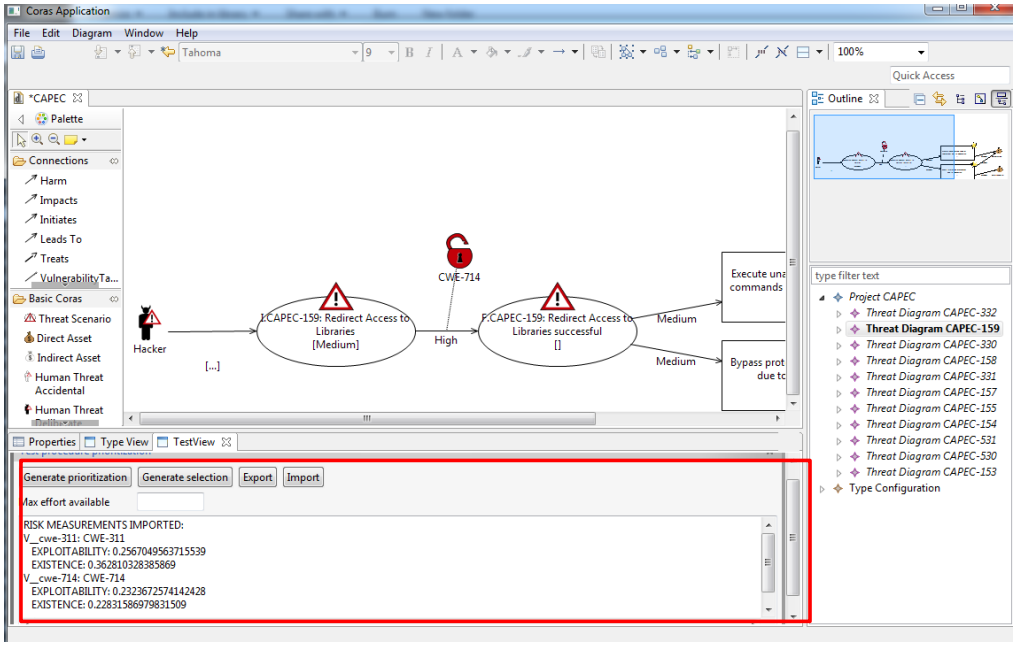

Figure 9 - Measurement imported from RASEN model

The main remaining tasks for the tool development are:

- Improve the measurement import functionality such that parts of the CORAS model can be automatically updated based on the measurements and such that security test measurements are aggregated into risk assessment measurements.
- Develop support for risk and uncertainty visualization
- Improve development of GUI-support for type specification, and test prioritization/selection.

4 Conclusion

In this report we have given an overview of the RASEN WP3 and WP4 tools. The tools presented in this deliverable are CORAS (SINTEF), RISKTest (Fraunhofer FOKUS), CertifyIt (Smartesting) and ARIS Business Architect (Software AG). In addition to give an introduction to the features and purposes of these tools, we have described the further development goals and the current status of these tools.