D5.5.A Automatic or semi-automatic techniques, methods and tools for revalidation

Fredrik Seehusen, Bjørnar Solhaug, Ketil Stølen (SIN)

Document information

Document Number	D5.5.A
Document Title	Automatic or semi-automatic techniques, methods and tools for revalidation
Version	1.0
Status	Final
Work Package	WP 5
Deliverable Type	Other (Appendix to D5.5 prototype)
Contractual Date of Delivery	N.A.
Actual Date of Delivery	2012-01-26
Responsible Unit	SIN
Contributors	SIN
Keyword List	Risk assessment, change, tool support, automation
Dissemination level	PU

Document change record

Version	Date	Status	Author (Unit)	Description
0.1	12-01-03	Draft	B. Solhaug (SIN)	Table of contents and first draft of all sections
0.2	12-01-11	Draft	F. Seehusen (SIN), B. Solhaug (SIN)	Updated all sections
0.3	12-01-23	Draft	B. Solhaug (SIN)	Added section on case study validation
0.4	12-01-24	Draft	B. Solhaug (SIN)	Prepared for quality check
0.5	12-01-24	Quality check	M. Angeli (UNITN)	First quality check completed; minor remarks
0.6	12-01-25	Draft	B. Solhaug (SIN)	Revised after quality check
0.7	12-01-26	Draft	F. Seehusen (SIN)	Added Section 3.1 and appendix
1.0	12-01-26	Final	B. Solhaug	Finalized for submission

Executive summary

D5.5 is a prototype risk assessment tool that provides automation and semi-automation of tasks that are conducted by following the SecureChange method for risk assessment of changing and evolving systems. D5.5 builds on the prototype tool delivered as D5.4. The latter is a diagram editor for building risk models to identify and estimate risks, as well as changes to risks as systems evolve. D5.4 moreover supports the specification and documentation of mapping rules between the target models and the risk models to facilitate the tracing of system changes to the risk models in a systematic way.

The D5.5 prototype tool provides automated support in several ways. First, the tool automatically generates the target model index that is used to create the mapping rules. Second, when the system changes and the target model is modified accordingly, the tool automatically updates the generated index and moreover identifies the changes by creating the delta. Third, based on the mapping rules and the delta, the tool flags the parts of the risk models that may be affected by the changes and therefore needs to be assessed anew. Fourth, the tool makes automatic syntax constraint checking that includes detecting inconsistencies in the risk models with respect to change. Together with the flagging of change-affected risk diagrams the latter feature gives automated support for systematically rippling changes from the target models and through all potentially affected parts of the risk models.

Index

1 Introduction

This document presents the prototype tool of SecureChange deliverable D5.5 by describing its main features and functionalities, as well as its main purposes in the setting of risk assessment of changing and evolving systems. The tool is closely aligned with the method for risk assessment of changing and evolving systems presented in deliverable D5.3 as well as the risk modeling language support presented in D5.2 and D5.3. The tool moreover builds on the previous tool deliverable, i.e. D5.4, by automating several of the risk assessment tasks that are supported by D5.4.

The approach of the research tasks of WP5 is to develop and deliver artifacts for risk assessment of changing systems that are general in the sense that they can be instantiated by several specific approaches to risk assessment. Such an instantiation is in D5.3 made in CORAS [1] to demonstrate and explain the more general approach of WP5 by a specific instance. The instantiation results in the generalization of the CORAS approach to the setting of the risk assessment of changing and evolving systems. The instantiation thereby offers the CORAS risk assessment process, risk assessment techniques and risk modeling language for change. In the development of the prototype tools we have implemented the artifacts in the CORAS instantiation.

In Section 2 we present the main objectives of the D5.5 prototype. In Section 3 we present the main functionality of the prototype, illustrated with screenshots and a running example drawn from the ATM domain. In Section 4 we briefly summarize the SecureChange case study activities involving the WP5 tools. Finally we conclude in Section 5 by summarizing. Some technical details are given in the appendix.

2 Objectives

The overall objective of the D5.5 prototype is to provide automated support for various tasks that are conducted by the CORAS method for the risk assessment of changing and evolving systems. In the context of SecureChange WP5, three main artifacts have already been developed, namely the method (D5.3), the language (D5.2 and D5.3) and the diagram editing tool (D5.4). These three artifacts are closely aligned and combine into the SecureChange approach to risk assessment of evolving systems: The language is actively used in conducting the phases and tasks of the method, and supports the modeling and evaluation of changes to risks; the editing tool is used for making all the diagrams that are used, and facilitates quick on-the-fly risk modeling during risk identification workshops.

Specifically, the D5.5 prototype builds on the D5.4 prototype by extending it with several new features as described in the following.

- In order to support traceability of changes from the target of analysis to the risk model the D5.4 tool comes with an indexing editor for indexing elements of the target model. The index is used for referring to the target model in the tool and specifying mapping rules for traceability. In D5.5 the tool automatically generates the target model index.
- When the system changes, the target model is modified accordingly. To handle this, the D5.5 tool automatically updates the generated index and moreover identifies the changes by creating the delta between the target model before changes and the target model after changes. The delta shows which parts of the target remain unchanged, which parts have been removed, and which parts are added under the change.
- Based on the mapping rules and the delta, the system changes can be traced from the target model to the risk model. This facilitates the identification of the risks that may be affected by the system changes and therefore need to be reassessed. To support this, the D5.5 tool automatically detects and flags these parts of the risk models.
- The risk modeling language supports the explicit modeling of changes to risks by distinguishing between risks that become obsolete after change, risks that persistent under change, and risks that emerge after change. With respect to this, the D5.5 tool makes automatic syntax constraint checking that includes detecting inconsistencies in the risk models with respect to change. Together with the flagging of change-affected risk diagrams the latter feature gives automated support for systematically rippling changes from the target models and through all potentially affected parts of the risk models.

In the next section we describe in more details the main functionality of the D5.5 prototype and give examples of its use.

3 Main Functionality of Tool

In order to describe the main functionality of the tool we give a small illustrative running example that shows the various automated features. The example is a fragment extracted from the SecureChange case studies drawn from the ATM domain. It should be noticed that a main objective of the tool is to support on-the-fly risk modeling during structured brainstorming sessions where risks are identified and assessed. The examples in this document give an idea of how the tool facilitates the handling of change in this setting.

3.1 Model schema loader

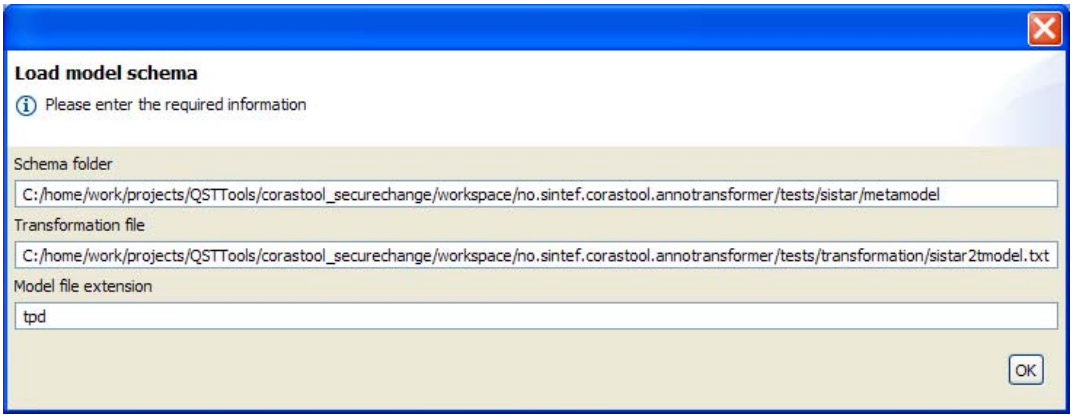

Figure 1- Load model schema dialog.

By default, the tool can generate indexes from models that conform to our intermediate target metamodel (defined in Deliverable D5.3). However, the tool can also generate indexes from any system model, as long as the system model is defined according to an Ecore metamodel (specified by a set of XSD schema files).

In order to generate indexes from an arbitrary system model *M*, the user first has to

- Load the Ecore metamodel *MM* that the system model *M* conforms to;
- Load a transformation specification which maps models conforming to *MM* into models conforming to our intermediate target metamodel.

Upon generating indexes from *M,* the tool will (1) load *M, (2)* transform *M* into an intermediate model *TM* conforming to our target metamodel, and (3) generate indexes from *TM.* See the appendix for a more detailed description of the transformation language.

The dialog window where the user can load metamodels and transformations is shown in Figure 1.

3.2 Index Generation

The WP5 risk identification and risk modeling use a description of the relevant parts of the target of analysis as input and basis. This description is built as part of the context establishment and documented in a suitable language such as the UML. In order to trace changes from the target model to the risk model, the tool supports the specification of mapping rules between risk model elements and target model elements. The pointers to the target model elements in the tool are in terms of a target model index that is automatically generated by the tool. The indexing is based on the meta-model of the chosen language for the modeling of the target of analysis.

In the ATM case study documented in D5.3 the target description was made by means of various UML models such as class diagrams, structured classifiers, activity diagrams and sequence diagrams. Figure 2 shows a fragment of these models and depicts a simplified specification of the sequence creation task that is part of the arrival management.

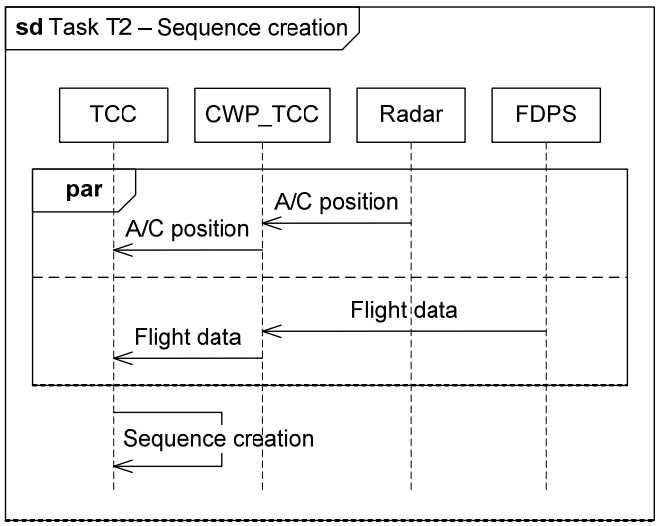

Figure 2 - Fragment of UML target model.

Given the finalized target models, as exemplified here, the tool takes their meta-model representation as input and automatically generates the index. The use of this feature is shown in Figure 3 where the index is generated for the chosen target model.

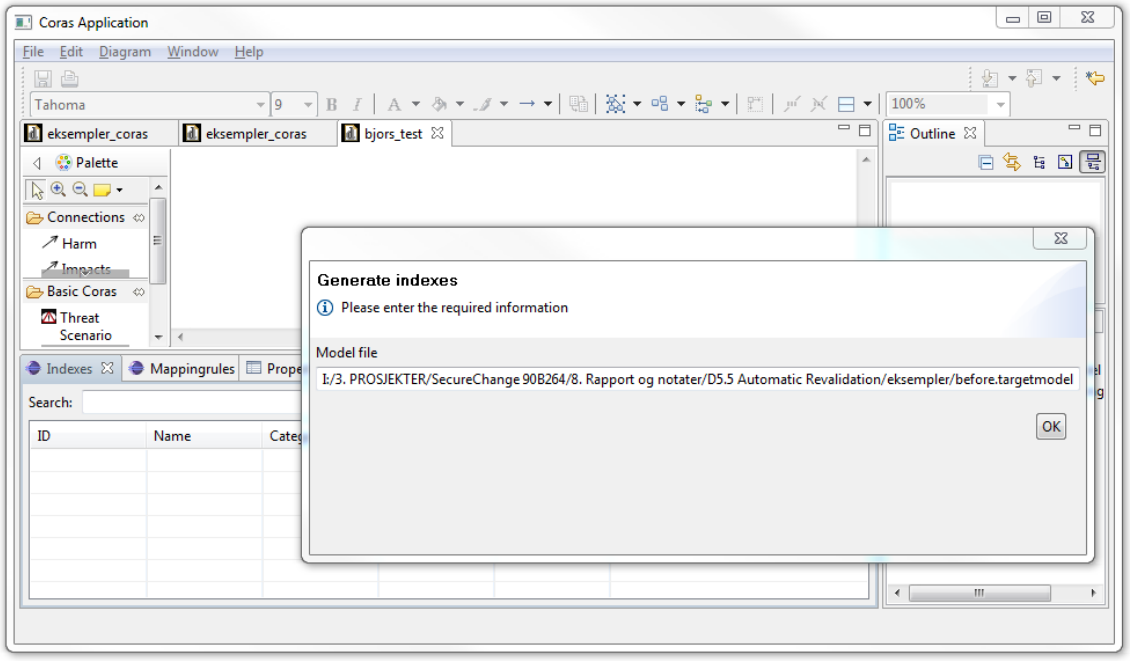

Figure 3 - Automatic index generation.

The index is represented in the tool as a set of tuples (*ID, Name, Category, Description, Mode*) as shown in Figure 4. *ID* is a unique automatically generated identifier, *Name* is the name of the element as specified in the target model, *Category* is the kind of target model element (*Actor*, *Event* or *Scenario*), and *Mode* specifies the mode of the target model element with respect to change. The mode is either *Before*, *After* or *Before-After*, denoting, respectively, that the element is part of the target of analysis only before changes, only after changes or both before and after changes. Note that before changes are introduced, the default mode is *Before*, as shown in the example. Finally, *Description* is an initially empty field that can be filled in by the user if further explanation is desired or needed. Notice that the set of indexes is only used for the purpose of traceability and is not used as part of the graphical risk models.

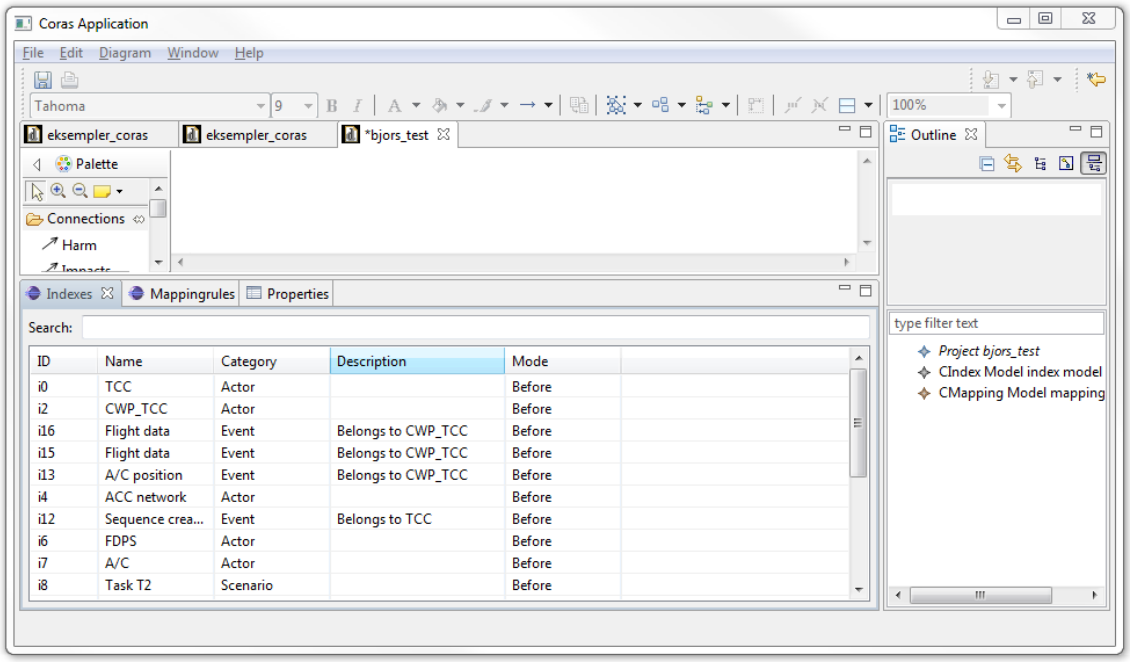

Figure 4 - Target model indexes.

So far the tasks are part of the context establishment before the actual risk identification and risk modeling begins. During risk identification, CORAS threat diagrams are made to do the assessment and documentation of the risks. An example is shown in Figure 5.

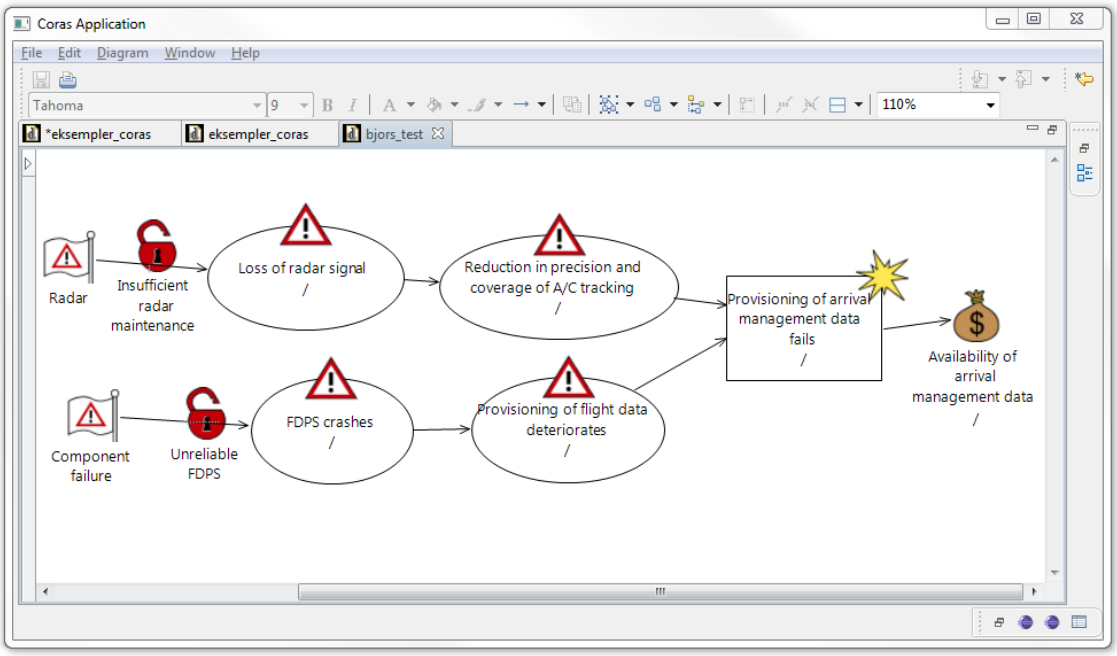

Figure 5 - Risk identification before changes.

3.3 Specifying Mapping Rules

Given the target model index as exemplified in Figure 4 and the threat diagrams as exemplified in Figure 5, mapping rules are created to specify the trace model that links the target model and the threat diagrams. A mapping rule is a pair of a target model index and a threat diagram identifier and is specified in the tool as shown in Figure 6. In the CORAS threat diagrams, the mapping rules are visualized by means of target element icons that are tagged with a chosen, intuitive name.

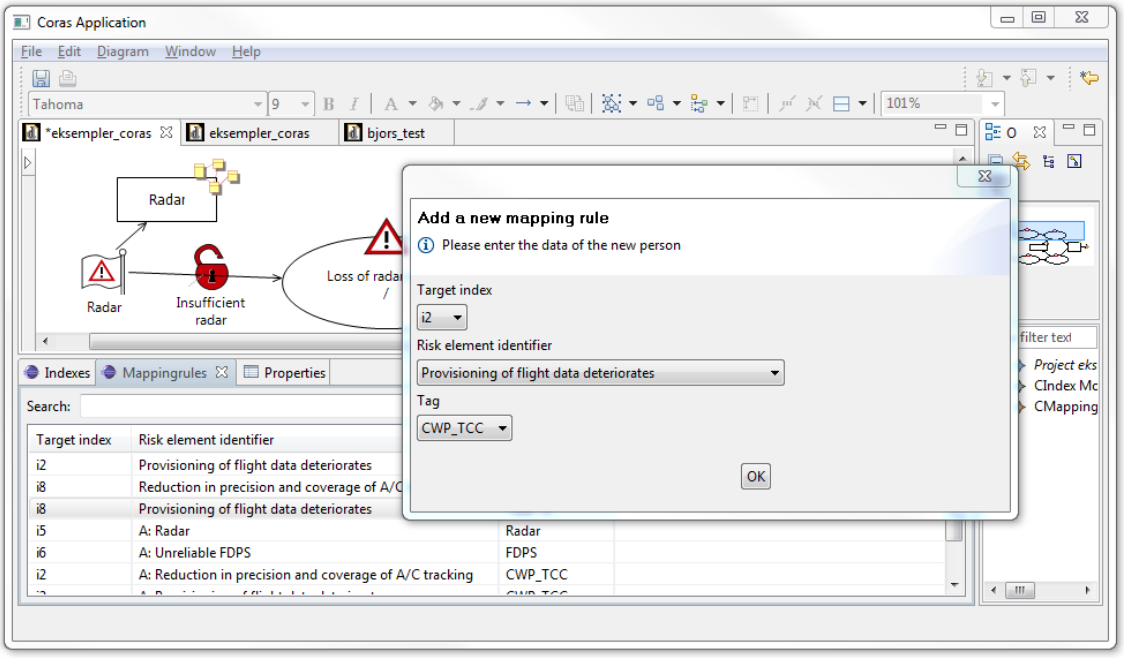

Figure 6 - Specify mapping rule.

The mapping rules and the visualization are exemplified in Figure 7, where, for example, the threat named *Radar* is mapped to the target model element of the same name. Figure 8 shows the full threat diagram in question with the visualization of the specified mapping rules.

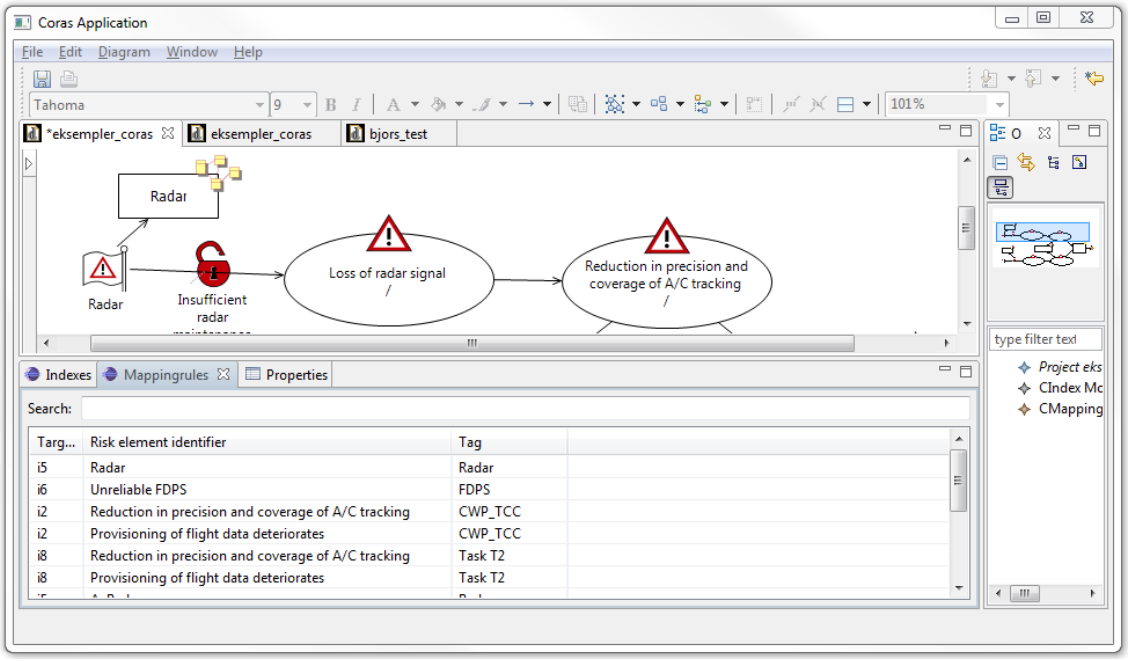

Figure 7 - Mapping rules and their visualization.

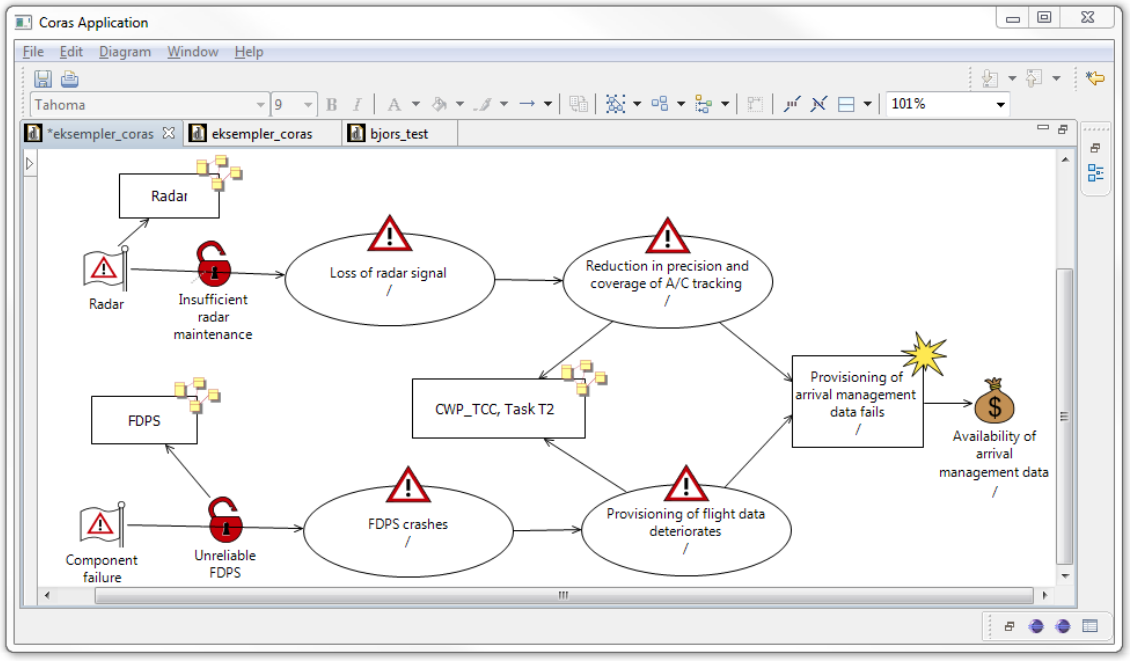

Figure 8 - Risks and trace model before changes.

3.4 Generation of Index and Diff after Change

The risk assessment tasks and the tool support as described above serve as a basis for systematically handling changes. To continue the running example, Figure 9 shows changes to the target description by the introduction of the AMAN and the ADS-B. (The

use of colors is only for highlighting.) The tool now gives support for identifying the delta of the target models and tracing changes from the target models to the current risk models that may need to be revised and updated.

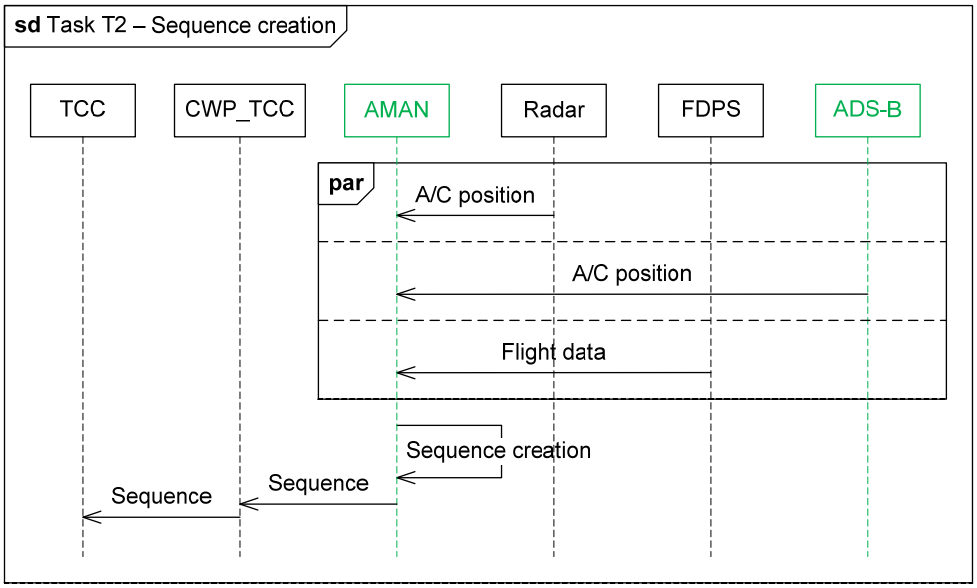

Figure 9 - Fragment of UML target model after change.

As shown by Figure 10, the tool takes the target model file for the target before and the file for the target after to generate the updated indexes. Based on these files and the current set of indexes, the indexes for the target models after change shows not only all target model elements before and after changes, but also their mode with respect to change, i.e. the diff (delta). The new index is exemplified in Figure 11. Notice, for example, that AMAN has mode *After* since it exists only after the changes.

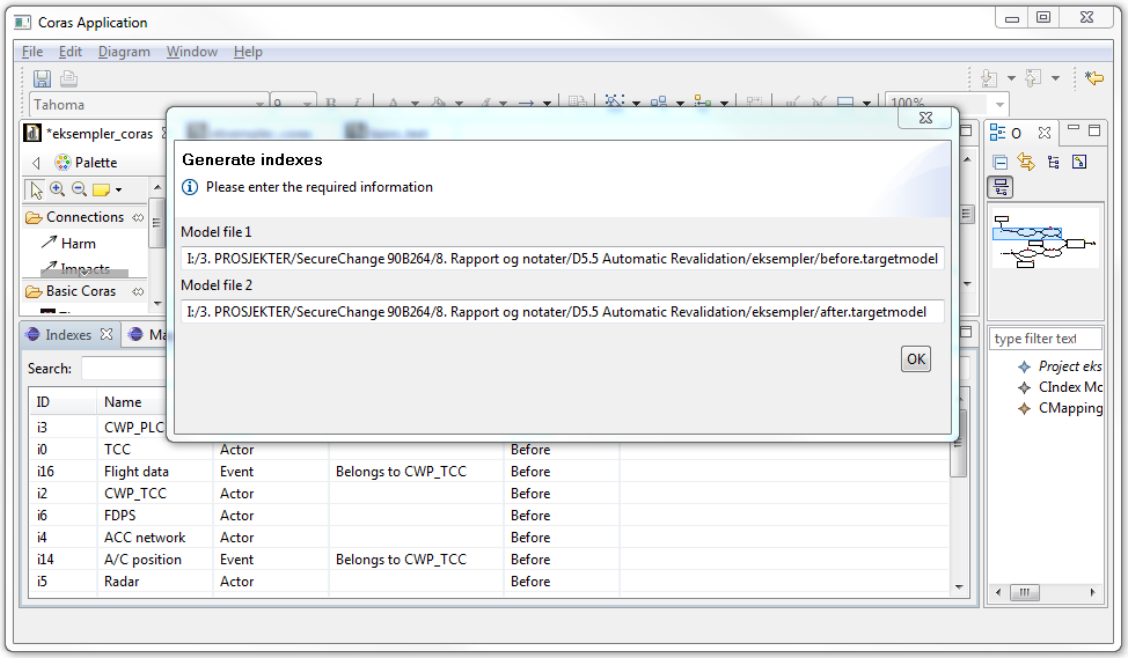

Figure 10 - Automatic generating index and delta after change.

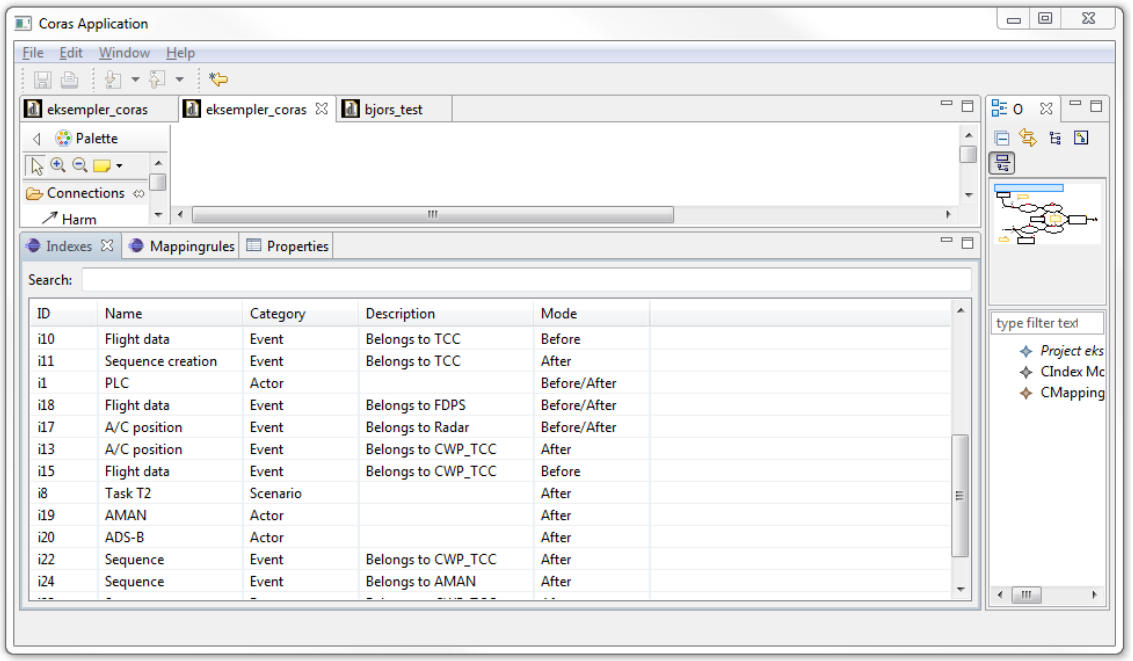

Figure 11 - Target model indexes after change.

3.5 Automatic Flagging of Risks Affected by Change

Based on the trace model specified before change and the new index with the automatically identified delta, the tool offers automatic flagging of risk model elements that may be affected by changes in the target of analysis. This is exemplified in Figure

12 with the message "The target element related to this risk element has been changed". This supports the systematic handling of change in the risk assessment, in particular when the number of CORAS threat diagrams is large.

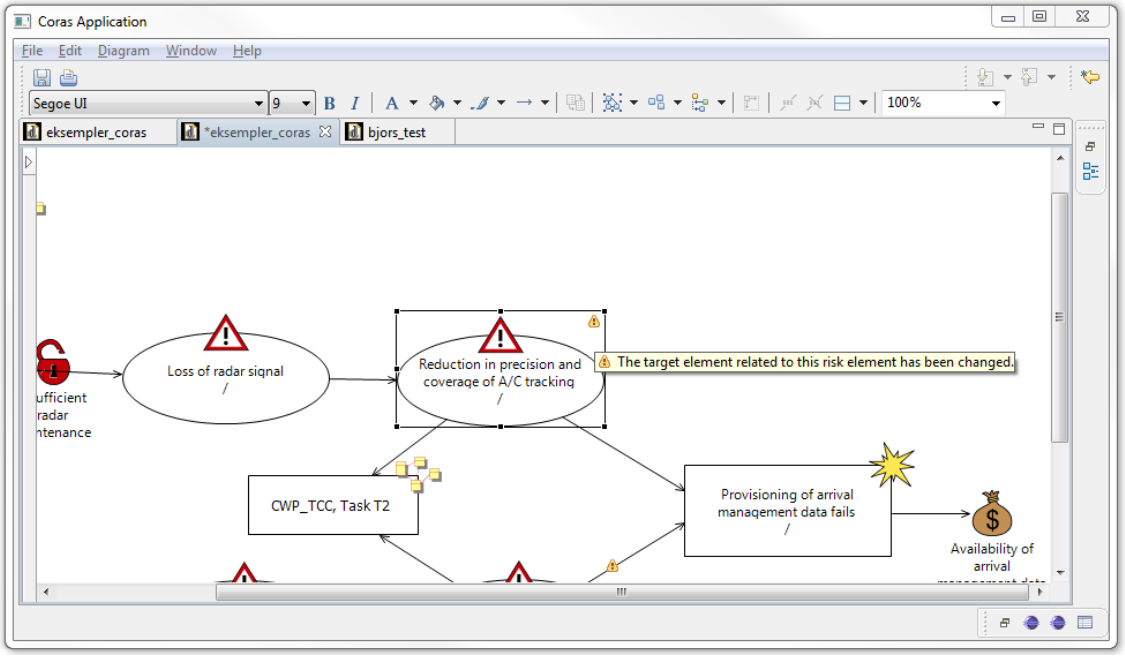

Figure 12 - Automatic flagging of risk model elements.

3.6 Automatic Syntax Checking

The identification, modeling and documentation of changes to risks is supported by the language and tool by the distinction between risks that are present only before changes, risk that are present only after changes, and risk that are present before and after changes. When updating the risk models to account for the changes, each risk model element is therefore assigned one of the modes *Before*, *After* and *Before-After*. Obviously, a risk element that is present only before changes cannot be related to a risk element that is present only after changes. When the mode of one risk model element is changed in order to account for changes in the target of analysis, this will usually impact the related risk model elements. In other words, changes usually ripple through the risk models. The automatic syntax checking is a valuable feature in systematically rippling the changes as the syntax error warnings progressively detects related model elements that may need to be reassessed and possibly assigned a new mode after change.

This is exemplified in Figure 13 where the mode of the threat scenario *Reduction in precision and coverage of A/C tracking* (mode *Before*) in inconsistent with each of the three elements it is related to (mode *After*). Each of these three inconsistencies is flagged with a red warning sign that, when the mouse is hovered above it, gives the warning messages "If source has model before or after, then target must have the same mode" and " If target has model before or after, then source must have the same mode ". If, for example, the inconsistency with the threat scenario *Loss of radar signal* is resolved by changing its mode from *After* to *Before*, the warning disappears, but a

new one appears on the relation between *Loss of radar signal* and the threat *Radar*. In this way the changes that are made in the model ripples through diagrams until the diagrams are consistent with respect to the modes.

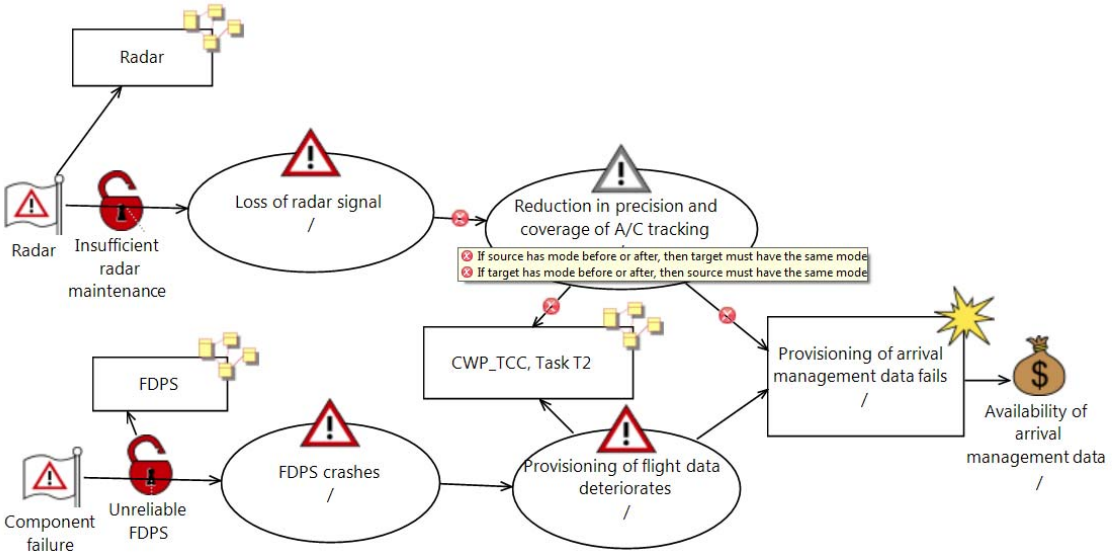

Figure 13 - Automatic syntax checking wrt change.

Finally, we show Figure 14 to exemplify a final CORAS threat diagram after change. Here the mode of all elements apart from the upper three is *Before-After* since they are present both before and after. The upper part is related to the ADS-B which is introduced as part of the change, and therefore has mode *After*.

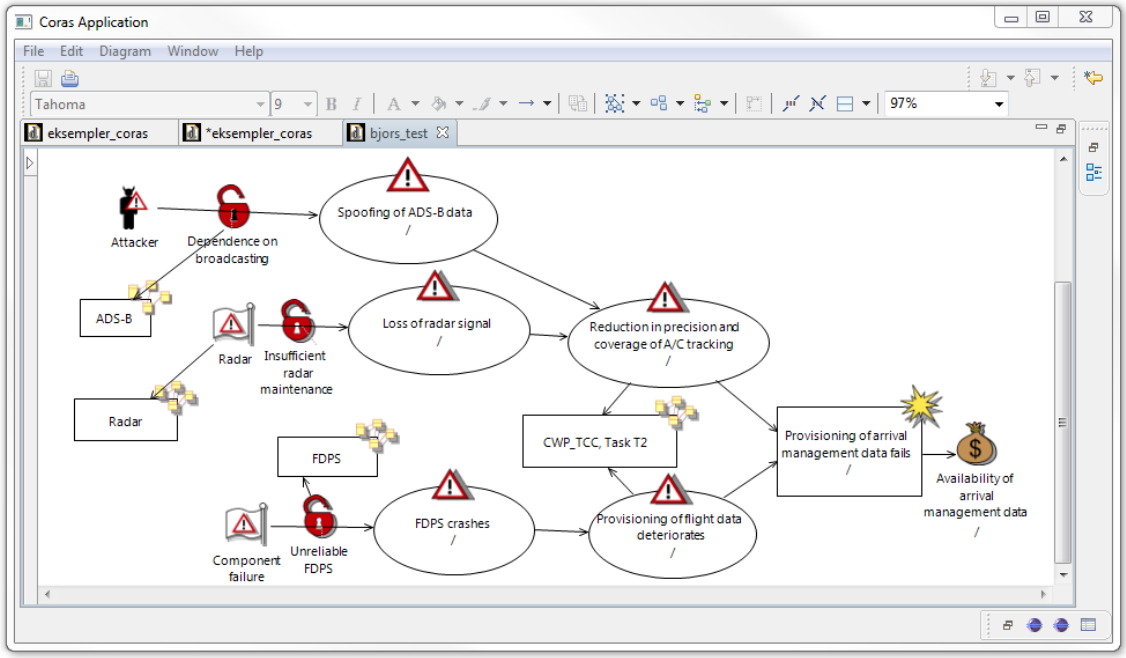

Figure 14 - Final threat diagram after change.

4 Validation

In this section we briefly summarize the validation activities that have involved the WP5 prototypes. For further details we refer to other SecureChange deliverables.

During the third year of the project, the WP5 tools were applied and evaluated in the ATM case study with dedicated workshops involving ATM experts. Separate workshops were held in June and September 2011. At the first workshop the risk identification was led by a representative from WP5 who also did the on-the-fly modeling using the tool support. At the second workshop the ATM experts themselves did the risk identification and risk modeling after being given an introduction to the method, language and tool. The ATM experts were moreover presented the target of analysis, the security properties, the risk evaluation criteria, and the change requirements. The results of these validation activities are documented in SecureChange deliverable D1.3.

The WP5 risk assessment tools were moreover used in validation activities that involved several steps of the security tailored V-model, as well as several tools from the SecureChange tool portfolio. The tools included SI* (part of SecMER), CORAS, Rinforzando and CARiSMA, each of which are depicted in the SecureChange tool roadmap shown in Figure 15. For details about these activities we refer to Appendix E of D1.3.

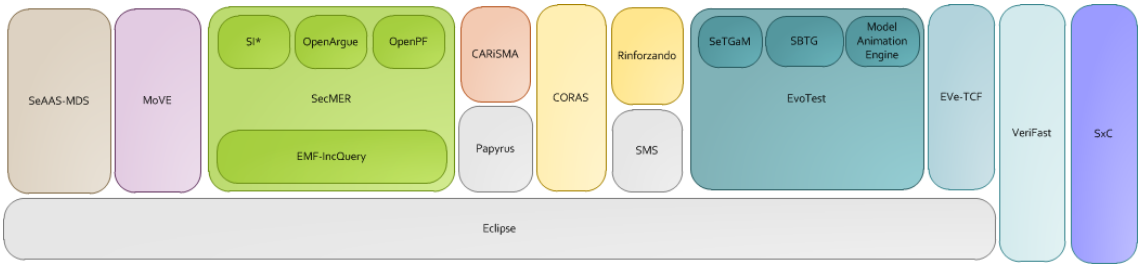

Figure 15 - SecureChange tool roadmap.

In the context of WP2 validation activities, the WP5 risk assessment method and tool were applied for risk identification and risk modeling in the HOMES case study. Deliverable D2.3 reports on how various SecureChange tools map on the Integrated SecureChange process that was presented in D2.2. In particular, D2.3 reports on the use of the MoVE tool as a backend tool to support collaboration in a change-driven engineering process involving different frontend tools.

5 Conclusion

The overall objective of WP5 is to develop an approach to risk assessment of changing and evolving systems. The three main kinds of artifacts that are delivered for this purpose are a method for risk assessment of changing systems (D5.3), languages for the modeling of changing risks (D5.2 and D5.3), and prototype tools to support the former two (D5.4 and D5.5). The risk modeling languages are tightly interwoven with the risk assessment method, and the prototype tools are designed to support and facilitate the use of the modeling languages during the various activities of the risk assessment tasks.

In this document we have presented and explained the main functionality and purposes of the SecureChange prototype deliverable D5.5. D5.5 builds on D5.4, a main purpose of which was to enable efficient on-the-fly modeling while ensuring that the diagrams that are made are clearly presented, easily understandable and syntactically correct.

As explained, exemplified and documented in this report, a main purpose of the D5.5 prototype is to provide automatic and semi-automatic support for several of the risk assessment tasks that are conducted by following the WP5 risk assessment method and techniques. The features are summarized as follows.

- The tool automatically generates the target model index that is used to create the mapping rules to enable traceability between the target models and the risk models.
- When the system changes and the target model is modified accordingly, the tool automatically updates the generated index and moreover identifies the changes by creating the delta.
- Based on the mapping rules and the delta, the tool flags the parts of the risk models that may be affected by the changes and therefore needs to be assessed anew.
- The tool makes automatic syntax constraint checking that includes detecting inconsistencies in the risk models with respect to change. Together with the flagging of change-affected risk diagrams the latter feature gives automated support for systematically rippling changes from the target models and through all potentially affected parts of the risk models.

References

[1] Lund, M. S., Solhaug, B., Stølen, K., ”Model-Driven Risk Analysis – The CORAS Approach”, Springer, 2011.

Appendix A: Schema loading

The transformation which can be loaded by the tool can

- Map elements of the source model to the target model. For example, if the source model has elements called “Lifeline”, then we can specify that “Lifeline” elements of the source model will be transformed to, e.g. “Actor” elements of the target model.
- Map attributes of the source model elements to attributes of elements the target model.
- Map references of elements in the source model to references of elements in the target model.

The transformation language used by the tool is defined by the following grammar:

```
Start    := <TRule>? (, <TRule>*)
TRule    := <Name> -> <Name> : <TAttrRef>* (, <TAttrRef>*)
TAttrRef := TAttr | TRef
TAttr    := <Name> => <Name>
TRef     := <Name> (. <Name>)* -> <Name>
```

As an example, consider the following transformation:

```
Lifeline -> Actor :
  name => name,
  messageEnds.message -> events
;
Message -> Event :
  name => name
;
```

This specifies the following mapping:

- All *Lifeline* elements are mapped to *Actor* elements, and
    - The *name* attribute of a *Lifeline* element will be mapped to the *name* attribute of the *Actor* element;
    - All elements reached by first traversing the *messageEnds* reference of a *Lifeline* element, and then traversing the *message* reference (of elements that are reached by traversing *messageEnds*), will mapped to the elements reached by the *Actor* element by traversing the *events* relation of *Actor*.
- All *Message* elements are mapped to *Event* elements, and
    - The *name* attribute of a *Message* element will be mapped to the *name* attribute of the *Event* element.