predict
prioritise
prevent

TRESPASS

Technology-supported Risk Estimation by Predictive Assessment of Socio-technical Security

Deliverable D6.1.2

Final requirements for tools integration

Project:	TRESPASS
Project Number:	ICT-318003
Deliverable:	D6.1.2
Title:	Final requirements for tools integration
Version:	1.0
Confidentiality:	Public
Editor:	H. Jonkers
Cont. Authors:	J. Barendse, R.R. Hansen, D. Ionita, H. Jonkers, M. Martins, W. Pieters, C.W. Probst, S. Saraiva
Date:	2015-10-30

SEVENTH FRAMEWORK PROGRAMME

Part of the Seventh Framework Programme
Funded by the EC-DG CONNECT

Members of the TRESPASS Consortium

1.	University of Twente	UT	The Netherlands
2.	Technical University of Denmark	DTU	Denmark
3.	Cybernetica	CYB	Estonia
4.	GMV Portugal	GMVP	Portugal
5.	GMV Spain	GMVS	Spain
6.	Royal Holloway University of London	RHUL	United Kingdom
7.	itrust consulting	ITR	Luxembourg
8.	Goethe University Frankfurt	GUF	Germany
9.	IBM Research	IBM	Switzerland
10.	Delft University of Technology	TUD	The Netherlands
11.	Hamburg University of Technology	TUHH	Germany
12.	University of Luxembourg	UL	Luxembourg
13.	Aalborg University	AAU	Denmark
14.	Consult Hyperion	CHYP	United Kingdom
15.	BizzDesign	BD	The Netherlands
16.	Deloitte	DELO	The Netherlands
17.	Lust	LUST	The Netherlands

***Disclaimer**: The information in this document is provided "as is", and no guarantee or warranty is given that the information is fit for any particular purpose. The below referenced consortium members shall have no liability for damages of any kind including without limitation direct, special, indirect, or consequential damages that may result from the use of these materials subject to any liability which is mandatory due to applicable law. Copyright 2015 by University of Twente, Technical University of Denmark, Cybernetica, GMV Portugal, GMV Spain, Royal Holloway University of London, itrust consulting, Goethe University Frankfurt, IBM Research, Delft University of Technology, Hamburg University of Technology, University of Luxembourg, Aalborg University, Consult Hyperion, BizzDesign, Deloitte, Lust.*

Document History

Authors		
Partner	Name	Chapters
AAU	René Rydhof Hansen	2, 3
BD	Henk Jonkers	1, 2, 3, 4, 5, 6
DTU	Christian W. Probst	2, 3
GMVP	Sérgio Saraiva	3.1
ITR	Miguel Martins	2, 4, 5
LUST	Jeroen Barendse	3.4
UT	Dan Ionita	3, 4, 5
TUD	Wolter Pieters	3.5

Quality assurance		
Role	Name	Date
Editor	Henk Jonkers	2015-10-30
Reviewer	Margaret Ford	2015-06-30
Reviewer	Claude Heath	2015-06-30
Task leader	Miguel Martins	2015-10-30
WP leader	Miguel Martins	2015-10-30
Coordinator	Pieter Hartel	2015-10-30

Circulation	
Recipient	2015-10-30
Project Partners	2015-10-30
European Commission	2015-10-30

Acknowledgement: *The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement no. 318003 (TRESPASS). This publication reflects only the authors' views and the Union is not liable for any use that may be made of the information contained herein.*

Contents

List of Figures

List of Tables

Management Summary

This deliverable documents the final requirements for integrating the tools developed and used in the different technical work packages of TRESPASS, leading to a single integrated tool environment for data collection, modelling, analysis and visualisation.

Key takeaways:

- This deliverable summarises the state-of-the-art in tool integration options. Given the specific characteristics of the TRESPASS environment, **loose coupling** based on a central integration component (the TRESPASS core system) has been considered the most appropriate option.
- A project-wide tool **architecture diagram** shows the tools from the different technical work packages to be integrated, as well as the main information exchanges between these tools. This is the basis for the requirements for the tool architecture, the core system, and the tools developed in the other work packages.
- Where appropriate, the data that is to be exchanged between modules will be represented in a **standardised XML-based format**; the receiving party is responsible for any data conversion that may be required.
- A project-wide **requirements task-force** is in place to guarantee a systematic approach to the refinement of requirements. This deliverable reflects the achievements of the task-force that are relevant for Work Package 6 (both requirements aimed at WP6 and requirements originating from WP6 aimed at other work packages).

1. Introduction

Appendix A provides the context for this deliverable in the TRESPASS project. It describes the overall summary of the project and the TRESPASS workflow.

The aim of Work Package 6 is the design and realisation of a prototype of an integrated TRESPASS toolset, supporting the integrated TRESPASS process as described by Work Package 5, by combining pre-existing tools and tools developed in the other technical work packages of the project.

This deliverable documents the final requirements for tool integration, originating from Work Package 6 internally, from other work packages, from the project management team, and from the project as a whole. This document builds upon the initial requirements deliverable of Work Package 6 (The TRESPASS Project, D6.1.1, 2013). The most relevant content of the previous deliverable has been summarised here. The initial requirements have been refined, clarified and extended based on the experience gained through validation of earlier prototypes in the project's case studies, and as a result of the efforts of the project-wide *requirements task-force* (see Chapter 4).

1.1. Foreground and background

All requirements concerning TRESPASS tool integration are foreground of the project. The TRESPASS project does not develop requirements engineering, software engineering, or systems engineering methods, and any such methods used in the project are therefore considered background.

1.2. Document structure

The remainder of this document is structured as follows:

- Chapter 2 summarises the relevant state of the art in tool integration approaches, including the different options that exist for tool integration and their advantages and drawbacks.
- Chapter 3 sketches the tool landscape of the project, which is an important source of input for the requirements on tool integration.

- Chapter 4 specifies the consolidated requirements for tool integration, based on the effort of the project-wide requirements task-force, extended with some additional explanation.
- Chapter 5 lists the requirements aimed at other work packages originating from WP6.
- Finally, in Chapter 6, we draw some conclusions with a view to the continuing work of TRESPASS.

2. State-of-the art in tool integration methods

This chapter presents a brief survey of the state of the art in the integration of software tools, which serves as a baseline for the requirements in this work package. First, we describe the main types or aspects of tool integration in Section 2.1, and the role of these aspects in the context of TRESPASS. In Section 2.2, we classify the different ways in which tool integration can be achieved, and explain the advantages and drawbacks of these options. In Section 2.3 we identify a number of commonly used industry standards and products for software tool integration. Finally, Section 2.4, shows how the tool integration task has been tackled in a number of other research projects.

2.1. Aspects of tool integration

Several authors, e.g., (Thomas & Nejmeh, 1992; Wicks, 2004; Schefström & van den Boek, 1993), have identified a number of different aspects or levels for tool integration. It is important to make explicit choices for the types of integration we will consider.

The most basic type of integration is *data integration*, which plays a central role in the TRESPASS project. Tools for data collection and management, modelling, analysis and visualisation will all have to exchange data. Data integration can be attained in a loosely coupled way, e.g., by using shared data files or a shared data repository, or in a more tightly coupled way, by exchanging messages using well-defined programming interfaces.

A next step is *control integration*, in which tools may hand over control to each other; for example, once a model has been completed or modified, a modelling tool may automatically trigger an analysis tool, or an analysis tool may trigger a visualisation tool to present the analysis results. Where appropriate, the Attack Navigator interface developed in the project will provide some facilities for this.

Process integration is needed if we want to (partially) automate the TRESPASS process ("workflow"), as defined in Work Package 5 ("Process Integration"). This is also part of the functionality of the Attack Navigator interface.

Finally, Work Package 4 ("Visualisation") is expected to play an important role in *presentation integration*, making sure that the results of the modelling and analysis efforts are presented to the different stakeholders in a consistent and integrated way, again making use of the Attack Navigator interface.

In (Amsden, 2001), five levels of integration are distinguished, ranging from no integration at all to a fully integrated user interface (UI):

None	No integration, tools are separate and independent
Invocation	Integration is through invocation of registered applications on resource types
Data	Integration is achieved through data sharing
API	Tools interact with other tools through APIs that abstract tool behavior and make it publicly available
UI	Tools and their user interfaces are dynamically integrated at runtime including window panes, menus, toolbars, properties, preferences, etc.

2.2. Integration options

There are a number of different options to realise tool integration, each of which has their own specific advantages and disadvantages. Which of these options is the most appropriate depends on the specific situation. We distinguish four types of integration, characterised by two dimensions. First, pairs of tools may be connected in an ad-hoc (1-to-1) way, or they may be connected through a centralised integration component (van Leeuwen, ter Doest, & Lankhorst, 2004; Lankhorst, 2004). Second, tools may be loosely coupled (sharing stored data), or tightly coupled (directly interchanging messages). This leads to the following four options, illustrated in the figure below:

- Ad hoc (1-to-1) integration
  - Loose coupling: tools mutually share data files
  - Tight coupling: tools exchange information via established interfaces
- Central integration component
  - Loose coupling: all tools make use of a shared data repository
  - Tight coupling: tools exchange information via a mediation component (a tool bus)

For asynchronous information exchanges, i.e., if the information is used an unspecified amount of time after the time that it is produced, the loose coupling options are usually the most appropriate. For synchronous information exchanges, both options can be used. For types of integration stronger than data integration, in particular control integration, usually tight coupling would be required. In the TRE$_S$PASS tool environment, the information exchanges between the different components play a central role. Also, these exchanges will often be asynchronous: e.g., data will be incorporated in models some time after it is collected, and model analysis will take place after a model has been created. Therefore, loose coupling is the most appropriate option.

The advantage of using a central integration component is that it minimises the worst-case number of data exchange formats or adapters that could be needed (growing linearly with

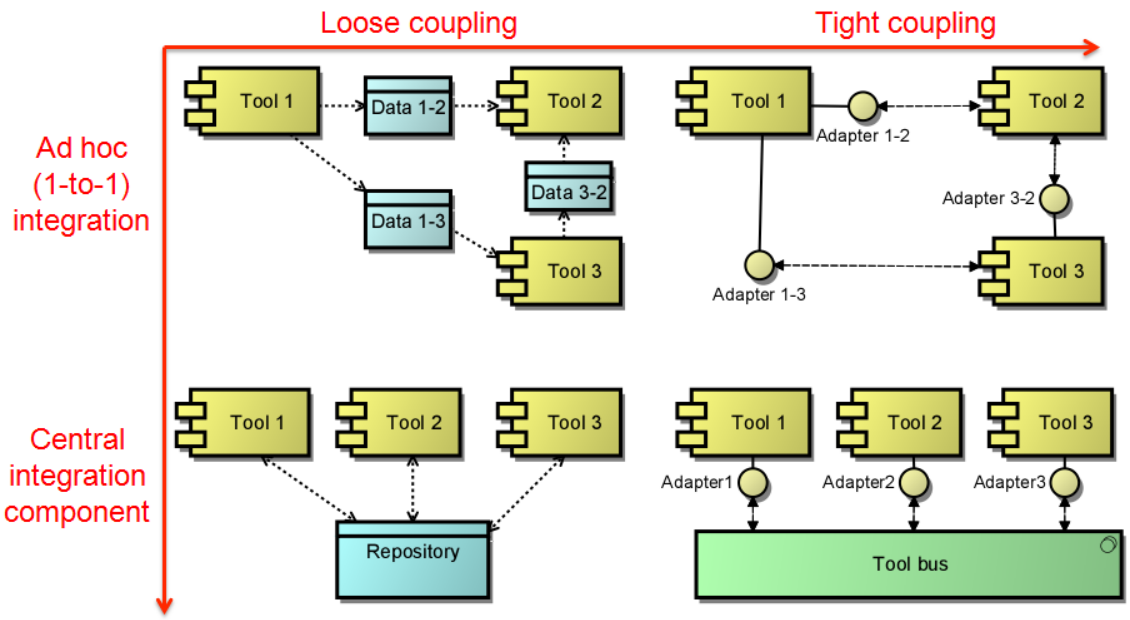

Figure 2.1.: Integration options

the number of tools to be integrated, in contrast to 1-to-1 integration, in which case the number of adapters required may grow quadratically with the number of tools). Another advantage, which is very relevant in the context of TRE$_S$PASS, is that security functions such as access control can be realised centrally. This is one of the reasons that the choice to include a centralised integration component (the "core system") in the tool architecture has been specified in the Description of Work. The most obvious disadvantage of a centralised solution is that it requires an additional component in the system. An additional drawback is that the central storage facility may become a performance bottleneck of the system, in cases where large amounts of data have to be exchanged. Therefore, it is important to consider the scalability of the central component.

2.3. Tool integration standards and products

In this section, we summarise a number of commonly used industry standards and products for software tool integration. Although not specifically aimed at modelling and analysis tools, they may provide inspiration for tool integration in TRE$_S$PASS.

Portable Common Tool Environment (PCTE)

The Portable Common Tool Environment (PCTE) is a public tool interface to a set of services for building integrated project support environments and for integration of tools within

such an environment (European Computer Manufacturers Association, 1997). The basis of PCTE is an object management system. PCTE is the result of a European Community research project initiated in 1984. A PCTE standard was adopted by the European Computer Manufacturers Association (ECMA) in 1990.

A Tools Integration Standard (ATIS)

A Tools Integration Standard (ATIS) is an object-oriented interface to a set of services that allows the saving, accessing and managing of information in a common repository. ATIS was developed by Atherton Technology and DEC, based on an extended version of the Software BackPlane, proposed as an ANSI industry standard.

Common Object Request Broker Architecture (CORBA)

The Common Object Request Broker Architecture (CORBA) is a standard defined by the Object Management Group (OMG) designed to facilitate the communication of systems that are deployed on diverse platforms (Object Management Group, 2015). CORBA is an open, vendor-independent architecture and infrastructure that computer applications use to work together over networks. CORBA enables collaboration between systems on different operating systems, programming languages, and computing hardware. CORBA is based on a central integration component, the Object Request Broker (ORB), although multiple distributed ORBs may interoperate. CORBA was introduced in 1991, while the most recent version, CORBA 3.3 (also known as CORBA/Ziob) was published in 2012.

Eclipse

Although primarily an (open source) integrated development environment, the Eclipse platform provides many facilities for integrating tools, as explained in (Amsden, 2001). For integration at the invocation level, it provides an operating system-independent registry mechanism for registering tools to be invoked on resource types. The platform also provides several different options for data sharing among tools. Eclipse's plugin mechanism can be employed to realise API-level integration of tools, while the Eclipse UI frameworks can be used to build integrated user interfaces. The Eclipse Modeling Framework (EMF) project adds specific functionality to the platform for building tools and other applications based on a structured data model, which makes it especially suitable for integrating different modeling tools.

2.4. Tool integration projects

The integration problem for modelling and analysis tools has been addressed in several other research projects, most of them in the late 1990's and early 2000's. In this section, we summarise a number of these approaches.

Web-based Open Tool Integration Framework (WOTIF)

In the Web-based Open Tool Integration (WOTIF) project (Vanderbilt University ISIS group, 2003), an open framework for integrating design tools for embedded system development was developed. The project was carried out at the Institute for Software Integrated Systems of Vanderbilt University. WOTIF provided a metamodel-driven infrastructure for design tool integration, facilitating semantic interoperability across the elements of a tool chain. The solution was based on the Eclipse integrated development environment.

The UniForM Workbench

The Universal Formal Methods (UniForM) Workbench was a project at the universities of Bremen and Oldenburg, together with industrial partners at ElPro LET GmbH, Berlin, funded by the German Ministry for Education and Research BMBF, running from 1995 until 1998 (Krieg-Brückner, Peleska, Olderog, & Baer, 1999; Karlsen, 1998). In the integration solution developed in this project, tools were integrated in a loosely coupled way. The workbench provided a subsystem interaction manager component that takes care of control integration. Data integration is realised by interfacing with a public domain implementation of the industry standard Portable Common Tool Environment (European Computer Manufacturers Association, 1997). For presentation integration, the workbench included a user interaction manager component.

ToolBus

The ToolBus is a software coordination architecture for the integration of components written in different languages running on different computers, which has been used in a variety of projects since 1994 (de Jong & Klint, 2003). A central principle supported by the ToolBus architecture is the separation of coordination, representation and computation.

ArchiMate tool integration workbench

One of the central premises of enterprise architecture is that it incorporates the specification of relations between different domains, each speaking their own languages and using their own tools. As a consequence, an enterprise architecture practice asks for the integration of existing modelling tools. Therefore, within the ArchiMate project (2002-2004),

research was conducted on the design and prototyping of a tool integration workbench (van Leeuwen et al., 2004), with a focus on model integration.

3. The TRESPASS tool landscape

In this chapter, we present an overview of the tool landscape in the project, which is the main source of requirements for tool integration. Figure 3.1 shows the project-wide integration diagram that has been established in a project-wide joint effort of all the work packages involved in tool development. It shows the tool functions (automated or interactive activities), the tools providing these functions, and the data flows between these functions.

The activities are grouped in four stages, that correspond to the four technical work packages in TRESPASS: data collection (WP2), model creation (WP1), analysis (WP3), and visualisation (WP4). Some tools are used in more than one stage and by more than one work package.

3.1. Stage 1 – Data collection tools (WP2)

Data collection is the process of gaining insights into the general model of an organisation as well as specific attributes of elements on the physical, virtual, and social level of the organisation. The different layers require different approaches to data collection, and may be manual or automated. Mechanisms to consolidate date from the different layers into a single model is also part of WP2 work.

Manual data collection may be supported by tangible modelling techniques, e.g., the use of LEGO®, and possibly other participatory research methods where scenarios can be collectively brainstormed and solutions co-designed.

For automated data collection, for example for the collection of IT infrastructure data, a large number of tools, both commercial and open source, are available, e.g., the open source vulnerability scanner OpenVAS. The raw data provided by these tools is usually available in some kind of binary format, an XML-based format, or as a comma-separated values (CSV) file.

WP2 is also responsible for a database structure for a central repository for the extracted data. The repository itself will be part of the central integration component to be developed in WP6, but WP2 will provide the required XML-based data format. The collected data is subsequently queried by the remaining stages for model creation, analysis, and visualisation. Another aspect is the ability to perform data transformation between formats (i.e. data stored on an entity/relationship database to data stored on XML-based files), without losing information during data transformation processes.

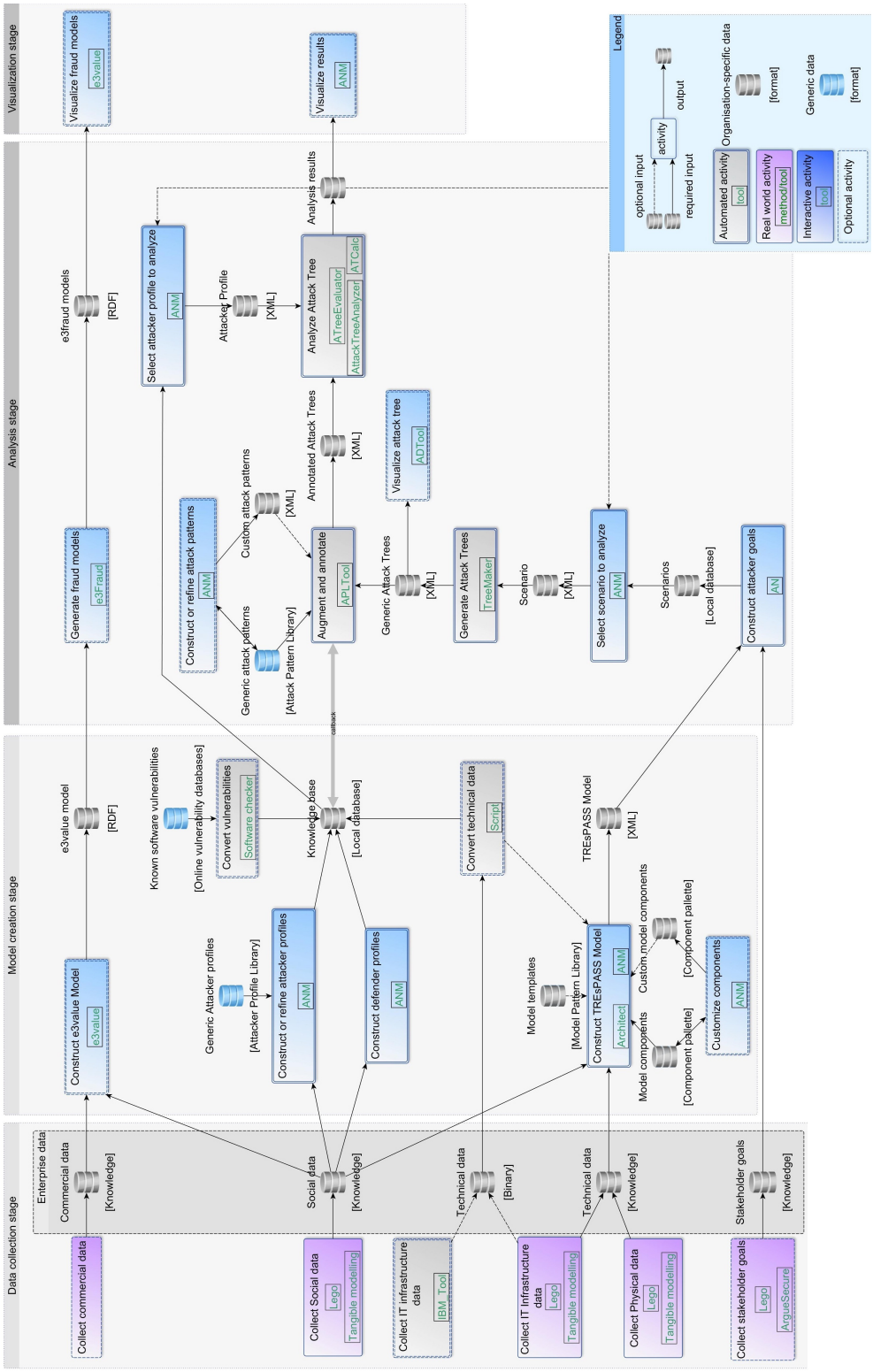

Figure 3.1.: Tool functions, tools and data flows.

Advanced WP2 data management processes might include other aspects of the data management process, like data inference (deducing data from the existing data) and data quality (includes assessing the value of the factors that comprise the data, like integrity check, data cleansing, correctness, completeness and data relevance).

3.2. Stage 2 – Model creation tools (WP1)

One model, or modelling formalism, is never able to capture every possible aspect that might be of interest in the modelled system. Therefore, a number of different, complementary modelling approaches will be used in the TRE$_S$PASS project, each potentially with its own specialised tool support. Choosing the best, or most appropriate, modelling technique for a given case depends on the specific characteristics of the case, as well as the properties and features of the case that are relevant for later analysis. Currently, the following three types of models have been used in the TRE$_S$PASS case studies:

- Enterprise architecture models,
- Socio-technical security models, and
- Value network and fraud models.

For enterprise architecture models, we use the enterprise architecture modelling standard of The Open Group, ArchiMate® (The Open Group, 2013). Several commercial tools support this modelling standard, some of them certified by The Open Group. A free open-source tool for ArchiMate modelling is Archi (`http://archi.cetis.ac.uk`). Most of the ArchiMate modelling tools use their own, often XML-based, storage format. Recently, The Open Group has proposed a draft of a common exchange format for ArchiMate models (). A hosted version of the commercial tool Architect from project partner BiZZdesign is freely available for project members during the course of the project.

In Work Package 1, the TRE$_S$PASS modelling formalism (The TRE$_S$PASS Project, D1.1.2, 2015; The TRE$_S$PASS Project, D1.3.1, 2013) for socio-technical security models has been defined, and a common XML-based storage and exchange format has been specified for this formalism. The creation of these socio-technical security models is supported by the Attack Navigator Map, part of the uniform user interface developed within the project. This tool is also used for the selection or creation of attacker and defender profiles, and for the creation and customisation of reusable model components. Also, a customised version of the hosted version of BiZZdesign Architect is available that supports the creation of socio-technical security models. This version includes a function to import and export ArchiMate models from and to a socio-technical security model. Tools for making socio-technical models should be capable of reading and writing the common XML-based storage format.

For value network and fraud models, the *e3value* formalism is used. A tool for creating *e3value* models, originating from the developers of the formalism, is freely available (Gordijn, Akkermans, Koks, & Schildwacht, 2004). Within TRE$_S$PASS, an extension of *e3value* to include modelling and analysis of fraud scenarios has been developed,

called *e3Fraud*, accompanied by a tool. The *e3value*-based tools use an RDF-based storage and exchange format. Currently, the *e3value* tools do not have links with the other TRESPASS tools.

3.3. Stage 3 – Analysis tools (WP3)

In the analysis stage, a number of primary activities or phases can be identified:

- Construction of an attack tree based on a socio-technical security model,
- Construction and refinement of attacker goals and attack patterns,
- Selection of attacker profiles and scenarios to be analysed, and
- Augmentation and annotation of attack trees,
- The actual analysis of attack trees.

Each phase is supported by one or several tools facilitating comprehensive and complementary analysis to be performed.

The first phase of (attack) analysis is to generate possible attacks from the underlying model. For socio-technical models, this task is performed by the TreeMaker tool through so-called *policy invalidation* (Ivanova, Probst, Hansen, & Kammüller, 2015b, 2015a). The analysis stage is continued in the second phase, where attacker goals and attack patterns are constructed, and a specific attacker profile is selected to determine attributes of the attacker, e.g., skill levels, risk appetite, and available resources. All of these tasks are performed using the Attack Navigator Map described in the previous section. The attack trees that are (automatically) generated from socio-technical models describe generic attacks at a high-level and does not contain (all of) the low-level steps required in a particular attack nor possible variations of an attack step. Elaborating such abstract attack trees, requires model-specific information and expansion/augmentation of high-level attack steps into concrete low-level attack steps. This annotation and expansion of attack trees is done through the use of the APLTool ().

For the actual analysis of the attack trees resulting from this process, several special tools have been developed in WP3. These include ATreeEvaluator (Aslanyan & Nielson, 2015), AttackTreeAnalyzer () and ATCalc (). All of these use an XML-based formats for inputs and outputs. In addition, also general purpose analysis tools are used, such as the UPPAAL and UPPAAL CORA model checkers (?, ?, ?).

For the analysis of fraud scenarios based on *e3value* models, the *e3Fraud* tool as described in the previous section can be used.

3.4. Stage 4 – Visualisation tools (WP4)

WP4 provides ways to visualise the inputs and results from modelling and analysis tools in an appropriate way. The Attack Navigator Map (ANM) tool, as described above, is the place where most of these visualisations are created. Model content is already available in this tool, and maps of organisations can be build by this tool or can be imported from an XML file, which can be analysed. Analysis results will have to be imported into the tool, and possibly matched to model elements. Once the analysis finishes, its results are visualised, with the possibility for the user to rank and filter the attacks. In a split-screen dashboard, there will be multiple views which visualises various aspects, besides the list of attacks. Overview will include showing the attack traces in the context of the intermediate attack tree, or highlight the relevant parts of the model/ANM that play a role in the attack(s). The ANM is able to deal with the visualisation of data input, the visualisation of navigator maps, the visualisation of attacker profiles and the visualisation of attack tree scenarios.

BiZZdesign Architect, as described above, also provides a number of ways to visualise ArchiMate models and socio-technical security models, and to visualise certain properties of these models (e.g., using colour views). Specifically for the visualisation of attack trees, ADTool (Kordy, Kordy, Mauw, & Schweitzer, 2013) can be used. *e3value* models can be visualised using the original *e3value* editor.

3.5. Sharing libraries (WP5)

Although WP5 is primarily focused on the processes rather than the tools, some specific tool innovations were proposed in WP5, in particular in relation to information sharing (The TRESPASS Project, D5.3.2, 2015). Specifically, the following libraries were proposed:

- Model Pattern Library. The Model Pattern Library stores reusable models (navigator maps), parts of models, or model templates. These can be used as a basis for developing new models.
- Attacker Profile Library. The Attacker Profile Library stores reusable attacker types along with their properties, such as skill and budget. The attacker profiles are used in combination with the navigator maps to provide attacker-specific security analyses.
- Attack Pattern Library. The attack pattern library stores subtrees of attack trees. These subtrees are used to expand attack trees generated in the analysis, providing more detailed attack steps as well as enabling calculation of parameters.

These shared libraries are used at key points in the tool chain, and they are therefore connected to the requirements of the tools from which they are referenced.

4. Requirements for tool integration

In order to address the definition of requirements in a systematic way, a project-wide task-force was put in place. In that context, all technical work packages were requested to identify their most relevant requirements. For a full overview of the requirements identified by the task-force, we refer to (The TRESPASS Project, D6.2.2, 2015). In this chapter, we list the requirements that are targeted at WP6, and discuss their implications.

Requirement R04

Requirement : Integration based on loose coupling and on a central integration component

Source WP: WP6

Target WP: WP6

Goals: Nature of data exchanges

Acceptance criteria: General integration approach, with flexibility to adapt to the tools being integrated

Status: Agreed

Dependencies: None

Comments: *In Section 2.2, the different integration options with their advantages and drawbacks have been explained. Loose coupling is best suited for TRESPASS because both synchronous and asynchronous communications have to be supported (see Requirements R05.1 and R05.2). Tight coupling is mainly suited for synchronous exchanges, not for asynchronous exchanges. A central integration component reduces the number of interfaces to be realized, and makes it easier to secure the information exchanged between the tools.*

Requirement R05.1

Requirement : Support for asynchronous data exchanges

Source WP: WP6

Target WP: WP6

Goals: Nature of tools from other WPs

Acceptance criteria: Tool integrated if requested

Status: Agreed

Dependencies: Partly conflicts R04

Comments: *Certain parts of the TRE$_S$PASS process require asynchronous communication between tools. E.g., data collection may take place prior to model creation, which means that the collected data needs to be stored first, or analysis may take place at a later stage than model creation.*

Requirement R05.2

Requirement : Support for synchronous data exchanges

Source WP: WP6

Target WP: WP6

Goals: At some point, there may be tools colecting and processing real time data. It will be implemented if needed

Acceptance criteria: Tool integrated if requested

Status: Agreed

Dependencies: None

Comments: *Certain parts of the TRE$_S$PASS process require synchronous communication between tools. E.g., in case of real-time visualisations of analysis results.*

Requirement R06

Requirement : “Apps style” user interface

Source WP: WP6

Target WP: WP6

Goals: Modern user-friendly style

Acceptance criteria: ?Apps? should work stand alone as well as in a tool chain (whenever appllicable)

Status: Agreed

Dependencies: None

Comments: *Users of the TRE$_S$PASS toolset are not necessarily experts in the use of modelling and analysis tools. An intuitive, user-friendly user interface, consistent with other contemporary applications, makes the tools more accessible to these user groups.*

Requirement R10

Requirement : Interface between analysis tools and visualisation tools

Source WP: WP6

Target WP: WP4,6

Goals: Needed to generate visualisations

Acceptance criteria: The output of the analysis tools is available for visualisations

Status: Agreed

Dependencies: None

Requirement R12

Requirement : Develop a visual language

Source WP: WP4

Target WP: WP4,6

Goals: Needed for development of interface

Acceptance criteria: Documented language accepted by reviewers in Deliverable 4.1.1 (achieved) Accepted interpretation by practitioner panel and data analysis designers (WP) in Attack Navigator Map (iterative and on-going)

Status: Agreed

Dependencies: None

Requirement R13

Requirement : Define a visualisation process

Source WP: WP4

Target WP: WP4,5,6

Goals: Needed for development of interface

Acceptance criteria: Documented process accepted by reviewers in Deliverable 4.1.1 (achieved) Accepted interpretation by practitioner panel and data analysis designers (WP) in Attack Navigator Map (iterative and on-going)

Status: Agreed

Dependencies: None

Requirement R18

Requirement : Programmatic interface in other components in the TREsPASS tool

Source WP: WP1

Target WP: WP6

Goals: Needed to provide access to model

Acceptance criteria: The model is successfully integrated in a workflow ANM-model-treemaker-analyses-visualisations.

Status: Agreed

Dependencies: None

Requirement R40

Requirement : Support for attack tree visualisation

Source WP: WP3

Target WP: WP4,6

Goals: Improved presentation of tool output

Acceptance criteria: Visual inspection

Status: Agreed

Dependencies: None

Requirement R52

Requirement : Users can drag and drop standard elements onto the map.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment) and Use-case 2 (Audit), specifically U1.5, U2.4, U2.5, U4.1, U4.2, U5.6 and U5.7

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training. Standard elements must include business processes.

Status: Agreed

Dependencies: None

Requirement R53

Requirement : Users can add connections between elements on the map.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U2.4 and U5.6.

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: Potentially conflicts R07, R38

Requirement R54

Requirement : Users can change parameters of map elements by clicking and entering new values

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U1.12, U2.4, U5.6.

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: Potentially conflicts R07, R38

Requirement R55

Requirement : Users can change parameters of map elements by clicking and requesting information from available data extraction tools.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U2.4 and U5.6

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to respond effectively to the suggested updates.

Status: Agreed

Dependencies: None

Requirement R56

Requirement : The system can suggest updates to the TRESPASS model based on scenarios and associated parameters.

Source WP: MT

Target WP: WP3,5,6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U1.12, U1.13, U2.4 U5.6.

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to respond effectively to the suggested updates

Status: Shelved

Dependencies: Reason for shelving: adaptations to the sociotechnical model will have to be done manually, not automagically.

Requirement R57

Requirement : The user can select attacker profiles from the library.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), specifically U1.9.

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: None

Requirement R58

Requirement : The user can build attacker profiles by editing their properties

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), specifically U1.6, U1.8 and U1.10

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: None

Requirement R59

Requirement : The user can assign asset values and properties to elements on the map.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U1.6, U2.4 and U5.6.

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training

Status: Agreed

Dependencies: None

Requirement R60

Requirement : The user should be able to replicate an entity in the model by clicking on it, selecting the replicate option, and indicate the number of replications and associated parameters.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U2.1, U2.4 and U5.6.

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training

Status: Agreed

Dependencies: None

Requirement R64

Requirement : The system should be able to handle push messages from external tools in order to generate push messages (updates) to users/models

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), specifically U1.1 and U1.13.

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Shelved

Dependencies: None

Requirement R68

Requirement : The TRESPASS tools enable the user to store a TRESPASS model, including navigator map, attacker profiles, and history of analyses

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 1 (Security Investment), Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U1.18, U2.15, U4.3 and U5.17.

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: R7

Requirement R69

Requirement : The TRESPASS tools enable the user to load a previously stored TRESPASS model, including navigator map, attacker profiles, and history of analyses.

Source WP: MT

Target WP: WP6

Goals: Derived from P1 (TRESPASS tools should assist the user in building a navigator map). Supports Use-case 4 (Product-service system), specifically U4.3

Acceptance criteria: 80 percent of the users specified in the Use Cases are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: R7

Requirement R80

Requirement : The TRESPASS tools can start analyses automatically after updates of the corresponding models.

Source WP: MT

Target WP: WP6

Goals: Supports Use-case 1 (Security Investment), specifically U1.1

Acceptance criteria: Results of the chosen analyses are available after the corresponding events.

Status: Shelved

Dependencies: None

Requirement R81

Requirement : Visualisation of social and technical data, maps, scenarios, countermeasures.

Source WP: MT

Target WP: WP4,6

Goals: Derived from P7 (TRESPASS tools should be able to visualise maps, scenarios, and countermeasures). Supports Use-case 1 (Security Investment), Use-case 2 (Audit), Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U1.2, U2.6, U2.13, U4.6, U5.9 and U5.15.

Acceptance criteria: The visualisations are sufficiently expressive.

Status: Agreed

Dependencies: R03, R06

Requirement R82

Requirement : TRESPASS tools should be accessible through a web interface

Source WP: MT

Target WP: WP6

Goals:

Acceptance criteria: The web interface functions correctly on the most common platforms (Android, iOS).

Status: Agreed

Dependencies: R02, R06

Requirement R83

Requirement : The TRESPASS integrated component should use loosely coupled tools that are also available individually

Source WP: MT

Target WP: WP6

Goals:

Acceptance criteria: 80 percent of the users specified in the Use Cases are satisfied upon use of each individual tool separately.

Status: Agreed

Dependencies: R01

Requirement R86

Requirement : Users can select base models from a Model Template Library.

Source WP: MT

Target WP: WP4,5,6

Goals: Supports Use-case 4 (Product-service system) and Use-case 5 (Quick scan), specifically U4.2, U5.4

Acceptance criteria: 80 percent specified in the case studies are able to use the interface effectively after receiving the specified training.

Status: Agreed

Dependencies: Assumes the Model Template Library exists

Requirement R87

Requirement : Users can (re-) run analyses manually or at scheduled times/intervals

Source WP: MT

Target WP: WP6

Goals: Supports Use-case 2 (Audit) and Use-case 5 (Quick scan), specifically U2.14, U5.16

Acceptance criteria: 80 percent specified in the case studies are able to use the interface effectively after receiving the specified training.

Status: Shelved

Dependencies: None

5. Requirements to other work packages

In this chapter, we list the requirements, established by the requirements task-force, that originate from WP6 and are aimed at one or more of the other work packages. If necessary, we clarify the requirements with some additional explanation.

Requirement R01

Requirement : Common XML shared model

Source WP: WP6

Target WP: WP2

Goals: Simplify data exchanges between tools

Acceptance criteria: Model available for current toolset

Status: Agreed

Dependencies: None

Comments: *It has been agreed by the technical work packages in the project that, as much as possible, tools will exchange data using a standard XML-based format (containing information such as the content of socio-technical security models, attacker goals, attacker and defender profiles, etc.) WP2 should define the structure of these XML files.*

Requirement R02

Requirement : Central database structure

Source WP: WP6

Target WP: WP2

Goals: Store data needed by several tools

Acceptance criteria: Structure available for current toolset

Status: Agreed

Dependencies: None

Requirement R03

Requirement : Availability of an attack pattern library

Source WP: WP6

Target WP: WP5

Goals: Reuse of sub-attack-trees

Acceptance criteria: One example per case study

Status: Agreed

Dependencies: None

Requirement R04

Requirement : Integration based on loose coupling and on a central integration component

Source WP: WP6

Target WP: WP6

Goals: Nature of data exchanges

Acceptance criteria: General integration approach, with flexibility to adapt to the tools being integrated

Status: Agreed

Dependencies: None

Requirement R05.1

Requirement : Support for asynchronous data exchanges

Source WP: WP6

Target WP: WP6

Goals: Nature of tools from other WPs

Acceptance criteria: Tool integrated if requested

Status: Agreed

Dependencies: Partly conflicts R04

Requirement R05.2

Requirement : Support for synchronous data exchanges

Source WP: WP6

Target WP: WP6

Goals: At some point, there may be tools colecting and processing real time data. It will be implemented if needed

Acceptance criteria: Tool integrated if requested

Status: Agreed

Dependencies: None

Requirement R06

Requirement : “Apps style” user interface

Source WP: WP6

Target WP: WP6

Goals: Modern user-friendly style

Acceptance criteria: ?Apps? should work stand alone as well as in a tool chain (whenever appllicable)

Status: Agreed

Dependencies: None

Requirement R10

Requirement : Interface between analysis tools and visualisation tools

Source WP: WP6

Target WP: WP4,6

Goals: Needed to generate visualisations

Acceptance criteria: The output of the analysis tools is available for visualisations

Status: Agreed

Dependencies: None

Requirement R19

Requirement : Socio-technical security models must provide the necessary abstraction to represent infrastructure and other components.

Source WP: WP2,6

Target WP: WP1

Goals: Needed to use the model

Acceptance criteria: XML Format defined

Status: Agreed

Dependencies: None

6. Conclusions

Work package 6 is concerned with the integration of the data collection, modelling, analysis and visualisation tools developed and used in the TRESPASS project. Building on an overview of the state of the art and best practices as described in this document, we have identified requirements that we believe to be also beneficial for similar projects. Given the specific characteristics of the TRESPASS environment, loose coupling based on a central integration component (the TRESPASS core system) has been considered the most appropriate option.

This deliverable describes the final set of requirements for integrating the tools used in the different stages of the TRESPASS project, leading to a single integrated platform for data collection, modelling, analysis and visualisation. A project-wide tool architecture diagram has been developed as a common foundation for this, showing the tools from the different work packages to be integrated and the main information flows between these tools. Also requirements for tools developed in other work packages have been identified, in order to enable or facilitate their integration in the overall platform. These requirements have been described in a uniform format as agreed in a project-wide requirements task-force.

References

Amsden, J. (2001). *evels of integration: Five ways you can integrate with the eclipse platform.* Retrieved from https://www.eclipse.org/articles/Article-Levels-Of-Integration/levels-of-integration.html

Aslanyan, Z., & Nielson, F. (2015). Pareto efficient solutions of attack-defence trees. In R. Focardi & A. C. Myers (Eds.), *Proceedings of the 4th international conference on principles of security and trust, POST 2015* (Vol. 9036, pp. 95–114). Springer.

de Jong, H., & Klint, P. (2003). ToolBus: The next generation. In F. de Boer, M. Bonsangue, S. Graf, & W.-P. de Roever (Eds.), *Formal methods for components and objects* (pp. 220–241).

European Computer Manufacturers Association. (1997). *Portable common tool environment (PCTE) - abstract specification* (4th ed.; Tech. Rep. No. ECMA-149).

Gordijn, J., Akkermans, H., Koks, A., & Schildwacht, J. (2004). *User manual* e3value *editor* (Tech. Rep.). Vrije Universiteit Amsterdam. Retrieved from http://e3value.few.vu.nl/docs/misc/manual_version2.pdf

Ivanova, M. G., Probst, C. W., Hansen, R. R., & Kammüller, F. (2015a). Attack tree generation by policy invalidation. In R. N. Akram & S. Jajodia (Eds.), *Proceedings of the 9th IFIP WG 11.2 international conference on information security theory and practice, WISTP 2015* (Vol. 9311, pp. 249–259). Springer.

Ivanova, M. G., Probst, C. W., Hansen, R. R., & Kammüller, F. (2015b). Transforming graphical system models to graphical attack models. In S. Mauw & B. Kordy (Eds.), *Proceedings of the 2nd international workshop on graphical models for security.* Springer.

Karlsen, E. (1998). The UniForM Workbench - a higher order tool integration framework. In *Proceedings international workshop on current trends in applied formal methods* (pp. 266–280). Springer.

Kordy, B., Kordy, P., Mauw, S., & Schweitzer, P. (2013). ADTool: Security analysis with attack-defense trees. In *Proceedings of the 10th international conference on quantitative evaluation of systems (QEST'13)* (pp. 173–176).

Krieg-Brückner, B., Peleska, J., Olderog, E.-R., & Baer, A. (1999). The UniForM workbench, a universal development environment for formal methods. In J. Wing, J. Woodcock, & J. Davies (Eds.), *Proceedings of FM'99, formal methods* (pp. 1186–1205). Springer.

Lankhorst, M. M. (2004, October). Enterprise architecture modelling-the issue of integration. *Adv. Eng. Inform.*, *18*(4), 205–216. Retrieved from http://dx.doi.org/10.1016/j.aei.2005.01.005 doi: 10.1016/j.aei.2005.01.005

Object Management Group. (2015). *Corba website.* Retrieved from http://www.corba.org

Schefström, D., & van den Boek, G. (Eds.). (1993). *Tool integration: environments and frameworks*. New York, NY, USA: John Wiley & Sons, Inc.

The Open Group. (2013). *ArchiMate® 2.1 specification*. Van Haren Publishing.

The TRESPASS Project, D1.1.2. (2015). *Final specifications and requirements for socio-technical security models.* (Deliverable D1.1.2)

The TRESPASS Project, D1.3.1. (2013). *Initial prototype of the socio-technical security model.* (Deliverable D1.3.1)

The TRESPASS Project, D5.3.2. (2015). *Best practices for model creation and sharing.* (Deliverable D5.3.2)

The TRESPASS Project, D6.1.1. (2013). *Initial requirements for tool integration.* (Deliverable D6.1.1)

The TRESPASS Project, D6.2.2. (2015). *Final refinement of functional requirements.* (Deliverable D6.2.2)

Thomas, I., & Nejmeh, B. (1992). Definitions of tool integration for environments. *IEEE Software*, 29–35.

Vanderbilt University ISIS group. (2003). *Web-based open tool integration framework.* Retrieved from http://w3.isis.vanderbilt.edu/Projects/WOTIF/default.html

van Leeuwen, D., ter Doest, H., & Lankhorst, M. (2004). Tool integration workbench for enterprise architecture. In *Proceedings of the 6th international conference on enterprise information systems.* Porto, Portugal.

Wicks, M. (2004). Tool integration in software engineering: The state of the art in 2004. *Integration, the VLSI Journal*, 1–26.

A. Project Summary

This chapter gives an overview of the TRE$_S$PASS project and its use cases. The section is shared by the public deliverables to provide the necessary background and to put the current deliverable in context.

Information security threats to organisations have changed completely over the last decade, due to the complexity and dynamic nature of infrastructures and attacks. Attacks like StuxNet involve technical and human factors, and they damage physical infrastructure. The recent attack on a German steel mill [1] was a combination of both targeted phishing emails and social engineering attacks. The phishing helped the hackers extract information they used to gain access to the plant's office network and then its production systems. As a result, the technical infrastructure of the mill suffered severe damage.

The attack on the German steel mill illustrates that we need to integrate the social and technical aspects of systems in assessing their security - and we need to do so today. Socio-technical systems pose new challenges by combining parts for which we often understand the security issues; the combined system is however much more complex due to interactions between these parts.

The main innovation of the TRE$_S$PASS project is the attack navigator, a tool and metaphor that enables defenders to predict and preventing attacks on socio-technical systems. The attack navigator supports current risk-assessment techniques with the TRE$_S$PASS process (developed in Work Package WP5), an analytical approach to identifying attacks and evaluating their impact.

The four main stages in the TRE$_S$PASS process are *data collection*, *modelling*, *analysis*, and *visualisation*. Data collection (WP2) is vital to understanding the nature of a scenario and providing input to subsequent tasks of modelling, analysis and visualisation. Within the project, the focus has been on collection and analysis of social, technical and physical data and the ways in which these relate to one another. Within each of these domains, different approaches have been taken to provide different viewpoints on the nature of the organisation being investigated.

The models (WP1) developed in TRE$_S$PASS can be adapted to the application scenario. We have developed physical modelling techniques in order to understand where further investigation may usefully be targeted. The TRE$_S$PASS model describes relevant aspects of the organisation and their connections. To explore contractual and commercial relationships, the e3value method has been adopted.

[1] BBC News, *Hack attack causes 'massive damage" at steel works*, http://www.bbc.com/news/technology-30575104, last visited October 31, 2015.

The analysis methods (WP3) developed in TRE$_S$PASS identify attacks in models and identify the most effective controls to prohibit these attacks. The analyses are supported by tools and together they provide the defender with a comprehensive understanding of properties attacks, *e.g.*, cost for the attacker, required skills, or required time.

The innovative visualisations (WP4) developed in TRE$_S$PASS focus focus particularly on visualising elements of the analysis, as this is key to the overall project goal of providing "decision support" to practitioners. However, visualisations contribute also to model development and data gathering.

Practitioners can access the TREsPASS toolkit via the attack navigator map interface, which provides an intuitive means of selecting appropriate tools (WP6) for data gathering, modelling, analysis and visualisation. These can be used, individually or in combination, to strengthen operational and strategic decision-making.

A.1. Case Studies

The TREsPASS process and tools are validated by means of case studies (WP7) in the area of cloud infrastructure, telecommunications infrastructure, ATM infrastructure, and an organisation processing privacy sensitive data.

A cloud infrastructure shares infrastructure within or across organisations, giving the cloud services provider and its employees full physical and logical access to all resources across the different consumers. In TRE$_S$PASS we formalise typical components in cloud infrastructures as well as human actors and their interrelationships, to identify their contribution to attacks on the organisation.

In telco infrastructure new products need to be launched under significant time pressure, often opening up loopholes for so-called knowledge insiders who know the market very well, trying to make as much monetary gain from the new products as possible. In TRE$_S$PASS we model both the infrastructure and contractual relationships to identify physical and monetary attacks.

The ATM infrastructure connects machines that are composed of a money safe and a computer that controls the ATM's devices. There are well protected ATMs installed inside bank branches, while others are deployed in the street and some are not even embedded in a wall. ATM attacks are common and include classic physical attacks and emerging digital attacks. In TRE$_S$PASS we model ATM installations, and identify attack likelihoods using geospatial data.

The organisation processing privacy sensitive data develops a system supporting primarily elderly and disabled people in performing online payments and managing their own money from their home. This case study involves from strictly technical security aspects, such as how information is protected while stored or transmitted, to socio-technical security aspects covering security issues arising from the use of and interaction with the technology. In TRE$_S$PASS we identify social-engineering and trust-based attacks on such systems.

A.2. Overview of TRESPASS Integration

The TRESPASS workflow involves several stages with various activities, some of which are optional. Figure A.2 shows the architecture diagram and Figure A.1 shows a visual description of the notation used. In practice, stages may not follow a linear order. For example, depending on the goal of the risk assessment, new data requests may be issued later in the process, or automatic updates of data may be supported.

The **Data collection stage** prepares for analysis and modelling steps, and may require the gathering of one or more of the following kinds of data.

Physical data collection provides knowledge about the physical layout of the organization including locations, buildings, rooms, doors, windows, etc.

Digital data collection gathers information about the organization's IT infrastructure.

Social data collection focuses on organisational and individual data, and results in actor profiles containing, *e.g.*, attributes of employees, stakeholders, or potential attackers.

Commercial data collection gathers information required for *e3fraud* analyses, which focus on potential fraud.

Stakeholder goal collection identifies assets and policies the protection of which is critical to one or more stakeholders.

The **model creation stage** handles the creation of the TRESPASS model and associated actor profiles. The e3value model creation process is complementary to the main TRESPASS model, for cases requiring a more specific financial focus:

TRESPASS model creation is a key activity result in a system model that can be further extended and analysed.

- **Components customization (optional)** takes place before or during the TRESPASS model creation to create specialized custom model components.
- **Attacker profile creation** creates the attacker profile that the TRESPASS model analysis should consider, based on ready-made attacker profiles.
- **Defender/target profile creation** creates similar profiles for the other actors in the model based on the social data gathered in the social data collection activity.

e3value model creation This interactive activity involves using the *e3value toolkit* [2] to create business value models. These models structure the commercial information gathered in the data collection stage in a formal way.

In the **analysis stage** different analyses are possible depending on the model chosen. The analysis of the TRESPASS model involves these steps:

1. In the **attacker profile selection**, the user selects the attacker profile to use in the analysis.

[2] http://e3value.few.vu.nl/tools/

2. The **attacker goals creation** provides the attack generation with the attacker goals. These can be derived by hand from the stakeholder goals or deduced automatically from the selected attacker profiles.
3. The **scenario selection** selects a scenario, consisting of a single pair of attacker and attacker goal, to run the TRESPASS analysis on.
4. To extend attack trees, **attack pattern creation and sharing** provides libraries with known attack steps. The attack tree generation can only reach a certain level of abstraction, which may not be sufficient for quantitative analyses.
5. **Attack generation** transforms the TRESPASS model to an attack tree.
6. **Attack tree annotation & augmentation** then extends the attack tree with attack patterns and decorates leaf nodes with parameter values from the data collection stage for quantitative analysis.
7. The **attack tree analyses** compute quantitative properties of attacks, *e.g.*, utility for the attacker or success probability of the attack.

The analysis of the **e3value model** is complementary to the core TRESPASS analysis and has only one step:

1. For the **fraud model generation**, the user needs select an attacker and an interval of expected occurrence rates of the commercial transactions specified by the e3value model. The e3fraud tool then identifies all possible violations of contracts, the loss for actors, and the delta in profit for the other actors.

The **visualisation stage** can be used continuously to provide practitioners with feedback regarding the results of their activities:

1. **Fraud model visualisation** shows the generated attacks as a ranked list of textual descriptions of the attack steps and displays charts showing the profitability for each actor.
2. **Attack tree visualisation** shows the intermediary attack trees.
3. **Attack tree analysis visualisation** visualises analysis results.

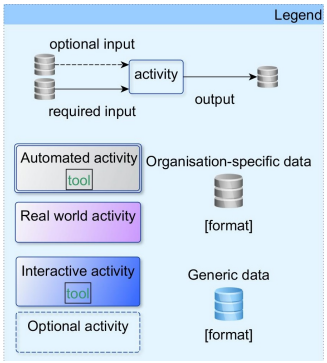

Figure A.1.: Legend for the Integration diagram in Figure A.2.

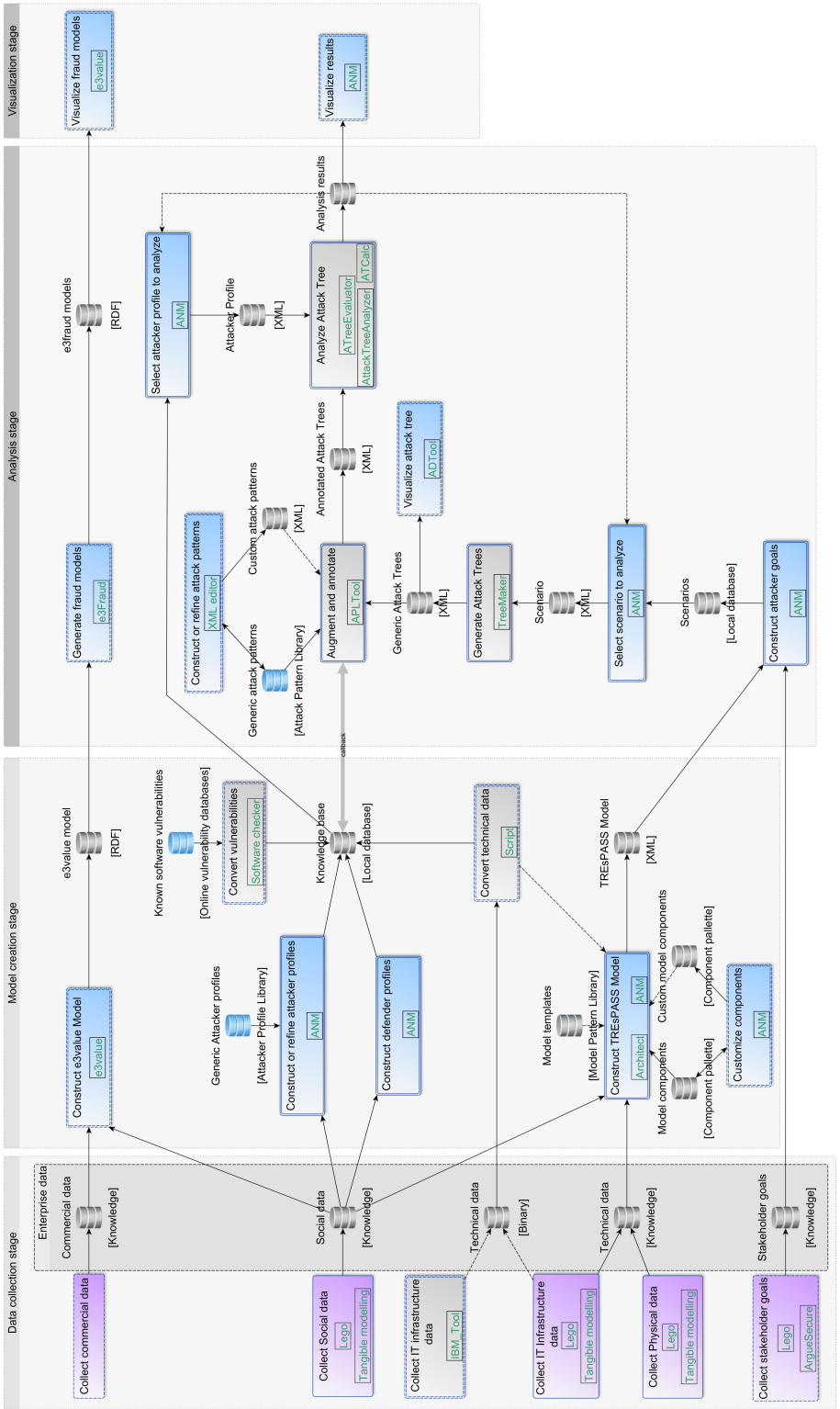

Figure A.2.: Integration diagram for the TRESPASS project.