

**MOSTRO Del.8, WP 4**  
**Methodology Evaluation**

University of Trento

November 13, 2007



# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
<b>2</b>	<b>Applying Tropos/Secure Tropos to e-voting scenario</b>	<b>9</b>
2.1	Methodological issues . . . . .	9
2.2	Modelling the e-voting process . . . . .	11
<b>3</b>	<b>Transfer data scenario: contribution analysis</b>	<b>15</b>
<b>4</b>	<b>Transfer data scenario: analyzing security properties</b>	<b>19</b>
4.1	Secure Tropos modelling . . . . .	20
4.2	First analysis results . . . . .	23
4.2.1	The input file . . . . .	23
4.2.2	The output file . . . . .	25
4.3	Improving the model . . . . .	26
<b>5</b>	<b>Conclusion</b>	<b>31</b>
	<b>Bibliography</b>	<b>33</b>



# Chapter 1

## Introduction

The aim of this deliverable is to evaluate the Secure Tropos methodology, proposed within the MOSTRO project [3] as a suitable candidate for modelling security concerns in an organizational setting. Such evaluation has been conducted empirically, by showing its application to a specific case study, used as a testbed, namely the ProVotE initiative [1].

The characteristics and activities of the ProVotE project have already been widely presented in Deliverable 7 [4], here it suffices to remind that it is a project whose purpose is that of allowing a smooth transition from traditional paper-based elections to e-voting elections in the Province of Trento.

Within the various activities of the ProVotE project, the one which is of interest from the MOSTRO project's perspective is the one related to modelling. This activity was performed in order to mitigate the risk of creeping security "holes" in the electronic procedures. The model of the existing procedures ("as is") provides a baseline for the definition of the new procedures, which describe the electoral process after the introduction of electronic procedures. Basic requirements for the system "to be" are to ensure the same security level of paper elections, to deal with new threats introduced by electronic systems, and to introduce as few changes as possible in the way of voting. As it is easy to see, these requirements concern both technical aspects (e.g. reliability of the system) and sociological aspects (e.g. citizens' acceptance).

This modelling activity was initially conducted using an UML-based approach, that was particularly helpful in explaining the dynamic of the whole process in all its phases with all the procedures. The UML, however, is weak in providing means of describing alternatives and, therefore, the methodology devised in [7] falls short in providing ways to describe the *why* of the transition from the "as is" to a specific "to be". Hence there is a need to complement UML modelling with some other

approach more suited to face these issues.

The synergy between ProVotE and MOSTRO arises from the idea that it is possible to fill the gap that is left after the application of the UML approach with the modelling and analysis techniques provided by Tropos [5], an agent-oriented software development methodology, and Secure Tropos [3, 9], an extension of Tropos that deals with trust and security analysis. Tropos is requirements driven, i.e. it is based on concepts used during early phases of the requirements analysis process. The main entities that populate models in Tropos come from the i\* modelling approach [8], and are actors, goals and dependencies (among actors and among goals); Tropos models both organizations and information systems as networks of interdependent actors endowed with goals. Finally, it covers also early requirements, i.e. it focuses on the environment in which the software has to be introduced and explains why it is introduced. It is thus in many senses complementary with respect to UML. In turn, Secure Tropos extends Tropos notation with the primitives for modelling trust, permission, ownership, etc. It also provides mechanisms of defining, verifying and enforcing security properties of a model, as well as the tool support to automate these activities (see Deliverable 6 [3] for the details).

In the approach firstly introduced in Deliverable 7 [4] we presented the integration of UML and Tropos/Secure Tropos perspectives in the following way. The UML models provide an exact snapshot of the procedures (independently from the motivations for which they have been devised in a specific way), while, at the same time, Tropos/Secure Tropos help to maintain track of the reasons for any change we had to introduce to support electronic elections. From a technical standpoint, this translates into an approach which produces an alternating sequence of UML and Tropos models (Figure 1.1). In particular, UML is used to model both “as is” and “to be” processes, while Tropos/Secure Tropos are used in between to reason about design alternatives with the purpose of providing a rationale for the solutions adopted for the implementation of the system “to be”. The latter is done by (a) modelling possible alternative ways of accomplishing a goal and their impact on non-functional system requirements, and (b) exploring trust and security issues related to the e-voting process. The results of the analysis allow, in turn, modifying the existing UML models and devise the new procedures that meet the requirements stated in the Tropos model. The steps described above are then iterated as needed.

In this deliverable, we present some of the results of the application of the above approach to the e-voting scenario. In particular, in chapter 2 we explain how and why Tropos/Secure Tropos is used as a tool for reasoning about security as well as the other non-functional issues involved in the choice of a particular solution against alternatives in the passage to the electronic procedure. In the second part of the same chapter we present the high level Tropos models for the e-voting

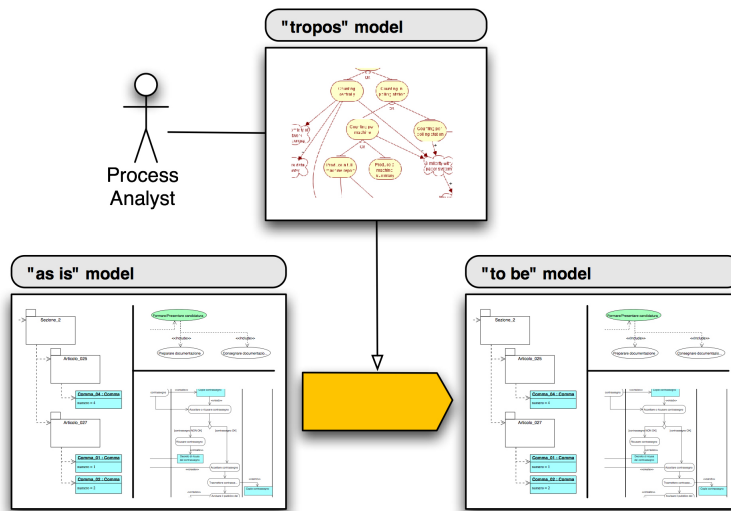


Figure 1.1: Combining UML and Tropos modelling to devise the “to be” model.

scenario. In chapters 3 and 4 we illustrate, respectively, the two ways in which we apply Tropos/Secure Tropos to the transfer data process of the e-voting scenario. In particular, in chapter 3 we present the modelling of non-functional requirements specific to the e-voting domain, followed by the contribution analysis. In chapter 4 we present the results of trust and execution dependency modelling, followed by the tool supported analysis of the model. Finally, in chapter 5 we draw some conclusions on the evaluation of the proposed methodology.



## Chapter 2

# Applying Tropos/Secure Tropos to e-voting scenario

### 2.1 Methodological issues

As discussed in Deliverable 7, our approach in combining UML and Tropos/Secure Tropos modelling is as follows. Firstly, UML model is used at the “object” level to model processes. Then, Tropos/Secure Tropos are used at the “meta” level, to reason about design alternatives with a twofold purpose:

1. to provide a rationale for the solutions adopted for the implementation of the system “to be”, by modelling possible alternative ways of accomplishing a goal;
2. to explore trust and security issues related to the e-voting process.

The former consists in identifying the non-functional requirements relevant to the scenario, and then performing contribution analysis to understand the impact of different functional alternatives (i.e. different ways of operationalizing this or that goal) on such non-functional characteristics of the system as, e.g., performance, usability, maintainability, security, etc. For the details on this type of analysis we refer the reader to Deliverable 7 [4].

The second use of Tropos pointed out above, refers to trust and security properties of an organizational model, which is an extremely important perspective to be taken into account.

The starting point of this inquiry is the analysis of the dependency relations holding between the actors involved in a process in terms of trust and delegation, which we conducted using Secure Tropos.

The main idea is that in a voting scenario there are many situations that involve the management of sensitive data and that are thus very critical with respect to security concerns and very often the actors that own or use these data should trust the actors that manage the data. Moreover, some operations need to be delegated and, even in this case, the actors to whom the operations are delegated must be trusted by the actors who are delegating. In particular, it is highly undesirable to delegate some critical operation to a distrusted (or just not entrusted) actor.

The approach to security analysis we propose in this deliverable, has two main objectives:

1. understanding which are the “problematic” trust/delegation relationships, for instance the cases in which an actor has to delegate the accomplishment of some important goal to another actor whom she doesn’t trust;
2. proposing a “secure” solution, for instance by monitoring the execution of such actions or by appointing a different (trusted) actor for the same action.

This is accomplished through the process depicted in Figure 2.1 and described in the following:

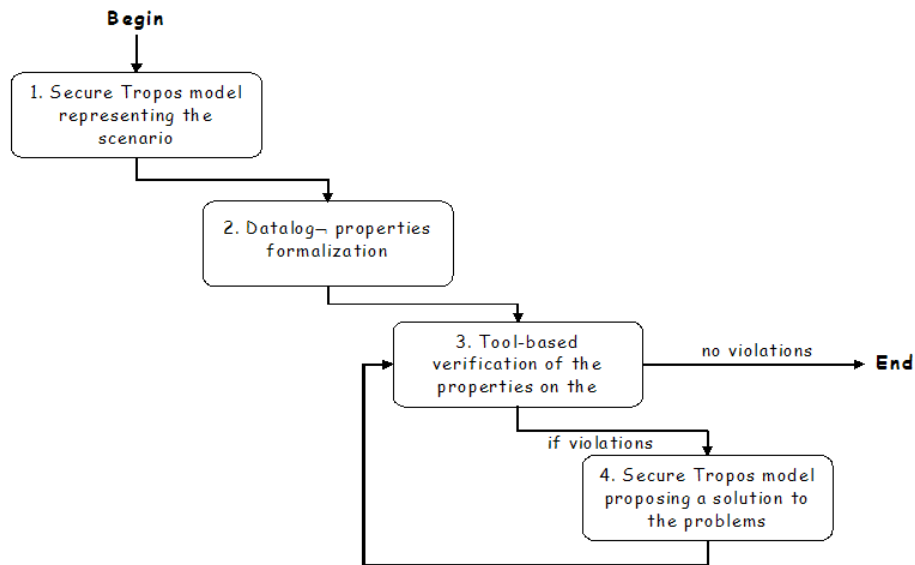


Figure 2.1: The analysis process used in this deliverable.

1. we build a Secure Tropos model of all the trust and delegation relationships involved in the various available alternative ways of accomplishing a certain goal, in order to verify whether there are untrusted actors performing important tasks or there are trust's conflicts;
2. we formally define (in Datalog) the security properties we want the model to comply with. Some security properties have already been formalized during the development of Secure Tropos [9];
3. we perform a check on the model using a tool [2] and we single out the "problematic" relations; the features of this tool have been explained in Deliverable 6 and its usefulness for the present work consists in allowing an automatization of the analysis. If not violation was detected, the process ends;
4. we build a new Secure Tropos model in which solutions are proposed, for instance changes of the addressee of these relations or adjuncts of monitoring relations over distrusted actors. We step back to the security check (step 3);

The last two operations are performed iteratively until the check is passed.

## 2.2 Modelling the e-voting process

Before turning to the examples that have the purpose of showing the approach just presented, we briefly remind how the voting process (considered in ProVotE) is organized. Here are the steps of the election process.

1. *Identification and registration of the voter.* At the polling station the voter is usually required to show his/her ID card and the electoral card. If the name of the voter is present in the electoral list of the polling station, the voter is registered, the electoral card stamped, and the voter is admitted to voting.
2. *Casting a vote.* The voter is given a ballot and a pencil and is shown a cabin where the vote can be cast in secrecy. Secrecy is both a right and a duty. The Italian law and procedures are aimed at ensuring that a voter cannot make his/her vote manifest to other people. At the end of the voting day, the ballot boxes are opened and the counting procedure starts:
3. *Counting.* Votes are counted and the results tabulated in special registers.
4. *Transmission of the results.* When all the ballots have been tabulated, the results are transcribed in various paper documents and transmitted to the offices responsible of aggregating all the data.

5. *Sum and proclamation of the elected representatives.* All the data coming from the different polling stations are counted and seats assigned according to algorithms defined by the law. Data are then made available to the general public.

In the following, we present the two preliminary Tropos models for the e-voting scenario, which serve as a starting point for the further analysis.

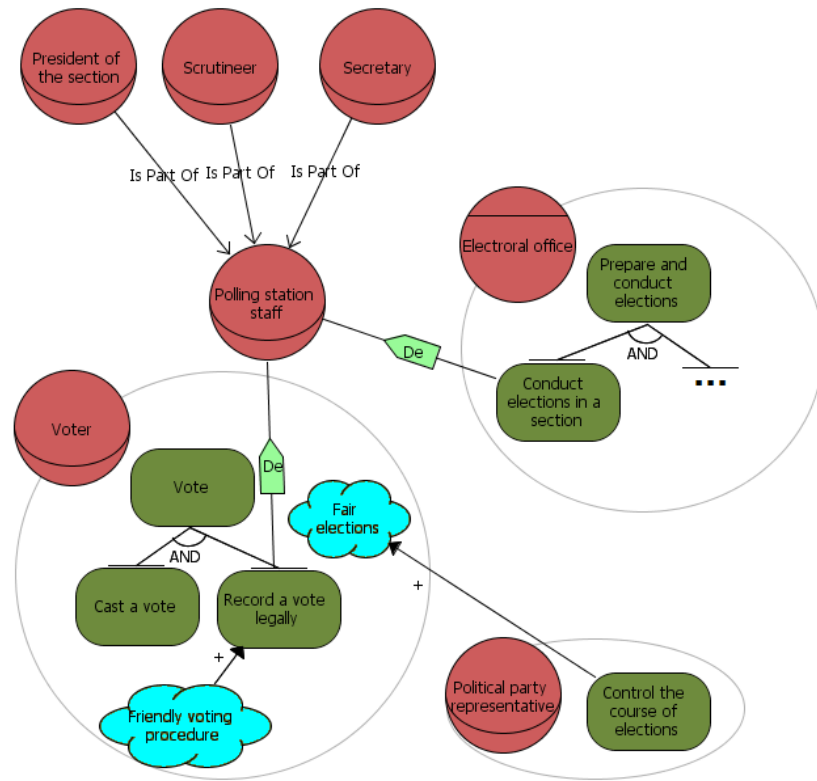


Figure 2.2: Secure Tropos model depicting the stakeholders of the e-voting scenario along with their top level goals.

Figure 2.2 presents the first high level model of the e-voting stakeholders (*voter*, *polling station staff*, *Electoral office* and *political party representative*) along with their goals<sup>1</sup>. The staff of each polling station staff consists of three

<sup>1</sup>Dots in the decomposition tree in this and further diagrams are used for the purpose of presentation simplicity, and stand for “and many other goals which are not relevant for this example”. Of course, dots are not an element of the Tropos modelling notation.

role entities: *president*, *secretary* and *scrutineer* (in fact, there can be several scrutineers in a station). Voter's concerns are in that the elections should be conducted in a fair way, and that the voting procedure should not be overcomplicated. A voter delegates to a certain polling station staff the job of organizing elections in a proper way. In turn, the central authority, i.e. Electoral office, delegates to the polling station staff (in fact, to the president) all what concerns the organization of the elections locally in the station. Finally, representatives of the involved political parties want to be given a right to (to some extent) monitor the election process.

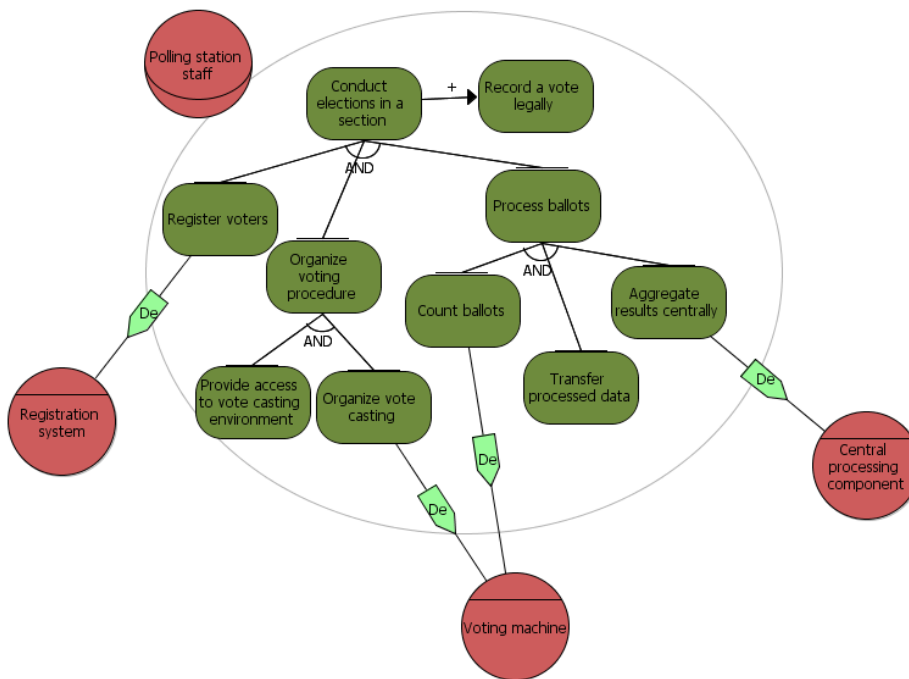


Figure 2.3: Secure Tropos diagram depicting responsibilities and major dependencies for the *polling stations staff* actor of the e-voting scenario.

Figure 2.3 presents the *polling station staff* actor and the details on the activities this actor is involved in on the day of elections. Note that the diagram does not contain preparation procedures, which precede the election day. Two soft/hardware entities support the polling station staff in place, namely, *voting machines* and *registration system*. The third entity, *central processing component*, is the same for all polling stations. An important activity, which involves both technological and organizational issues, is transferring the resulting material to the electoral office at the end of the election day. In the following two sections we will analyze this

process in details.

## Chapter 3

# Transfer data scenario: contribution analysis

At the end of the election day, the results of the election in each section should be sent to the Electoral office. In the e-voting system developed within ProVotE, three kinds of resulting artifacts represent the voting results: electronic ballots copied from a voting machine on an USB key<sup>1</sup>, the USB keys, and the paper ballots produced by the voting machine printer. Electronic data are transferred through a Virtual Private Network (VPN) to the central server, while USB keys and paper ballots are either physically sent to the Electoral office or not (e.g. they might be kept somewhere locally, and used only if some problems with the electronic data occur).

There are various ways for carrying out the data transfer process. The actor responsible for this duty is the president of the section. Three basic alternatives are the following ones:

1. *Transfer electronic data*: this option is the most innovative one, because it removes the need any physical data transfer, and it requires all the actors to trust the reliability of both electronic data means and transfer channels;
2. *Transfer USB keys and electronic data*: this option is different from the previous one in providing the redundancy in the transferred data. It still requires that all the actors highly trust the electronic data means and channels;
3. *Transfer electronic and paper data*: in this solution paper data is still considered important, which is important in the cases where certain (old fash-

---

<sup>1</sup>Note that here we assume that one technological choice is already made – at the end of an election day voting data (either processed or raw) are copied from each voting machine on a special USB key, one for each machine.

ioned) actors trust much more the reliability of paper based data means than the electronic ones;

4. *Transfer all data means*: all the data means are transferred, in order to provide the highest level of such security requirements as, e.g., data integrity and the reliability of the transfer procedure.

In Figure 3.1 the (partial) contribution analysis of alternative choices and non-functional requirements of the data transfer process is represented.

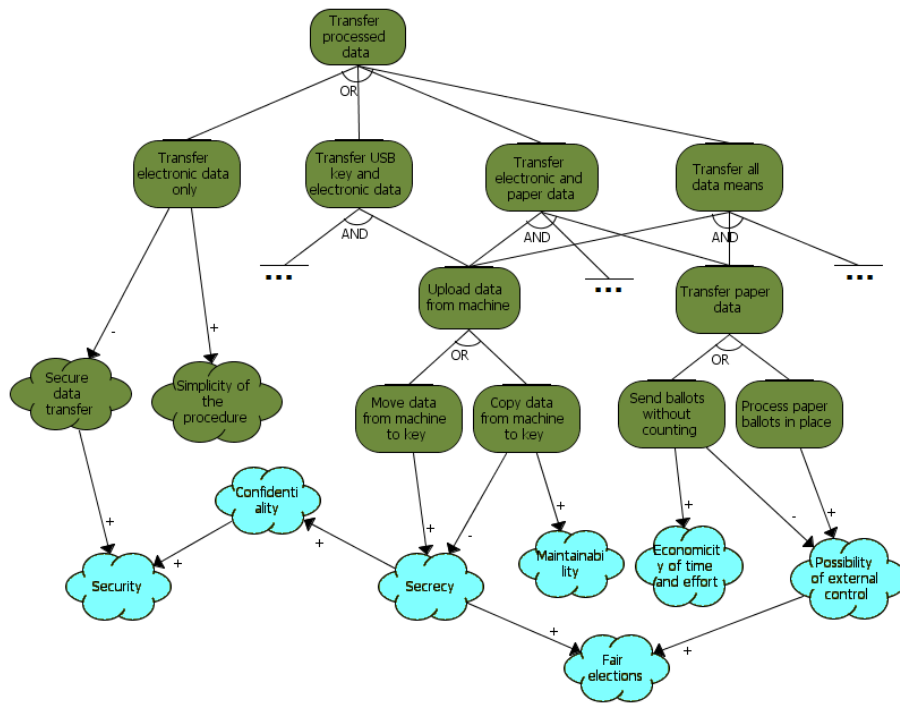


Figure 3.1: Secure Tropos model depicting the results of contribution analysis for the data transfer scenario.

Transferring only electronic voting results makes the procedure easy to organize and control, but, at the same time, makes the security issues very crucial. Namely, the connection and the procedure of uploading the data on the server should be secure enough to avoid malicious user intervention. This observation is the key motivation for not adopting the alternative design system in which only electronic data are transferred to the Electoral office. Of course, security should

not be ignored when both the electronic data and USB keys and/or paper ballots are transferred, but in this case it is less crucial as data inconsistencies can be detected and fixed by comparing data on several media.

If a decision to send an USB key (one per each voting machine) to the Electoral office is taken, another choice needs to be made. Should we copy the data from a voting machine to the key, leaving a copy on the machine hard disk, or should we move it, erasing the machine hard disk content? Moving data means to reduce the number of media which should be carefully protected against unauthorized access, whereas copying data means to double the number of vulnerable points. On the other side, leaving the data copy on a voting machine contributes positively to system maintainability as it becomes easier to detect and recover from errors. The latter motivates the choice of copying the data to USB keys rather than moving.

A number of other choices concern paper ballots, which are sent to the Electoral office either on a regular basis, or only if there is a problem in transferring the other data media, or the results appear to be inconsistent. The choice we consider here is about the processing of paper ballots: should they be counted in a polling station, or only later, in the Electoral office, or even both? Unlike the previous discussed choice, related to moving or copying data from an USB key, these alternatives concern the organization of the new e-based voting process, rather than technological issues. Counting the paper ballots in a polling station requires more time and human resources, however, unlike counting centrally, allows the external observers to control the process. This last point might be crucial as the interests of the representatives of political parties should also be taken into account while designing the new voting system. The final decision with respect to this point is not taken yet, as the project has not yet produced the instructions on the new procedures, being focused more on the technological issues.

In [6] the analogous contribution analysis of the other e-voting subprocess, namely, vote counting, is presented.



## Chapter 4

# Transfer data scenario: analyzing security properties

This section examines organizational security concerns related to the transfer of voting data from the voting section to the central electoral office. This post-voting procedure is particularly critical in the context of security and trust, because collected data should not be manipulated, corrupted, or lost during its transfer.

Before showing how the Secure Tropos methodology can help to deeply analyze the case study, we would like to give details on the alternatives we consider for carrying out the data transfer process, which are the following three

1. Transfer USB keys and electronic data;
2. Transfer electronic and paper data;
3. Transfer all data means.

These three possible solutions need to be further examined. It can be easily seen that the alternatives are a combination of three data transfer elementary processes: (a) USB keys transfer, (b) electronic data transfer, and (c) paper data transfer. These basic processes have to be refined, showing the steps to be carried out:

(a) USB keys transfer:

1. send data to electoral office via PC.

(b) Transfer USB keys:

1. upload data from the machines;

2. prepare USB keys for sending;
  3. physically transport the keys to the electoral office.
- (c) Transfer paper ballots:
1. extract paper ballots from machines;
  2. prepare ballots for sending;
  3. physically transport the ballots to the electoral office.

Case (a) represents a secure electronic transmission of data from a PC to the electoral office. This should be performed by the president, so the central electoral office should only trust the president. However, data must also be transferred from machines to the PC from which they are sent. This transfer (presumably via USB key) must be performed by the president, and the central electoral office must trust the president also for this step.

In case (b) the president uploads data on keys, while the secretary prepares USB keys for sending. The old paper system required the usage of packages with ballots that were sealed and signed by two scrutineers, before the sending. In order to keep a similar integrity mechanism, scrutineers must trust the secretary. The following step is the physical transportation of the keys to the central office; this activity is performed by a policeman, who must be trusted by the central office.

In case (c) the scrutineer extracts paper ballots from machines and passes them to the secretary (who must trust him/her) and the package containing them must still be signed by the two Scrutineers and physically transported to the central office (as in (b)).

## 4.1 Secure Tropos modelling

An initial attempt of modelling the scenario drawing a diagram type supported by the Secure Tropos methodology is shown in Figure 4.1, where some assumptions have been made concerning trust and distrust relationships.

The actors can be identified directly from the natural language description of the scenario; in this case the actors are the following ones:

- Section president (Role);
- Electoral office (Agent);
- Policeman (Role);
- Secretary (Role);

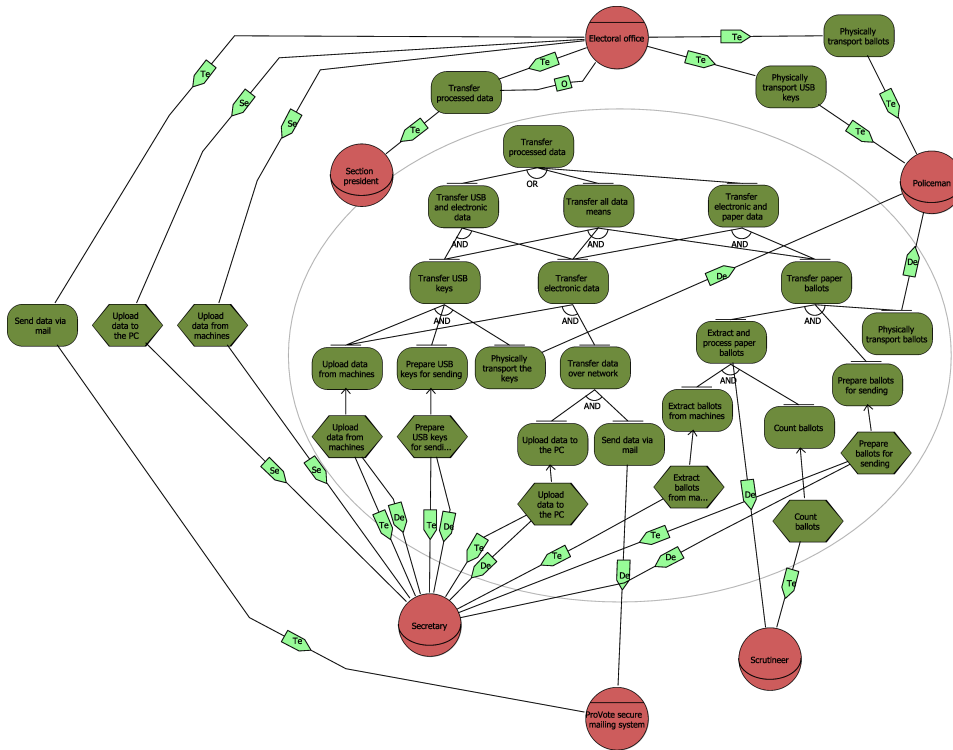


Figure 4.1: Secure Tropos model depicting the initial scenario (data transfer to the electoral office).

- Scrutineer (Role);
- ProVote secure mailing system (Agent).

Roles are used to identify those actors that have a certain function in the process, but whose characteristics do not depend on the specific agent playing that role; agents are used when the specificity of the actor is relevant, and we have an actual instantiation. Roles are classes, while agents are instances.

The main actor that has been modelled is the Section president, because she is responsible for the correct execution of the data transfer procedure. If something goes wrong, she can be held responsible, even if the problematic activity was not performed directly by herself. Her main goal is to *transfer processed data*, and she can achieve this goal by achieving one of the subgoals *transfer USB and electronic data*, *transfer all data means*, and *transfer electronic and paper data*. These are exactly the three alternatives proposed in the natural language description, translated to the Secure Tropos notation (the Secure i\* language). The further decomposition

process follows the description, using three sub-goals whose various combinations represent the three alternatives. The sub-goals are *transfer USB keys*, *transfer electronic data*, and *transfer paper ballots*. Some basic goals are decomposed into tasks via means-end, that is a goal can be achieved following the plan described by the task; some other goals are directly delegated to other agents or roles, that is the president does not mind the way the goal will be achieved (the exact plan the actor will follow).

Let's now examine the delegations and the trust relations in the model:

- goal *transfer processed data*: the electoral office owns this goal, and trusts the section president for the execution of this goal;
- task *upload data from machines*: the section president delegates the execution of the task to the secretary, and there is also a trust of execution relation between the section president and the secretary. The electoral office has a distrust of execution relation to the secretary for that task. This case is realistic (even if there are ways of avoiding this distrust), because the secretary does not have direct contacts with the electoral office;
- task *prepare USB keys for sending*: there are both a trust of execution and a delegation of execution relations between the section president and the secretary;
- task *upload data to the PC*: the situation is analogous to that of *upload data to the PC*, with delegation of execution and trust of execution between the section president and the secretary, and with distrust of execution between the electoral office and the secretary;
- task *extract ballots from machine*: the section president trusts the execution of this task if made by the secretary. In the whole scenario, there is a strong trust relation between the president and the secretary, since the secretary is nominated by the president;
- task *prepare ballots for sending*: there are both a trust of execution and a delegation of execution relations between the section president and the secretary;
- task *count ballots*: the section president trusts the scrutineer for the execution of this task;
- goal *send data via email*: the section president delegates the execution of this goal (the achievement, to be more precise) to the ProVote secure mailing system, and the electoral office trusts the ProVote secure mailing system for

the execution of that goal. Nothing is said about trust relations between the section president and the ProVotE secure mailing system for that goal;

- goal *extract and process paper ballots*: the section president delegates the execution of that goal to the scrutineer;
- goal *physically transport the keys*: the section president delegates the execution of this goal to the policeman, and the electoral office trusts the policeman for the execution of that goal;
- goal *physically transport ballots*: the section president delegates the execution of this goal to the policeman, and the electoral office trusts the policeman for the execution of that goal.

A lot of choices have been made in order to make this case study both realistic and interesting for the analysis, and we believe that the presented model is quite interesting because it accurately describes trust and dependency relations in the examined context.

## 4.2 First analysis results

After modelling the data transfer process, the following step is using the tool for verifying security properties over the model. The size of the depicted scenario is quite small, and analysis could be performed by hand, but if we consider the whole e-Voting scenario the situation becomes much more complicated, and then tool support becomes fundamental.

The tool supporting the methodology is based on Datalog. The model is translated to a set of Datalog predicates (*input file*), that exactly represent the scenario. A set of security properties is specified using the same language (Datalog). The security properties are then verified against the Datalog representation of the scenario, producing an *output file* that points out the violated properties in the model. A visual representation of the violated properties is also available.

### 4.2.1 The input file

A commented excerpt of the translation of the model to the input file is now presented.

```
role(section_president). goal(transfer_processed_data).
goal(transfer_electronic_and_paper_data).
goal(transfer_all_data_means).
goal(transfer_usb_and_electronic_data). goal(transfer_usb_keys).
```

...

These predicates are used to formalize the fact that the *section president* is a role, that *transfer processed data* is a goal, and so on.

```
task(prepare_usb_keys_for_sending).
task(prepare_ballots_for_sending).
```

...

With a similar syntax tasks can be defined (*prepare ballots for sending* is a task).

```
or_decomposition3(transfer_processed_data,transfer_usb_and_electronic_data,
transfer_electronic_and_paper_data,transfer_all_data_means).
and_decomposition2(transfer_electronic_and_paper_data,
transfer_electronic_data,transfer_paper_ballots).
```

...

The definition of and/or decomposition is quite easy. Datalog does not allow predicates with an arbitrary number of arguments, and hence different predicates are needed to support and/or decomposition. For instance, the first predicate specifies an or-decomposition involving three sub-goals: the goal *transfer processed data* is or-decomposed into *transfer usb and electronic data*, *transfer electronic and paper data* and *transfer all data means*.

```
agent(electoral_office).
```

...

The definition of agents is as trivial as the one of roles.

```
means_end(prepare_ballots_for_sending,prepare_ballots_for_sending).
means_end(upload_data_to_the_pc,upload_data_to_the_pc).
```

...

Means-end decomposition is done using the `means_end` predicate, where the first argument is a task, the second is a goal.

```
del_exec(section_president,policeman,physically_transport_the_keys).
del_exec(section_president,policeman,physically_transport_ballots).
trust_exec(electoral_office,section_president,transfer_processed_data).
trust_exec(electoral_office,policeman,physically_transport_ballots).
```

...

Delegation of execution and trust of execution can be expressed with a ternary predicate, where the first argument is the delegator (the one that delegates), the second argument is the delegatee (the one that receives the delegation), the third argument is the delegated service (goal, resource, or task).

ID	Violation description
1	untrustedDelExec(section_president,scrutineer,extract_and_process_paper_ballots)
2	untrustedDelExec(section_president,policeman,physically_transport_the_keys)
3	untrustedDelExec(section_president,policeman,physically_transport_ballots)
4	untrustedDelExec(section_president,provote_secure_mailing_system,send_data_via_mail)
5	trust_conflict_exec(electoral_office,secretary,upload_data_from_machines)
6	trust_conflict_exec(electoral_office,secretary,upload_data_to_the_pc)

Table 4.1: The security violations identified by running the tool on the original data transfer scenario.

```
own(electoral_office,transfer_processed_data).
distrust_exec(electoral_office,secretary,upload_data_from_machines).
distrust_exec(electoral_office,secretary,upload_data_to_the_pc).
```

...

Predicates like `own` and `distrust` of execution are expressed with the same logic used by the previous predicate types.

#### 4.2.2 The output file

The output file represents the results of the analysis, and contains both the scenario model and the derived predicates, among which there are the violated properties. Security properties are expressed via a negation, that is the output file shows the identified security failures.

In this section we just present the security properties that do not hold, because the rest of the file is identical to the input file. We also enumerate the problems (Table 4.1), since in the following we will propose a table associating the problems to the possible ways of achieving the main goal of transferring the voting data. In fact, the security analysis can help us in two ways:

1. discover missing trust/dependency/monitoring relations and enact them;
2. classify some solutions as unfeasible, because of missing relations that cannot be established.

There are four problems of type **untrusted delegation of execution**, namely some delegation of execution relations do not have an adequate trust. For instance, the section president does not trust the scrutineer for the execution of `extract` and `process paper ballots`. The same problem occurs between the section president and the policeman, for the physical transport of both keys and ballots. There are also **trust of execution conflict** problems, that is some actor trusts another actor for the execution of a service (typically the delegator), while a third actor does not. For instance, the electoral office does not trust the secretary for the upload of data

from machines (problem number 5); the modelled scenario shows that the section president has that trust of execution relation.

Table 4.2 shows how the identified problems are related to the various solutions. Sending all data means is the worst case, because that solution has the problems coming from both the USB and electronic solution and from the electronic and paper solution. The obvious advantages in terms of reliability are not shown, because the framework is not intended to verify those properties; in terms of security having three kinds of data transport is more risky.

	USB and electronic	All data means	Electronic and paper
1		✓	✓
2	✓	✓	
3		✓	✓
4	✓	✓	✓
5	✓	✓	
6	✓	✓	

Table 4.2: Mapping between the identified security property violations and the possible data transfer strategies. The numbers of the security violations are consistent with Table 4.1

### 4.3 Improving the model

This section shows some ways of improving the model, in order to avoid the security property violations shown in the previous section.

This improvement is not always feasible, because certain types of trust problems cannot be solved: the model should reflect the real world, and when real trust cannot be added also the model should depict this lack. However, a remedy could consist in finding a solution, propose it to the actors involved in the scenario, and ask them to enact it in order to avoid organizational security violations.

The first enhancement to the scenario tries to face the trust of execution conflicts between the electoral office and the secretary, for the tasks *upload data to the PC* and *upload data from machines*. We adopted two different solutions to solve the violations: in the first case (*upload data to the PC*) the electoral office requires that the secretary signs a contract by which the secretary guarantees the integrity of the data (she becomes responsible for possible violations); in the second case (*upload data from machines*) the section president takes charge of the task, removing the delegation of execution from the secretary. The model obtained after adding the monitoring of execution can be seen in Figure 4.2.



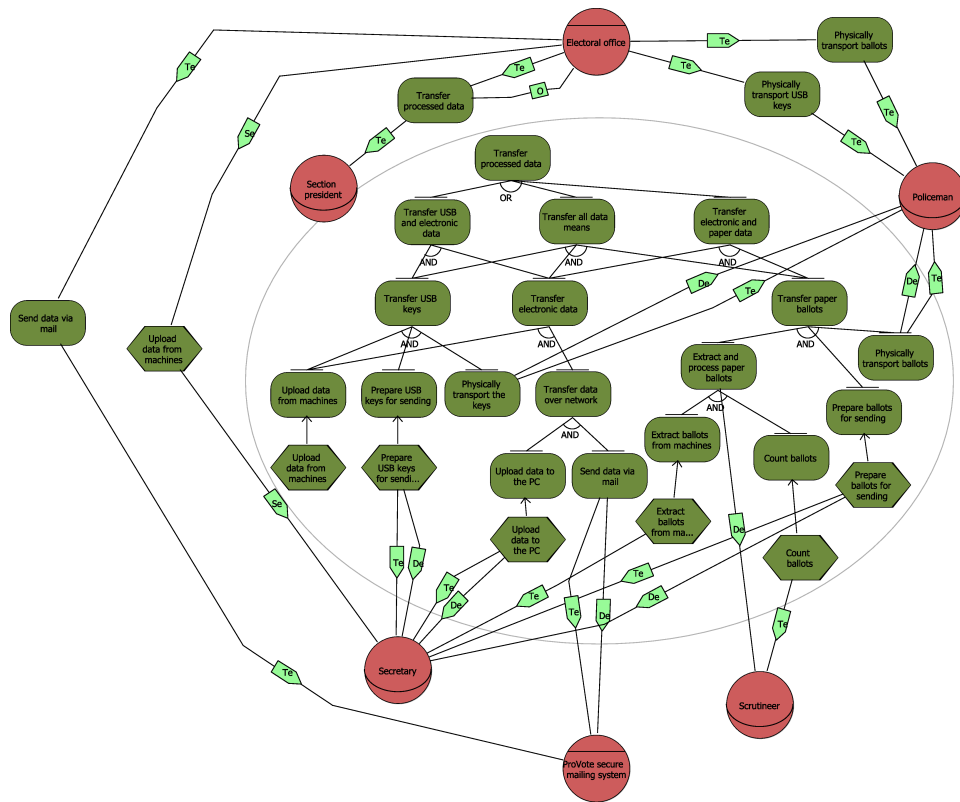


Figure 4.3: Secure Tropos model depicting the scenario after removing violations 2,3,4

step.

The last problem to be solved is the untrusted delegation of execution of the goal *extract and process the ballots* from the section president to the scrutineer. In fact, the execution of the whole goal is delegated, but trust is assured only for the plan *count ballots*. The section president trusts the secretary for the execution of the task *extract the ballots from the machines*, but she is not delegated for that task. The solution we suggest is to remove the delegation of execution of the main goal, and add two delegations of execution relations for the sub-goals *extract ballots from machines* and *count ballots*, where the delegates are the secretary and the scrutineer, respectively. This solves the identified security violations, and the scenario obtained after enacting these improvements can be seen in Figure 4.4.





## Chapter 5

# Conclusion

In this deliverable we have tried to show, with the help of a detailed example, the usefulness of the methodology presented in Deliverable 6 applied to an e-voting scenario, namely that of the ProVotE project. In particular, Tropos and Secure Tropos have revealed their effectiveness in the analysis of the rationale of choices among alternatives and of security concerns.

In the ProVotE case study, Tropos has been used to motivate the choice of a particular procedure based on electronic means to substitute a procedure in the traditional “paper and pencil” framework. Secure Tropos has instead been used to analyze whether relations that were critical from the security standpoint involved untrusted actors.

The example has precisely the purpose of empirically showing how a methodology like Tropos, involving both a formal modelling analysis and the possibility to apply a tool to automatize the analysis can result in a deeper understanding of very complex scenarios.

Obviously, this methodology is not given once and for all, it needs to be each time adapted to the specific domain at hand. This feature, that at a first sight can be seen as a drawback, is actually an asset. This is because this makes Tropos/Secure Tropos a flexible methodology, whose application can be extended to different domains. There are also other modelling activities (presented in Deliverable 6) that are not particularly important in the case of e-voting, but that are relevant for other domains. Take for example e-government: in Public Administration is often of primary importance to understand who can be held responsible for the accomplishment of some activities or for the behavior of someone else; in these cases supervision modelling becomes central. E-voting is just one among the many possible domains that can be analyzed in the way reported in this deliverable. Under this perspective, the presented work on the e-voting case study can give some insights

on how such a customization (of a methodology to the specific domain) can be done.

To sum up, we could say that the methodology proposed within the MOSTRO project is a powerful and flexible tool for modelling and analyzing organizations, and, in particular, security within organizations, and it can be adapted for a variety of real-life domains.

# Bibliography

- [1] ProVotE project. <http://sra.itc.it/provote/>.
- [2] SI\* Tool. <http://sesa.dit.unitn.it/sttool/>.
- [3] MOSTRO Del.6, WP 3. Whole Methodology Specification. Available at <http://www.loa-cnr.it/mostro/files/MostroDel6.pdf>, 2007.
- [4] MOSTRO Del.7, WP 4. Case Study Description. Available at <http://www.loa-cnr.it/mostro/files/MostroDel7.pdf>, 2007.
- [5] P. Bresciani, P. Giorgini, F. Giunchiglia, J. Mylopoulos, and A. Perini. Tropos: An agent-oriented software development methodology. *JAAMAS*, 8(3):203–236, 2004.
- [6] V. Bryl, F. Dalpiaz, R. Ferrario, A. Mattioli, and A. Villaflorita. Evaluating Procedural Alternatives. A Case Study in E-Voting. In *Proceedings of the 1st International Conference on Methodologies, Technologies and Tools enabling e-Government*, 2007.
- [7] A. Mattioli. Analisi dei Processi Elettorali in ambito di voto elettronico per le Elezioni in Provincia di Trento, 2006.
- [8] E. S.-K. Yu. *Modelling strategic relationships for process reengineering*. PhD thesis, University of Toronto, 1996.
- [9] N. Zannone. *A Requirements Engineering Methodology for Trust, Security, and Privacy*. PhD thesis, University of Trento, 2006.