UNIVERSITY OF TRENTO - Italy

IDENTIFYING CONFLICTS IN SECURITY REQUIREMENTS

Elda Paja, Fabiano Dalpiaz, Paolo Giorgini

Re-Seminar

February 28th 2013

Outline

- Introduction
  - TasLab Case Study
  - Baseline: STS-ml
- Formal framework
  - Conflicts among authorisations
  - Conflicts among business policies and security requirements
- Evaluation
  - Findings from the case study
  - Scalability study

Introduction

***Conflicting requirements** are requirements that **cannot** possibly be satisfied at the same time*

- Requirements are inherently prone to conflicts
  - They originate from different stakeholders with different needs
- Security requirements are no exception!
  - Their violation leads to severe consequences
    - Non-compliance: privacy laws infringements, loss of reputation, and monetary sanctions
  - Critical in STS: each actor defines its individual policy independently
- Non-compliance is not an option!
  - Coping with such conflicts at requirements-time avoids designing and implementing a non-compliant and hard-to-change system

Introduction

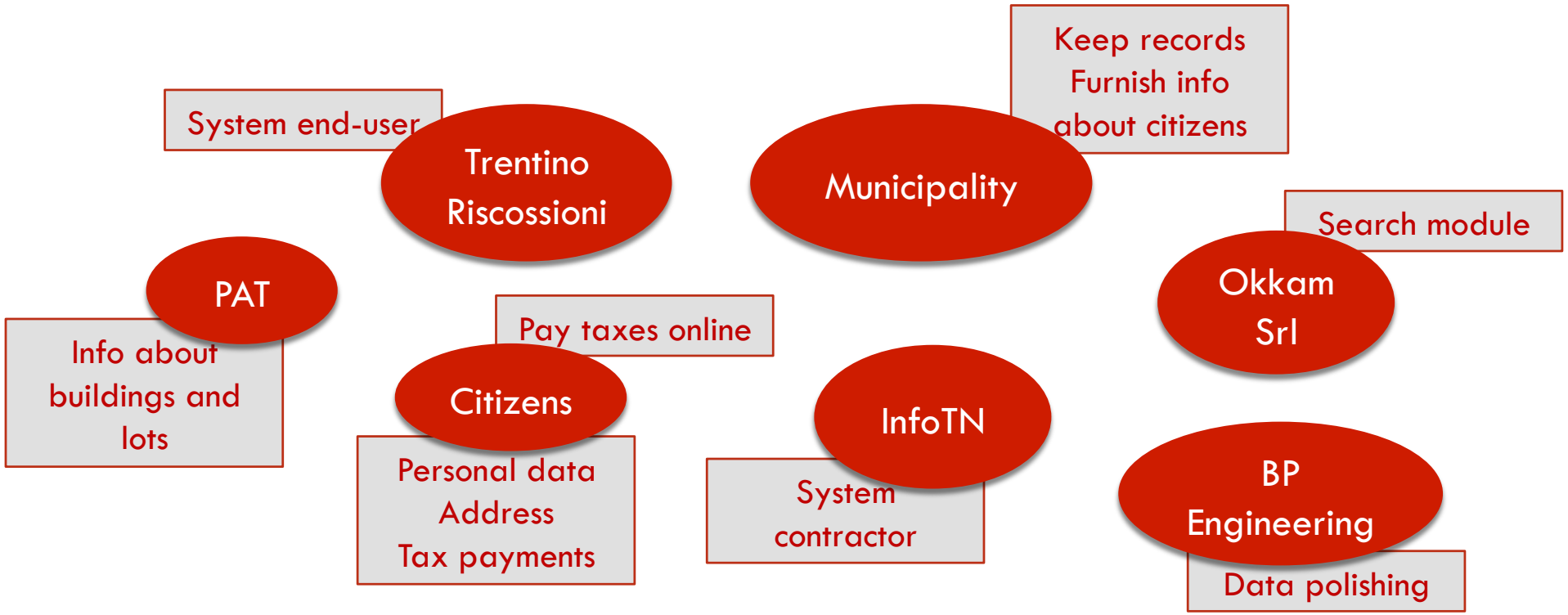
- The problem

- Challenges
  - Conflicts (inconsistencies) not trivial to spot
  - Models are often large, cannot be effectively analysed manually

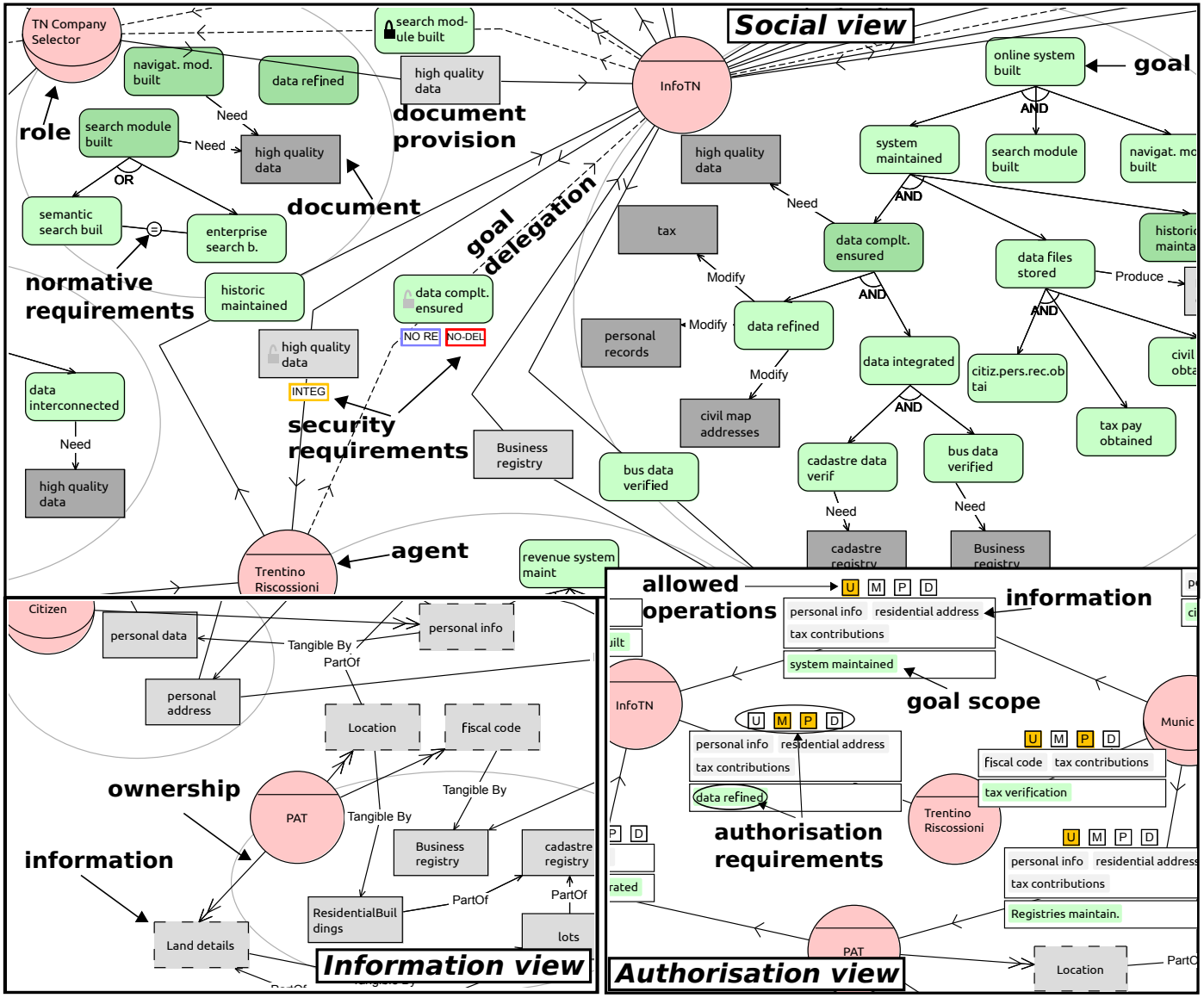
Automated reasoning techniques *are needed to identify conflicts among security requirements, and between business policies and security requirements*

TasLab Case Study

- Trentino as a Lab: online collaborative platform to foster ICT innovation in Trentino
  - Ongoing project: tax collection and verification in Trentino



Baseline: STS-ml



- 3 views: different perspectives of the same model
- Business policies
- Security requirements
  - Interaction (security) requirements
  - Normative requirements
  - Authorisation requirements

Supported security policies

- Interaction (security) requirements
  - non-repudiation (3 types): non-repudiation of delegation, of acceptance, of delegation and acceptance;
    no-delegation;
    redundancy (4 types): fallback redundancy single, fallback redundancy multi, true redundancy single, true redundancy multi;
    integrity of transmission
    availability
    trustworthiness level
- Normative requirements
  - separation of duties, binding of duties: among roles and goals
- Authorisation requirements
  - non-usage, non-modification, non-production, non-disclosure, need-to-know, non-reauthorisation

Formal Framework

- A framework to detect conflicts
  - Conflicts not trivial to find
  - Scalability is an issue
- Formal language to support automated reasoning about the expressed security policies (requirements)
- Formally Defined
  - Security requirements supported by STS-ml (derived by the security policies)
    - Are the security requirements (policies) violated in the modelled STS?
    - Key question: Is the specification consistent?
- Built on top of DLV
  - Define transformation rules from STS-ml concepts and relations into Datalog predicates
  - Define propagation rules

Security requirements in STS-ml

Interaction (security) requirements

a property that an actor requires another to comply with, related to a social relationship between them: goal delegation (Del= delegates (A1,A2,G)) or document provision (Prov=provides(A1,A2,D))

- r-not-repudiated-del(A2,A1,*Del*), r-not-repudiated-acc(A2,A1,*Del*)
- r-ts-red-ensured(A1,A2,G), r-tm-red-ensured(A1,A2,G), r-fs-red-ensured(A1,A2,G), r-fm-red-ensured(A1,A2,G)
- r-not-redelegated(A1,A2,G)
- r-availability-ensured(A1,A2,G)
- r-integrity-ensured(A1,A2,*Prov*)
- r-availability-ensured(A1,A2,D)

Security requirements in STS-ml

Normative requirements

a property that the STS – intended as the legal context – requires any participating actor:

- r-not-played-both(STS,A,R1,R2) – *A cannot play both roles R1 and R2*
- r-not-pursued-both(STS,A,G1,G2) – *A cannot pursue both goals G1 and G2*
- r-played-both(STS,A,R1,R2) – *if A plays role R1 (R2) shall also play R2 (R1)*
- r-pursued-both(STS,A,G1,G2) – *if A pursues goal G1 (G2) shall also pursue G2 (G1) too*

Security requirements in STS-ml

Authorisation requirements

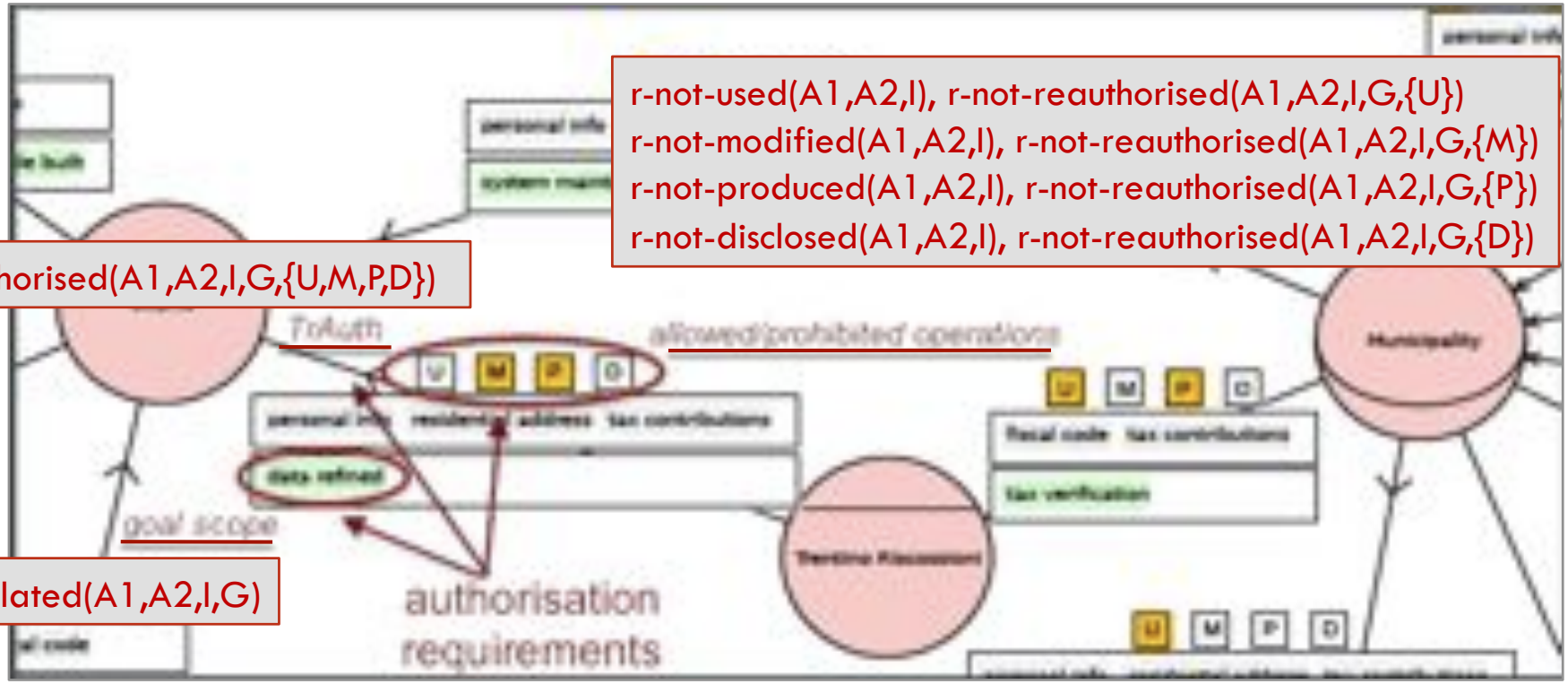
a requirement derived from an authorisation relationship

Auth=authorises(A1,A2,I,G,OP,TrAuth)

r-not-used(A1,A2,I), r-not-reauthorised(A1,A2,I,G,{U})
r-not-modified(A1,A2,I), r-not-reauthorised(A1,A2,I,G,{M})
r-not-produced(A1,A2,I), r-not-reauthorised(A1,A2,I,G,{P})
r-not-disclosed(A1,A2,I), r-not-reauthorised(A1,A2,I,G,{D})

r-not-reauthorised(A1,A2,I,G,{U,M,P,D})

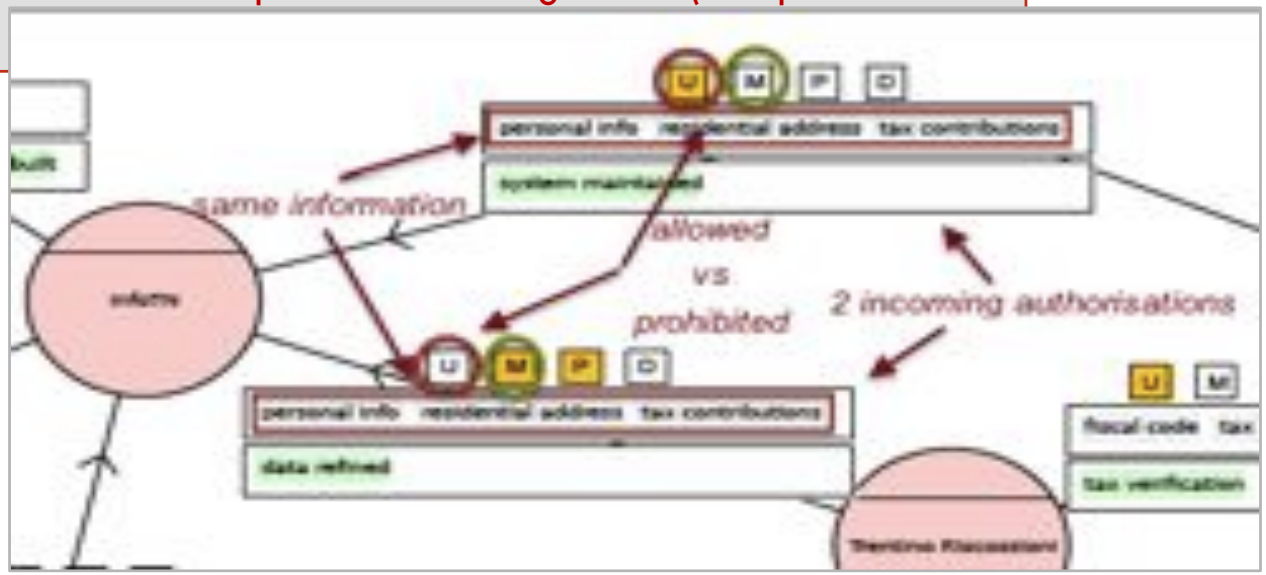
r-not-ntk-violated(A1,A2,I,G)



Identifying conflicts

- Step 1. Authorisations conflict
  - Before reasoning on conflicts between Bus. Policies and security requirements
  - Ensure authorisations are consistent

An authorisation conflict occurs for every pair of authorisation relationships, if
(1) Both authorisations apply to the same information, and either
  i. One authorisation restricts the permission to a goal scope, while the other does not, or
  ii. The scopes are intersecting, and different permissions are granted (on operations or transferability)



Identifying conflicts

- Step 2. Bus – Sec Conflict
  - Over an authorisation – consistent STS-ml model
  - Verify if any security requirement is violated by actors' business policies
    - Actors do some action they are required not to do
    - Actors do not perform some action they are required to

- But, STS-ml models contain variability
  - Intentional or social relationships define the actions an actor can possibly do
  - Security requirements imply commitments about (not) performing certain actions

STS-ml Variant: defines the *exact* set of actions actors *do carry out* to pursue their goals

Identifying conflicts

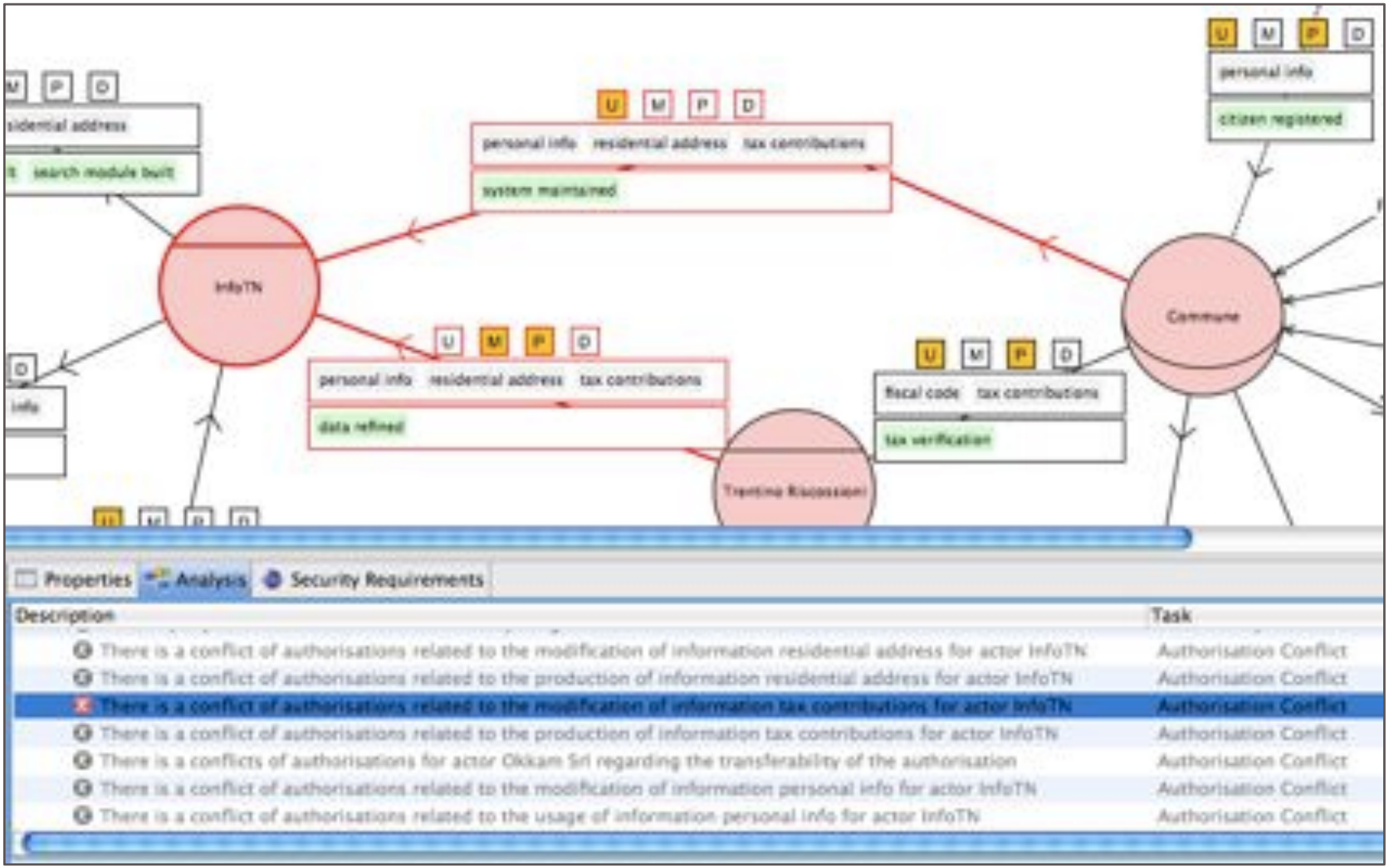
Requirement	Verification at design-time
Interaction requirements	
R_1 : r-not-repudiated-del(A_2, A_1, Del)	No
R_2 : r-not-repudiated-acc(A_1, A_2, Del)	No
R_3 : r-ts-red-ensured(A_1, A_2, G)	Partial. A_2 pursues goals in $\mathcal{V}_M$ that define at least two disjoint ways to support G
R_4 : r-fs-red-ensured(A_1, A_2, G)	
R_5 : r-tm-	
R_6 : r-fm-	
R_7 : r-not-	
R_8 : r-inte	
Authorisation requirements	
R_9 : r-not-ntk-violated$(A_1, A_2, \mathcal{I}, \mathcal{G})$	$\exists$needs/modifies/produces$(A_2, G, D) \in \mathcal{V}_M$. D makes tangible (part of) $I \in \mathcal{I}$ and $G \notin \mathcal{G}$
R_{10} : r-not-used$(A_1, A_2, \mathcal{I})$	$\exists$needs$(A_2, G, D) \in \mathcal{V}_M$. D makes tangible (part of) $I \in \mathcal{I}$
R_{11} : r-not-modified$(A_1, A_2, \mathcal{I})$	$\exists$modifies$(A_2, G, D) \in \mathcal{V}_M$. D makes tangible (part of) $I \in \mathcal{I}$
R_{12} : r-not-produced$(A_1, A_2, \mathcal{I})$	$\exists$produces$(A_2, G, D) \in \mathcal{V}_M$. D makes tangible (part of) $I \in \mathcal{I}$
R_{13} : r-n	
R_{14} : r-n	
Normative requirements	
R_{15} : r-not-played-both(STS, A, R_1, R_2)	$\{\text{plays}(A, R_1), \text{plays}(A, R_2)\} \not\subseteq \mathcal{V}_M$
R_{16} : r-played-both(STS, A, R_1, R_2)	$\{\text{plays}(A, R_1), \text{plays}(A, R_2)\} \subseteq \mathcal{V}_M$
R_{17} : r-not-pursued-both(STS, A, G_1, G_2)	A is not the final performer for both G_1 and G_2 or their subgoals
R_{18} : r-pursued-both(STS, A, G_1, G_2)	A is the final performer for both G_1 and G_2 or their subgoals

Evaluation

- 2 ways to evaluate our approach
  - Show effectiveness of our reasoning applying it to the TasLab Case study
  - Assess efficiency performing a scalability study

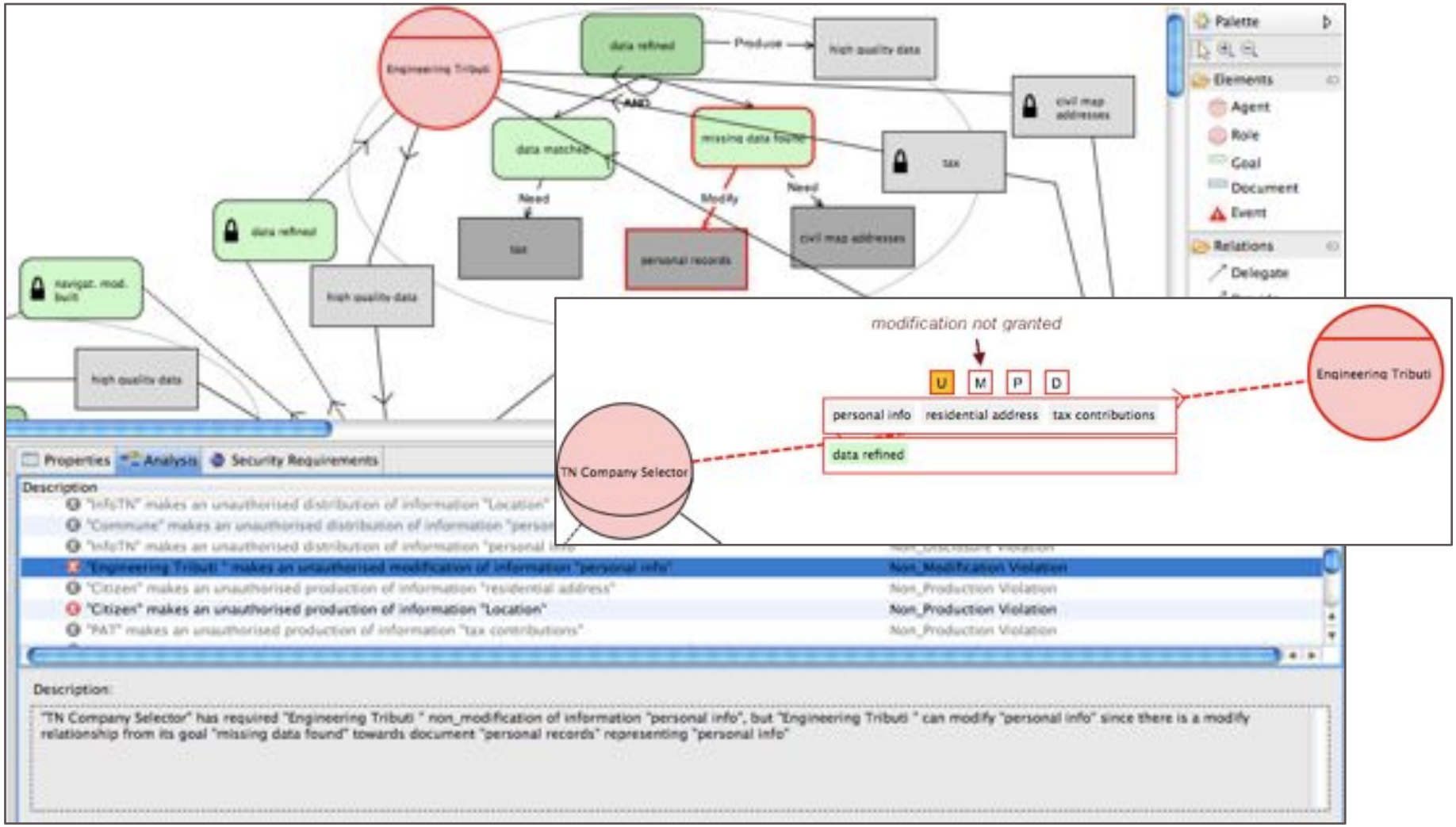
Findings from the case study

- Authorisation Conflicts



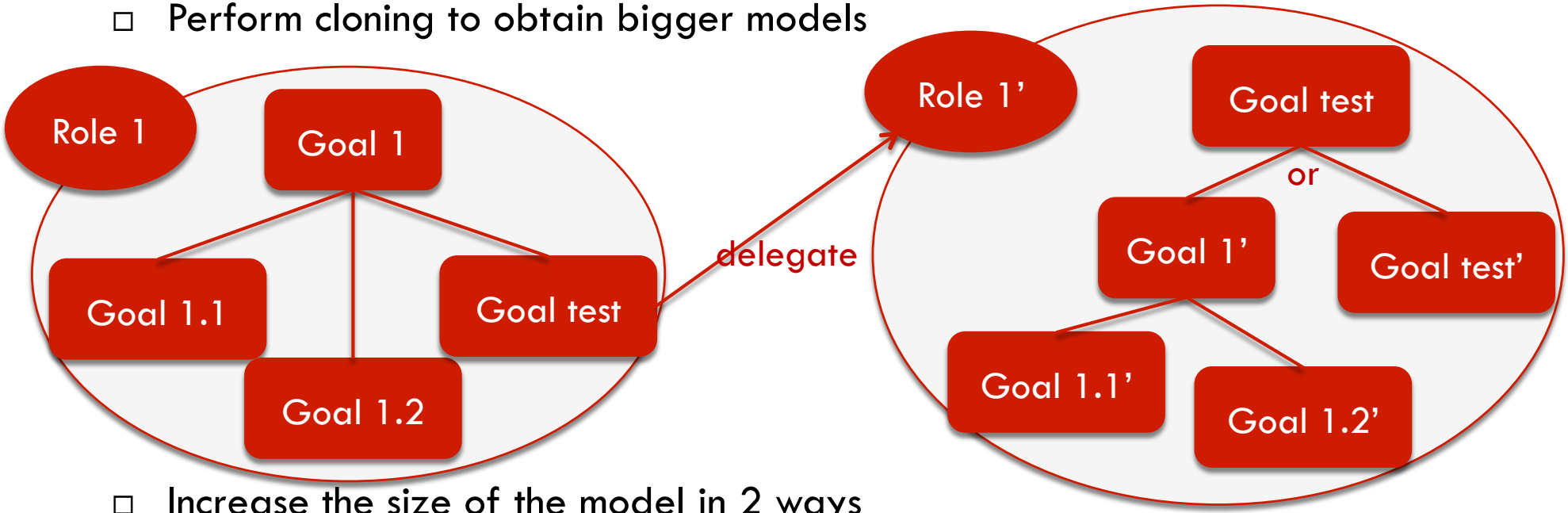
Findings from the case study

- Bus – Sec Conflicts



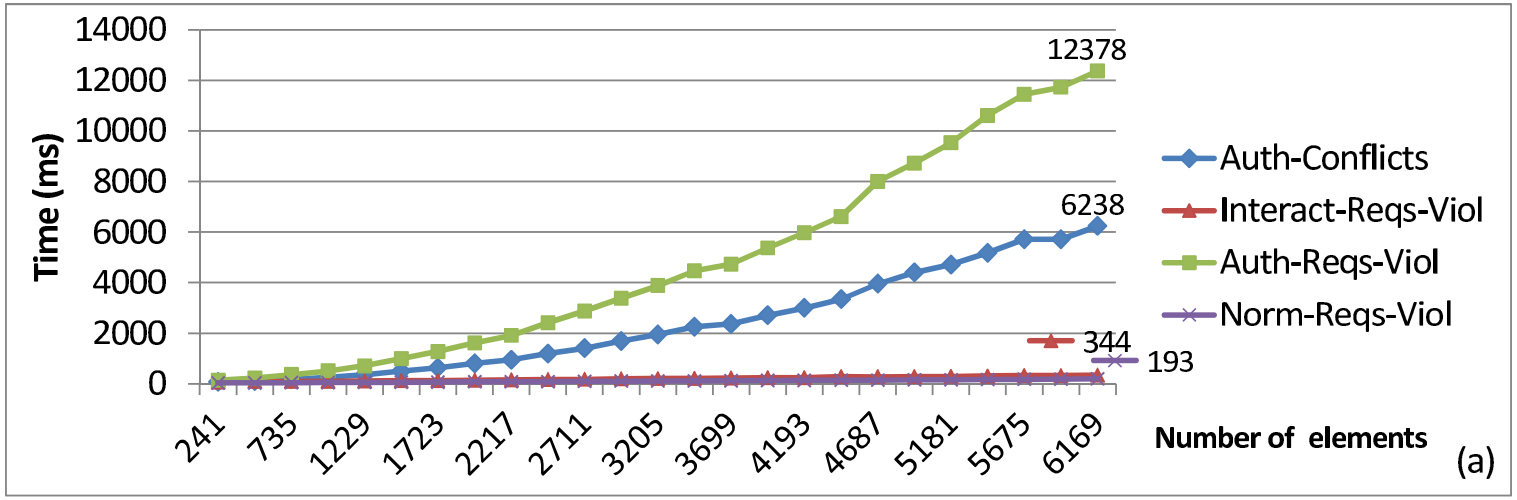
Scalability study

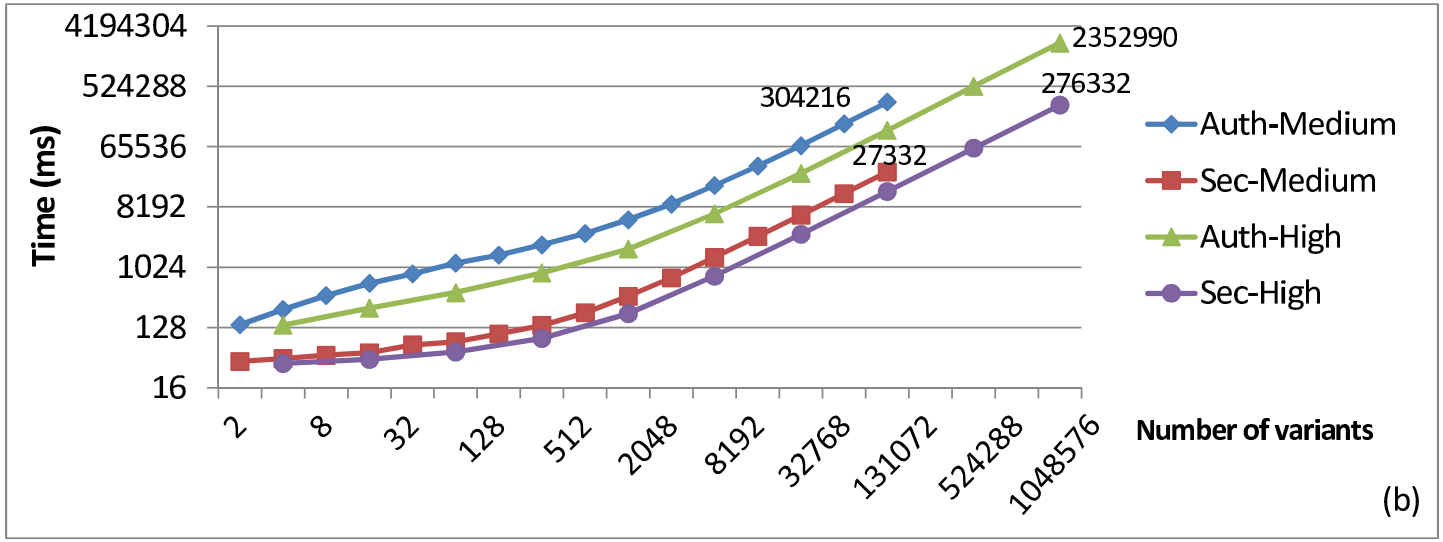
- Consider the *TasLab* case study model as a basic building block
- Perform cloning to obtain bigger models



- Increase the size of the model in 2 ways
  - Augment the number of elements (nodes and relationships) in the model
    - Models with zero variability – all decompositions considered AND-Dec
  - Increase the number of variants in the model (reasoning technique relies upon generating STS-ml model variants)
    - Models with zero, medium, and high variability and a considerate number of elements
    - *The cloning process itself also influences the model variability!*

Experimental results





Ongoing and Future Work

- Devise further analysis techniques to identify conflicts among all types of security requirements
  - For now only authorisation requirements
- Explore possible ways to resolve the identified conflicts
  - Perhaps through trade-off analysis or negotiation
- Evaluation
  - 2 different industrial case studies
    - Air Traffic Control Management
    - eGoverment

The end

Thank you!

Contact: paja@disi.unitn.it

ANIKETOS

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant no 257930 (Aniketos)