



SW'2011
Joint Work with the
SecureChange
Consortium



UNIVERSITY
OF TRENTO

ORCHESTRATING SECURITY AND SYSTEM ENGINEERING FOR EVOLVING SYSTEMS

Fabio Massacci
University of Trento

www.securechange.eu
www.disi.unitn.it/~massacci

SECURITY ENGINEERING PROCESS

- The science (and art) of transforming customer idea into concrete security artifacts for designing, coding, testing, auditing...
- Complex Process Involving Different “Viewpoints”
 - ✓ Many actors contribute to develop different aspects of the system
 - Architectural Viewpoints
 - “System” model generated by the analyst talking to the client
 - Engineering Viewpoints
 - Generates requirements lists for developers
 - Testing Viewpoint etc.
 - ✓ They use “models” to communicate and keep viewpoints in synch
 - ✓ Certification and assurance part is heavy burden
- And now some little change
 - ✓ “service 1027 is outsourced”, “We add card authorization service”
 - and you re-start from scratch...



THE MAIN IDEA OF SECURECHANGE

□ In Calculus

- ✓ calculate $f(x)$ from x
- ✓ Then a little change Δ
- ✓ do NOT calculate $f(x+\Delta)$ from scratch (i.e. from $x+\Delta$)
- ✓ Use derivative! \rightarrow do $f(x+\Delta) = f(x) + df/dx * \Delta$

□ But in Software we have many dimensions!

- ✓ $f(x,y,z,w,...)$ = x requirements, y risk, z system, w testing, ...
- ✓ Still the same we have changes. $\Delta_x, \Delta_y, \Delta_z, \Delta_w$
- ✓ Use partial derivatives!
- ✓ $f(x + \Delta_x, y + \Delta_y, z + \Delta_z, w + \Delta_w) = f(x) + \partial f / \partial x * \Delta_x + \partial f / \partial y * \Delta_y + \partial f / \partial z * \Delta_z + \dots$

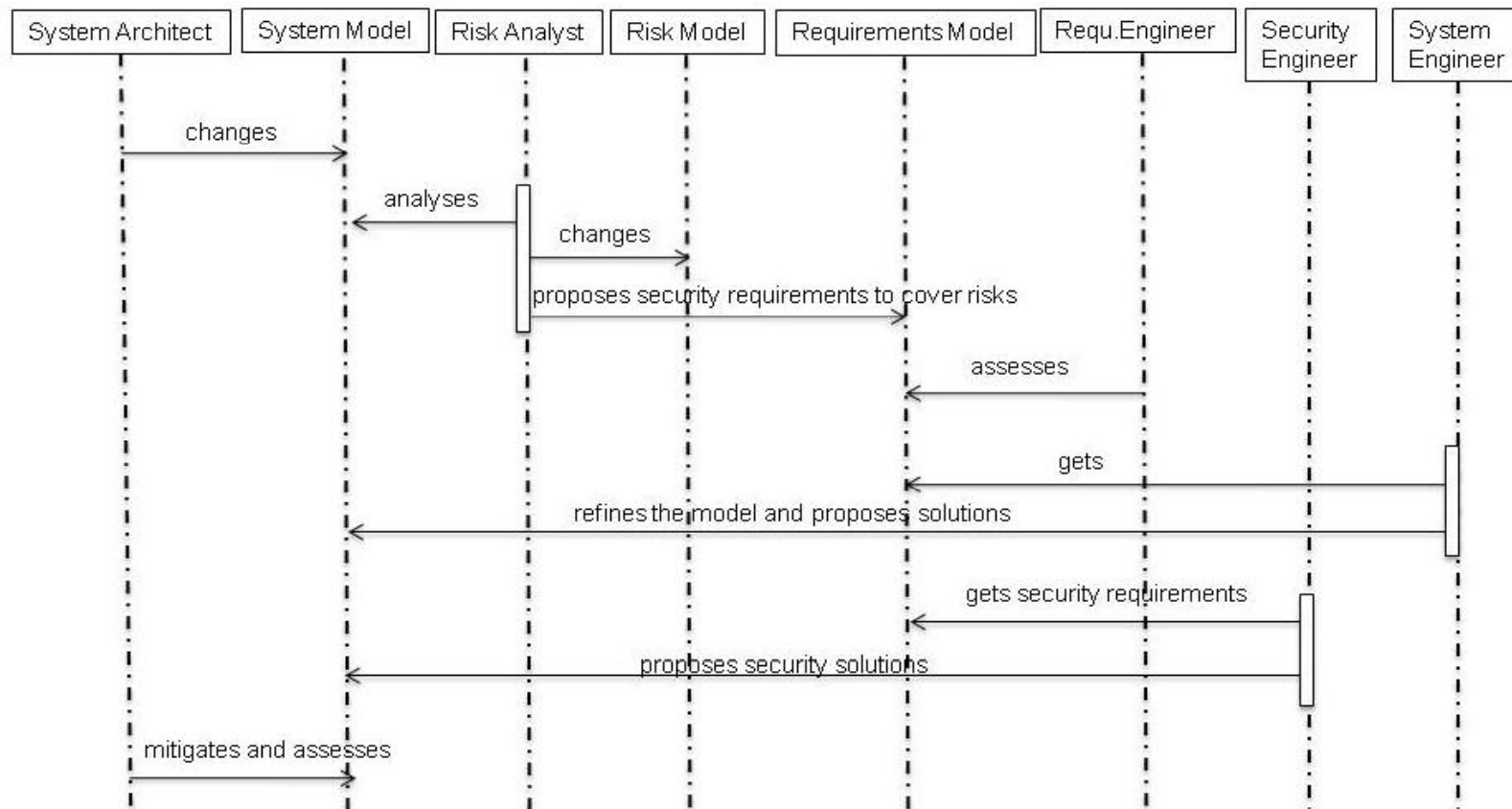
□ SecureChange does it for Service Security Engineering

- ✓ Compute (partial) derivatives
- ✓ Propagate results automatically and telling stakeholders what to change
- ✓ Automatically compute results (eg verification) when possible



SECURITY ENGINEERING PROCESS - INTERACTION

- ❑ Managing Changes requires back and forth communication
- ✓ Slides courtesy of Thales TRT



SECURITY ENGINEERING PROCESS - ROLES

Architectural Design Viewpoints (“System” Model)

- ✓ System Architect (the one closer to the customer...)
- ✓ Risk Analysts
 - Analyse “system” model proposed by System Architect to find what can hamper (threat) the important objectives (asset/goal) of the system and propose additional security goals (or security requirements)

Engineering Design Viewpoints (Requirements Lists)

- ✓ System Engineer
- ✓ Security Engineer
 - Propose security solutions that address the security requirements within the framework proposed by the System Architect
- ✓ Requirements Engineer

Each time something changes we must re-synch the viewpoints

SO WHAT'S THE PROBLEM?

- ❑ Traceability methods exists but... assume perfect visibility of model elements across viewpoints
 - ✓ Requires risk, security requirements, architectural solutions and testing to be integrated within the same tooling environment
- ❑ Doesn't match real industry scenarios
 - ✓ Legacy or Proprietary modelling techniques are used by different actors
 - ✓ Actors don't know and don't want to learn methods outside their viewpoint
 - ✓ Model IPR restrict access to designer outside the viewpoint
 - Eg smart-Card Security Certification for Gemalto is done by TrustedLabs
- ❑ Details of Models NOT shared among different viewpoint holders
 - ✓ Need a different solution → Orchestration (as opposed to integration)

SECURECHANGE SOLUTION

❑ Basically Object-Oriented Incapsulation for Design Models

- ✓ Access to different models should happen through APIs.
- ✓ Whatever happens inside you don't care

❑ Separation of Concerns

- ✓ Identify mapping concepts among requirements, risk, architectural, testing domains
- ✓ Little knowledge of the "alien" domains is required

❑ Orchestration

- ✓ Orchestrate the processes using the mapped concepts as interface
- ✓ Changes affecting a concept of the interface are propagated to the other domains



EXAMPLE WITH REQUIREMENTS AND TESTING

- ❑ Requirement Engineer introduce a new step in smart-card lifecycle for authentication
- ❑ Test Engineer push button and classification automatic
- ✓ Stagnation suite
 - Outdated "Test" → could run them to check things actually changed
 - Failed "Test" → should just not work
- ✓ Regression Suite
 - Un-impacted "Tests" → nothing to be done, just keep them
 - Re-executed "Test" → impacted but their value shouldn't change
- ✓ Evolution Suite
 - Updated "test" → same as before but expected output should change ...
 - New "test" → ... to be supplied...
- ❑ Now use Model-Based testing to regenerate just new and updated



EXAMPLE WITH SYSTEM AND RISK INTERACTION

❑ Example: Socio-Technical Systems Analysis

- ✓ Air Traffic Management under SESAR introduction of Arrival Manager
- ✓ Requires modelling both human actors and system actors
- ✓ Requires understanding complex risk scenarios

❑ System model

- ✓ Use notion of goals, resources and assignments of responsibility
- ✓ Has its own graphical notation
- ✓ Has its own methodology to identify solutions

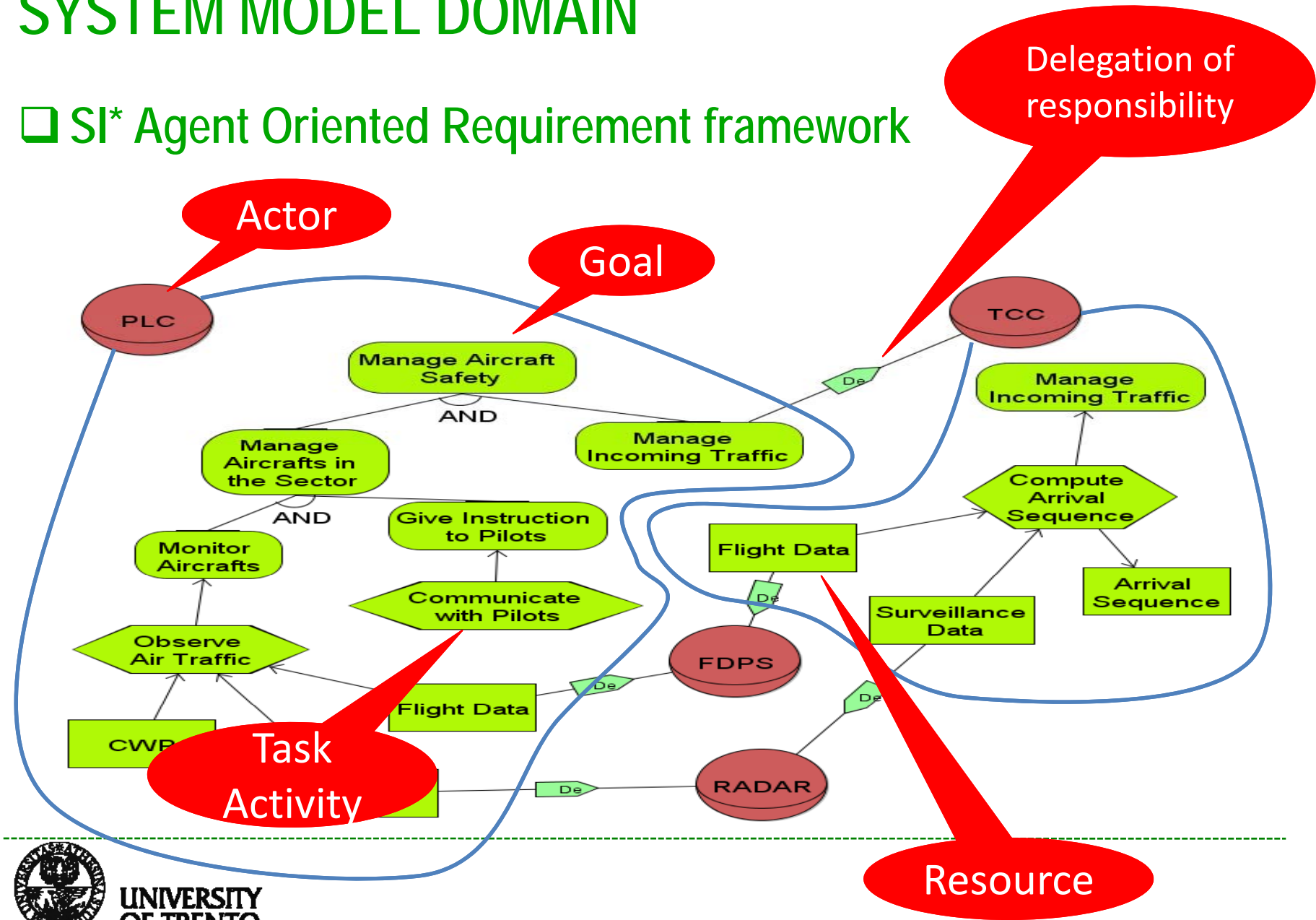
❑ Risk Model

- ✓ Use notions of target, threat and damage, risk
- ✓ Has its own ...



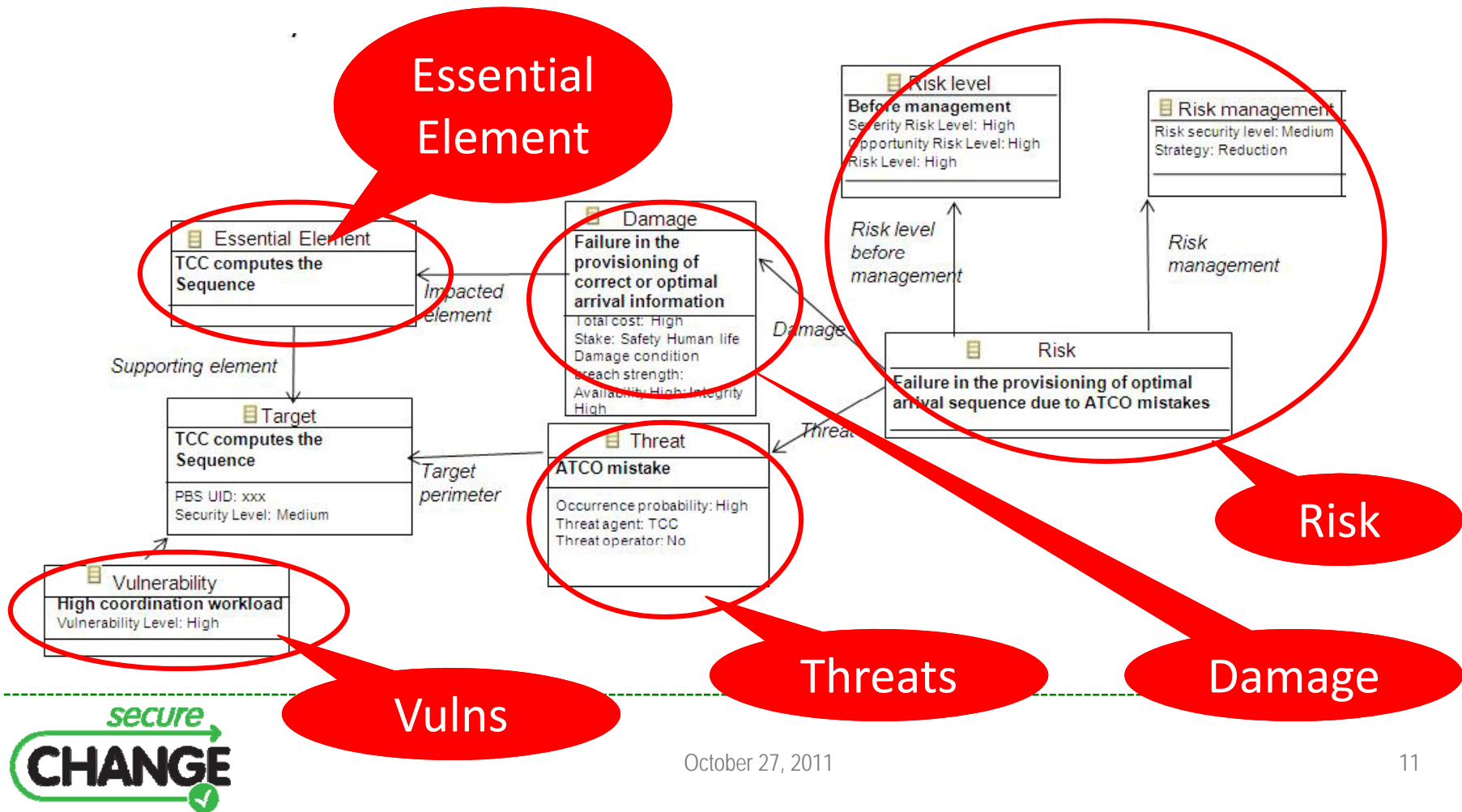
SYSTEM MODEL DOMAIN

□ SI* Agent Oriented Requirement framework



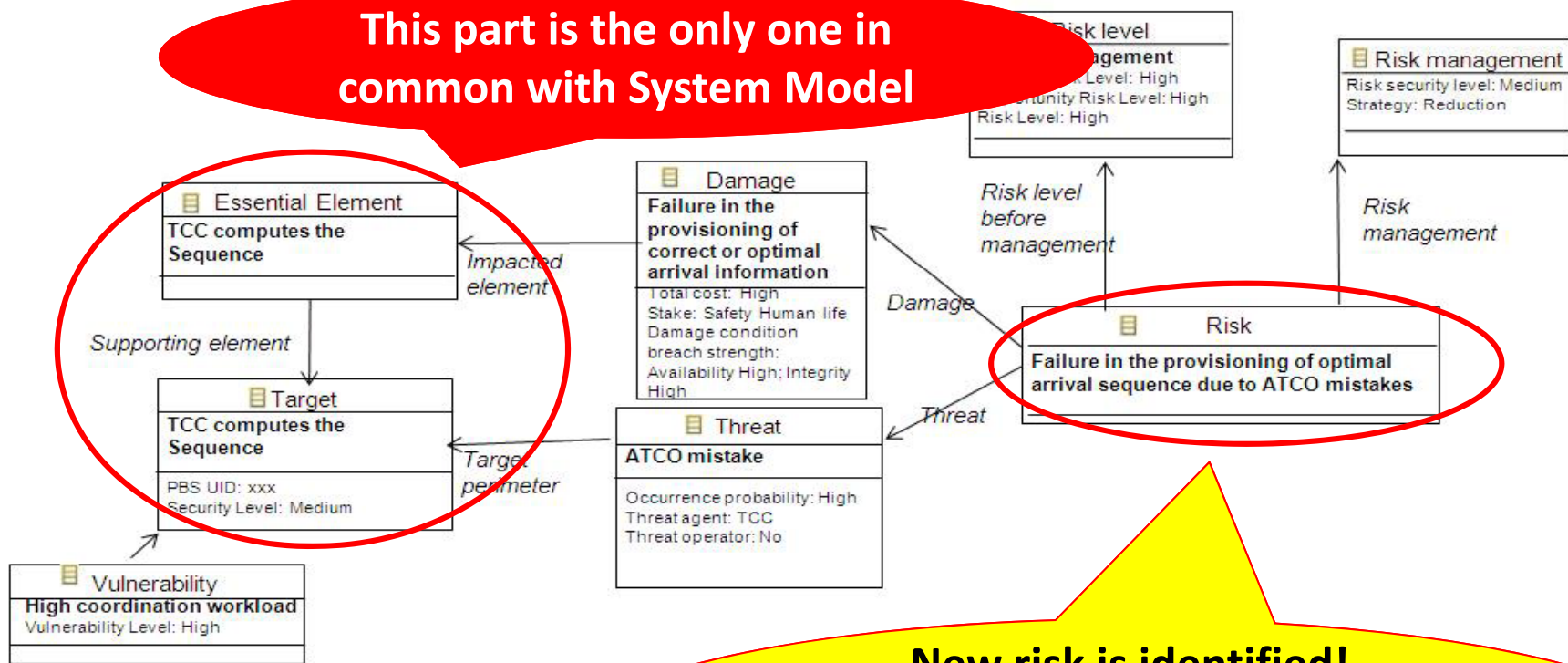
RISK ASSESSMENT VIEWPOINT

□ Risk Assessment DSML capture risk analysis concepts



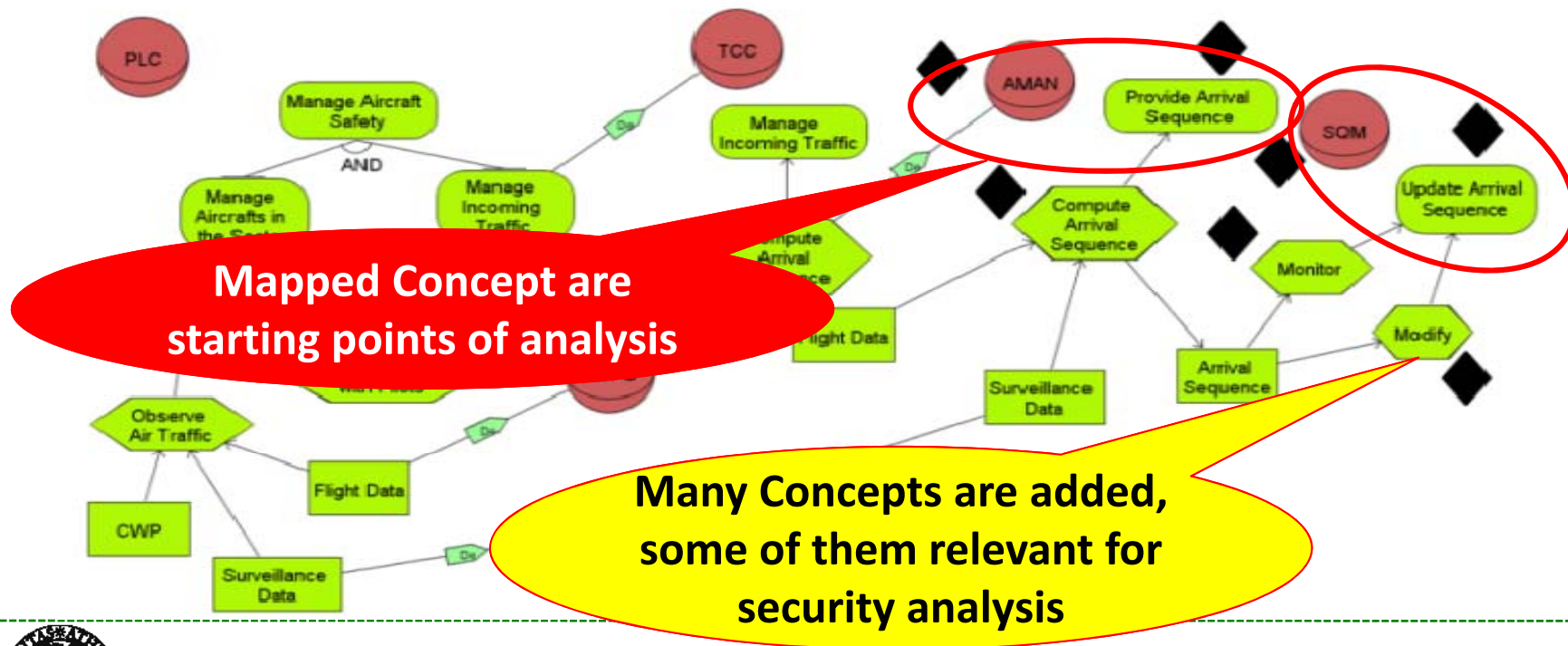
ORCHESTRATED CHANGE MANAGEMENT PROCESS

1. Proposed Solution is sent to Risk Management
2. Risk analyst analyze the model



ORCHESTRATED CHANGE MANAGEMENT PROCESS III

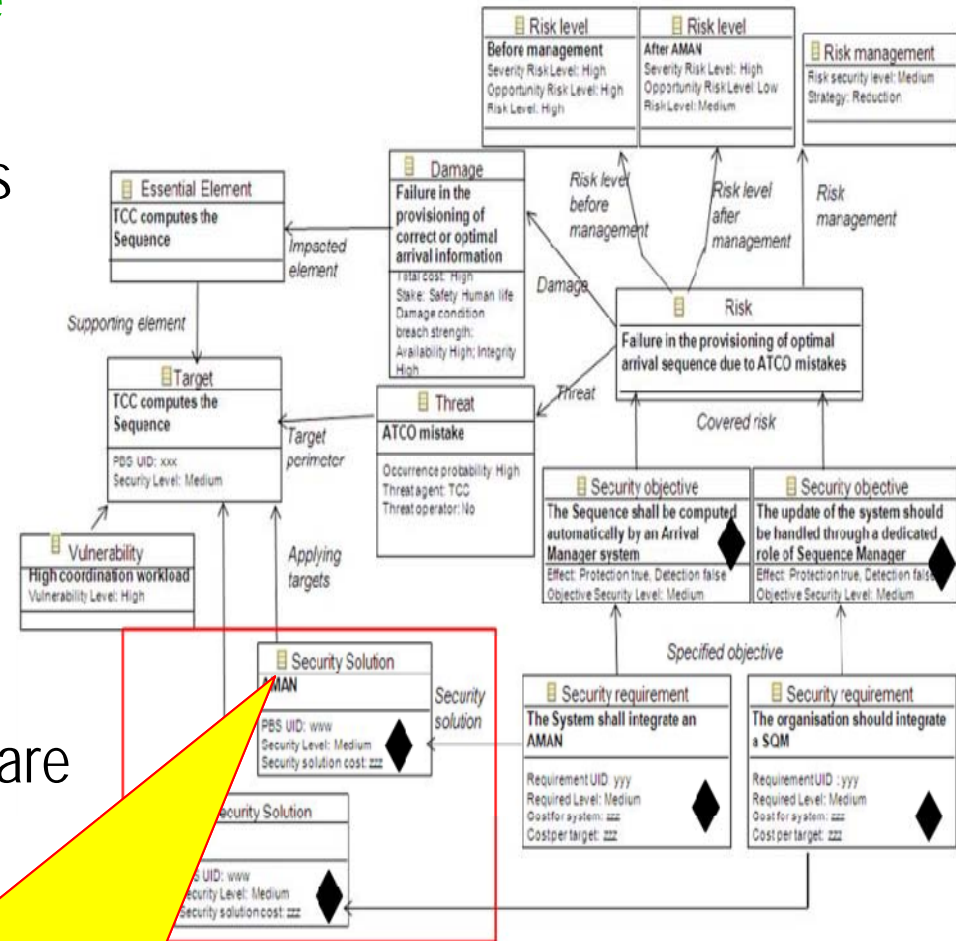
- 5. The requirements analyst updates the SI* model based on the new security objectives
- 6. Provide implementation of security solutions



ORCHESTRATED CHANGE MANAGEMENT PROCESS IV

❑ Process is now back to the risk analyst

- ✓ Receives the mapped concepts and might introduce new concepts to adapt the risk analysis
- ✓ Requirements Engineer can happily ignore what's going on
- ✓ ... If risk mitigation is sufficient
- ✓ ... else if new security controls are necessary he can be involved again



Risk is covered as the target now includes the security solutions

CONCLUSIONS

❑ A change management framework for security engineering processes

- ✓ Requirement, Risk, Design, Service Deployment, Testing, Verification
- ✓ Coding only one we are missing

❑ The framework is based on two ideas

- ✓ Reason about the Δ s, don't start from scratch
- ✓ Separation of concerns
- ✓ Orchestration of changes

❑ Open issues

- ✓ Does it really scale to industrial practice?
- ✓ Currently well evaluated in the domain of organizational changes for Air Traffic Management, code changes in Smart-Card certification

