Computer Science
Technical Report

USE Tool Analysis of Activity Theory Models

Geri Georg[1] and Gunter Mussbacher[2]

[1]Computer Science Department, Colorado State University, Fort Collins, Colorado

[2]School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Canada

georg@cs.colostate.edu,
gunterm@eecs.uottawa.ca

March 13, 2013

Technical Report CS-13-102

Computer Science Department
Colorado State University
Fort Collins, CO 80523-1873

Phone: (970) 491-5792 Fax: (970) 491-2466
WWW: http://www.cs.colostate.edu

USE Tool Analysis of Activity Theory Models

Geri Georg[1] & Gunter Mussbacher[2]
March 2013

Introduction

This document contains AT, GRL, URN, and USE models and analysis results for the Premises Surveillance portion of the vector-borne disease example program. A simplified version of the meta-model we have defined for an Activity Theory language, along with its relations and OCL constraints, is also included. The purpose of this AT DSL is for use in the requirements elicitation process, and also during evolution of a system that is already in place.

We envision that a requirements engineer or system analyst will create Activity Theory System Diagrams (ASDs) from stakeholder input. The basic form of an ASD is taken from [Engeström]. The elements and their relations as defined in this document can be used to determine whether such an ASD is well-formed, through structural validation such as is provided by the USE tool. Missing relations can be used to decide where further input is needed from stakeholders to resolve vagueness or contradictory information. Once ASDs are defined, trace-link mappings can be used to transform the ASD into a URN [ITU-T] goal model (using the GRL notation) in the jUCMNav tool [jUCMNav]. Goal and trade-off analyses are available in jUCMNav and these can be used to further refine the goal model. jUCMNav supports powerful tracing capabilities, that can be used when high level designs (as URN use case maps) are created from the goal model.

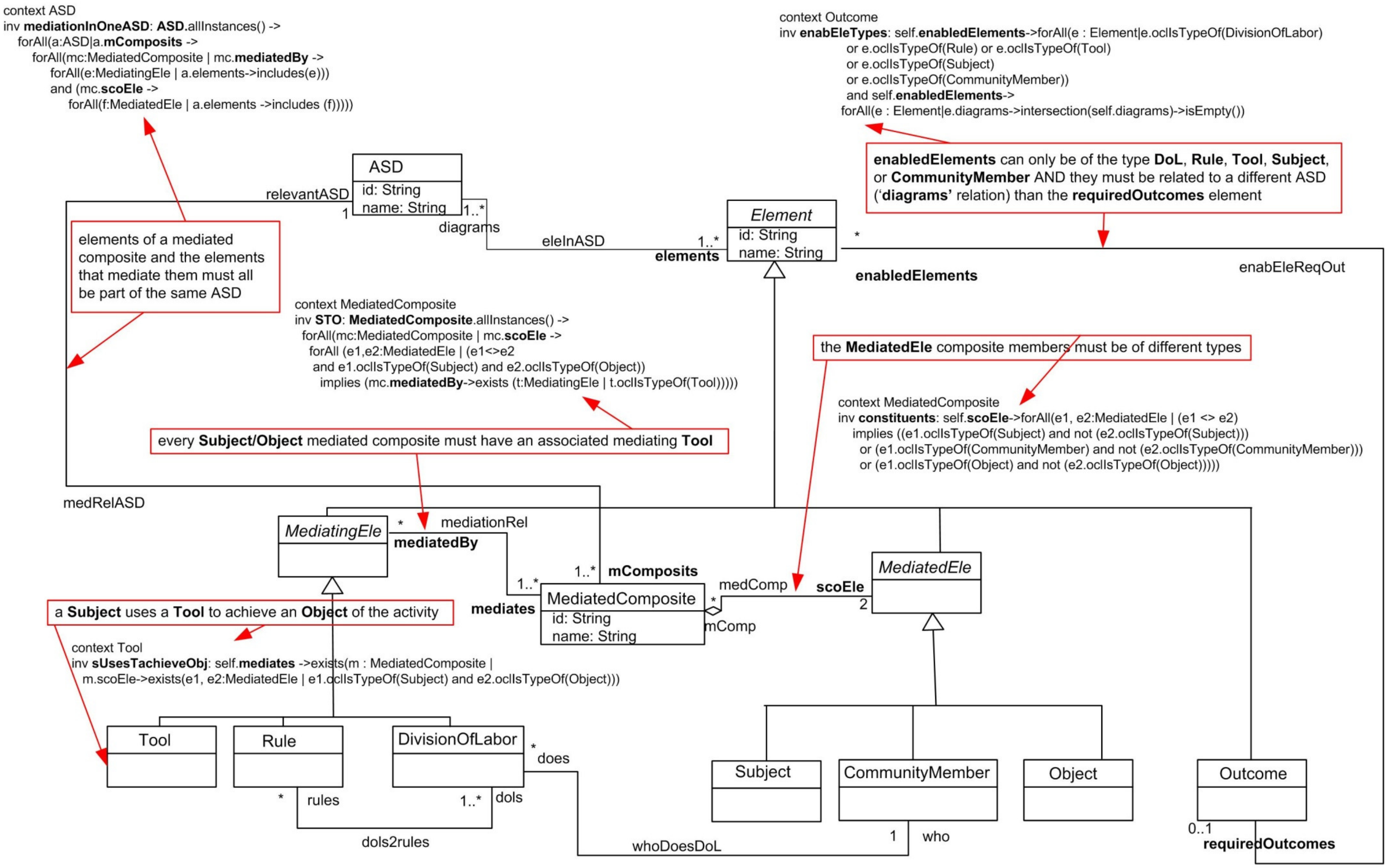
When a system is evolved, if the designs are specified with URN use case maps, then we can utilize the trace-links to re-construct equivalent ASDs. Once again these ASDs can be analyzed with the USE tool, and inconsistencies identified. Requirements Engineers may choose to resolve some inconsistencies themselves, while others may require additional stakeholder dialog. Beyond this resolution, the evolved ASDs can be used as the basis for dialog to determine if the constraints they show are still valid. Irrelevant constraints and the designs they dictate can be abandoned with impunity. We believe that knowing what parts of a system design can be changed freely will be of great benefit to designers, and help in situations where the reasons for a design decision are buried in time and everyone is afraid to change the design because of potential unforeseen consequences.

The example system used in this document revolves around the Dengue virus data capture and interpretation system that was developed between the Microbiology, Immunology, and Pathology department at CSU, and a University and Public Health Ministry in Mexico. A portion of this was field tested in the city of Mérida, Yucatán. The ASDs used in this document describe the context of the vector surveillance portion of this system. Vector surveillance entails looking for potential breeding sites of the mosquito that carries Dengue, *Aedes aegypti*, in residential areas. The related ASD is the *Survey Premises* ASD. It is an ASD at a rather low level of abstraction, so its scope is quite limited. It makes a good example since it relies on other activities to provide tools essential to achieving it, and also because it was derived from activities that take place at a higher level of abstraction. This activity was the target of a technical solution, a cell phone application that field agents can take to residences and use to directly enter data regarding how many and of what type of breeding sites are found. Data is uploaded by this application to a centralized database where it is available for interpretation and policy decision-making.

An AT Language Meta-model: Elements of an ASD, their Relations, and Networked Systems

ASDs can be networked together (via their Outcomes) to provide elements of other ASDs. These relations are included in the meta-model shown in Figure 1. Definitions and constraints are presented in natural language (outlined in red), with red arrows pointing to the equivalent OCL and related portions of the meta-model. The AT language also supports hierarchical decomposition, but for simplification this relation is not shown in Figure 1.

[1] Computer Science Department, Colorado State University, Fort Collins, USA
[2] School of Electrical Engineering and Computer Science, University of Ottawa, Ottawa, Canada



```
context ASD
inv mediationInOneASD: ASD.allInstances() ->
    forAll(a:ASD|a.mComposits ->
        forAll(mc:MediatedComposite | mc.mediatedBy ->
             forAll(e:MediatingEle | a.elements->includes(e)))
             and (mc.scoEle ->
                  forAll(f:MediatedEle | a.elements ->includes (f)))))
```

elements of a mediated composite and the elements that mediate them must all be part of the same ASD

```
context Outcome
inv enabEleTypes: self.enabledElements->forAll(e : Element|e.oclIsTypeOf(DivisionOfLabor)
                  or e.oclIsTypeOf(Rule) or e.oclIsTypeOf(Tool)
                  or e.oclIsTypeOf(Subject)
                  or e.oclIsTypeOf(CommunityMember))
                  and self.enabledElements->
                 forAll(e : Element|e.diagrams->intersection(self.diagrams)->isEmpty())
```

enabledElements can only be of the type **DoL**, **Rule**, **Tool**, **Subject**, or **CommunityMember** AND they must be related to a different ASD (‘**diagrams’** relation) than the **requiredOutcomes** element

```
context MediatedComposite
inv STO: MediatedComposite.allInstances() ->
  forAll(mc:MediatedComposite | mc.scoEle ->
   forAll (e1,e2:MediatedEle | (e1<>e2
    and e1.oclIsTypeOf(Subject) and e2.oclIsTypeOf(Object))
     implies (mc.mediatedBy->exists (t:MediatingEle | t.oclIsTypeOf(Tool)))))
```

every **Subject/Object** mediated composite must have an associated mediating **Tool**

the **MediatedEle** composite members must be of different types

```
context MediatedComposite
inv constituents: self.scoEle->forAll(e1, e2:MediatedEle | (e1 <> e2)
   implies ((e1.oclIsTypeOf(Subject) and not (e2.oclIsTypeOf(Subject)))
    or (e1.oclIsTypeOf(CommunityMember) and not (e2.oclIsTypeOf(CommunityMember)))
    or (e1.oclIsTypeOf(Object) and not (e2.oclIsTypeOf(Object)))))
```

a **Subject** uses a **Tool** to achieve an **Object** of the activity

```
context Tool
inv sUsesTachieveObj: self.mediates ->exists(m : MediatedComposite |
  m.scoEle->exists(e1, e2:MediatedEle | e1.oclIsTypeOf(Subject) and e2.oclIsTypeOf(Object)))
```

Figure 1. ASD elements, decomposition, networks

Initial Vector Surveillance ASD Network

The Survey Premises, or *SP* ASD, is based on the cell phone application. It relies on another ASD in a network to provide tools – in particular a coordinator activity (Assign Surveillance Tasks to Personnel, or *ASTP*) that creates field agent premises surveillance lists.

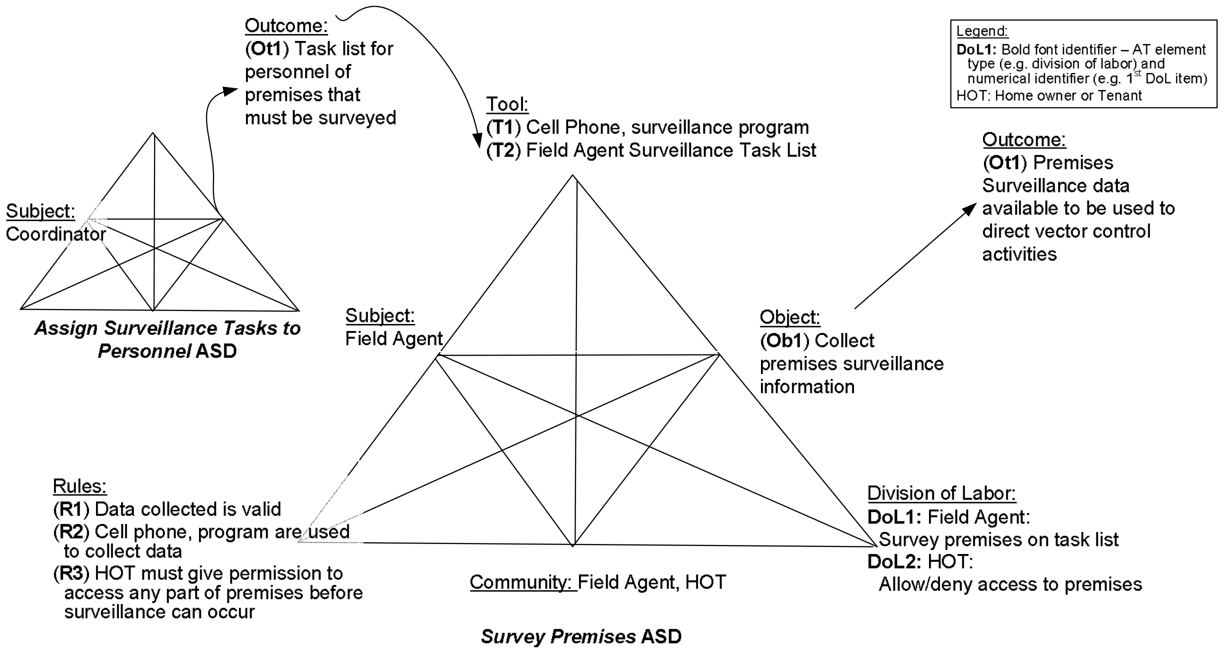

Outcome:
(Ot1) Task list for personnel of premises that must be surveyed
Subject:
Coordinator
Assign Surveillance Tasks to Personnel ASD
Tool:
(T1) Cell Phone, surveillance program
(T2) Field Agent Surveillance Task List
Legend:
DoL1: Bold font identifier – AT element type (e.g. division of labor) and numerical identifier (e.g. 1st DoL item)
HOT: Home owner or Tenant
Outcome:
(Ot1) Premises Surveillance data available to be used to direct vector control activities
Subject:
Field Agent
Object:
(Ob1) Collect premises surveillance information
Rules:
(R1) Data collected is valid
(R2) Cell phone, program are used to collect data
(R3) HOT must give permission to access any part of premises before surveillance can occur
Community: Field Agent, HOT
Division of Labor:
DoL1: Field Agent:
Survey premises on task list
DoL2: HOT:
Allow/deny access to premises
Survey Premises ASD

Figure 2. Vector Surveillance ASD network

Figure 2 shows all the elements of the *SP* ASD, but only the subject and outcome elements of the *ASTP* ASD for simplification purposes. The network relation between the two ASDs is shown by the outcome of the *ASTP* activity being used as a tool in the *SP* activity. The *SP* ASD has a single subject, the field agent, who achieves the object of collecting premises data through the use of two mediating tools, the cell phone application and a list of premises to be surveyed. The object is transformed into the outcome. The work associated with the activity is divided into two parts – the field agent performs the actual survey, and the home owner or tenant living at the premises gives permission for the field agent to enter their residence to perform it. Three rules are shown for this activity.

Transformation to User Requirements Notation (URN)

We use trace-links [Paige] to map from the AT meta-model to the Goal Requirements Language (GRL) meta-model of URN. Trace-links allow models developed in a modeling language that differs significantly from another to be transformed into a model in that other language. The types of links that are needed must be identified between the relevant portions of the meta-models, and must also be enriched with constraints that need to hold across models being mapped from one meta-model to the other. Figure 3 shows an example of such trace-links between AT and GRL.

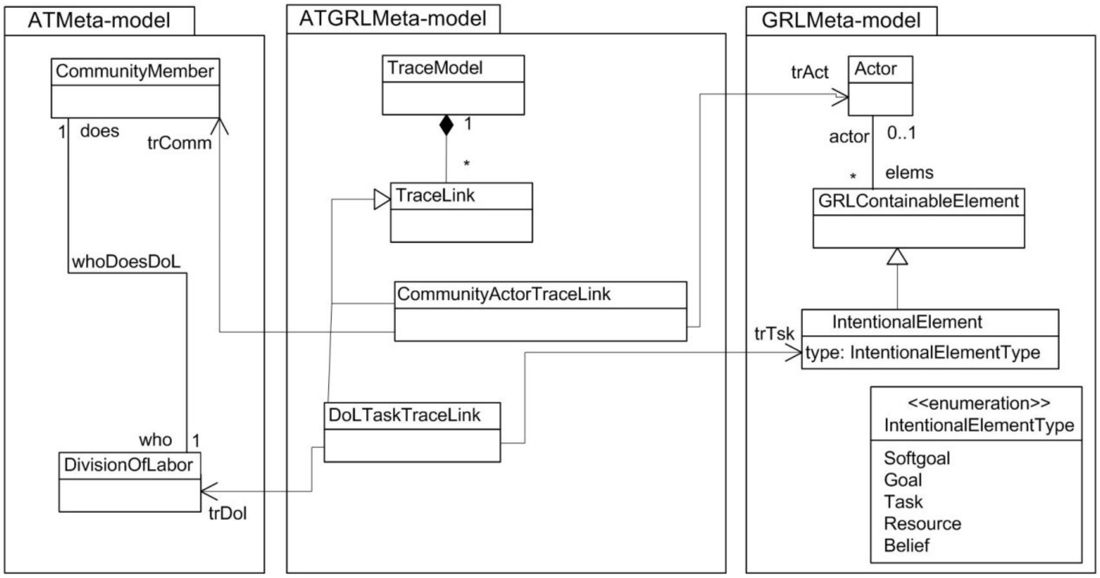


Figure 3. Trace-link specification from AT to GRL.

The transformations we use are that AT community members (subjects are also community members) become GRL actors and division of labor items in general become task intentional elements. Rules, objects, and outcomes are transformed into goals or softgoals, depending on how subjective the conditions to meet them are defined. Usually tools are transformed into GRL resources.

Initial Vector Surveillance Network Goal Model and Use Case Map Design

Figure 4 shows the initial goal model for the ASD network from Figure 2.

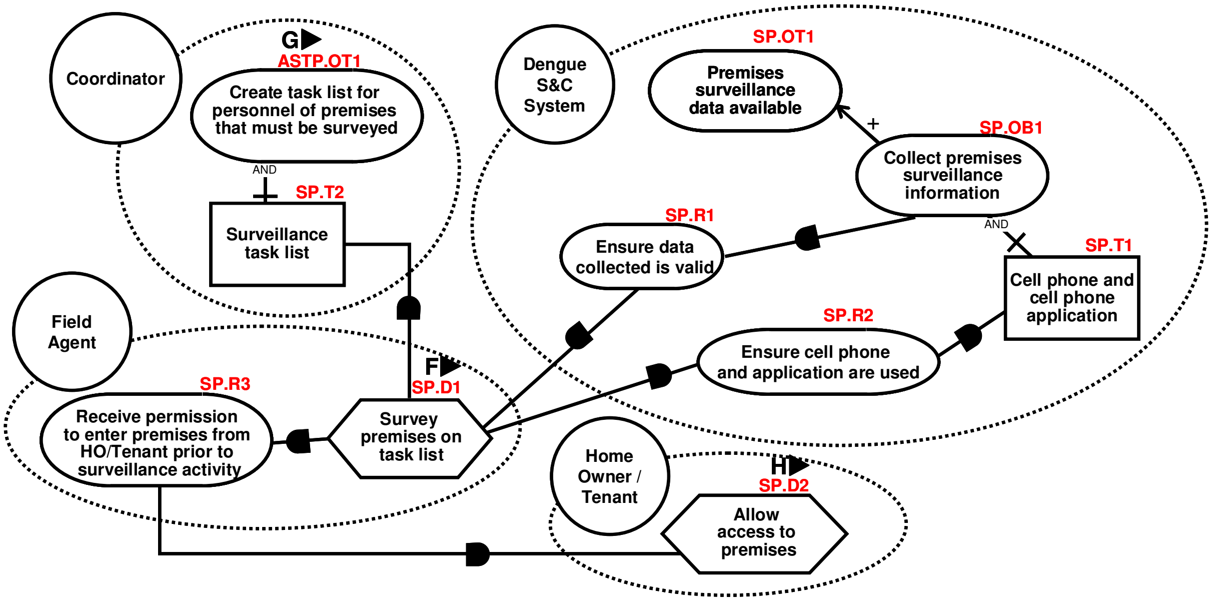

Coordinator
G
ASTP.OT1
Create task list for personnel of premises that must be surveyed
AND
SP.T2
Surveillance task list
Dengue S&C System
SP.OT1
Premises surveillance data available
+
SP.OB1
Collect premises surveillance information
SP.R1
Ensure data collected is valid
AND
SP.T1
Cell phone and cell phone application
SP.R2
Ensure cell phone and application are used
Field Agent
SP.R3
Receive permission to enter premises from HO/Tenant prior to surveillance activity
F
SP.D1
Survey premises on task list
Home Owner / Tenant
H
SP.D2
Allow access to premises

Figure 4. Goal Model derived from ASD network

The goal model shows several stakeholders as *actors* (⬭, e.g., Dengue S&C System or Coordinator). The system stakeholder is implied by the ASD network as it is about a system that is either in place or to be put in place as hinted by the cell phone and surveillance program tools. The stakeholders *depend* (—■—) on each other which is typically expressed by rules in an ASD, e.g., the system depends on the field agent's *task* (⬭) "Survey premises…" to result in valid data (SP.R1) in order to achieve its "Collect premises…" *goal* (⬭). At the top of the goal model in the system actor, the "Premises surveillance..." goal corresponds to an outcome in the ASD (i.e., "Premises Surveillance data…"). This high-level goal is further refined into the object from the ASD with the help of a *contribution* (→), i.e., the object contributes positively (+) to the outcome. Eventually, at the lower levels of the goal model, tasks represent the division of labor (DoL) items from the ASD (e.g., the DoL1 and DoL2). DoL items are assigned to the community member to which they belong. Finally, tools are represented by resources (□, e.g., Surveillance task list) which may be a decomposition (+—) of a goal, i.e., they are an integral part of the goal.

The design created from this goal model includes the UCM scenario for the field agent performing premises surveillance as shown in Figure 5. *URN links* (▶) connect the goal model in Figure 4 with scenario model elements in Figure 5, e.g., the allow access to premises task of the Home Owner/Tenant relates to the deny access and grant access *responsibilities* (✕) of the Home Owner/Tenant *component* (□).

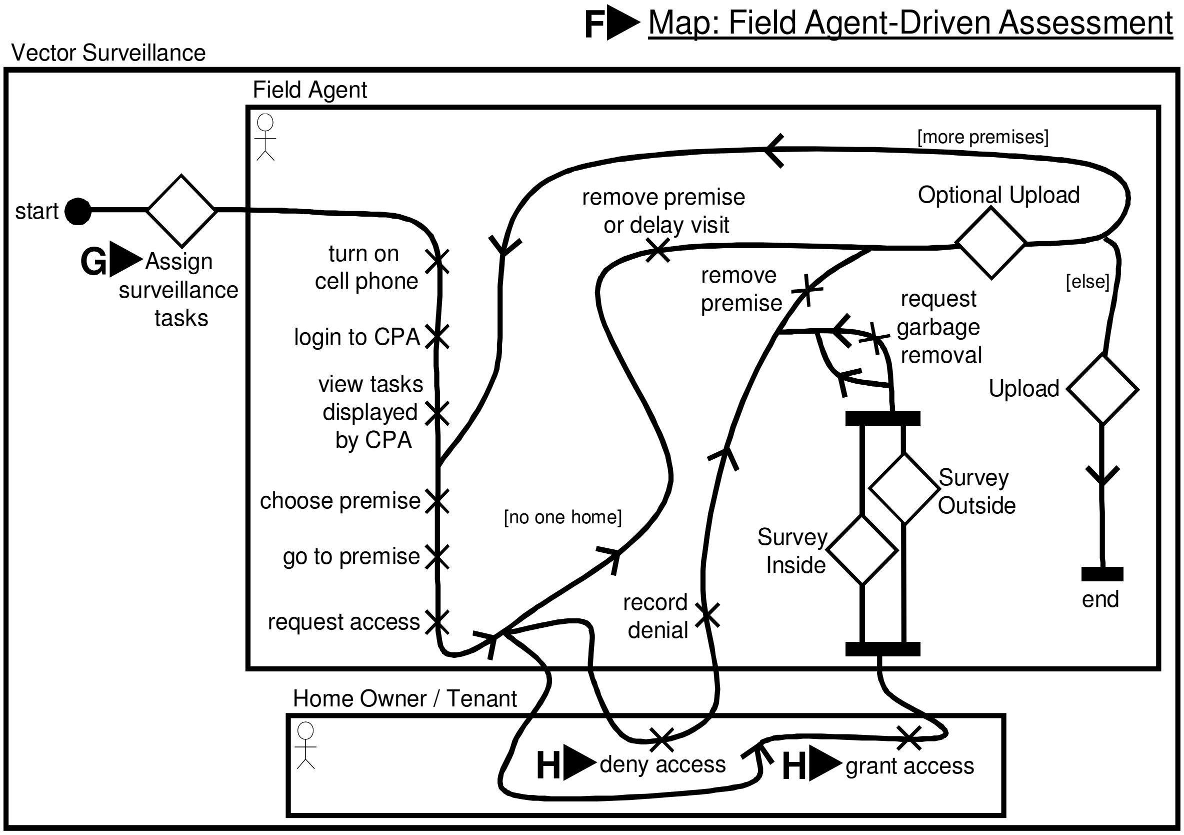


Figure 5. UCM for field agent premises surveillance activity

Evolved UCM Design and Related Goal Model

The design of premise surveillance can be evolved to include self-surveillance on the part of home owners and tenants (HOT). This addition is hypothesized to lead to an increase in the valid surveillance data that can be collected, over that available from field agent surveillance only. This is an issue that was identified during field testing of the cell phone application although it is by no means unique to that application. Less than 35% of all premises surveillances attempted were completed, either because no one was home to give permission to the field agents to enter or because permission was denied [Lozano-Fuentes]. The UCM design for the new capability is shown in Figure 6.

The UCM scenario in Figure 6 has three maps. The left map shows that, initially, a coordinator must provide permission for a self-survey. If this survey is the first for a HOT, then the HOT needs to be trained. Otherwise, the survey may go ahead. If, however, there are problems additional training may be required.

Training is shown on the right map. It must be scheduled by the coordinator, and a field agent must be available to provide training. It is dependent on materials that the field agent can use to show the HOT how to perform the surveillance, and that can be left for future surveys. Thus, an entomologist must be available to create such materials.

A variation of the existing cell phone application must be available to download and use for self-surveys. When this application is available, the HOT can perform the survey. The field agent will provide guidance and check results until the validity of the data is acceptable or a HOT is blocked from performing self-surveys. Valid data includes the number of containers with water in them, the subset that has target larvae, and the subset that has target pupae. Data can be uploaded as in the Field Agent variation of surveillance. The application will then be deactivated or removed until it is needed again, and the field agent will report that the HOT has been successfully trained.

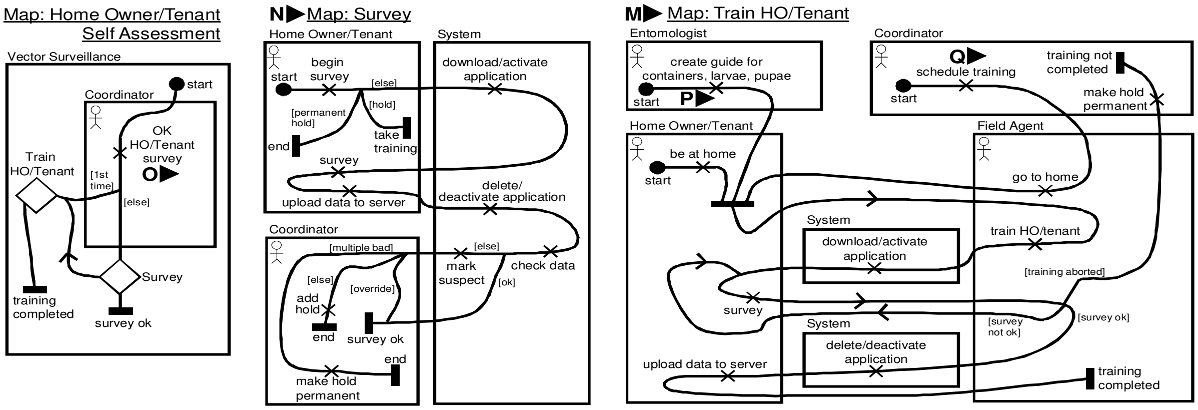


Figure 6. UCM representing design change to allow home owner self-survey

Permission for subsequent self-surveys as shown on the middle map must again be given by the coordinator. Unless the HOT has been restricted, the application must be obtained or reactivated for the HOT to perform the survey and upload the data to the central server where checks can occur to assure data validity. In the case of problems, the coordinator will be alerted and the data marked as suspect. The coordinator can then allow the survey result to be taken into account, enforce retraining, or simply deny the HOT further self-surveillance capabilities.

Goal Model Associated with Evolved System

New elements must be added to the goal model, which are associated with the modified UCM. While tracing built into jUCMNav can be used heuristically to determine where new elements should be placed, the analyst must make final decisions regarding these additions. For example, a new task called

"Perform self survey" can be added to the HOT actor. However, the new task "Create larvae/pupae guide" entails adding another actor to the goal model, the entomologist, who was created as a new component in the evolved UCM. Various softgoals, goals, and additional tasks must be added to the goal model, which is shown in Figure 7. Many of these goals and softgoals may now be used in goal analysis, to determine if the proposed design changes are worth their cost.

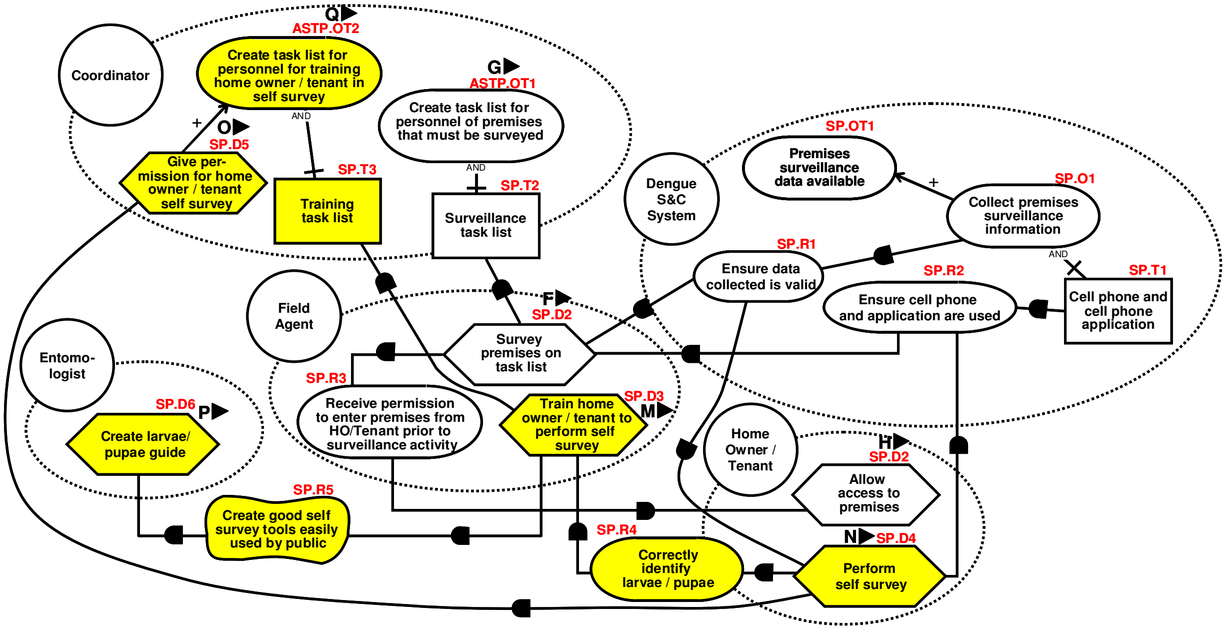

Q
ASTP.OT2
Coordinator
Create task list for personnel for training home owner / tenant in self survey
AND
G
ASTP.OT1
Create task list for personnel of premises that must be surveyed
+
O
SP.D5
Give permission for home owner / tenant self survey
SP.T3
Training task list
AND
SP.T2
Surveillance task list
SP.OT1
Premises surveillance data available
+
SP.O1
Collect premises surveillance information
Dengue S&C System
SP.R1
Ensure data collected is valid
AND
SP.R2
Ensure cell phone and application are used
SP.T1
Cell phone and cell phone application
Field Agent
F
SP.D2
Survey premises on task list
Entomologist
SP.R3
Receive permission to enter premises from HO/Tenant prior to surveillance activity
Train home owner / tenant to perform self survey
SP.D3
M
SP.D6
P
Create larvae/ pupae guide
Home Owner / Tenant
H
SP.D2
Allow access to premises
SP.R5
Create good self survey tools easily used by public
SP.R4
Correctly identify larvae / pupae
N
SP.D4
Perform self survey

Figure 7. Modified goal model to include home owner self-survey; new elements are shaded

Evolved ASD Network

The AT/URN trace-links can now be used to create an evolved ASD network from the evolved goal model. The evolved ASD network is presented in Figure 8.

The items in normal font in Figure 8 result from transformation back to AT from GRL. Items in bold font are items that were omitted from the GRL model, but were identified through analysis of the ASD network with the USE tool or follow-up discussions with stakeholders (i.e., T4, R6, and R7). Note that we neglected to add the entomologist to the community based on the goal model, and this was discovered by the USE tool. The item in strikethrough font is a rule (R5) that either does not belong in this ASD or is missing a related DoL. The issues with T4 and R5 were identified through analysis using the USE tool, while those related to R6 and R7 can only be identified though stakeholder discussion, as directed by this ASD. These analyses are discussed below.

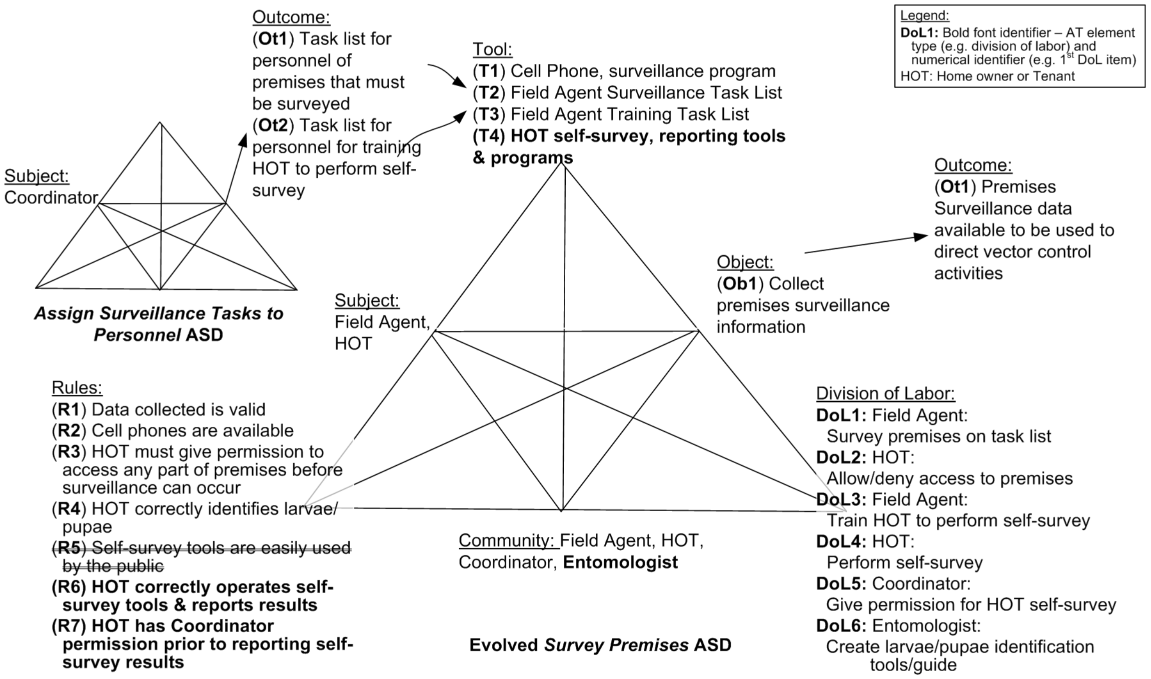

Outcome:
(Ot1) Task list for personnel of premises that must be surveyed
(Ot2) Task list for personnel for training HOT to perform self-survey
Subject:
Coordinator
Assign Surveillance Tasks to Personnel ASD
Tool:
(T1) Cell Phone, surveillance program
(T2) Field Agent Surveillance Task List
(T3) Field Agent Training Task List
(T4) HOT self-survey, reporting tools & programs
Legend:
DoL1: Bold font identifier – AT element type (e.g. division of labor) and numerical identifier (e.g. 1st DoL item)
HOT: Home owner or Tenant
Outcome:
(Ot1) Premises Surveillance data available to be used to direct vector control activities
Object:
(Ob1) Collect premises surveillance information
Subject:
Field Agent, HOT
Rules:
(R1) Data collected is valid
(R2) Cell phones are available
(R3) HOT must give permission to access any part of premises before surveillance can occur
(R4) HOT correctly identifies larvae/pupae
~~(R5) Self-survey tools are easily used by the public~~
(R6) HOT correctly operates self-survey tools & reports results
(R7) HOT has Coordinator permission prior to reporting self-survey results
Community: Field Agent, HOT, Coordinator, Entomologist
Evolved Survey Premises ASD
Division of Labor:
DoL1: Field Agent:
Survey premises on task list
DoL2: HOT:
Allow/deny access to premises
DoL3: Field Agent:
Train HOT to perform self-survey
DoL4: HOT:
Perform self-survey
DoL5: Coordinator:
Give permission for HOT self-survey
DoL6: Entomologist:
Create larvae/pupae identification tools/guide

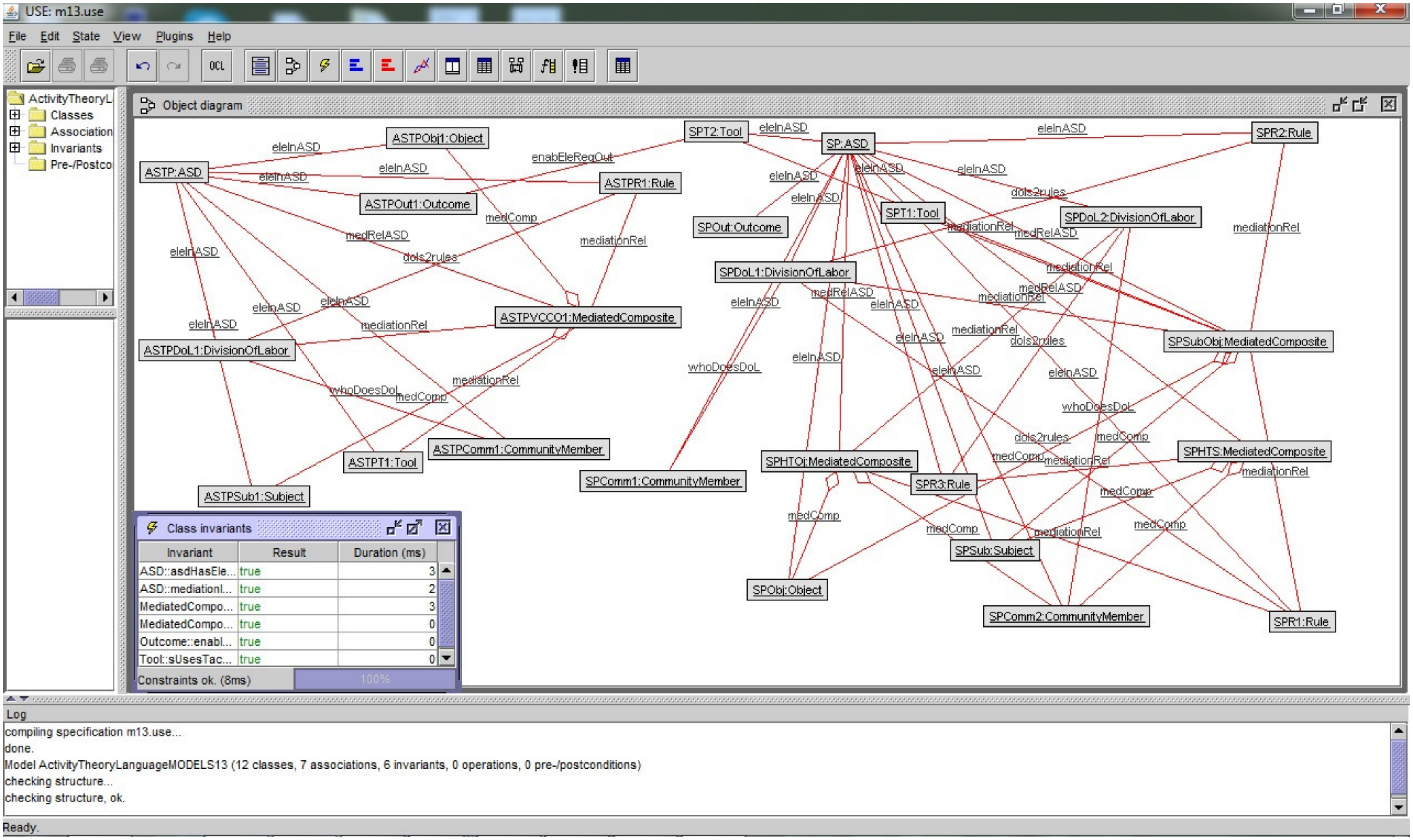
Figure 8. Evolved ASD Network.

USE Analysis: Original ASD Network

We first present the results of the USE analysis of the original vector surveillance ASD network shown in Figure 2. Figure 9 shows a screen shot of the USE tool after the AT meta-model has been loaded as a USE model, followed by instance creation of the *ASTP* and *SP* ADSs, and the network connection between them. The main window shows the object model of the ASD network. Objects are displayed as rectangles labeled with an identifier and class (e.g., ASTP:ASD at the upper left of the object view). Association instances are shown as links between objects, labeled with the association name (e.g., eleInASD between ASTP and ASTPObj1, located above and to the right of the ASTP object). Composite relations are also shown (e.g. ASTPObj1 is a member of the MediatedComposite called ASTPVCCO1, along with ASTPSub1 which is located at the lower left of the object diagram).

The result of the invariant evaluations are shown in the smaller window located in the same area as the object view, and called 'Class invariants'. All of the constraints evaluate to TRUE. Finally, the Log area below the main window shows messages from the tool and also the results of structural checks.

Note that the structural and constraint analyses do not identify any problems in the model.



Invariant	Result	Duration (ms)
ASD::asdHasEle...	true	3
ASD::mediationI...	true	2
MediatedCompo...	true	3
MediatedCompo...	true	0
Outcome::enabl...	true	0
Tool::sUsesTac...	true	0

Constraints ok. (8ms) 100%

```
compiling specification m13.use...
done.
Model ActivityTheoryLanguageMODELS13 (12 classes, 7 associations, 6 invariants, 0 operations, 0 pre-/postconditions)
checking structure...
checking structure, ok.
```

Ready.

Figure 9. USE output of original ASD network

USE Analysis: Evolved ASD Network

We next present the USE analysis of the evolved network, as shown in Figure 10. The USE tool was restarted, the AT USE model was loaded and the evolved instances of the two ASDs and the network relations between them were loaded.

There is an additional network relation between the two ASDs, from the training list outcome of ASTP to the third tool of the SP ASD. Once again the results of constraint testing are also shown. This time however, USE analysis output shows errors in the structural check and also in the constraint checks.

There are 4 log errors shown in the USE window:

1. Rule 5 of the *SP* ASD (Self-survey tools are easily used by the public) is not related to any DoL through the *dols2rules* association. R5 may therefore be an over-specification of the *SP* ASD. If it is not, then there is a missing DoL. This rule probably belongs in an entirely different ASD, but one that is networked to the *SP* ASD through an outcome of producing a tool for self-surveillance linking to the T4 tool of the evolved SP ASD.

2. DoL 6 (Entomologist creates larvae/pupae identification tools/guide) is not related to any mediated composite through the *mediationRel* association. Similar to R5, this DoL may not belong in the *SP* ASD, but if the requirements engineer decides that it does belong in this ASD then the relation needs to be created since every DoL should be a mediating element between two mediated elements. In this case the mediating elements are the HOT subject and the object, and they are mediated by this DoL.

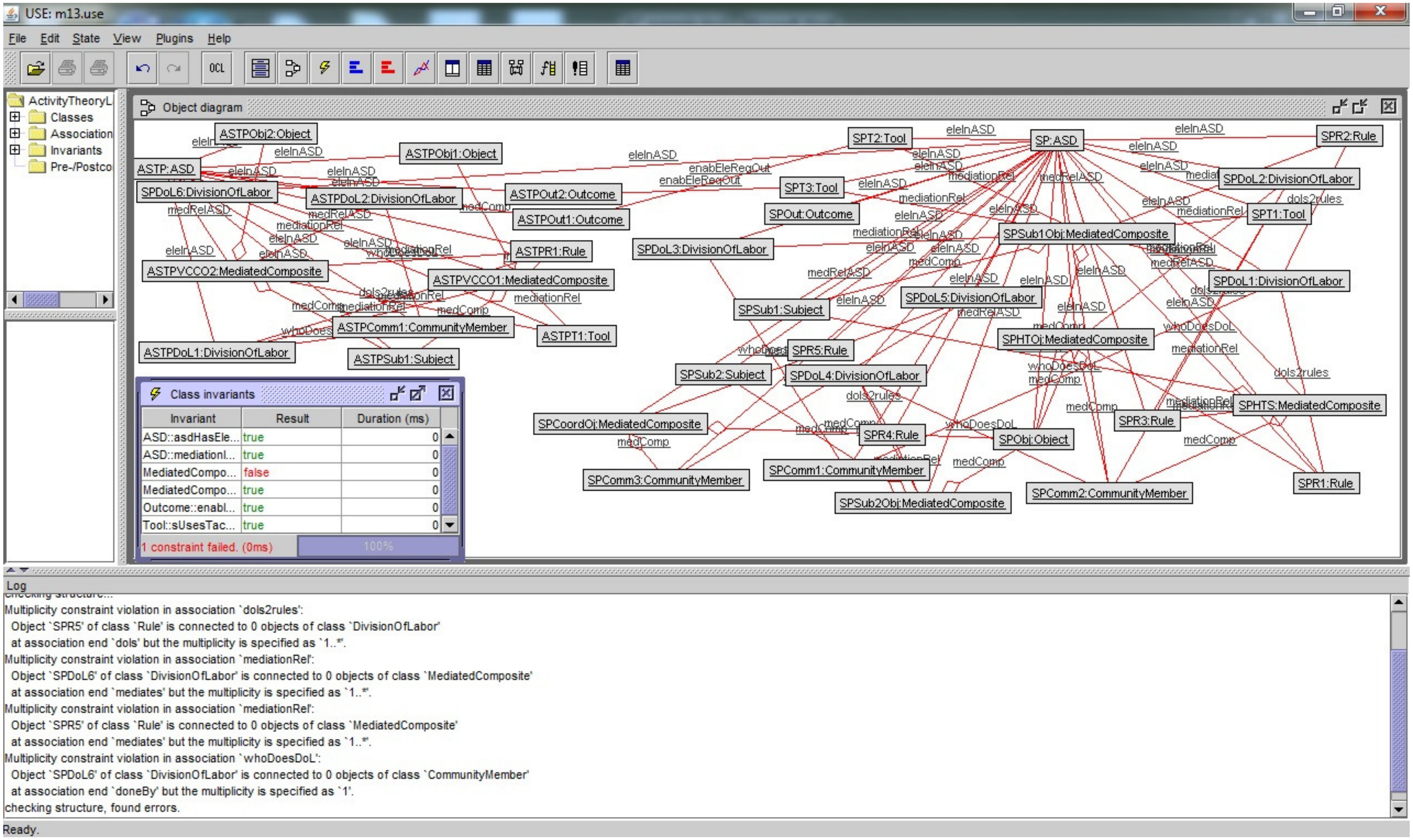
3. Rule 5 is also not involved in any mediating relation. Similar to the previous error with this rule, a Requirements Engineer might choose to relate it to the HOT subject and the activity object, or to decide that it does not belong in the *SP* ASD and remove it.

4. DoL 6 is not related to any community member through the *whoDoesDol* association. This error indicates that the Entomologist is really part of the community and needs to be added as a community member, and the relation created between this new community member and the DoL item.

There is one constraint that fails, which is shown in the Class Invariants window. This is the STO constraint which states that every subject-object mediated composite must be mediated by at least one tool. In this case, the HOT subject and object of the activity have no relation with any tool. Therefore, the ASD is missing a tool, T4 (HOT self-survey, reporting tools & programs).

While Figure 8 shows two additional Rules (R6 and R7), they cannot be identified through the USE analysis. There are no general constraints that can be added to the metamodel that could find these omissions. Instead, they need to be identified through discussions with the (now augmented) stakeholder members of the community. This is one of the strengths of using the AT framework, concepts, and notations – to provide a vehicle for discussions that identify rules such as these and make them explicit. A benefit of making these rules explicit is that in future evolution discussion with stakeholders they can be tested for continued relevance. Just as in the ASD of Figure 8, all the previous elements of the ASD network can be reviewed for relevance and constraints no longer needed, along with associated designs that realize them, can be changed without consequences.

Furthermore, validated changes to the ASD network because of the results of the analysis with the USE tool and the stakeholder discussions need to be retransformed back into the goal model to ensure that the design is not missing anything or has elements in it that cannot be traced to the ASD network.



Invariant	Result	Duration (ms)
ASD::asdHasEle...	true	0
ASD::mediationI...	true	0
MediatedCompo...	false	0
MediatedCompo...	true	0
Outcome::enabl...	true	0
Tool::sUsesTac...	true	0

1 constraint failed. (0ms) 100%

```
checking structure...
Multiplicity constraint violation in association `dols2rules':
  Object `SPR5' of class `Rule' is connected to 0 objects of class `DivisionOfLabor'
  at association end `dols' but the multiplicity is specified as `1..*'.
Multiplicity constraint violation in association `mediationRel':
  Object `SPDoL6' of class `DivisionOfLabor' is connected to 0 objects of class `MediatedComposite'
  at association end `mediates' but the multiplicity is specified as `1..*'.
Multiplicity constraint violation in association `mediationRel':
  Object `SPR5' of class `Rule' is connected to 0 objects of class `MediatedComposite'
  at association end `mediates' but the multiplicity is specified as `1..*'.
Multiplicity constraint violation in association `whoDoesDoL':
  Object `SPDoL6' of class `DivisionOfLabor' is connected to 0 objects of class `CommunityMember'
  at association end `doneBy' but the multiplicity is specified as `1'.
checking structure, found errors.
```

Figure 10. USE output of evolved ASD network

USE Models

The following sections contain listings of the complete USE models and commands used in the analysis of the vector surveillance ASD network. USE requires a model to be specified and loaded into the tool, and then object models can be created through a graphical or textual interface using commands. We have created textual versions of the model and command files for simplicity.

AT Meta-model as a USE Model

The first listing gives the USE model for the AT meta-model. This defines all the classes, associations, and invariants in the meta-model.

```
model ActivityTheoryLanguage
-- This is a USE model for an activity theory language

class Tool < MediatingEle
attributes
operations
end

class Rule < MediatingEle
attributes
operations
end

class DivisionOfLabor < MediatingEle
attributes
operations
end

class Subject < MediatedEle
attributes
operations
end

class CommunityMember < MediatedEle
attributes
operations
end

class Object < MediatedEle
attributes
operations
end

class Outcome < Element
attributes
operations
end

abstract class MediatingEle < Element
attributes
operations
end

abstract class MediatedEle < Element
attributes
operations
end

abstract class Element
attributes
operations
end

class MediatedComposite
attributes
operations
end

class ASD
attributes
operations
end

aggregation medComp between
MediatedComposite[0..*] role mComp
MediatedEle[2] role scoEle
end

association mediationRel between
MediatingEle[0..*] role mediatedBy
MediatedComposite[1..*] role mediates
end

association enabEleReqOut between
Element[0..*] role enabledElements
Outcome[0..1] role requiredOutcomes
end

association eleInASD between
ASD[1..*] role diagrams
Element[7..*] role elements
end

association medRelASD between
ASD[1] role relevantASD
MediatedComposite[1..*] role mComposits
end

association dols2rules between
DivisionOfLabor[1..*] role dols
Rule[0..*] role rules
end
```

```

association whoDoesDoL between
DivisionOfLabor[0..*] role commDoes
CommunityMember[1..1] role doneBy
end

constraints

context Outcome
inv enabledEleTypes: self.enabledElements->forAll(e : Element|e.oclIsTypeOf(DivisionOfLabor)
or e.oclIsTypeOf(Rule) or e.oclIsTypeOf(Tool)
or e.oclIsTypeOf(Subject)
or e.oclIsTypeOf(CommunityMember))
and self.enabledElements->forAll(e : Element|e.diagrams->intersection(self.diagrams)->isEmpty())

context ASD
inv asdHasEleTypes: self.elements->exists(e : Element|e.oclIsTypeOf(Tool))
and self.elements->exists(e : Element|e.oclIsTypeOf(Rule))
and self.elements->exists(e : Element|e.oclIsTypeOf(DivisionOfLabor))
and self.elements->exists(e : Element|e.oclIsTypeOf(Subject))
and self.elements->exists(e : Element|e.oclIsTypeOf(CommunityMember))
and self.elements->exists(e : Element|e.oclIsTypeOf(Object))
and self.elements->exists(e : Element|e.oclIsTypeOf(Outcome))

context MediatedComposite
inv constituents: self.scoEle->forAll(e1,e2:MediatedEle | (e1 <> e2)
implies ((e1.oclIsTypeOf(Subject) and not (e2.oclIsTypeOf(Subject)))
or (e1.oclIsTypeOf(CommunityMember) and not (e2.oclIsTypeOf(CommunityMember)))
or (e1.oclIsTypeOf(Object) and not (e2.oclIsTypeOf(Object)))))

context Tool
inv sUsesTachieveObj: self.mediates->exists(m:MediatedComposite |
m.scoEle->exists(e1,e2:MediatedEle|e1.oclIsTypeOf(Subject)
and e2.oclIsTypeOf(Object)))

context ASD
inv mediationInOneASD: ASD.allInstances() ->
forAll (a:ASD | a.mComposits ->
forAll (mc:MediatedComposite | (mc.mediatedBy ->
forAll (e:MediatingEle | a.elements -> includes (e)) )
and (mc.scoEle ->
forAll (f:MediatedEle | a.elements -> includes (f)) ) ) )

context MediatedComposite
inv STO: MediatedComposite.allInstances() ->
forAll(mc:MediatedComposite | mc.scoEle ->
forAll (e1,e2:MediatedEle | (e1<>e2 and e1.oclIsTypeOf(Subject)
and e2.oclIsTypeOf(Object))
implies (mc.mediatedBy->exists (t:MediatingEle | t.oclIsTypeOf(Tool)))))
```

USE Commands to Create ASD Network Object Model

USE allows object models to be created using a scripting language. We used this feature to break up the commands needed to create the entire vector surveillance network into three files; one each for the two ASDs and one for creating the network relation(s) between them. Commands to produce the *ASTP* instance are given first, followed by the *SP* ASD, and finally the network relations between these ASDs. The files containing commands for the original versions are presented first, followed by those used to create the evolved instances.

USE Commands to Create Original ASTP ASD Object Model

```
-- read ASTP-orig.x
-- This script creates the Assign Surveillance Tasks to Personnel ASD,
-- The outcome of this activity is a TOOL for the Survey Premises ASD

!create ASTP : ASD
!create ASTPObj1 : Object
!insert (ASTP,ASTPObj1) into eleInASD
!create ASTPOut1 : Outcome
!insert (ASTP,ASTPOut1) into eleInASD
!create ASTPT1 : Tool
!insert (ASTP,ASTPT1) into eleInASD
!create ASTPSub1 : Subject
!insert (ASTP,ASTPSub1) into eleInASD
!create ASTPDoL1 : DivisionOfLabor
!insert (ASTP,ASTPDoL1) into eleInASD
!create ASTPComm1 : CommunityMember
!insert (ASTP,ASTPComm1) into eleInASD
!create ASTPR1 : Rule
!insert (ASTP,ASTPR1) into eleInASD

-- Coordinator is related to
-- surveillance tasks object: SO
-- and is related to Rule 1, DoL1, T1
!create ASTPVCCO1 : MediatedComposite
!insert(ASTPVCCO1,ASTPSub1) into medComp
!insert(ASTPVCCO1,ASTPObj1) into medComp
!insert(ASTPR1,ASTPVCCO1) into mediationRel
!insert(ASTPDoL1,ASTPVCCO1) into mediationRel
!insert(ASTPT1,ASTPVCCO1) into mediationRel
!insert (ASTP, ASTPVCCO1) into medRelASD

!insert (ASTPDoL1,ASTPR1) into dols2rules

!insert (ASTPDoL1, ASTPComm1) into whoDoesDoL
```

USE Commands to Create Original SP ASD Object Model

```
-- read SP-orig.x
-- This script creates the Survey Premise ASD,
-- A TOOL of this activity is an OUTCOME from the ASTP ASD

!create SP : ASD
-- Obj: collect premise data
!create SPObj : Object
!insert (SP,SPObj) into eleInASD
!create SPOut : Outcome
!insert (SP,SPOut) into eleInASD
-- T1: CPA
-- T2: list
!create SPT1 : Tool
!create SPT2 : Tool
!insert (SP,SPT2) into eleInASD
!insert (SP,SPT1) into eleInASD
-- Sub: Field Agent
!create SPSub : Subject
!insert (SP,SPSub) into eleInASD
-- DoL1: Field Agent perform survey
-- DoL2: HOT give access
!create SPDoL1 : DivisionOfLabor
!insert (SP,SPDoL1) into eleInASD
!create SPDoL2 : DivisionOfLabor
!insert (SP,SPDoL2) into eleInASD
-- Comm1: Field Agent
-- Comm2: HOT
!create SPComm1 : CommunityMember
!create SPComm2 : CommunityMember
!insert (SP,SPComm1) into eleInASD
!insert (SP,SPComm2) into eleInASD
-- R1: valid data
-- R2: use CPA
-- R3: permission
!create SPR1 : Rule
!create SPR2 : Rule
!create SPR3 : Rule
!insert (SP,SPR1) into eleInASD
!insert (SP,SPR2) into eleInASD
!insert (SP,SPR3) into eleInASD

-- HOT is related to the subject: CS
-- and is related to Rule 3
!create SPHTS : MediatedComposite
!insert (SPHTS,SPComm2) into medComp
!insert (SPHTS,SPSub) into medComp
!insert (SPR3,SPHTS) into mediationRel
!insert (SP, SPHTS) into medRelASD

-- Subject is related to the object: SO
-- and is related to DoL1, R1, R2, T1, T2
!create SPSubObj : MediatedComposite
!insert (SPSubObj,SPObj) into medComp
!insert (SPSubObj,SPSub) into medComp
!insert (SPR1,SPSubObj) into mediationRel
!insert (SPR2,SPSubObj) into mediationRel
!insert (SPT1,SPSubObj) into mediationRel
!insert (SPT2,SPSubObj) into mediationRel
!insert (SPDoL1,SPSubObj) into mediationRel
!insert (SP, SPSubObj) into medRelASD

-- HOT is related to the object: CO
-- and is related to Rule 1
-- and is related to DoL2
!create SPHTOj : MediatedComposite
!insert (SPHTOj,SPComm2) into medComp
!insert (SPHTOj,SPObj) into medComp
!insert (SPR1,SPHTOj) into mediationRel
!insert (SPDoL2,SPHTOj) into mediationRel
!insert (SP, SPHTOj) into medRelASD

-- DoL1 related to R1, R2
-- DoL2 related to R3
!insert (SPDoL1,SPR1) into dols2rules
!insert (SPDoL1,SPR2) into dols2rules
!insert (SPDoL2,SPR3) into dols2rules

-- Field Agent does DoL1
-- HOT does DoL2
!insert (SPDoL1, SPComm1) into whoDoesDoL
!insert (SPDoL2, SPComm2) into whoDoesDoL
```

USE Commands to Create Original Network Relations

```
-- read ASTP-SPnet-orig.x
-- This script creates a network relation between the ASTP and SP ASDs

-- SP is Survey Premises
-- ASTP is Assign Surveillance Tasks to Personnel

-- create networked relation between ASTPOut1 (survey list) and SPT2 (survey list)
!insert(SPT2, ASTPOut1) into enabEleReqOut
```

USE Commands to Create Evolved ASTP ASD Object Model

```
-- read ASTP-evol.x
-- This script creates the EVOLVED Assign Surveillance Tasks to Personnel ASD,
-- The OUTCOMES of this activity are 2 TOOLS for the EVOLVED Survey Premises ASD

!create ASTP : ASD
-- OBJ1: manage surveillance tasks
-- OBJ2: manage training tasks
!create ASTPObj1 : Object
!insert (ASTP,ASTPObj1) into eleInASD
!create ASTPObj2 : Object
!insert (ASTP,ASTPObj2) into eleInASD
-- OT1: task list
-- OT2: training list
!create ASTPOut1 : Outcome
!insert (ASTP,ASTPOut1) into eleInASD
!create ASTPOut2 : Outcome
!insert (ASTP,ASTPOut2) into eleInASD
!create ASTPT1 : Tool
!insert (ASTP,ASTPT1) into eleInASD
!create ASTPSub1 : Subject
!insert (ASTP,ASTPSub1) into eleInASD
-- DOL1: create the task list
-- DOL2: create the training list
!create ASTPDoL1 : DivisionOfLabor
!insert (ASTP,ASTPDoL1) into eleInASD
!create ASTPDoL2 : DivisionOfLabor
!insert (ASTP,ASTPDoL2) into eleInASD
!create ASTPComm1 : CommunityMember
!insert (ASTP,ASTPComm1) into eleInASD
!create ASTPR1 : Rule
!insert (ASTP,ASTPR1) into eleInASD

-- Coordinator is related to
-- surveillance tasks object: SO
-- and is related to R1, DoL1, T1
!create ASTPVCCO1 : MediatedComposite
!insert(ASTPVCCO1,ASTPSub1) into medComp
!insert(ASTPVCCO1,ASTPObj1) into medComp
!insert(ASTPR1,ASTPVCCO1) into mediationRel
!insert(ASTPDoL1,ASTPVCCO1) into mediationRel
!insert(ASTPT1,ASTPVCCO1) into mediationRel
!insert (ASTP, ASTPVCCO1) into medRelASD

-- Coordinator is related to
-- training tasks object: SO
!create ASTPVCCO2 : MediatedComposite
!insert(ASTPVCCO2,ASTPSub1) into medComp
!insert(ASTPVCCO2,ASTPObj2) into medComp
!insert(ASTPR1,ASTPVCCO2) into mediationRel
!insert(ASTPDoL2,ASTPVCCO2) into mediationRel
```

```
!insert(ASTPT1,ASTPVCCO2) into mediationRel
!insert (ASTP, ASTPVCCO2) into medRelASD

!insert (ASTPDoL1,ASTPR1) into dols2rules

!insert (ASTPDoL1, ASTPComm1) into whoDoesDoL
!insert (ASTPDoL2, ASTPComm1) into whoDoesDoL
```

USE Commands to Create Evolved SP ASD Object Model

```
-- read SP-evol.x
-- This script creates the Survey Premise ASD,
-- A TOOL of this activity is an OUTCOME from the ASTP ASD
!create SP : ASD
-- Obj: collect premise data
!create SPObj : Object
!insert (SP,SPObj) into eleInASD
!create SPOut : Outcome
!insert (SP,SPOut) into eleInASD
-- T1: CPA
-- T2: surveillance list
-- T3: training list
!create SPT1 : Tool
!create SPT2 : Tool
!create SPT3 : Tool
!insert (SP,SPT1) into eleInASD
!insert (SP,SPT2) into eleInASD
!insert (SP,SPT3) into eleInASD
-- S1: Field Agent
-- S2: HOT
!create SPSub1 : Subject
!insert (SP,SPSub1) into eleInASD
!create SPSub2 : Subject
!insert (SP,SPSub2) into eleInASD
-- DOL1: survey
-- DOL2: permission
-- DOL3: train
-- DOL4: self-survey
-- DOL5: OK to self-survey
-- DOL6: make tools
!create SPDoL1 : DivisionOfLabor
!insert (SP,SPDoL1) into eleInASD
!create SPDoL2 : DivisionOfLabor
!insert (SP,SPDoL2) into eleInASD
!create SPDoL3 : DivisionOfLabor
!insert (SP,SPDoL3) into eleInASD
!create SPDoL4 : DivisionOfLabor
!insert (SP,SPDoL4) into eleInASD
!create SPDoL5 : DivisionOfLabor
!insert (SP,SPDoL5) into eleInASD
!create SPDoL6 : DivisionOfLabor
!insert (SP,SPDoL6) into eleInASD
-- Comm1: Field Agent
-- Comm2: HOT
-- Comm3: Coord
!create SPComm1 : CommunityMember
!create SPComm2 : CommunityMember
!create SPComm3 : CommunityMember
!insert (SP,SPComm1) into eleInASD
!insert (SP,SPComm2) into eleInASD
!insert (SP,SPComm3) into eleInASD
-- R1: valid data
-- R2: use CPA
-- R3: permission
-- R4: valid bug id
-- R5: tools good
!create SPR1 : Rule
!create SPR2 : Rule
!create SPR3 : Rule
!create SPR4 : Rule
!create SPR5 : Rule
!insert (SP,SPR1) into eleInASD
!insert (SP,SPR2) into eleInASD
!insert (SP,SPR3) into eleInASD
!insert (SP,SPR4) into eleInASD
!insert (SP,SPR5) into eleInASD

-- Homeowner/Tenant is related to the subject: CS
-- and is related to R3
!create SPHTS : MediatedComposite
!insert (SPHTS,SPComm2) into medComp
!insert (SPHTS,SPSub1) into medComp
!insert (SPR3,SPHTS) into mediationRel
!insert (SP, SPHTS) into medRelASD

-- Field Agent Subject is related to the object: SO
-- AND is related to DoL1, DoL3, R1, R2, T1, T2, T3
!create SPSub1Obj : MediatedComposite
!insert (SPSub1Obj,SPObj) into medComp
!insert (SPSub1Obj,SPSub1) into medComp
!insert (SPR1,SPSub1Obj) into mediationRel
!insert (SPR2,SPSub1Obj) into mediationRel
!insert (SPT1,SPSub1Obj) into mediationRel
!insert (SPT2,SPSub1Obj) into mediationRel
!insert (SPT3,SPSub1Obj) into mediationRel
!insert (SPDoL1,SPSub1Obj) into mediationRel
!insert (SPDoL3,SPSub1Obj) into mediationRel
!insert (SP, SPSub1Obj) into medRelASD

-- Homeowner/Tenant is related to the object: CO
-- and is related to Rule 1
-- AND is related to DoL2
!create SPHTOj : MediatedComposite
!insert (SPHTOj,SPComm2) into medComp
!insert (SPHTOj,SPObj) into medComp
!insert (SPR1,SPHTOj) into mediationRel
!insert (SPDoL2,SPHTOj) into mediationRel
!insert (SP, SPHTOj) into medRelASD

-- HOT Subject is related to the object: SO
-- AND is related to DoL4, R4
!create SPSub2Obj: MediatedComposite
!insert (SPSub2Obj,SPObj) into medComp
!insert (SPSub2Obj,SPSub2) into medComp
!insert (SPR4,SPSub2Obj) into mediationRel
!insert (SPDoL4,SPSub2Obj) into mediationRel
!insert (SP, SPSub2Obj) into medRelASD

-- Coordinator (SPComm3) is related to the object: CO
-- AND is related to DoL5
!create SPCoordOj : MediatedComposite
!insert (SPCoordOj,SPComm3) into medComp
!insert (SPCoordOj,SPObj) into medComp
!insert (SPDoL5,SPCoordOj) into mediationRel
!insert (SP,SPCoordOj) into medRelASD

-- DoL1 related to R1, R2
-- DoL2 related to R3
-- DoL4 related to R4
!insert (SPDoL1,SPR1) into dols2rules
!insert (SPDoL1,SPR2) into dols2rules
!insert (SPDoL2,SPR3) into dols2rules
!insert (SPDoL4,SPR4) into dols2rules

-- Field Agent: DoL1, DoL3
-- HOT: DoL2, DoL4
-- Coord: DoL5
!insert (SPDoL1, SPComm1) into whoDoesDoL
!insert (SPDoL3, SPComm1) into whoDoesDoL
!insert (SPDoL2, SPComm2) into whoDoesDoL
!insert (SPDoL4, SPComm2) into whoDoesDoL
!insert (SPDoL5, SPComm3) into whoDoesDoL
```

USE Commands to Create Evolved Network Relations

```
-- read ASTP-SPnet-evol.x
-- This script creates a network relation between the ASTP and SP ASDs

-- SP is Survey Premises
-- ASTP is Assign Surveillance Tasks to Personnel

-- create networked relation between surveillance lists
!insert(SPT2, ASTPOut1) into enabEleReqOut

-- create networked relation between training lists
!insert(SPT3, ASTPOut2) into enabEleReqOut
```

References

[Engeström] Y. Engeström, Learning by expanding, Helsinki: Orienta-Konsultit, 1987.

[ITU-T] ITU-T, *User Requirements Notation (URN) – Language definition*, ITU-T Recommendation Z.151 (10/12), Geneva, Switzerland, October 2012; http://www.itu.int/rec/T-REC-Z.151/en (acc. March 2013).

[jUCMNav] jUCMNav website, http://softwareengineering.ca/jucmnav (accessed March 2013).

[Lozano-Fuentes] S. Lozano-Fuentes, S. Ghosh, J.M. Bieman, D. Sadhu, L. Eisen, F. Wedyan, E. Hernandez-Garcia, J. Garcia-Rejon, and D. Tep-Chel, "Using Cell Phones for Mosquito Vector Surveillance and Control", *24th International Conference on Software Engineering & Knowledge Engineering (SEKE'12)*, Knowledge Systems Institute Graduate School, pp. 763–767 (2012).

[Paige] R.F. Paige, N. Drivalos, D.S. Kolovos, K.J. Fernandes, C. Power, G. K. Olsen, and S. Zschaler, "Rigorous identification and encoding of trace-links in model-driven engineering", *Software & Systems Modeling*, vol. 10, no. 4, pp. 469–487 (2011).