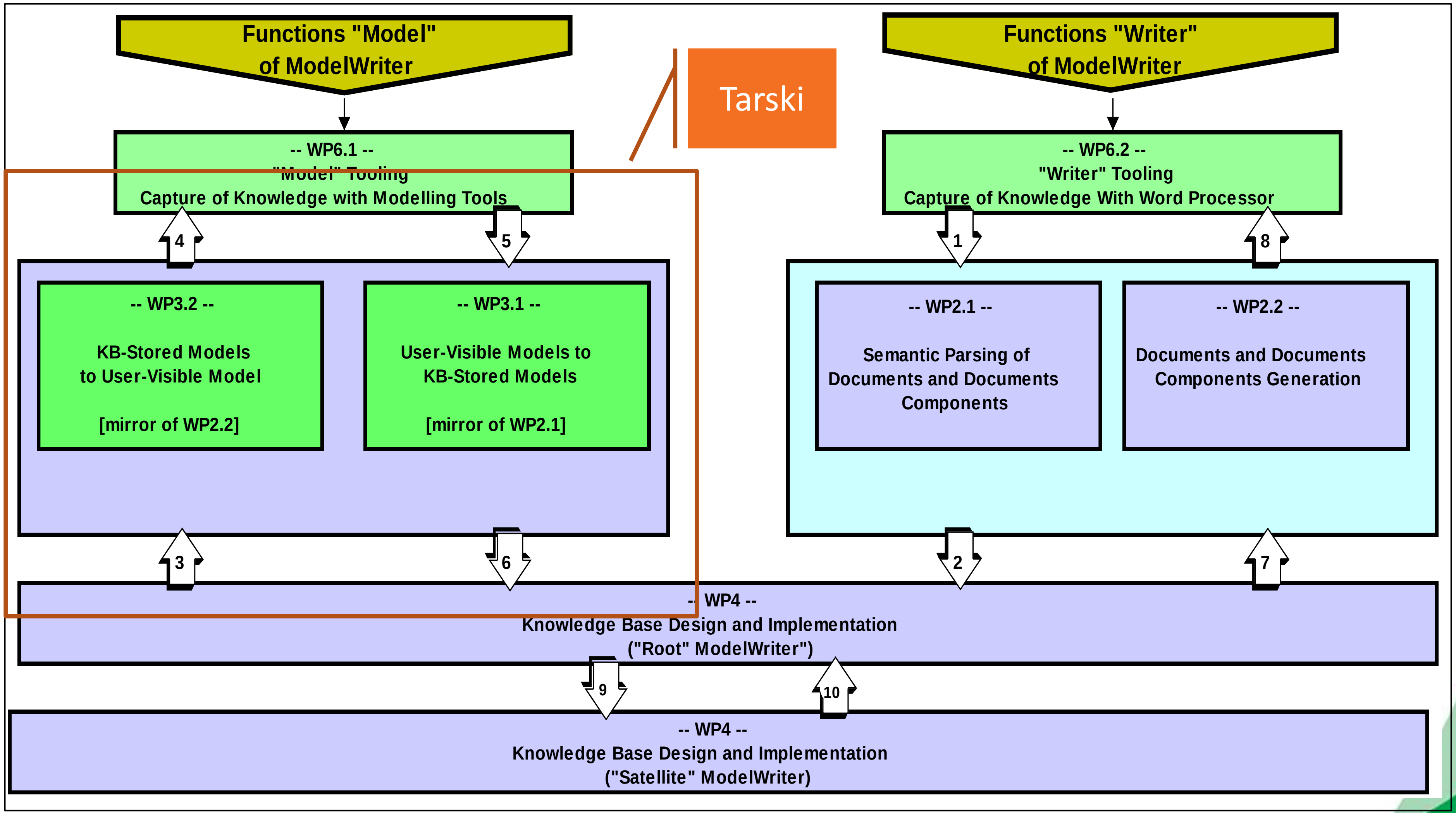
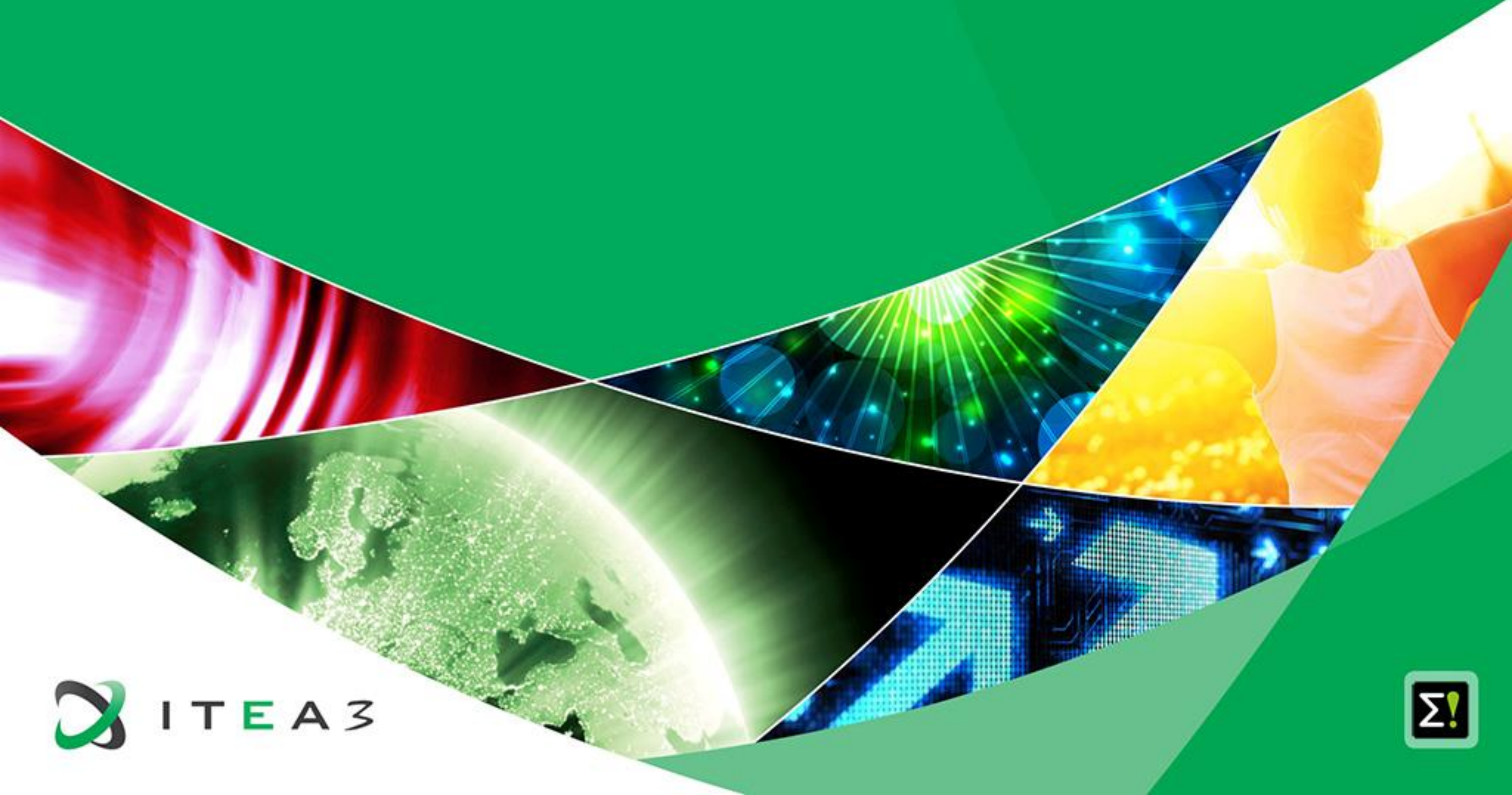
WP3 - Model to/from Knowledge Base Synchronization Mechanism

Ferhat Erata, Work Package Leader
UNIT Information Technologies R&D Ltd.

Technological components & interactions
Collaboration by WP interactions


Functions "Model" of ModelWriter
Functions "Writer" of ModelWriter
Tarski
-- WP6.1 -- "Model" Tooling Capture of Knowledge with Modelling Tools
-- WP6.2 -- "Writer" Tooling Capture of Knowledge With Word Processor
-- WP3.2 -- KB-Stored Models to User-Visible Model [mirror of WP2.2]
-- WP3.1 -- User-Visible Models to KB-Stored Models [mirror of WP2.1]
-- WP2.1 -- Semantic Parsing of Documents and Documents Components
-- WP2.2 -- Documents and Documents Components Generation
-- WP4 -- Knowledge Base Design and Implementation ("Root" ModelWriter")
-- WP4 -- Knowledge Base Design and Implementation ("Satellite" ModelWriter)
1
2
3
4
5
6
7
8
9
10

Tarski: A Platform for Automated Analysis of Dynamically Configurable Semantics of Traceability



Challenges of Traceability in Industry

Semantically meaningful traceability

- traceability relations should have a rich semantic meaning instead of being simple bi-directional referential relation

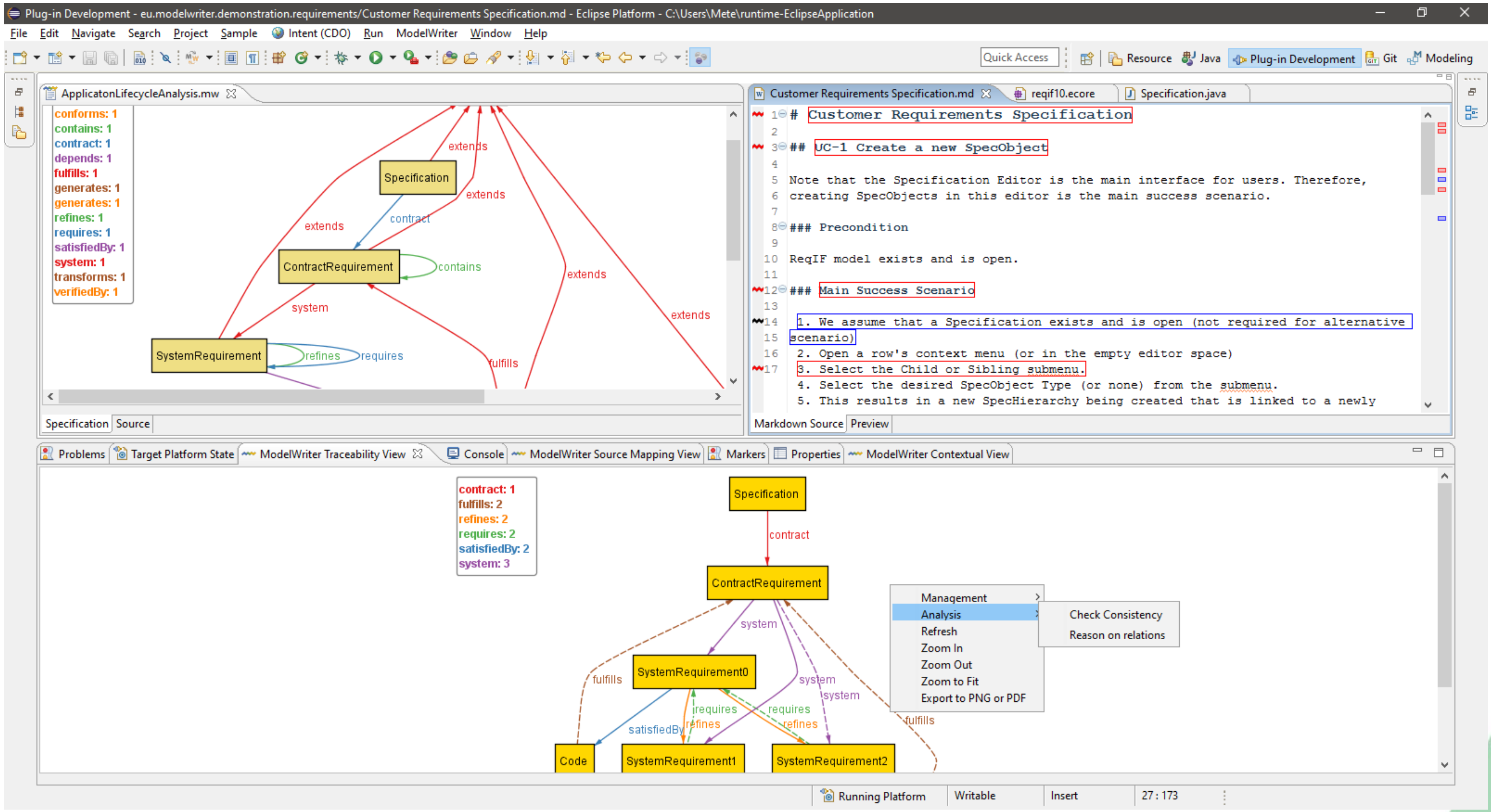
Configurability of traceability (possibly dynamically)

- the semantics of traceability is often statically defined
- the semantics cannot be easily adapted for the needs of different projects.
- different traceable elements and the types of relations exist in industrial settings.

Several industries demands formal proofs of traceability

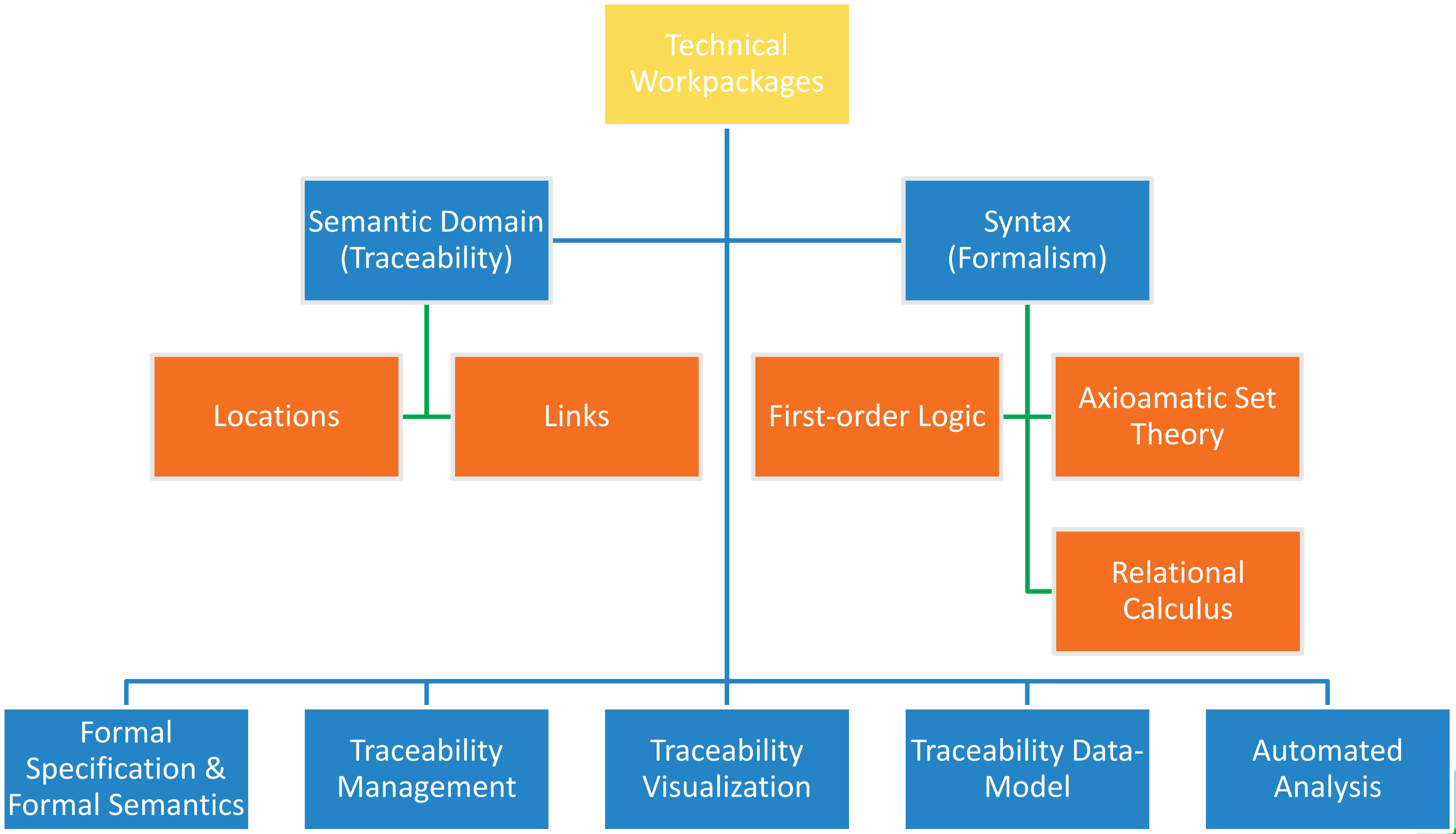
Consistency checking and repairing broken trace links

Technical Contributions @Tarski

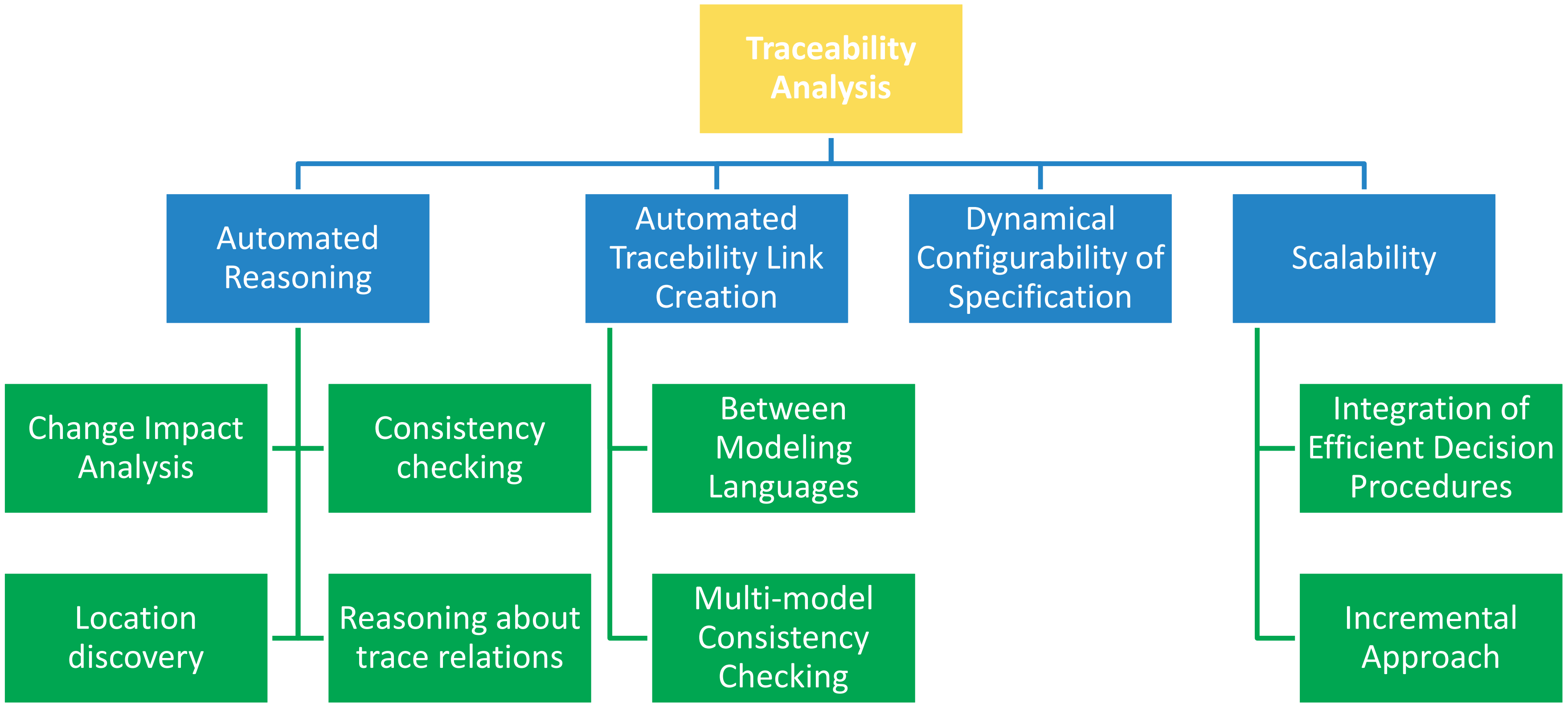


```
# Customer Requirements Specification

## UC-1 Create a new SpecObject

Note that the Specification Editor is the main interface for users. Therefore,
creating SpecObjects in this editor is the main success scenario.

### Precondition

ReqIF model exists and is open.

### Main Success Scenario

 1. We assume that a Specification exists and is open (not required for alternative
scenario)
 2. Open a row's context menu (or in the empty editor space)
 3. Select the Child or Sibling submenu.
 4. Select the desired SpecObject Type (or none) from the submenu.
 5. This results in a new SpecHierarchy being created that is linked to a newly
```

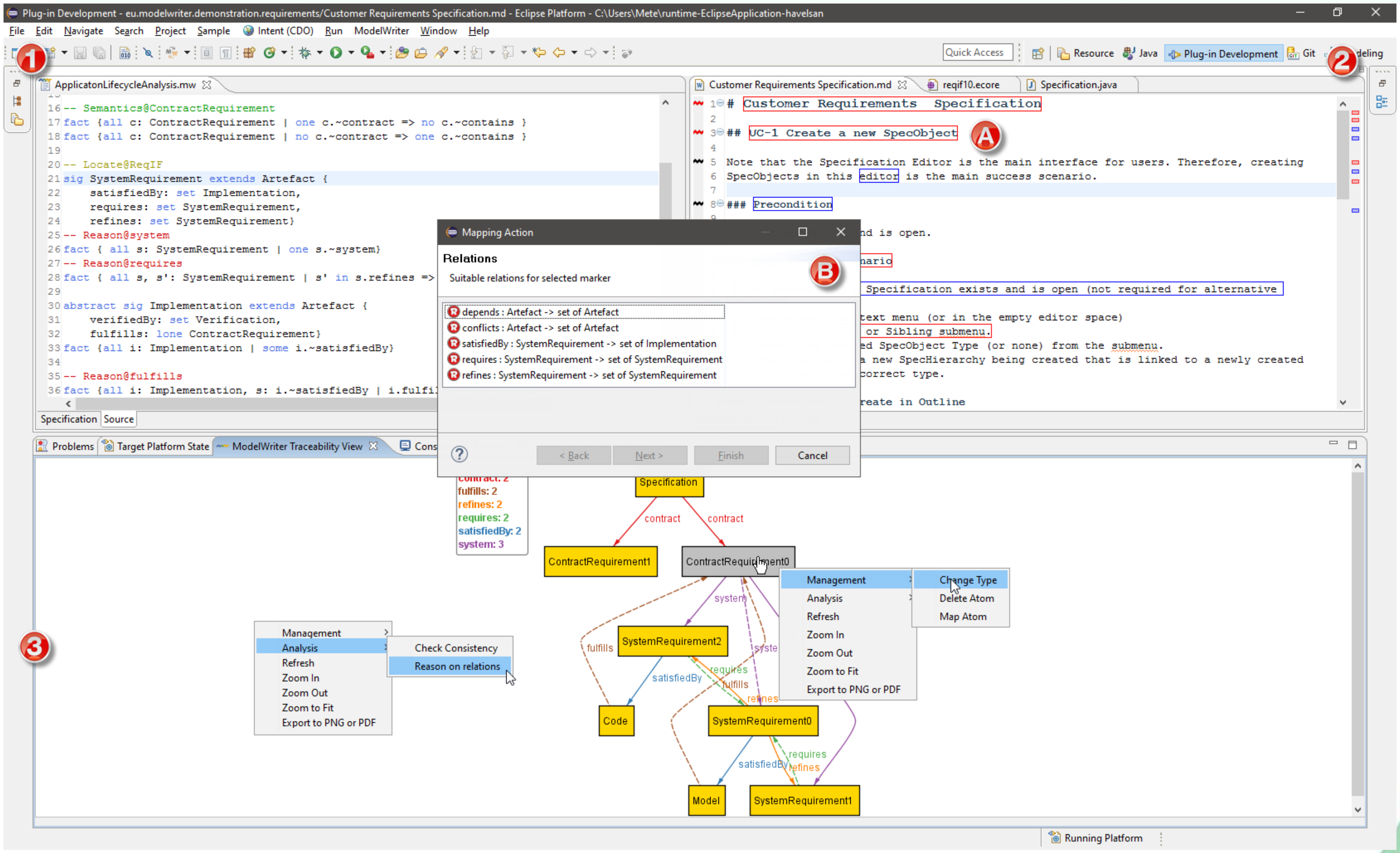
Overview of Technical Contributions @Tarski



Traceability Analysis


Traceability Analysis
Automated Reasoning
Automated Tracebility Link Creation
Dynamical Configurability of Specification
Scalability
Change Impact Analysis
Consistency checking
Location discovery
Reasoning about trace relations
Between Modeling Languages
Multi-model Consistency Checking
Integration of Efficient Decision Procedures
Incremental Approach

Tarski Approach



Plug-in Development - eu.modelwriter.demonstration.requirements/Customer Requirements Specification.md - Eclipse Platform - C:\Users\Mete\runtime-EclipseApplication-havelsan

ApplicatonLifecycleAnalysis.mw

```
-- Semantics@ContractRequirement
fact {all c: ContractRequirement | one c.~contract => no c.~contains }
fact {all c: ContractRequirement | no c.~contract => one c.~contains }

-- Locate@ReqIF
sig SystemRequirement extends Artefact {
    satisfiedBy: set Implementation,
    requires: set SystemRequirement,
    refines: set SystemRequirement}
-- Reason@system
fact { all s: SystemRequirement | one s.~system}
-- Reason@requires
fact { all s, s': SystemRequirement | s' in s.refines =>

abstract sig Implementation extends Artefact {
    verifiedBy: set Verification,
    fulfills: lone ContractRequirement}
fact {all i: Implementation | some i.~satisfiedBy}

-- Reason@fulfills
fact {all i: Implementation, s: i.~satisfiedBy | i.fulfi
```

Customer Requirements Specification.md

```
# Customer Requirements  Specification

## UC-1 Create a new SpecObject

Note that the Specification Editor is the main interface for users. Therefore, creating
SpecObjects in this editor is the main success scenario.

### Precondition
nd is open.

nario

 Specification exists and is open (not required for alternative

text menu (or in the empty editor space)
 or Sibling submenu.
ed SpecObject Type (or none) from the submenu.
a new SpecHierarchy being created that is linked to a newly created
correct type.

reate in Outline
```

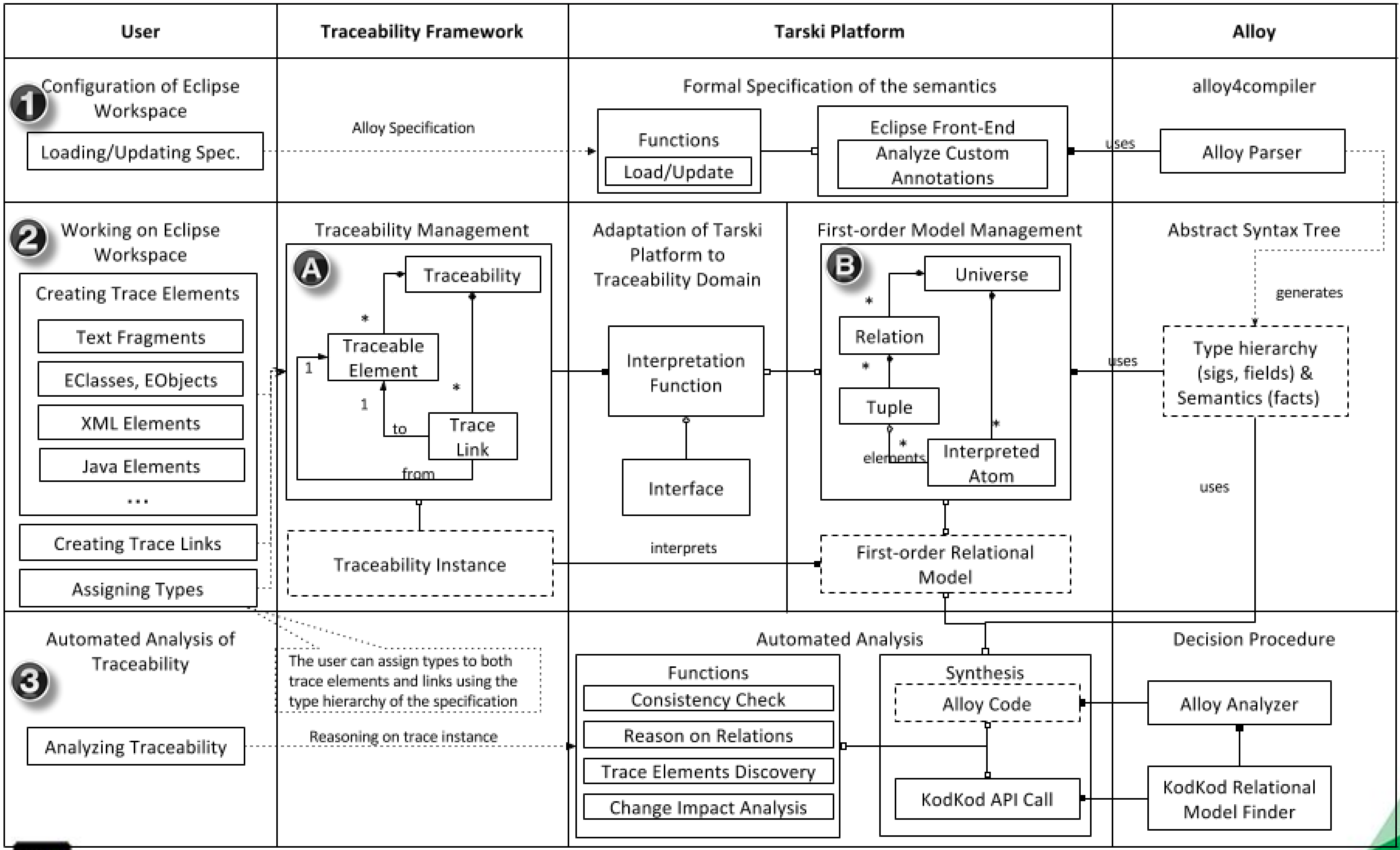
Mapping Action

Relations

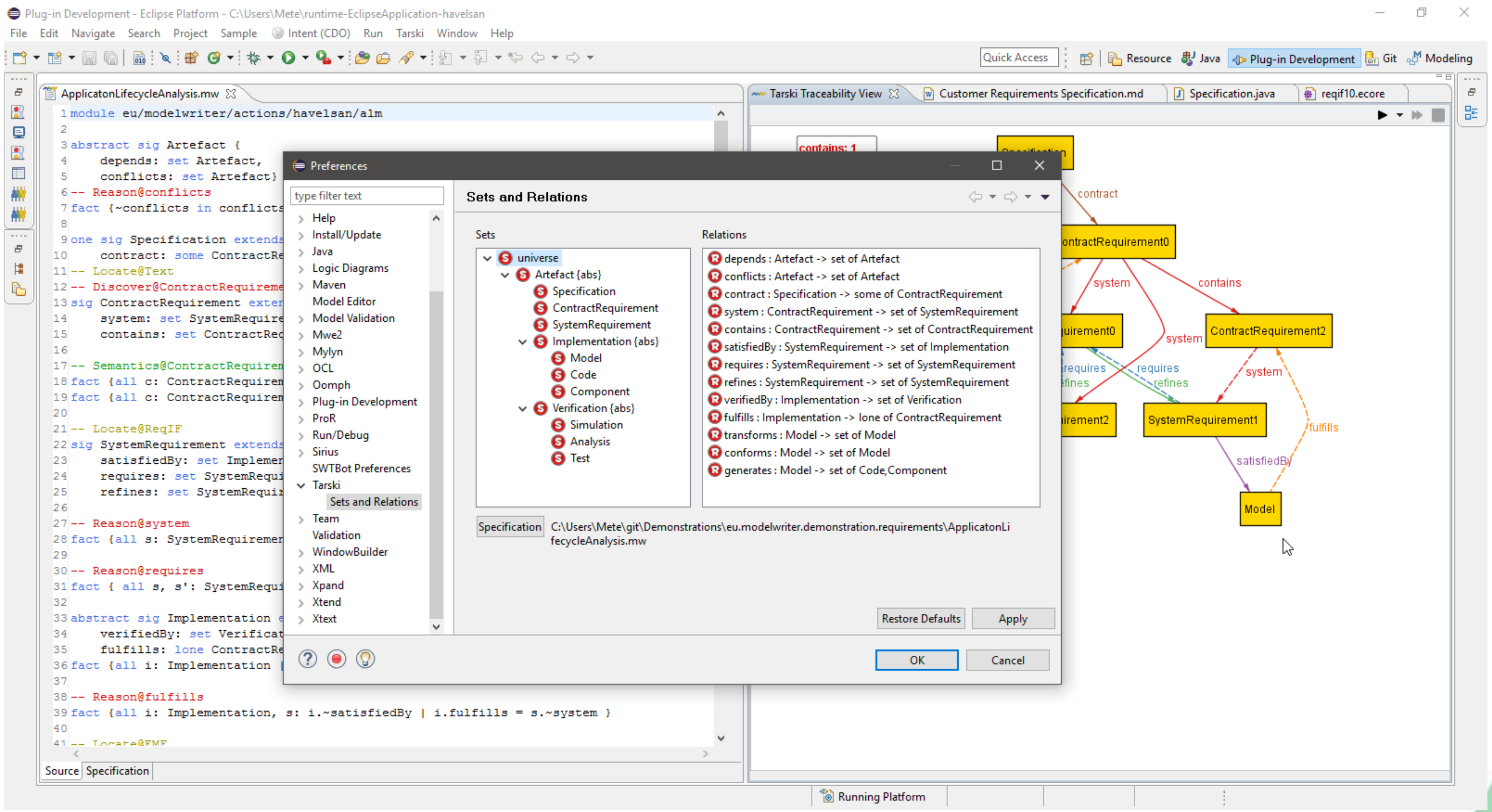
Suitable relations for selected marker

- depends : Artefact -> set of Artefact
- conflicts : Artefact -> set of Artefact
- satisfiedBy : SystemRequirement -> set of Implementation
- requires : SystemRequirement -> set of SystemRequirement
- refines : SystemRequirement -> set of SystemRequirement

Tarski
Approach



Types/Component Ontology derived from the specification



Plug-in Development - Eclipse Platform - C:\Users\Mete\runtime-EclipseApplication-havelsan

ApplicatonLifecycleAnalysis.mw

```
module eu/modelwriter/actions/havelsan/alm

abstract sig Artefact {
    depends: set Artefact,
    conflicts: set Artefact}
-- Reason@conflicts
fact {~conflicts in conflicts

one sig Specification extends
    contract: some ContractRe
-- Locate@Text
-- Discover@ContractRequireme
sig ContractRequirement exter
    system: set SystemRequire
    contains: set ContractReq

-- Semantics@ContractRequirem
fact {all c: ContractRequirem
fact {all c: ContractRequirem

-- Locate@ReqIF
sig SystemRequirement extends
    satisfiedBy: set Implemen
    requires: set SystemRequi
    refines: set SystemRequir

-- Reason@system
fact {all s: SystemRequiremer

-- Reason@requires
fact { all s, s': SystemRequi

abstract sig Implementation e
    verifiedBy: set Verificat
    fulfills: lone ContractRe
fact {all i: Implementation |

-- Reason@fulfills
fact {all i: Implementation, s: i.~satisfiedBy | i.fulfills = s.~system }

-- Locate@EMF
```

Preferences

Sets and Relations

Sets

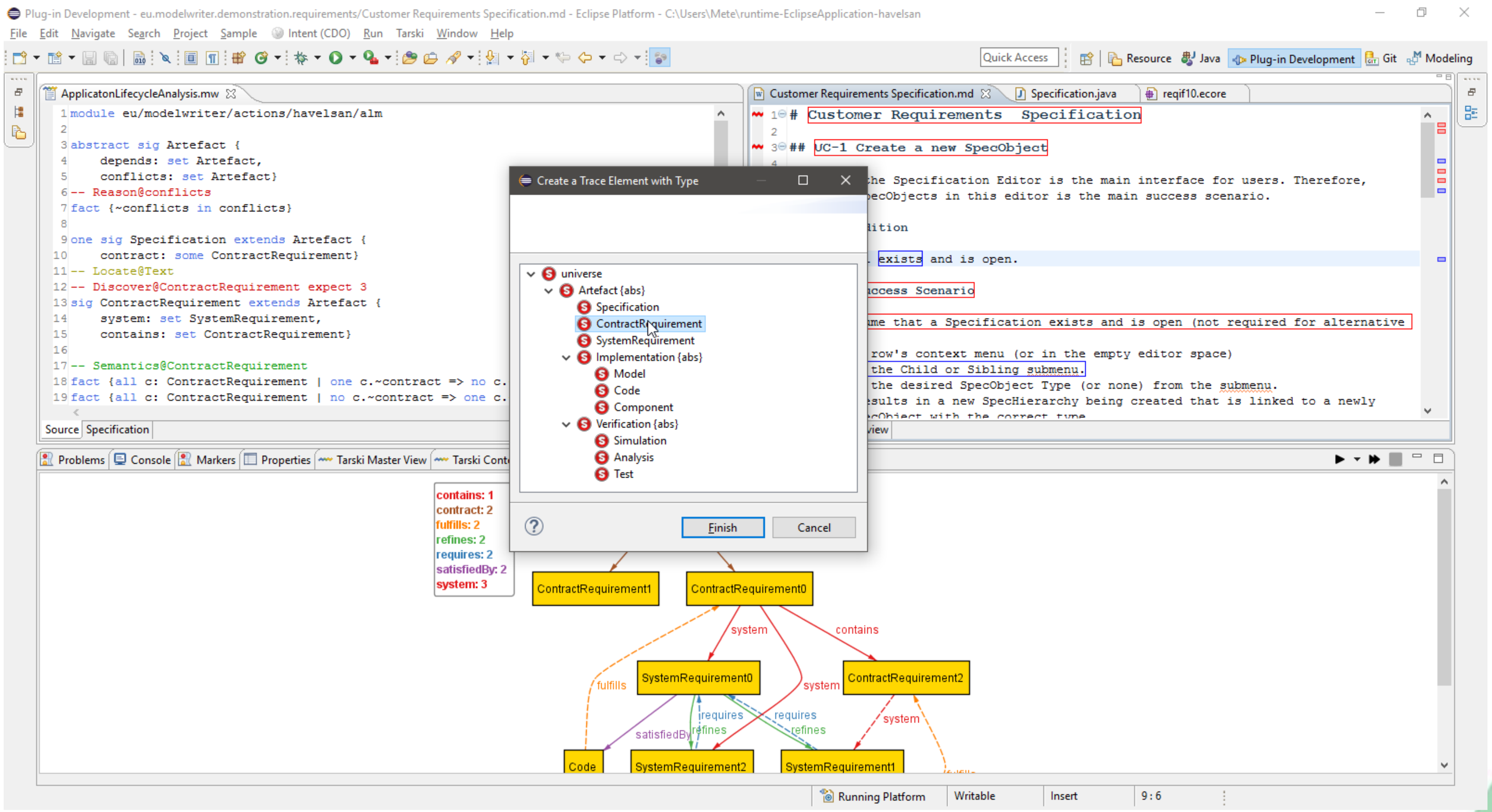
- universe
  - Artefact {abs}
    - Specification
    - ContractRequirement
    - SystemRequirement
    - Implementation {abs}
      - Model
      - Code
      - Component
    - Verification {abs}
      - Simulation
      - Analysis
      - Test

Relations

- depends : Artefact -> set of Artefact
- conflicts : Artefact -> set of Artefact
- contract : Specification -> some of ContractRequirement
- system : ContractRequirement -> set of SystemRequirement
- contains : ContractRequirement -> set of ContractRequirement
- satisfiedBy : SystemRequirement -> set of Implementation
- requires : SystemRequirement -> set of SystemRequirement
- refines : SystemRequirement -> set of SystemRequirement
- verifiedBy : Implementation -> set of Verification
- fulfills : Implementation -> lone of ContractRequirement
- transforms : Model -> set of Model
- conforms : Model -> set of Model
- generates : Model -> set of Code,Component

Specification C:\Users\Mete\git\Demonstrations\eu.modelwriter.demonstration.requirements\ApplicatonLifecycleAnalysis.mw

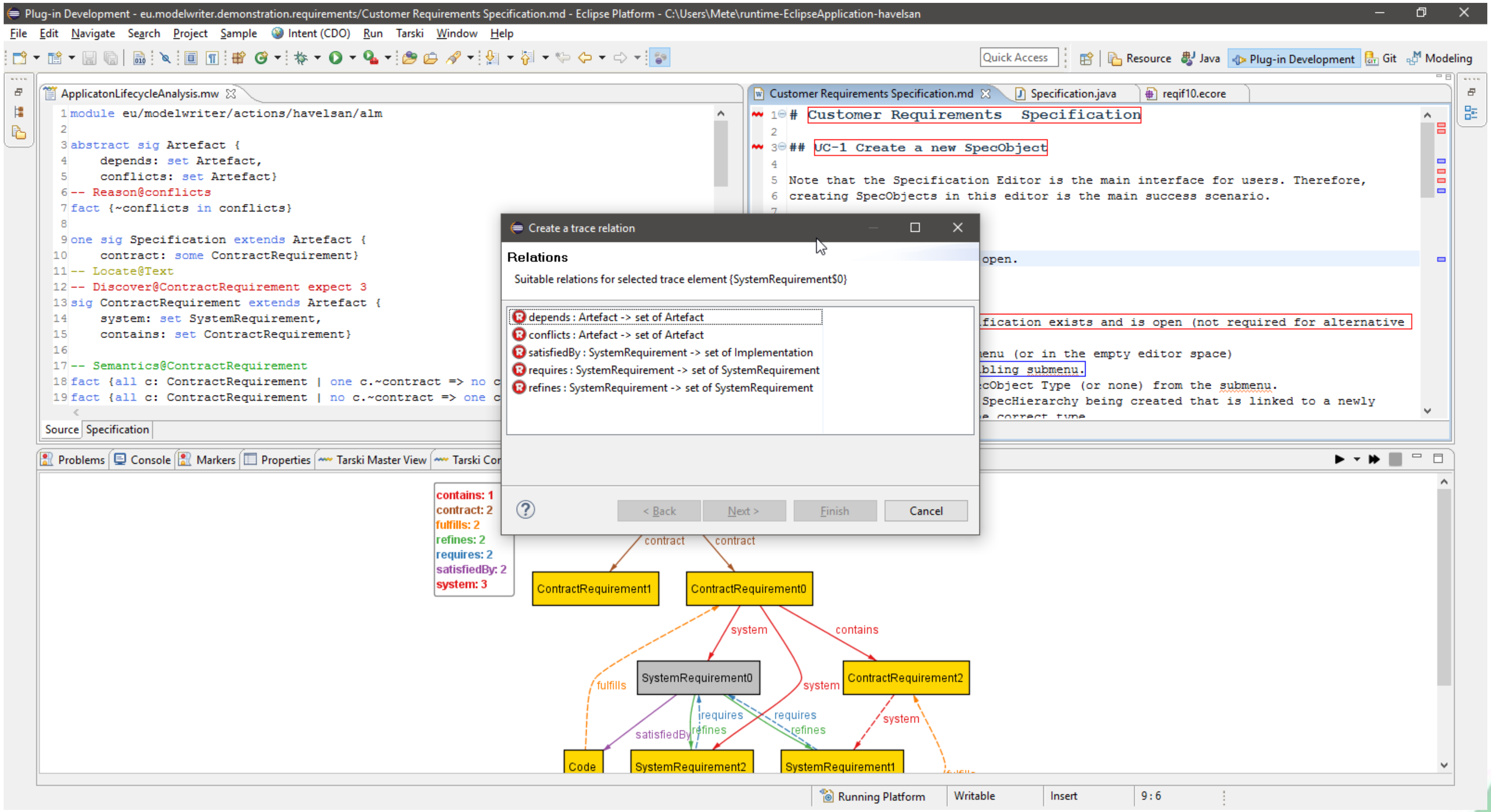
Assigning Unary Relations to a Traceable Elements



```
module eu/modelwriter/actions/havelsan/alm

abstract sig Artefact {
    depends: set Artefact,
    conflicts: set Artefact}
-- Reason@conflicts
fact {~conflicts in conflicts}

one sig Specification extends Artefact {
    contract: some ContractRequirement}
-- Locate@Text
-- Discover@ContractRequirement expect 3
sig ContractRequirement extends Artefact {
    system: set SystemRequirement,
    contains: set ContractRequirement}

-- Semantics@ContractRequirement
fact {all c: ContractRequirement | one c.~contract => no c.
fact {all c: ContractRequirement | no c.~contract => one c.
```

```
# Customer Requirements  Specification

## UC-1 Create a new SpecObject

he Specification Editor is the main interface for users. Therefore,
ecObjects in this editor is the main success scenario.

ition

 exists and is open.

ccess Scenario

me that a Specification exists and is open (not required for alternative

 row's context menu (or in the empty editor space)
 the Child or Sibling submenu.
 the desired SpecObject Type (or none) from the submenu.
sults in a new SpecHierarchy being created that is linked to a newly
```

Assigning Binary Relations to a Trace Link



Plug-in Development - eu.modelwriter.demonstration.requirements/Customer Requirements Specification.md - Eclipse Platform - C:\Users\Mete\runtime-EclipseApplication-havelsan

File Edit Navigate Search Project Sample Intent (CDO) Run Tarski Window Help

Quick Access | Resource | Java | Plug-in Development | Git | Modeling

ApplicatonLifecycleAnalysis.mw

```
module eu/modelwriter/actions/havelsan/alm

abstract sig Artefact {
    depends: set Artefact,
    conflicts: set Artefact}
-- Reason@conflicts
fact {~conflicts in conflicts}

one sig Specification extends Artefact {
    contract: some ContractRequirement}
-- Locate@Text
-- Discover@ContractRequirement expect 3
sig ContractRequirement extends Artefact {
    system: set SystemRequirement,
    contains: set ContractRequirement}

-- Semantics@ContractRequirement
fact {all c: ContractRequirement | one c.~contract => no c
fact {all c: ContractRequirement | no c.~contract => one c
```

Source | Specification

Customer Requirements Specification.md | Specification.java | reqif10.ecore

```
# Customer Requirements  Specification

## UC-1 Create a new SpecObject

Note that the Specification Editor is the main interface for users. Therefore,
creating SpecObjects in this editor is the main success scenario.
open.
fication exists and is open (not required for alternative
enu (or in the empty editor space)
bling submenu.
cObject Type (or none) from the submenu.
SpecHierarchy being created that is linked to a newly
e correct type
```

Create a trace relation

Relations

Suitable relations for selected trace element {SystemRequirement$0}

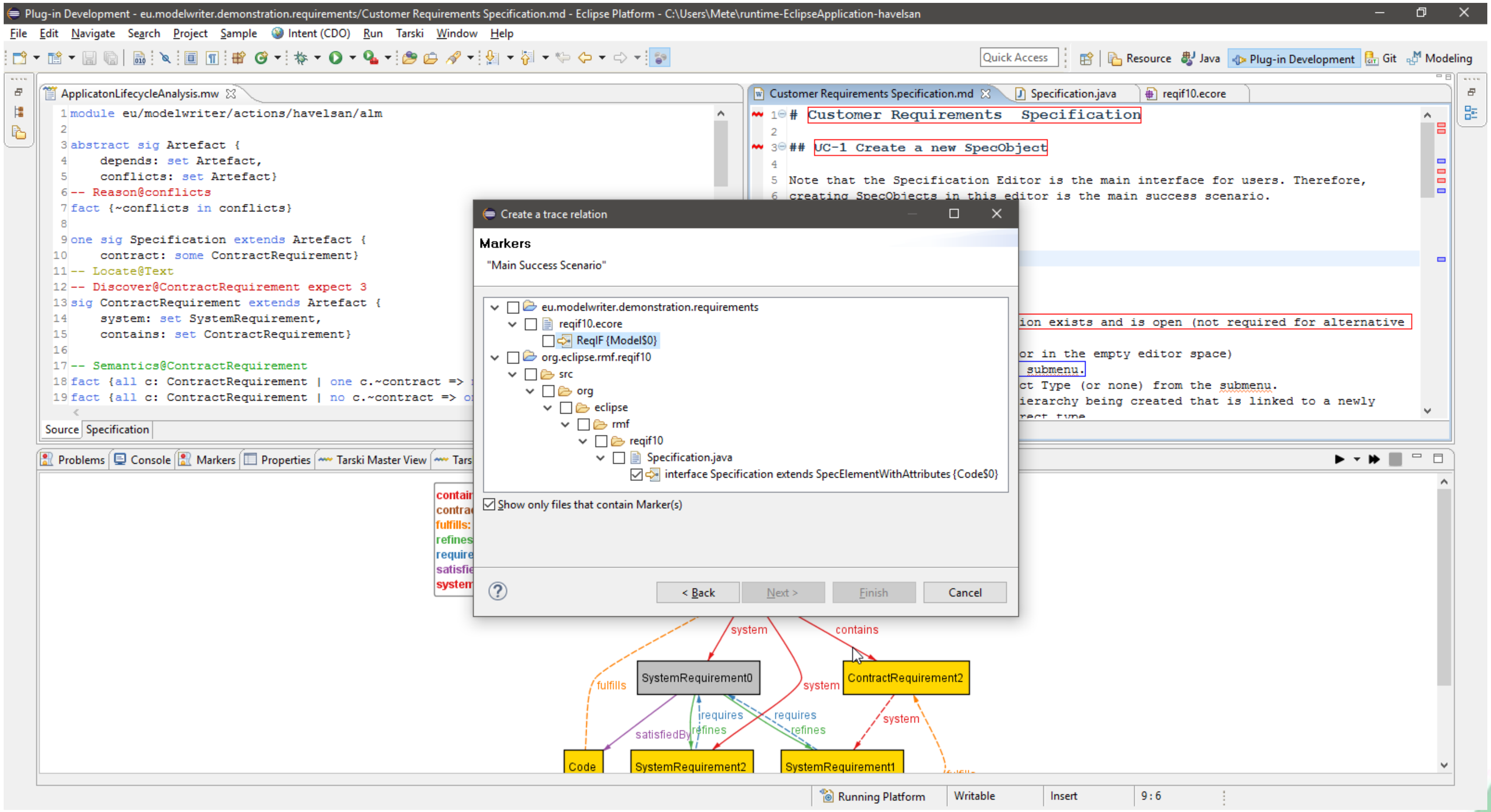
- depends : Artefact -> set of Artefact
- conflicts : Artefact -> set of Artefact
- satisfiedBy : SystemRequirement -> set of Implementation
- requires : SystemRequirement -> set of SystemRequirement
- refines : SystemRequirement -> set of SystemRequirement

< Back | Next > | Finish | Cancel

Problems | Console | Markers | Properties | Tarski Master View | Tarski Cor

Running Platform | Writable | Insert | 9 : 6

Selecting a range for a binary relation from an existing traceable elements



Plug-in Development - eu.modelwriter.demonstration.requirements/Customer Requirements Specification.md - Eclipse Platform - C:\Users\Mete\runtime-EclipseApplication-havelsan

ApplicatonLifecycleAnalysis.mw

```
module eu/modelwriter/actions/havelsan/alm

abstract sig Artefact {
     depends: set Artefact,
     conflicts: set Artefact}
-- Reason@conflicts
fact {~conflicts in conflicts}

one sig Specification extends Artefact {
     contract: some ContractRequirement}
-- Locate@Text
-- Discover@ContractRequirement expect 3
sig ContractRequirement extends Artefact {
     system: set SystemRequirement,
     contains: set ContractRequirement}

-- Semantics@ContractRequirement
fact {all c: ContractRequirement | one c.~contract =>
fact {all c: ContractRequirement | no c.~contract => o
```

Customer Requirements Specification.md | Specification.java | reqif10.ecore

```
# Customer Requirements  Specification

## UC-1 Create a new SpecObject

Note that the Specification Editor is the main interface for users. Therefore,
creating SpecObjects in this editor is the main success scenario.
```

```
ion exists and is open (not required for alternative
or in the empty editor space)
 submenu.
ct Type (or none) from the submenu.
ierarchy being created that is linked to a newly
```

Create a trace relation

Markers

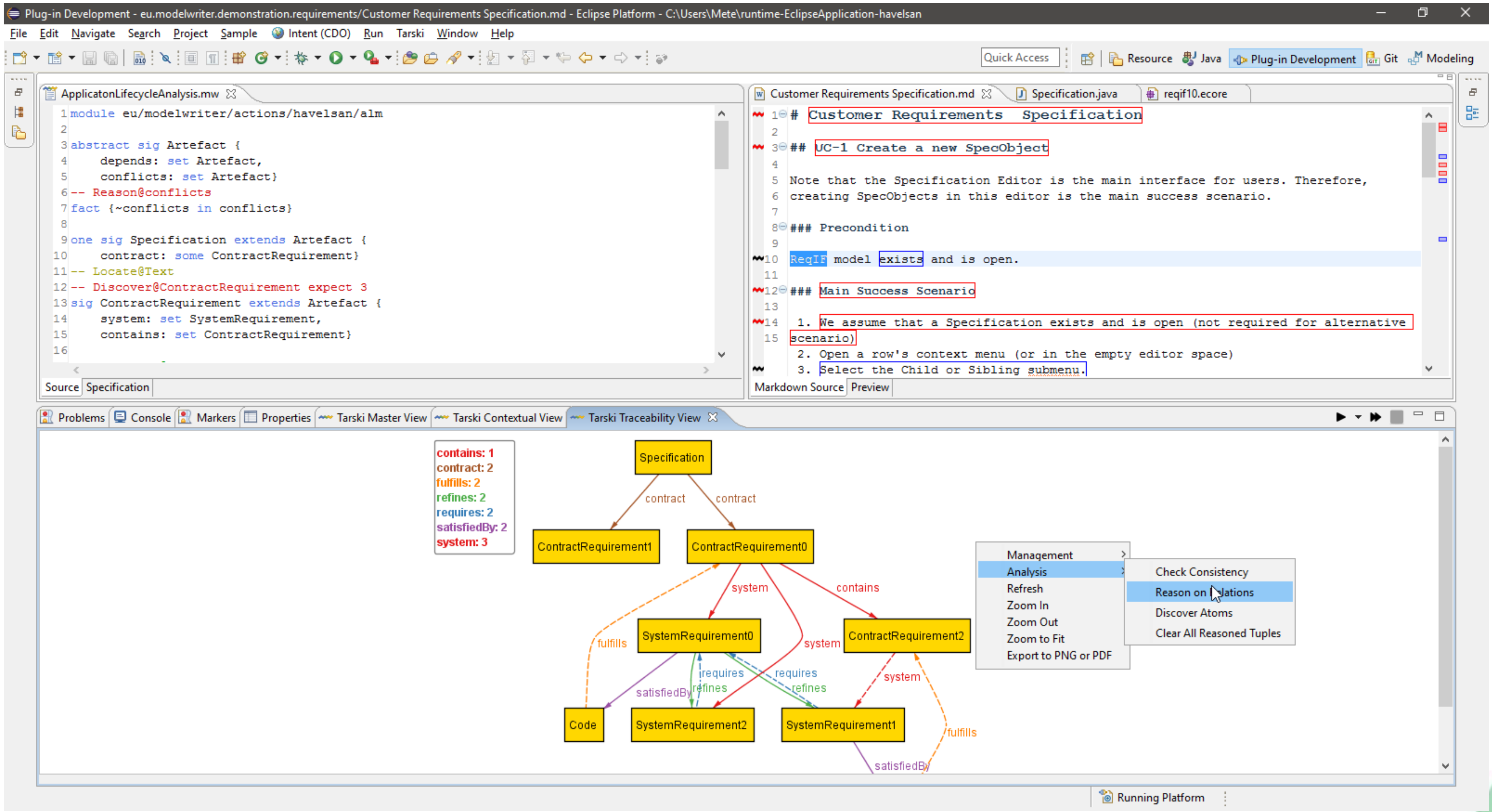
"Main Success Scenario"

- ☐ eu.modelwriter.demonstration.requirements
  - ☐ reqif10.ecore
    - ☐ ReqIF {Model$0}
- ☐ org.eclipse.rmf.reqif10
  - ☐ src
    - ☐ org
      - ☐ eclipse
        - ☐ rmf
          - ☐ reqif10
            - ☐ Specification.java
              - ☑ interface Specification extends SpecElementWithAttributes {Code$0}

☑ Show only files that contain Marker(s)

< Back | Next > | Finish | Cancel

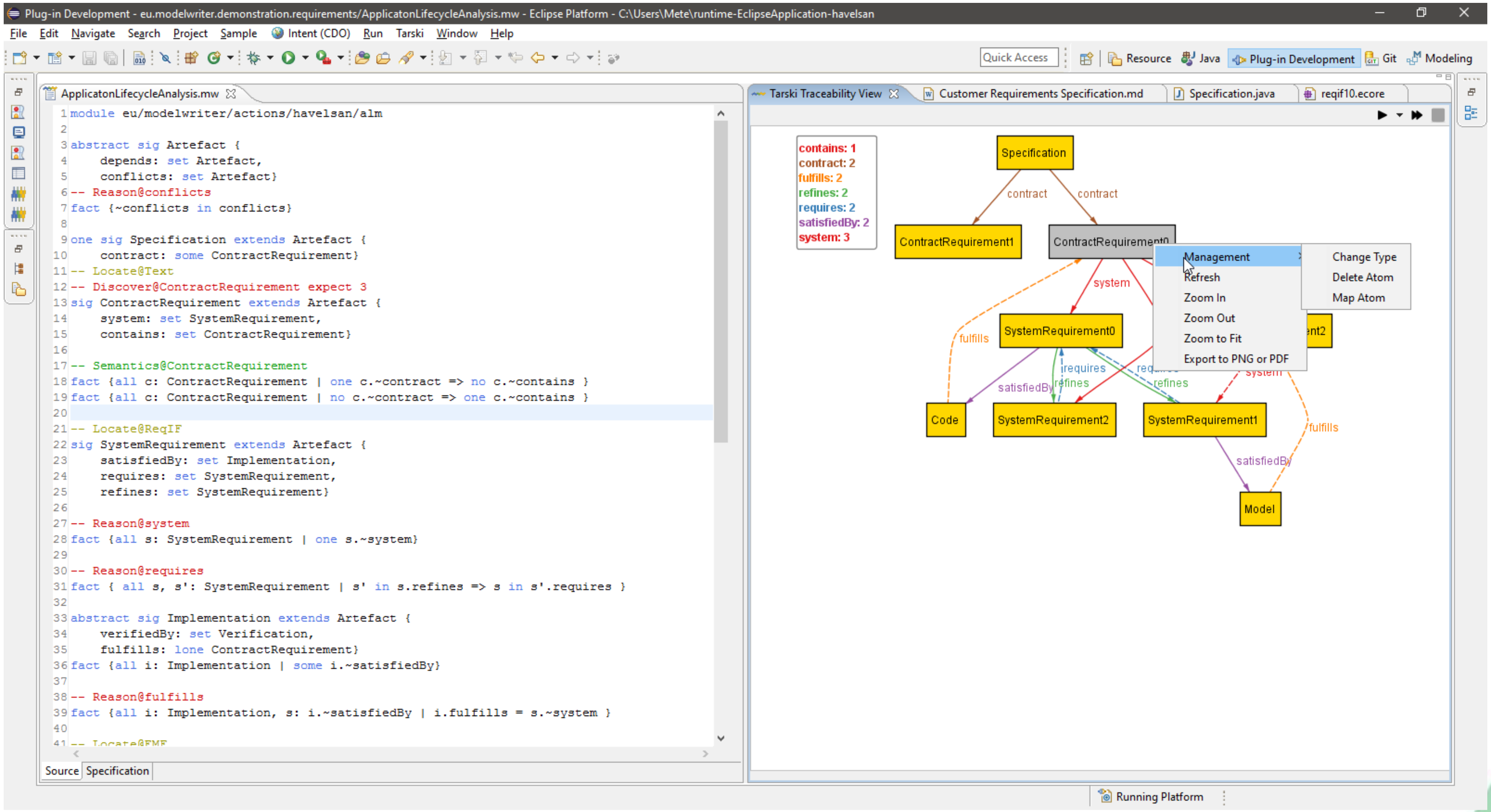
Automated Analysis of Traceability


Plug-in Development - eu.modelwriter.demonstration.requirements/Customer Requirements Specification.md - Eclipse Platform - C:\Users\Mete\runtime-EclipseApplication-havelsan
File Edit Navigate Search Project Sample Intent (CDO) Run Tarski Window Help
Quick Access
Resource Java Plug-in Development Git Modeling
ApplicatonLifecycleAnalysis.mw
Customer Requirements Specification.md
Specification.java
reqif10.ecore

```
module eu/modelwriter/actions/havelsan/alm

abstract sig Artefact {
    depends: set Artefact,
    conflicts: set Artefact}
-- Reason@conflicts
fact {~conflicts in conflicts}

one sig Specification extends Artefact {
    contract: some ContractRequirement}
-- Locate@Text
-- Discover@ContractRequirement expect 3
sig ContractRequirement extends Artefact {
    system: set SystemRequirement,
    contains: set ContractRequirement}
```

```
# Customer Requirements  Specification

## UC-1 Create a new SpecObject

Note that the Specification Editor is the main interface for users. Therefore,
creating SpecObjects in this editor is the main success scenario.

### Precondition

ReqIF model exists and is open.

### Main Success Scenario

 1. We assume that a Specification exists and is open (not required for alternative
scenario)
 2. Open a row's context menu (or in the empty editor space)
 3. Select the Child or Sibling submenu.
```

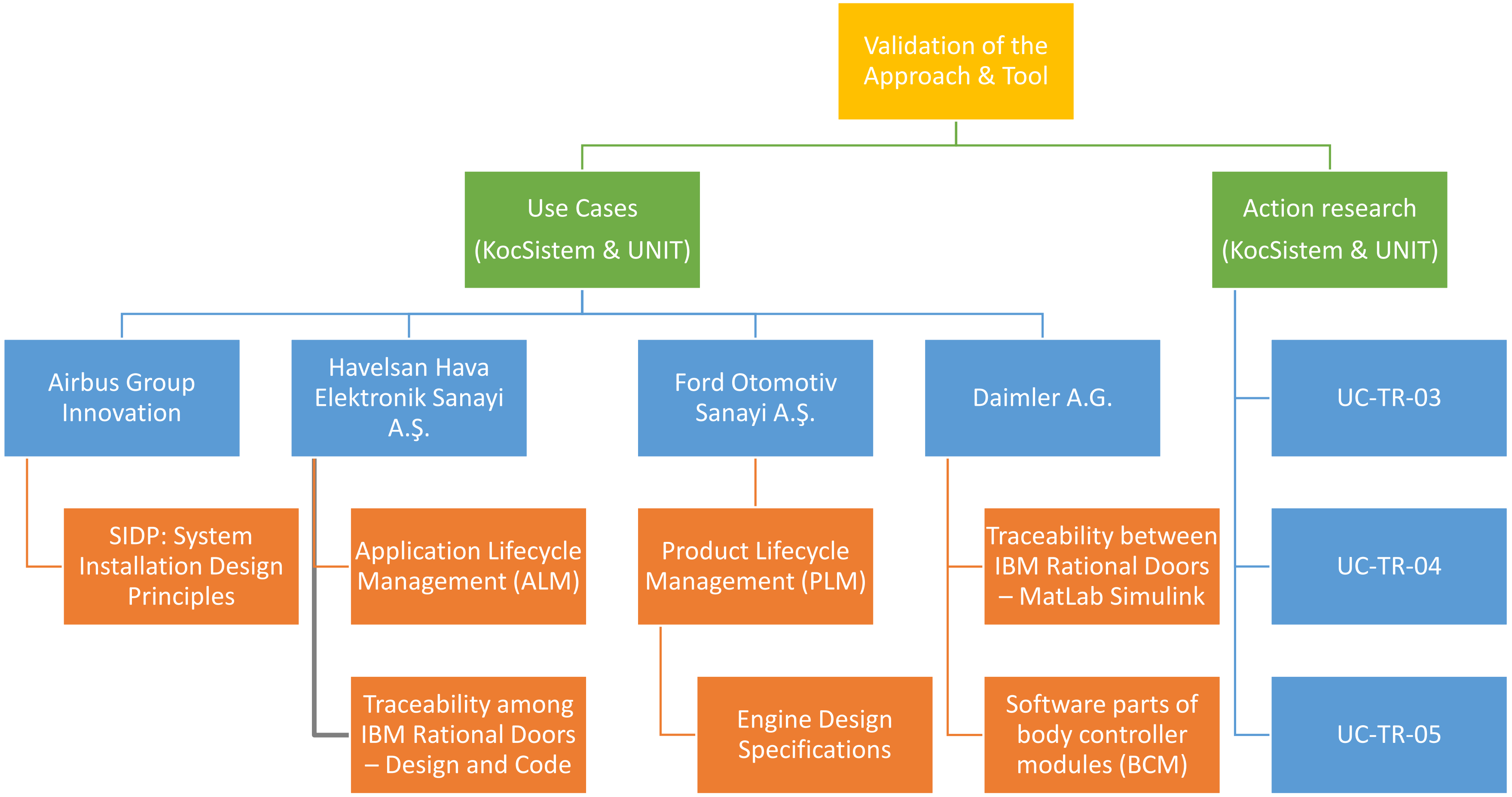
Dynamical Configuration & Model Management



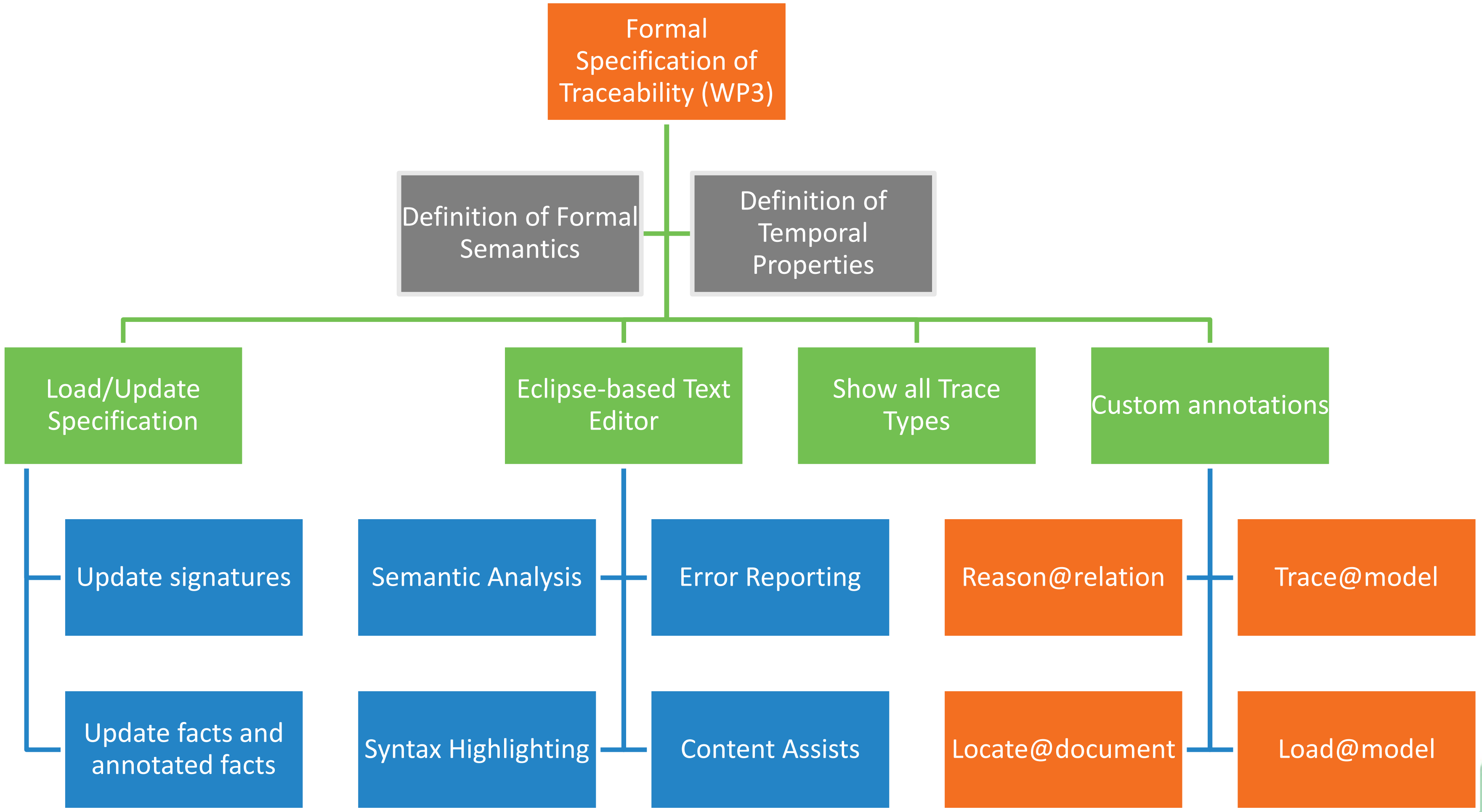
```
module eu/modelwriter/actions/havelsan/alm

abstract sig Artefact {
    depends: set Artefact,
    conflicts: set Artefact}
-- Reason@conflicts
fact {~conflicts in conflicts}

one sig Specification extends Artefact {
    contract: some ContractRequirement}
-- Locate@Text
-- Discover@ContractRequirement expect 3
sig ContractRequirement extends Artefact {
    system: set SystemRequirement,
    contains: set ContractRequirement}

-- Semantics@ContractRequirement
fact {all c: ContractRequirement | one c.~contract => no c.~contains }
fact {all c: ContractRequirement | no c.~contract => one c.~contains }

-- Locate@ReqIF
sig SystemRequirement extends Artefact {
    satisfiedBy: set Implementation,
    requires: set SystemRequirement,
    refines: set SystemRequirement}

-- Reason@system
fact {all s: SystemRequirement | one s.~system}

-- Reason@requires
fact { all s, s': SystemRequirement | s' in s.refines => s in s'.requires }

abstract sig Implementation extends Artefact {
    verifiedBy: set Verification,
    fulfills: lone ContractRequirement}
fact {all i: Implementation | some i.~satisfiedBy}

-- Reason@fulfills
fact {all i: Implementation, s: i.~satisfiedBy | i.fulfills = s.~system }

-- Locate@EMF
```

contains: 1
contract: 2
fulfills: 2
refines: 2
requires: 2
satisfiedBy: 2
system: 3

Validation of the Approach and Tool

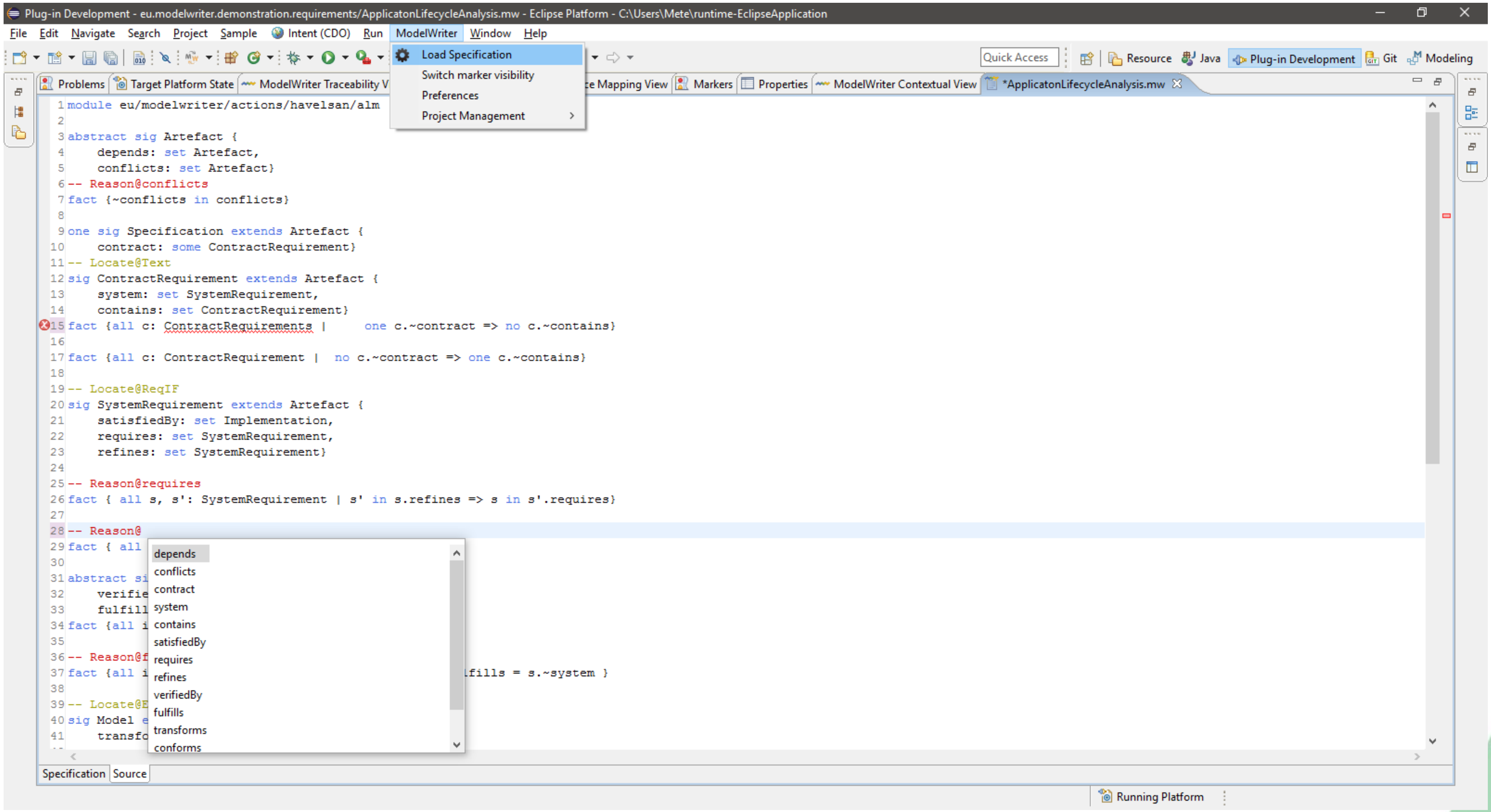

Validation of the Approach & Tool
Use Cases (KocSistem & UNIT)
Action research (KocSistem & UNIT)
Airbus Group Innovation
Havelsan Hava Elektronik Sanayi A.Ş.
Ford Otomotiv Sanayi A.Ş.
Daimler A.G.
UC-TR-03
SIDP: System Installation Design Principles
Application Lifecycle Management (ALM)
Product Lifecycle Management (PLM)
Traceability between IBM Rational Doors – MatLab Simulink
UC-TR-04
Traceability among IBM Rational Doors – Design and Code
Engine Design Specifications
Software parts of body controller modules (BCM)
UC-TR-05

Formal Specification of Traceability (WP3) [UNIT]



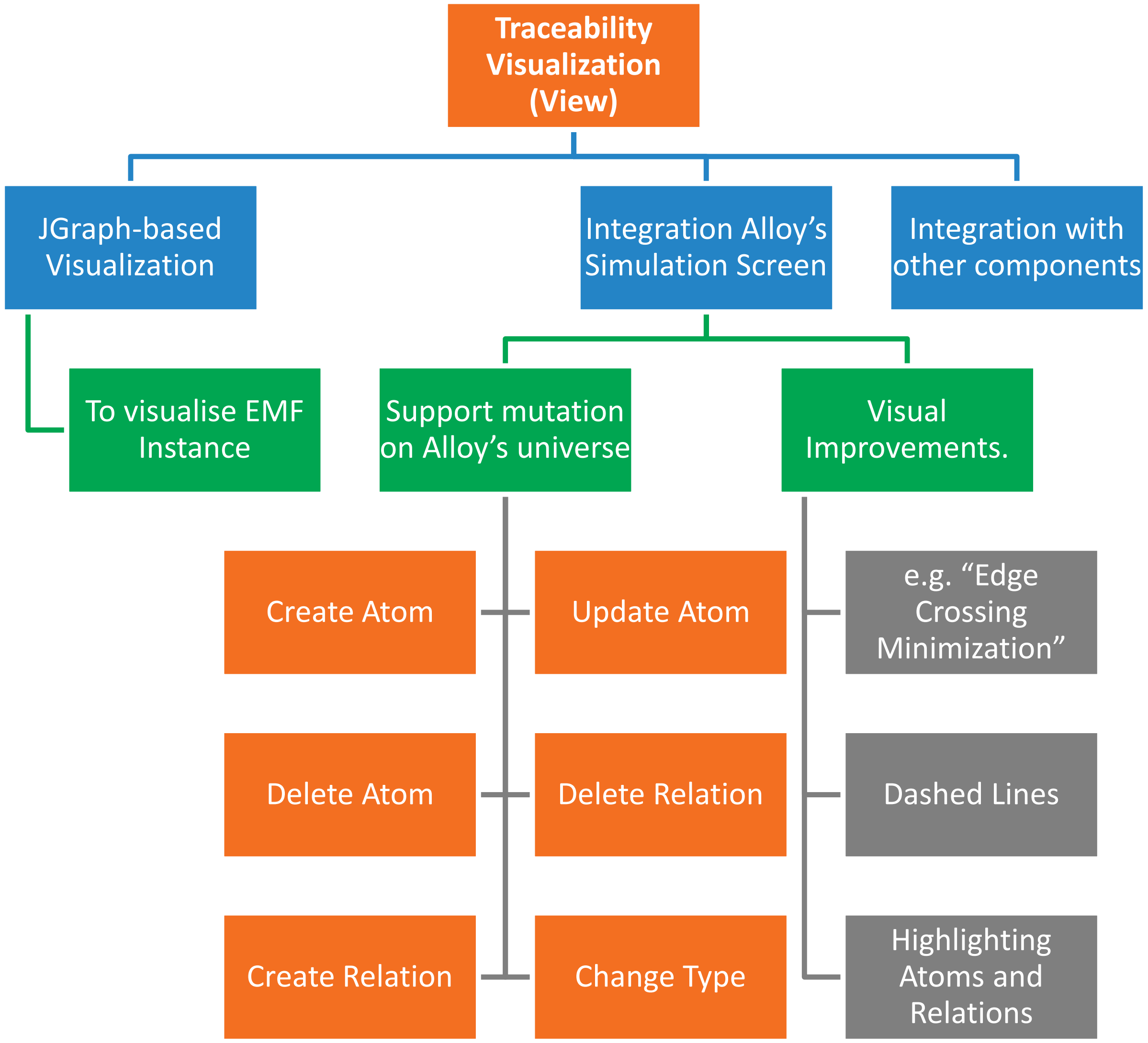
Formal Specification of Traceability (WP3)
Definition of Formal Semantics
Definition of Temporal Properties
Load/Update Specification
Eclipse-based Text Editor
Show all Trace Types
Custom annotations
Update signatures
Update facts and annotated facts
Semantic Analysis
Syntax Highlighting
Error Reporting
Content Assists
Reason@relation
Locate@document
Trace@model
Load@model

Demonstration
Textual Editor in Action

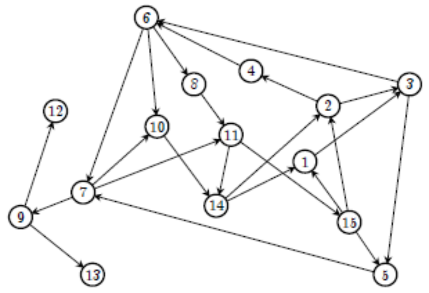

Plug-in Development - eu.modelwriter.demonstration.requirements/ApplicatonLifecycleAnalysis.mw - Eclipse Platform - C:\Users\Mete\runtime-EclipseApplication
File Edit Navigate Search Project Sample Intent (CDO) Run ModelWriter Window Help
Load Specification
Switch marker visibility
Preferences
Project Management
Quick Access
Resource Java Plug-in Development Git Modeling
Problems Target Platform State ModelWriter Traceability V
e Mapping View Markers Properties ModelWriter Contextual View *ApplicatonLifecycleAnalysis.mw
depends
conflicts
contract
system
contains
satisfiedBy
requires
refines
verifiedBy
fulfills
transforms
conforms
Specification Source
Running Platform

```
module eu/modelwriter/actions/havelsan/alm

abstract sig Artefact {
    depends: set Artefact,
    conflicts: set Artefact}
-- Reason@conflicts
fact {~conflicts in conflicts}

one sig Specification extends Artefact {
    contract: some ContractRequirement}
-- Locate@Text
sig ContractRequirement extends Artefact {
    system: set SystemRequirement,
    contains: set ContractRequirement}
fact {all c: ContractRequirements |     one c.~contract => no c.~contains}

fact {all c: ContractRequirement |  no c.~contract => one c.~contains}

-- Locate@ReqIF
sig SystemRequirement extends Artefact {
    satisfiedBy: set Implementation,
    requires: set SystemRequirement,
    refines: set SystemRequirement}

-- Reason@requires
fact { all s, s': SystemRequirement | s' in s.refines => s in s'.requires}

-- Reason@
fact { all

abstract si
    verifie
    fulfill
fact {all i

-- Reason@f
fact {all i                                        lfills = s.~system }

-- Locate@E
sig Model e
    transfo
```

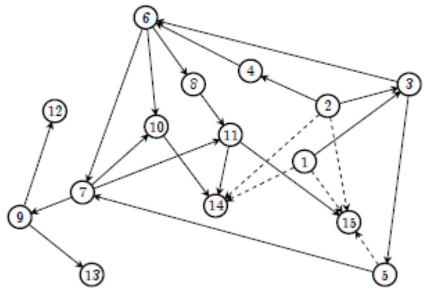
Traceability Visualization/View (WP3) [KoçSistem]


Traceability Visualization (View)
JGraph-based Visualization
Integration Alloy’s Simulation Screen
Integration with other components
To visualise EMF Instance
Support mutation on Alloy’s universe
Visual Improvements.
Create Atom
Update Atom
e.g. “Edge Crossing Minimization”
Delete Atom
Delete Relation
Dashed Lines
Create Relation
Change Type
Highlighting Atoms and Relations

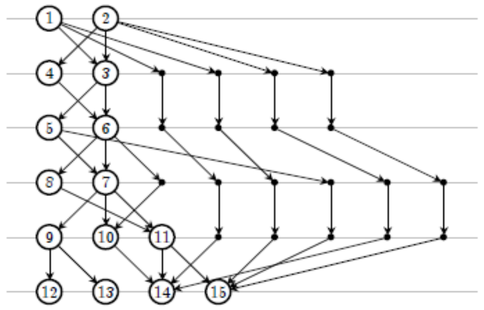
Layered/Hierarchical Graph Drawing Sugiyama Framework



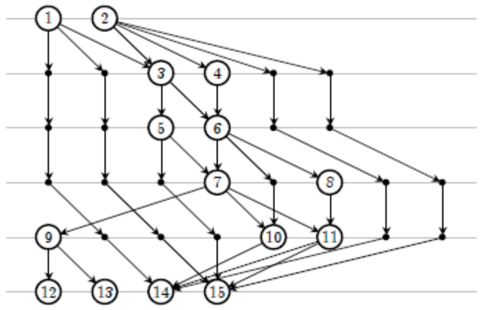
(a) Input digraph, G.



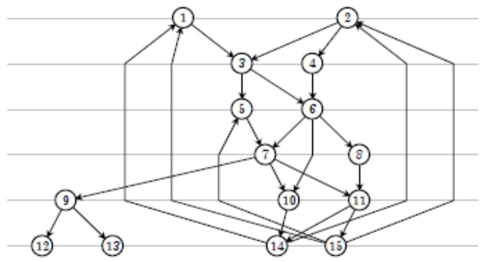
(b) Cycles removed.



(c) After leveling.

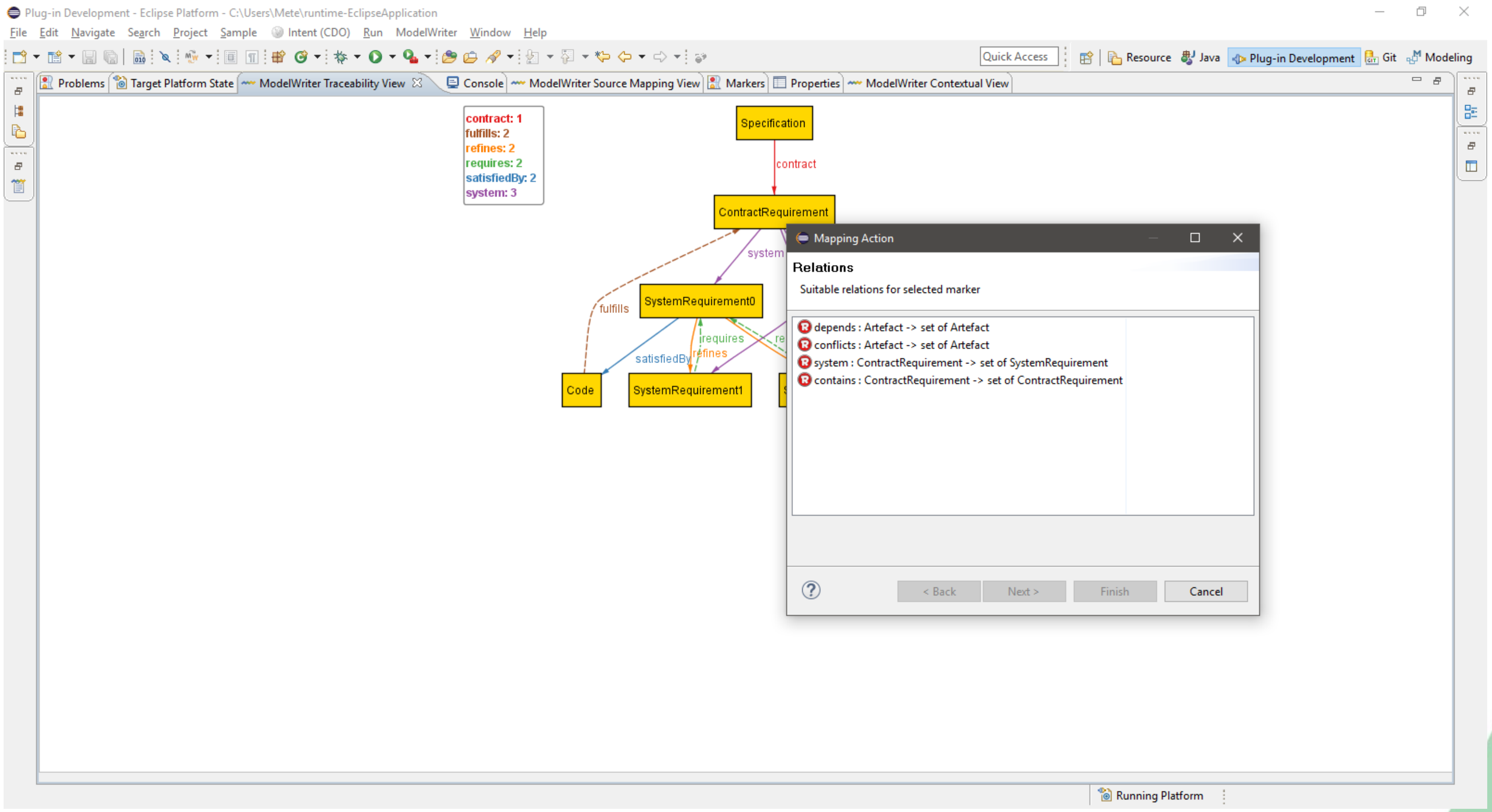


(d) Edge crossings minimized.



(e) Edges straightened.

Demonstration
Visualization in Action



Plug-in Development - Eclipse Platform - C:\Users\Mete\runtime-EclipseApplication

contract: 1
fulfills: 2
refines: 2
requires: 2
satisfiedBy: 2
system: 3

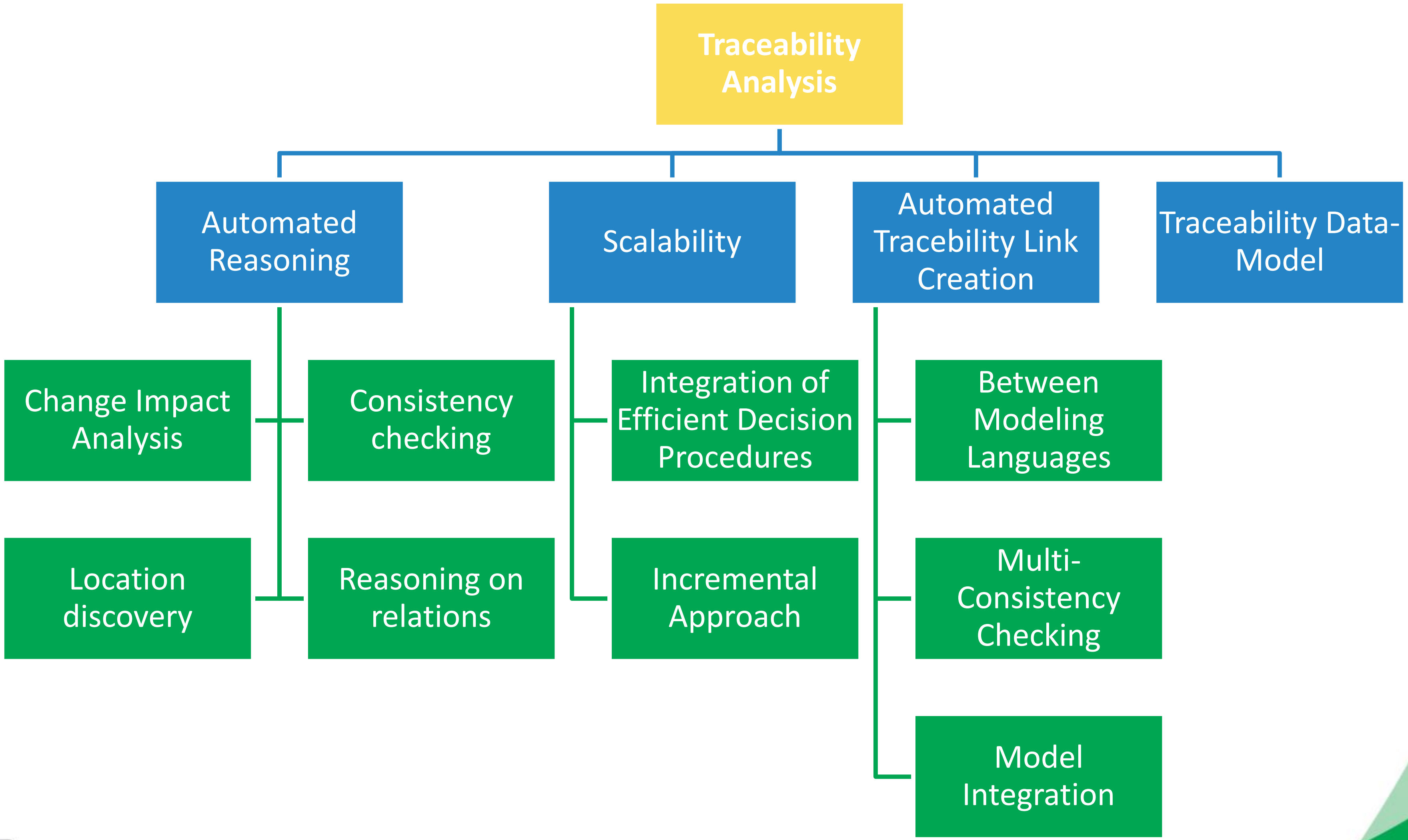
Mapping Action

Relations

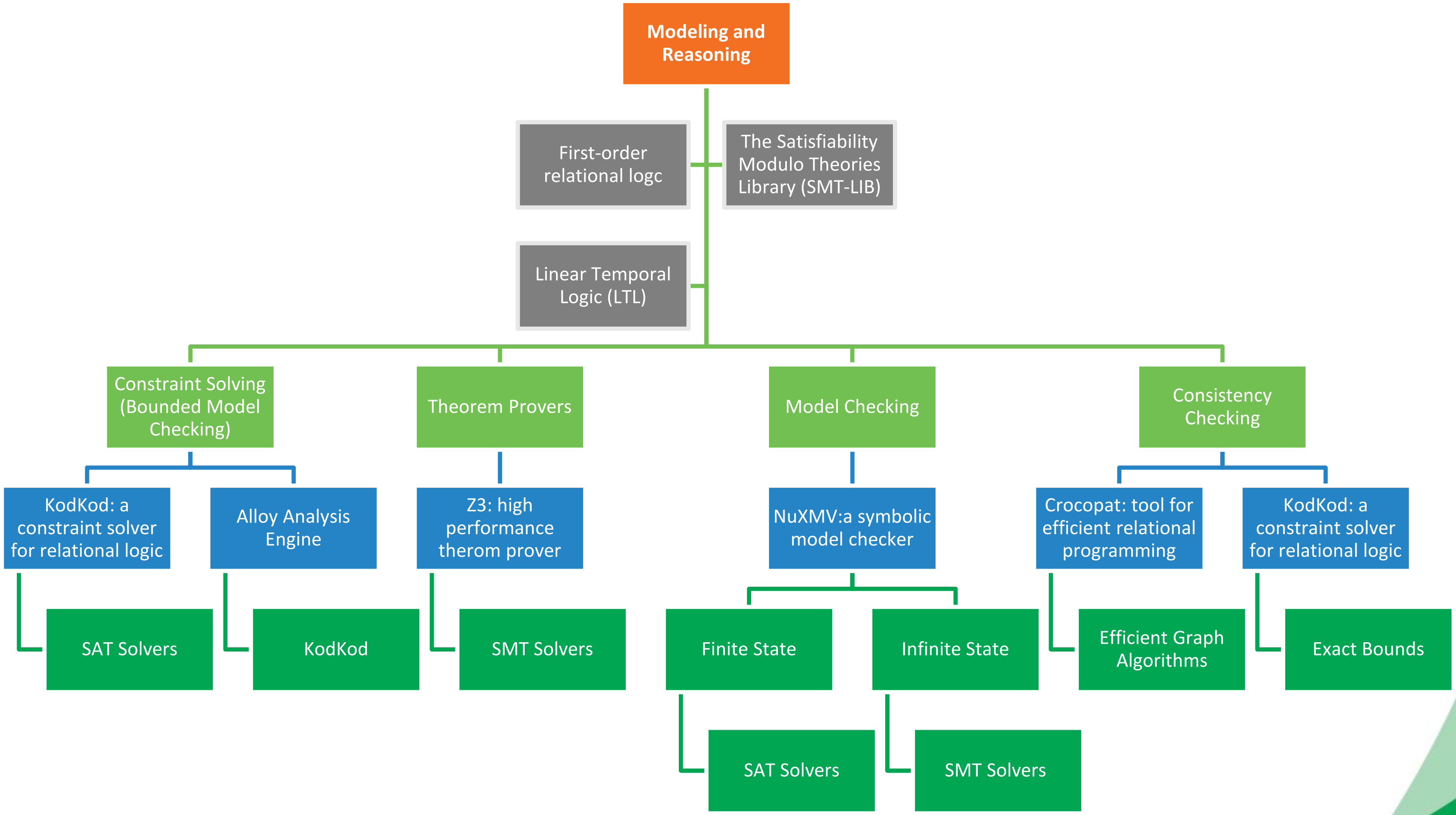
Suitable relations for selected marker

- depends : Artefact -> set of Artefact
- conflicts : Artefact -> set of Artefact
- system : ContractRequirement -> set of SystemRequirement
- contains : ContractRequirement -> set of ContractRequirement

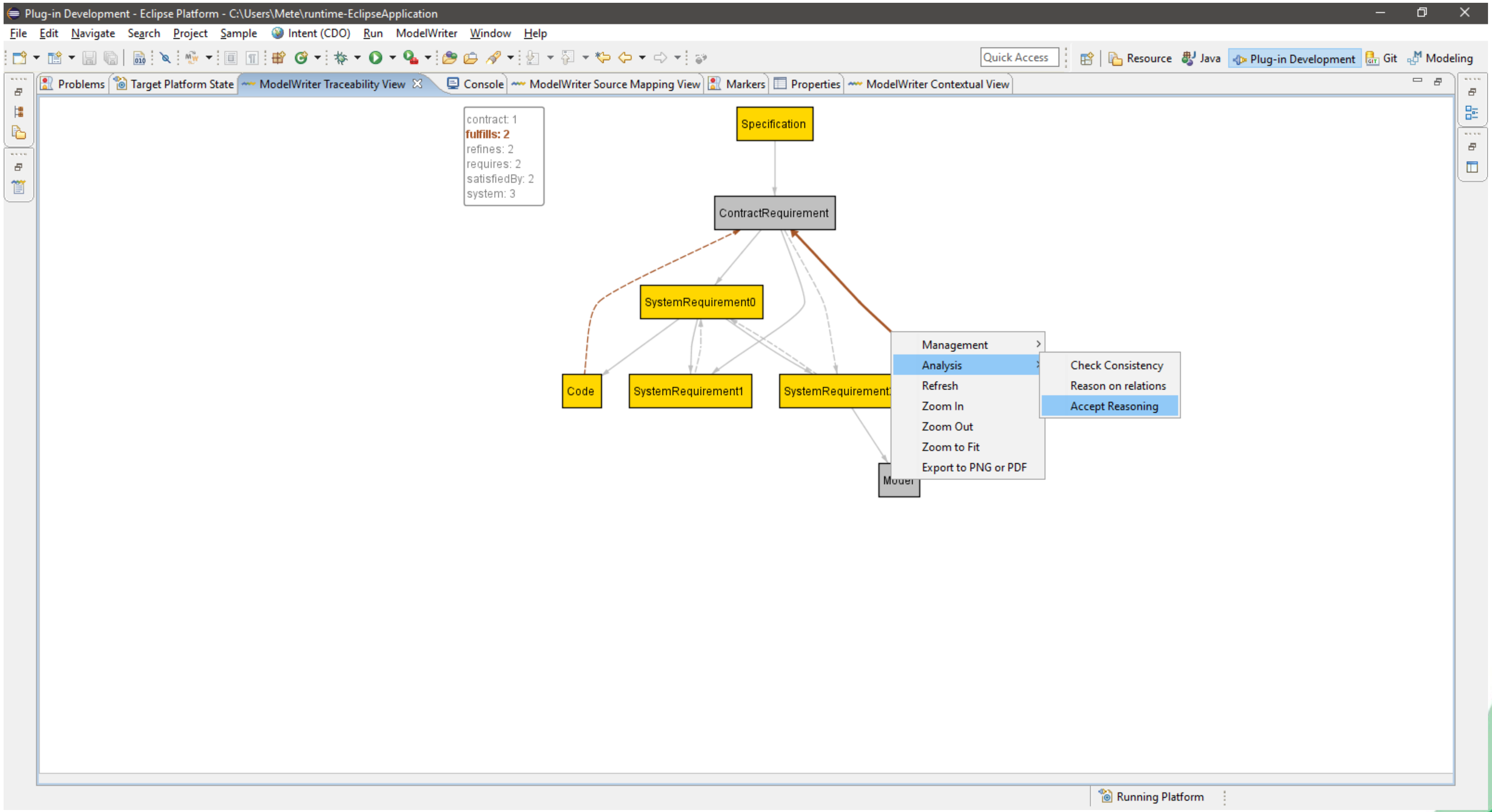
Traceability Analysis (WP4 & WP3) [UNIT]



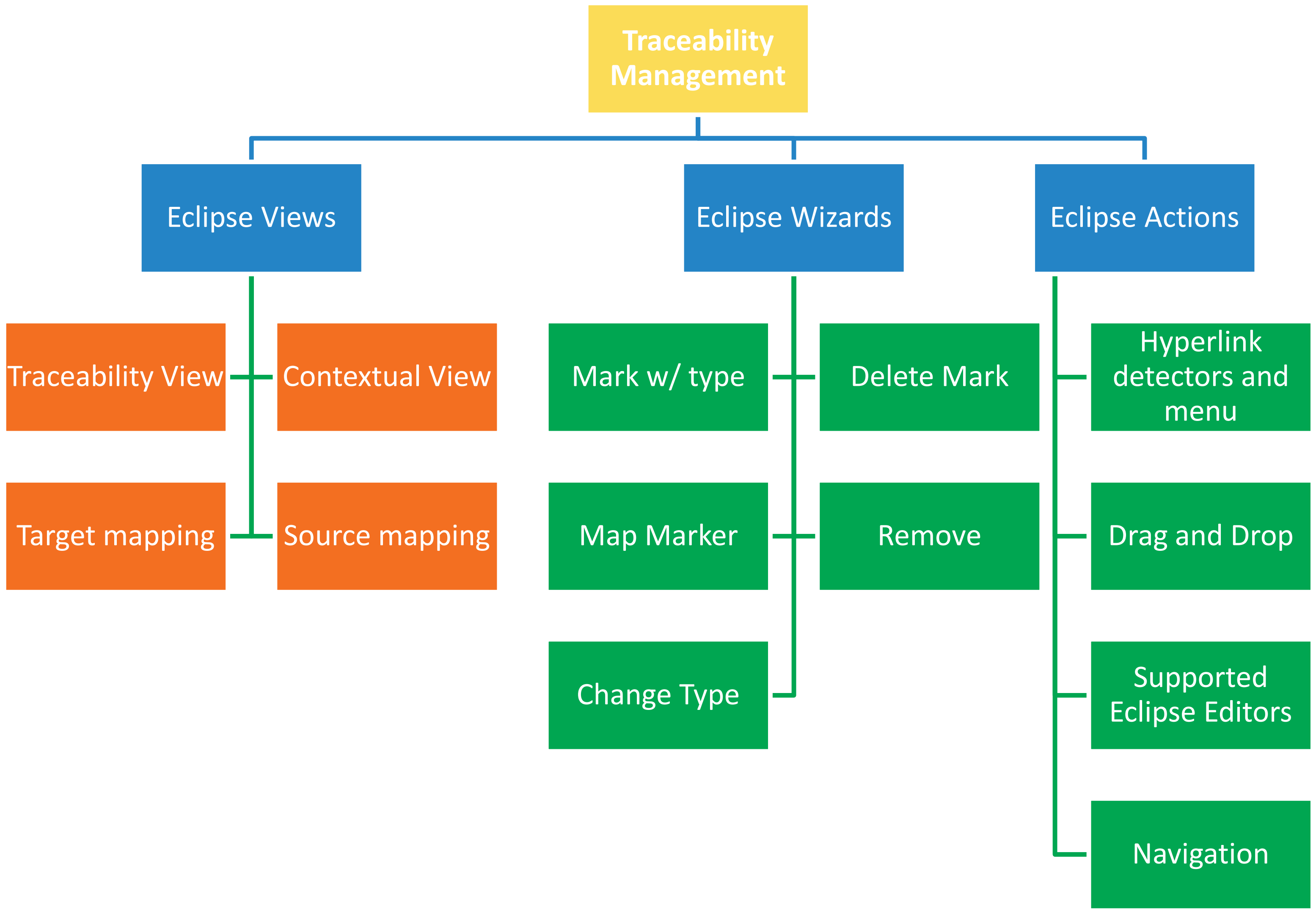
Modeling and Reasoning Approaches (WP3) [UNIT]


Modeling and Reasoning
First-order relational logc
The Satisfiability Modulo Theories Library (SMT-LIB)
Linear Temporal Logic (LTL)
Constraint Solving (Bounded Model Checking)
Theorem Provers
Model Checking
Consistency Checking
KodKod: a constraint solver for relational logic
Alloy Analysis Engine
Z3: high performance therom prover
NuXMV:a symbolic model checker
Crocopat: tool for efficient relational programming
KodKod: a constraint solver for relational logic
SAT Solvers
KodKod
SMT Solvers
Finite State
Infinite State
Efficient Graph Algorithms
Exact Bounds
SAT Solvers
SMT Solvers

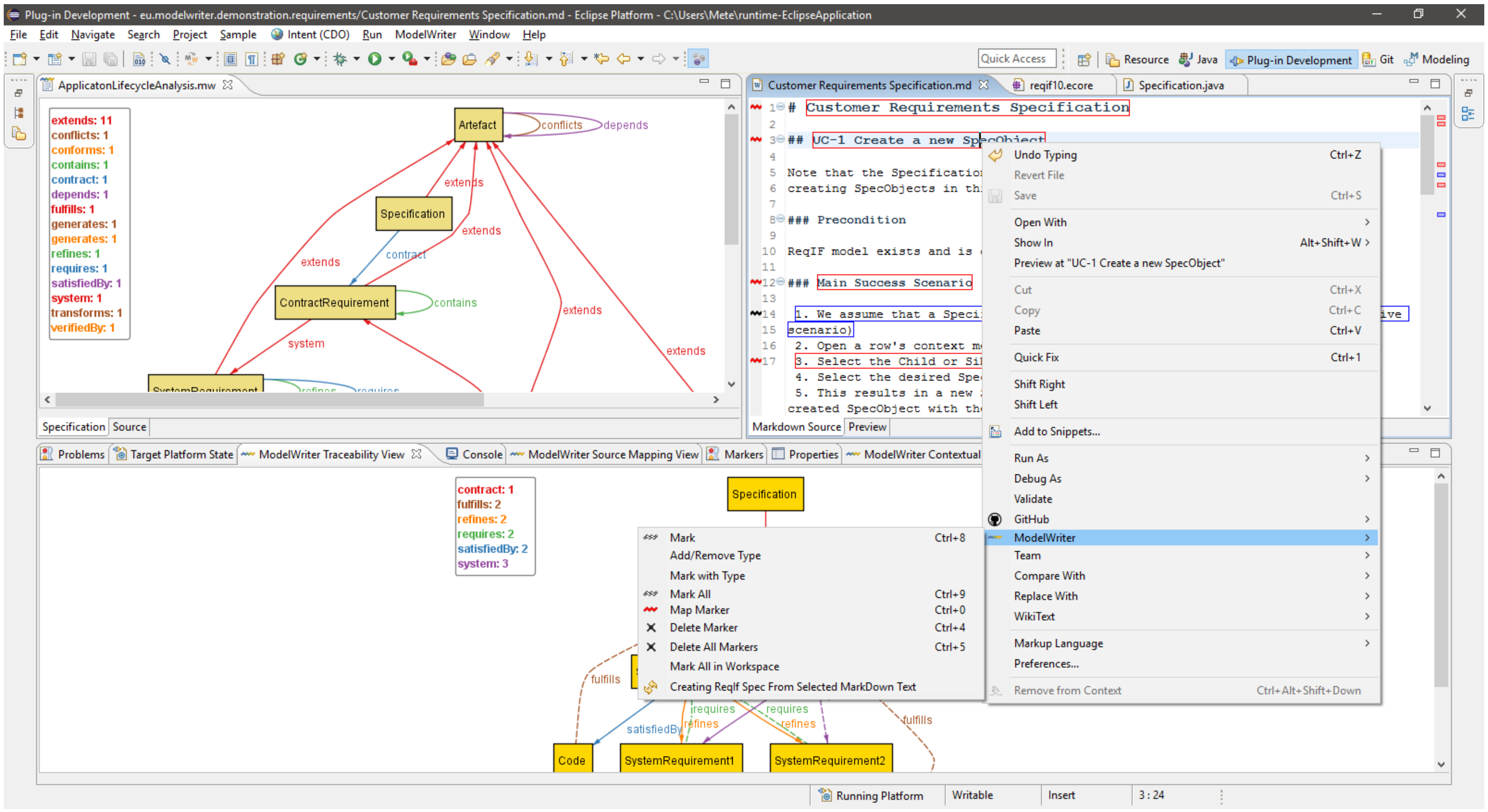
Demonstration
Traceability Analysis in Action


Plug-in Development - Eclipse Platform - C:\Users\Mete\runtime-EclipseApplication
File Edit Navigate Search Project Sample Intent (CDO) Run ModelWriter Window Help
Quick Access
Resource Java Plug-in Development Git Modeling
Problems Target Platform State ModelWriter Traceability View Console ModelWriter Source Mapping View Markers Properties ModelWriter Contextual View
contract: 1
fulfills: 2
refines: 2
requires: 2
satisfiedBy: 2
system: 3
Specification
ContractRequirement
SystemRequirement0
Code
SystemRequirement1
SystemRequirement
Model
Management
Analysis
Refresh
Zoom In
Zoom Out
Zoom to Fit
Export to PNG or PDF
Check Consistency
Reason on relations
Accept Reasoning
Running Platform

Traceability Management (WP6) [KoçSistem]



Demonstration
Traceability Management in Action



```
# Customer Requirements Specification

## UC-1 Create a new SpecObject

Note that the Specificatio
creating SpecObjects in th

### Precondition

ReqIF model exists and is

### Main Success Scenario

 1. We assume that a Speci                                            ive
scenario)
 2. Open a row's context m
 3. Select the Child or Si
 4. Select the desired Spe
 5. This results in a new
created SpecObject with th
```

Thank you for your attention We value your opinion and questions.