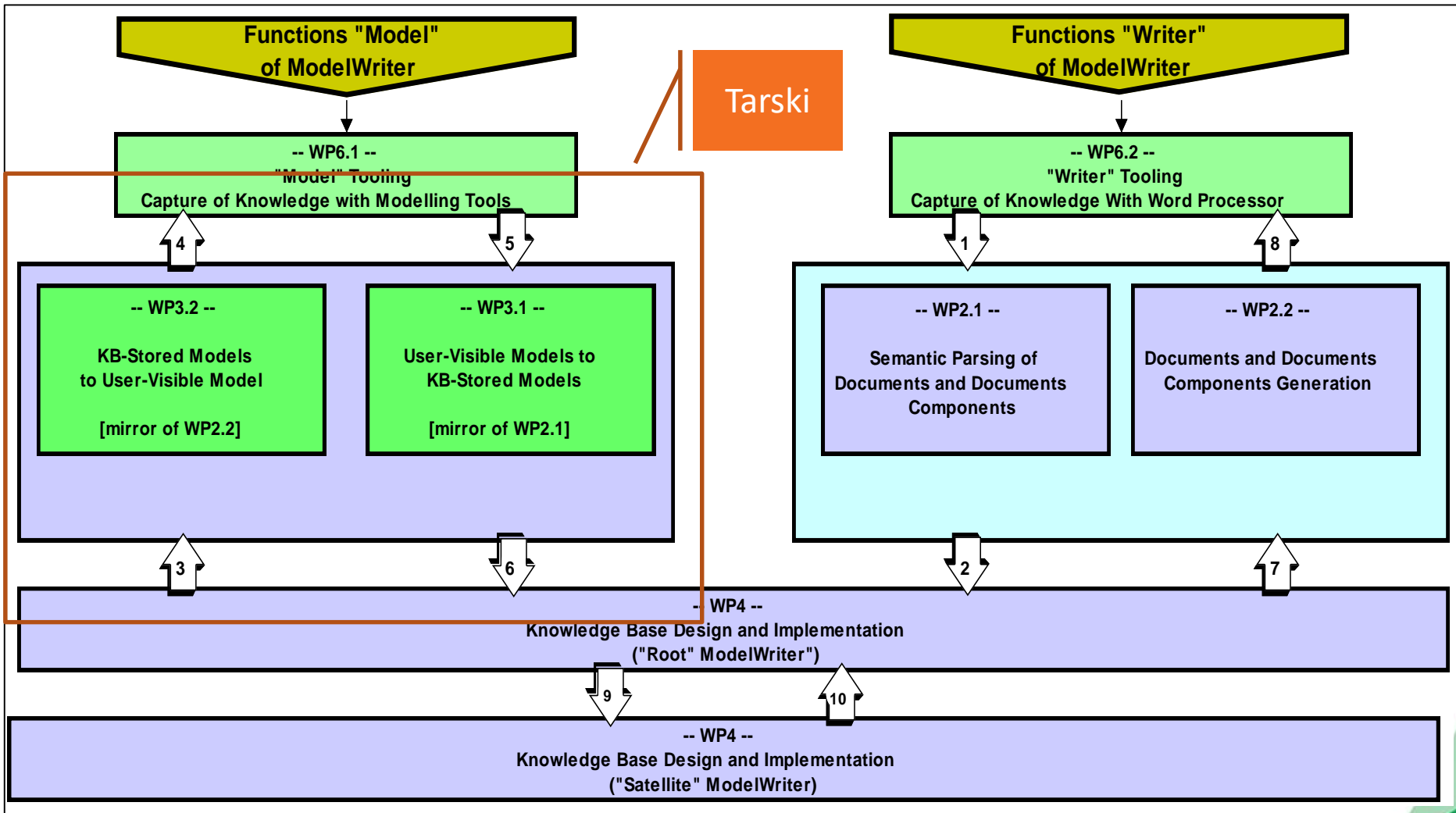


WP3 - Model to/from Knowledge Base Synchronization Mechanism

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

Technological components & interactions

Collaboration by WP interactions



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

The Paper is accepted by “The 32nd ACM Symposium on Applied Computing (SAC’2017), Programming Languages Track”.

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

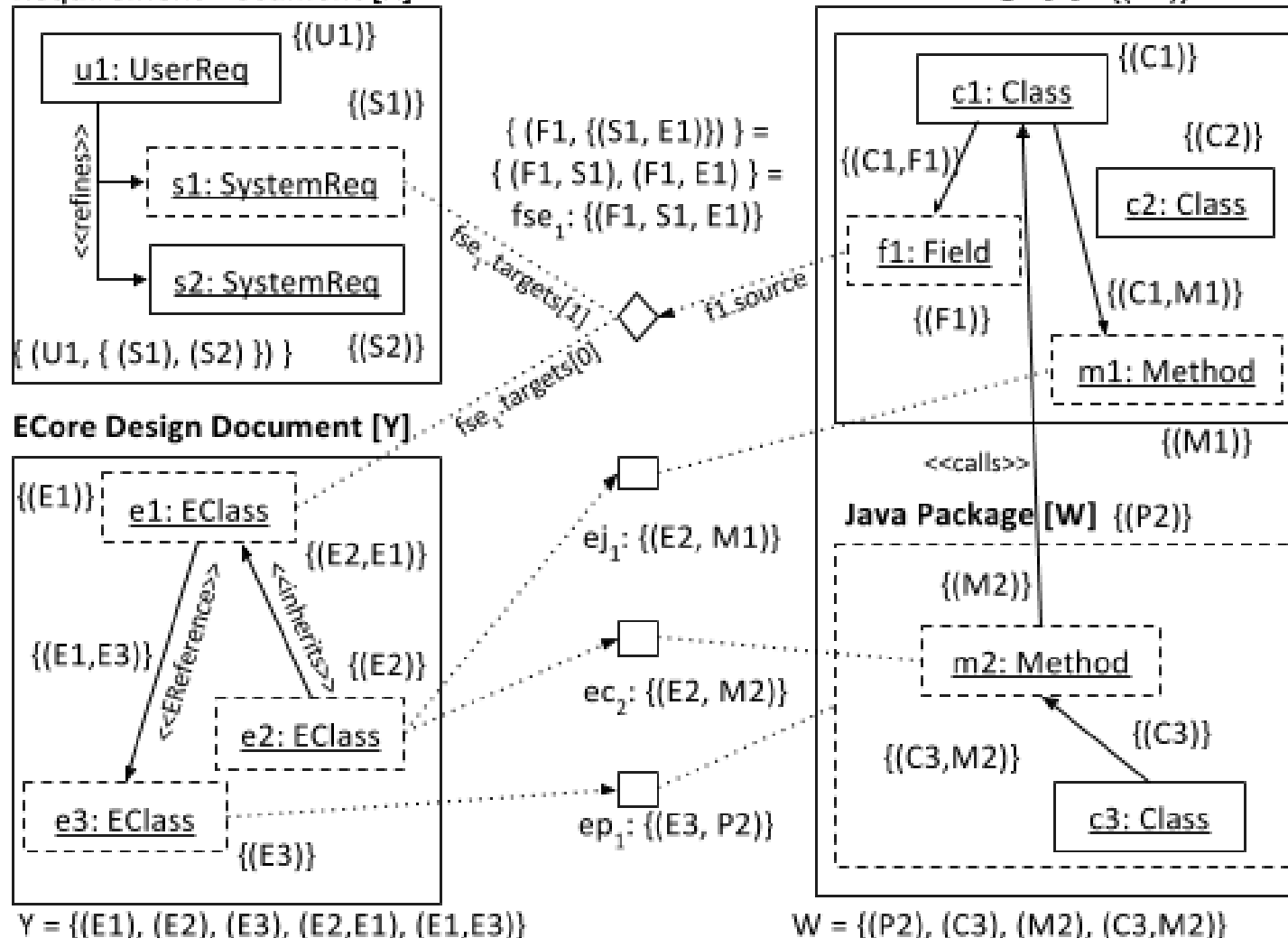
Consistency checking and repairing broken trace links



Technical Contributions @Tarski

Formalization of Traceability Semantics

$X = \{(U1), (S1), (S2), (U1,S1), (U1,S2)\}$ $Z = \{(P1), (C1), (C2), (F1), (M1), (C1,F1), (C1,M1)\} +$
Requirement Document [X] W **Java Package [Z]** $\{(P1)\}$



Technical Contributions @Tarski

Plug-in Development - eu.modelwriter.demonstration.requirements/Custom Requirements Specification.md - Eclipse Platform - C:\Users\Mete\run-time-EclipseApplication

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

Quick Access Resource Java Plug-in Development Git Modeling

AppliconLifecycleAnalysis.mw

conforms: 1
contains: 1
contract: 1
depends: 1
fulfills: 1
generates: 1
refines: 1
requires: 1
satisfiedBy: 1
system: 1
transforms: 1
verifiedBy: 1

Specification

ContractRequirement

SystemRequirement

extends

contract

contains

system

refines

requires

fulfills

Customer Requirements Specification.md

```

1 # Customer Requirements Specification
2
3 ## UC-1 Create a new SpecObject
4
5 Note that the Specification Editor is the main interface for users. Therefore,
6 creating SpecObjects in this editor is the main success scenario.
7
8 ### Precondition
9
10 ReqIF model exists and is open.
11
12 ### Main Success Scenario
13
14 1. We assume that a Specification exists and is open (not required for alternative
15 scenario)
16 2. Open a row's context menu (or in the empty editor space)
17 3. Select the Child or Sibling submenu.
18 4. Select the desired SpecObject Type (or none) from the submenu.
19 5. This results in a new SpecHierarchy being created that is linked to a newly
  
```

Markdown Source Preview

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

SystemRequirement2

fulfills

satisfiedBy

requires

refines

system

Management

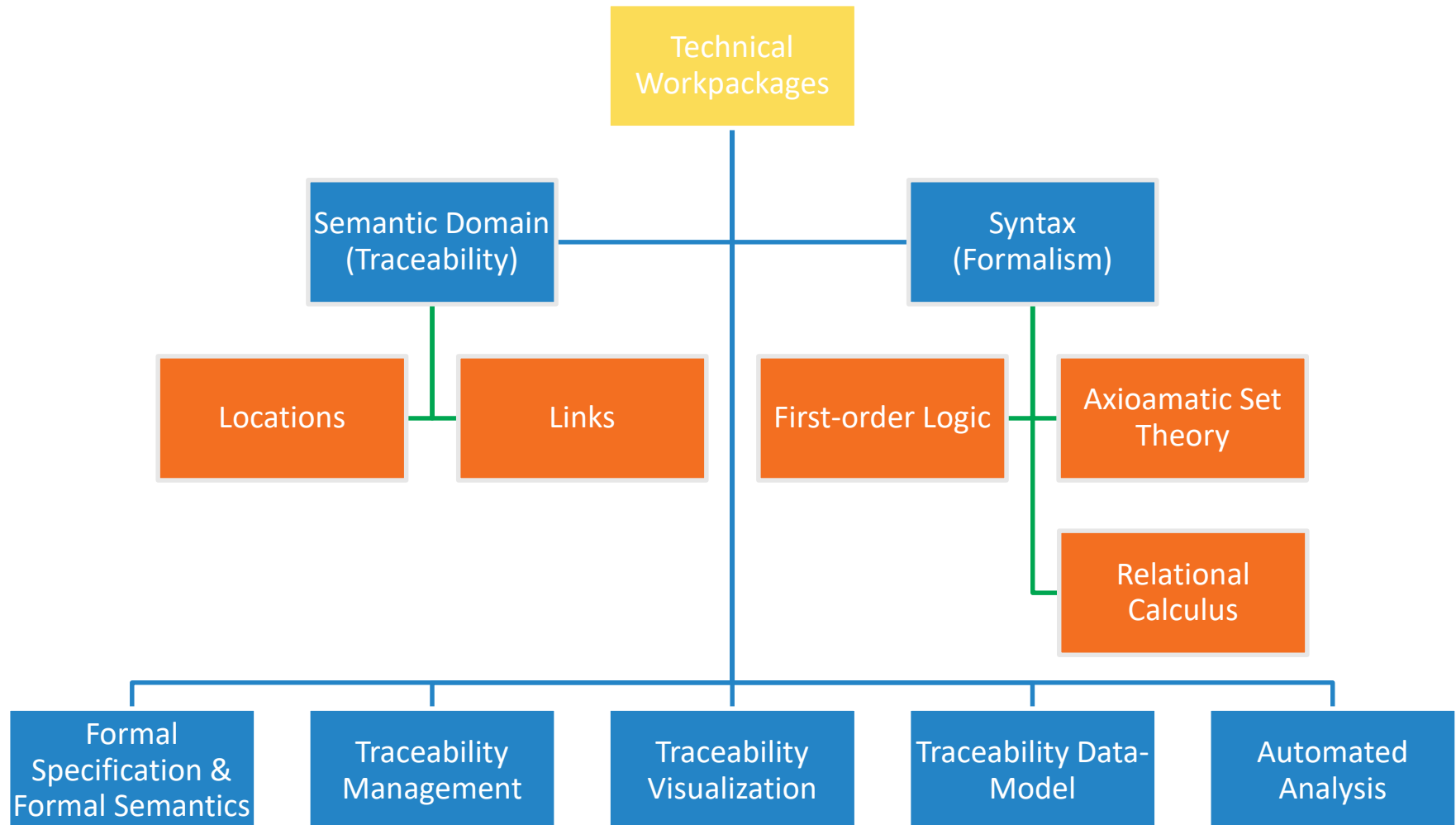
- Analysis
- Refresh
- Zoom In
- Zoom Out
- Zoom to Fit
- Export to PNG or PDF

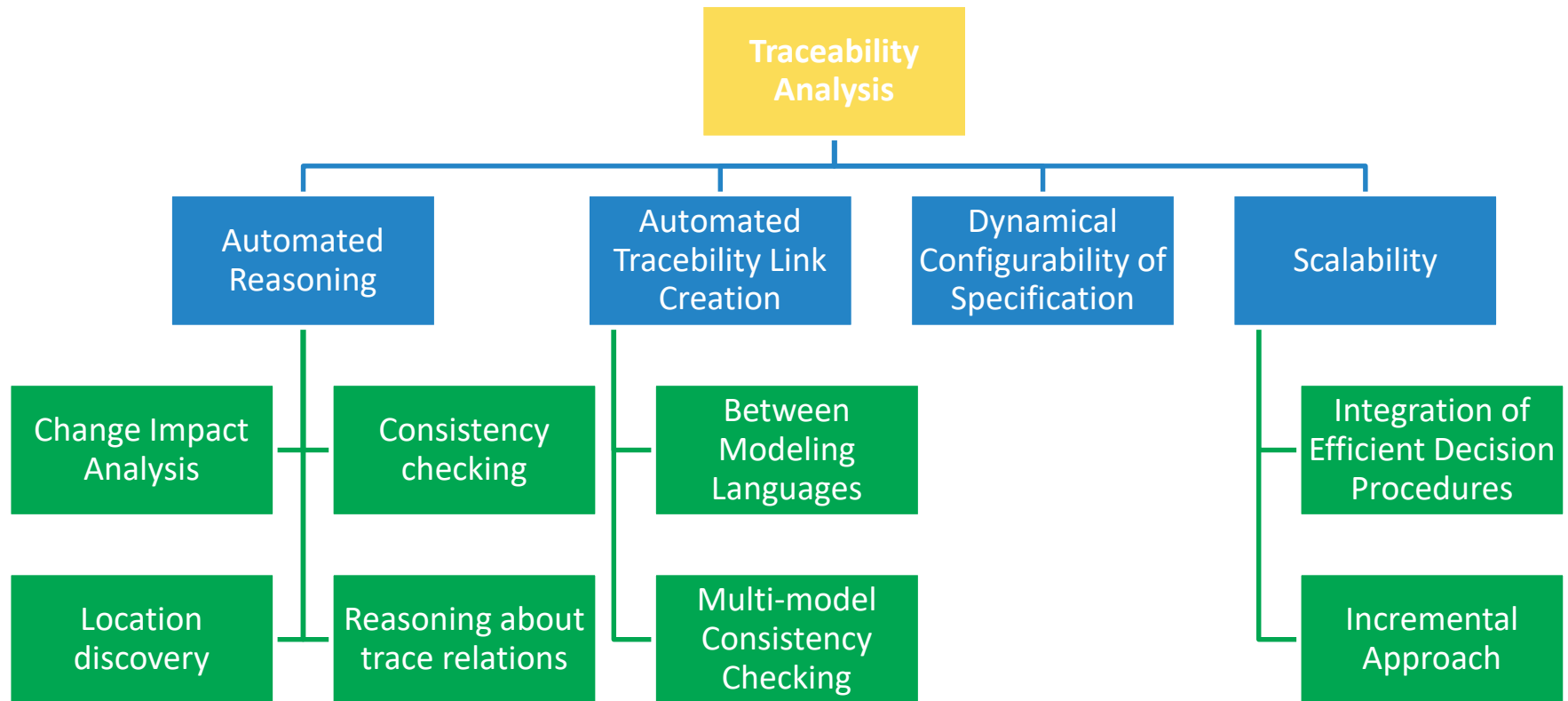
Check Consistency

Reason on relations

Running Platform Writable Insert 27: 173

Overview of Technical Contributions @Tarski





Tarski Approach

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

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

1

ApplicationLifecycleAnalysis.mw

```
16 -- Semantics@ContractRequirement
17 fact { all c: ContractRequirement | one c.~contract => no c.~contains }
18 fact { all c: ContractRequirement | no c.~contract => one c.~contains }
19
20 -- Locate@ReqIF
21 sig SystemRequirement extends Artefact {
22   satisfiedBy: set Implementation,
23   requires: set SystemRequirement,
24   refines: set SystemRequirement
25 }
26 -- Reason@system
27 fact { all s: SystemRequirement | one s.~system }
28 -- Reason@requires
29 fact { all s, s': SystemRequirement | s' in s.refines =>
30   abstract sig Implementation extends Artefact {
31     verifiedBy: set Verification,
32     fulfills: lone ContractRequirement
33   }
34   fact { all i: Implementation | some i.~satisfiedBy }
35   -- Reason@fulfills
36   fact { all i: Implementation, s: i.~satisfiedBy | i.fulfills s }
```

2

Customer Requirements Specification.md

```
1 # Customer Requirements Specification
2
3 ## UC-1 Create a new SpecObject A
4
5 Note that the Specification Editor is the main interface for users. Therefore, creating
6 SpecObjects in this editor is the main success scenario.
7
8 ### Precondition
9
```

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

3

Management

- Analysis
- Refresh
- Zoom In
- Zoom Out
- Zoom to Fit
- Export to PNG or PDF

Check Consistency

Reason on relations

ContractRequirement1

ContractRequirement0

SystemRequirement2

Code

SystemRequirement0

Model

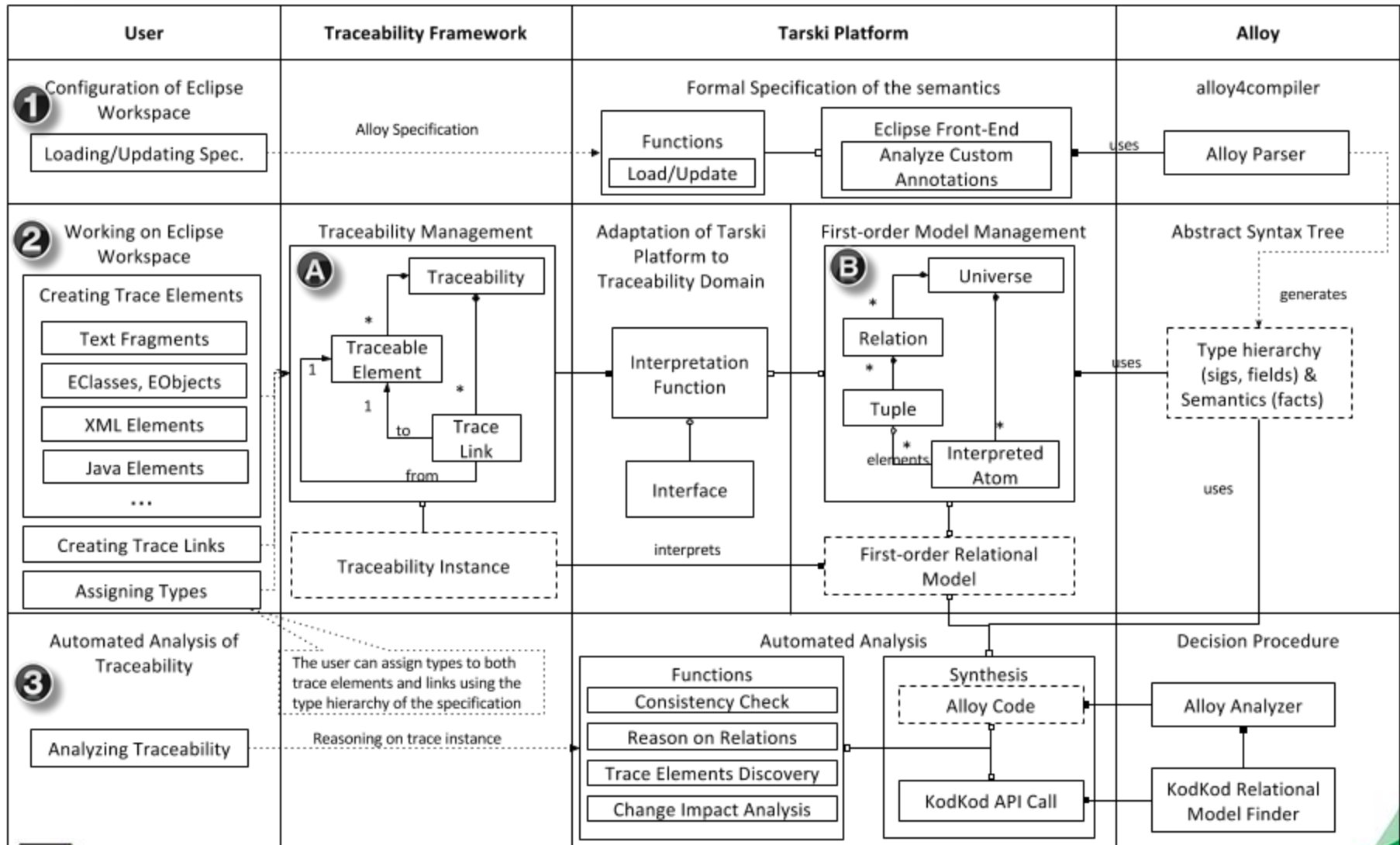
SystemRequirement1

Management

- Change Type
- Delete Atom
- Map Atom

Running Platform

Tarski Approach



Types/Component Ontology derived from the specification

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

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

Quick Access Resource Java Plug-in Development Git Modeling

AppliconLifecycleAnalysis.mw

```
1 module eu.modelwriter.actions.havelsan/alm
2
3 abstract sig Artefact {
4   depends: set Artefact,
5   conflicts: set Artefact
6 }
7 fact { ~conflicts in conflicts
8
9 one sig Specification extends
10  contract: some ContractRequirement
11 }
12 fact { ~Locate@Text
13 }
14 sig ContractRequirement extends
15  system: set SystemRequirement
16  contains: set ContractRequirement
17 }
18 fact { ~Semantics@ContractRequirement
19 }
20 fact { all c: ContractRequirement
21 }
22 sig SystemRequirement extends
23  satisfiedBy: set Implementation
24  requires: set SystemRequirement
25  refines: set SystemRequirement
26 }
27 fact { ~Reason@system
28 }
29 fact { all s: SystemRequirement
30 }
31 fact { all s, s': SystemRequirement
32 }
33 abstract sig Implementation extends
34  verifiedBy: set Verification
35  fulfills: lone ContractRequirement
36 }
37 fact { all i: Implementation
38 }
39 fact { all i: Implementation, s: i.~satisfiedBy | i.fulfills = s.~system }
40
41 }
42 }
```

Preferences

type filter text

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\workspace\Demonstrations\eu.modelwriter.demonstration.requirements\AppliconLifecycleAnalysis.mw

Restore Defaults Apply OK Cancel

contractRequirement0

ContractRequirement2

SystemRequirement1

Model

requires

refines

system

contains

fulfills

satisfiedBy

Running Platform

Assigning Unary Relations to a Traceable Elements

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

```
1 module eu/modelwriter/actions/havelsan/alm
2
3 abstract sig Artefact {
4   depends: set Artefact,
5   conflicts: set Artefact
6 -- Reason@conflicts
7 fact {~conflicts in conflicts}
8
9 one sig Specification extends Artefact {
10  contract: some ContractRequirement
11 -- Locate@Text
12 -- Discover@ContractRequirement expect 3
13 sig ContractRequirement extends Artefact {
14  system: set SystemRequirement,
15  contains: set ContractRequirement
16
17 -- Semantics@ContractRequirement
18 fact {all c: ContractRequirement | one c.~contract => no c.
19 fact {all c: ContractRequirement | no c.~contract => one c.
```

Customer Requirements Specification.md

```
1 ## Customer Requirements Specification
2
3 ## UC-1 Create a new SpecObject
```

Create a Trace Element with Type

- universe
 - Artefact (abs)
 - Specification
 - ContractRequirement**
 - SystemRequirement
 - Implementation (abs)
 - Model
 - Code
 - Component
 - Verification (abs)
 - Simulation
 - Analysis
 - Test

Finish Cancel

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

ContractRequirement1 ContractRequirement0

SystemRequirement0 ContractRequirement2

Code SystemRequirement2 SystemRequirement1

fulfills system contains requires refines satisfiedBy

Running Platform Writable Insert 9:6

Assigning Binary Relations to a Trace Link

Plug-in Development - eu.modelwriter.demonstration.requirements/Custom 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

```
1 module eu/modelwriter/actions/havelsan/alm
2
3 abstract sig Artefact {
4   depends: set Artefact,
5   conflicts: set Artefact
6 -- Reason@conflicts
7 fact {~conflicts in conflicts}
8
9 one sig Specification extends Artefact {
10  contract: some ContractRequirement
11 -- Locate@Text
12 -- Discover@ContractRequirement expect 3
13 sig ContractRequirement extends Artefact {
14  system: set SystemRequirement,
15  contains: set ContractRequirement
16
17 -- Semantics@ContractRequirement
18 fact {all c: ContractRequirement | one c.~contract => no c}
19 fact {all c: ContractRequirement | no c.~contract => one c}
```

Customer Requirements Specification.md

```
1 ## Customer Requirements Specification
2
3 ## UC-1 Create a new SpecObject
4
5 Note that the Specification Editor is the main interface for users. Therefore,
6 creating SpecObjects in this editor is the main success scenario.
```

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

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

ContractRequirement1 ContractRequirement0 ContractRequirement2

SystemRequirement0

Code SystemRequirement2 SystemRequirement1

contract contract system contains fulfills requires refines satisfiedBy

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/Custom 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

```
1 module eu/modelwriter/actions/havelsan/alm
2
3 abstract sig Artefact {
4   depends: set Artefact,
5   conflicts: set Artefact
6 -- Reason@conflicts
7 fact {~conflicts in conflicts}
8
9 one sig Specification extends Artefact {
10   contract: some ContractRequirement
11 -- Locate@Text
12 -- Discover@ContractRequirement expect 3
13 sig ContractRequirement extends Artefact {
14   system: set SystemRequirement,
15   contains: set ContractRequirement
16
17 -- Semantics@ContractRequirement
18 fact {all c: ContractRequirement | one c.~contract => }
19 fact {all c: ContractRequirement | no c.~contract => }
```

Customer Requirements Specification.md

```
1 # Customer Requirements Specification
2
3 ## UC-1 Create a new SpecObject
4
5 Note that the Specification Editor is the main interface for users. Therefore,
6 creating SpecObjects in this editor is the main success scenario.
```

Create a trace relation

Markers

"Main Success Scenario"

- eu.modelwriter.demonstration.requirements
 - reqif10.ecore
 - ReqIf (Model\$0)
 - org.eclipse.rmfm.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

contain
contr
fulfills:
refines
require
satisfie
system

fulfills

system

contains

requires

refines

satisfies

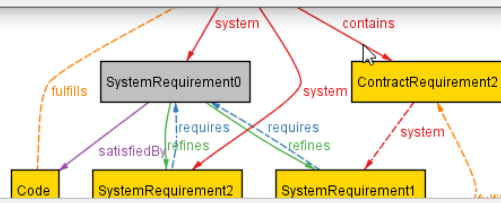
Code

SystemRequirement2

SystemRequirement1

ContractRequirement2

Running Platform Writable Insert 9:6



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

```

1 module eu/modelwriter/actions/havelsan/alm
2
3 abstract sig Artefact {
4   depends: set Artefact,
5   conflicts: set Artefact
6 -- Reason@conflicts
7 fact {~conflicts in conflicts}
8
9 one sig Specification extends Artefact {
10  contract: some ContractRequirement
11 -- Locate@Text
12 -- Discover@ContractRequirement expect 3
13 sig ContractRequirement extends Artefact {
14  system: set SystemRequirement,
15  contains: set ContractRequirement
16

```

Customer Requirements Specification.md

```

1 ## Customer Requirements Specification
2
3 ## UC-1 Create a new SpecObject
4
5 Note that the Specification Editor is the main interface for users. Therefore,
6 creating SpecObjects in this editor is the main success scenario.
7
8 ### Precondition
9
10 Req1: model exists and is open.
11
12 ### Main Success Scenario
13
14 1. We assume that a Specification exists and is open (not required for alternative
15 scenario)
16
17 2. Open a row's context menu (or in the empty editor space)
18
19 3. Select the Child or Sibling submenu.

```

Source Specification

Problems Console Markers Properties Tarski Master View Tarski Contextual View Tarski Traceability View

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

Specification

ContractRequirement1

ContractRequirement0

SystemRequirement0

Code

SystemRequirement2

ContractRequirement2

SystemRequirement1

Management

- Analysis
- Refresh
- Zoom In
- Zoom Out
- Zoom to Fit
- Export to PNG or PDF

Check Consistency

Reason on Relations

Discover Atoms

Clear All Reasoned Tuples

Running Platform

Dynamical Configuration & Model Management

Plug-in Development - eu.modelwriter.demonstration.requirements/ApplicatonLifecycleAnalysis.mw - 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

```
1 module eu/modelwriter/actions/havelsan/alm
2
3 abstract sig Artefact {
4   depends: set Artefact,
5   conflicts: set Artefact
6 } -- Reason@conflicts
7 fact { ~conflicts in conflicts }
8
9 one sig Specification extends Artefact {
10  contract: some ContractRequirement
11 } -- Locate@Text
12 -- Discover@ContractRequirement expect 3
13 sig ContractRequirement extends Artefact {
14  system: set SystemRequirement,
15  contains: set ContractRequirement
16 }
17 -- Semantics@ContractRequirement
18 fact { all c: ContractRequirement | one c.~contract => no c.~contains }
19 fact { all c: ContractRequirement | no c.~contract => one c.~contains }
20
21 -- Locate@ReqIF
22 sig SystemRequirement extends Artefact {
23  satisfiedBy: set Implementation,
24  requires: set SystemRequirement,
25  refines: set SystemRequirement
26 }
27 -- Reason@system
28 fact { all s: SystemRequirement | one s.~system }
29
30 -- Reason@requires
31 fact { all s, s': SystemRequirement | s' in s.refines => s in s'.requires }
32
33 abstract sig Implementation extends Artefact {
34  verifiedBy: set Verification,
35  fulfills: lone ContractRequirement
36 }
37 fact { all i: Implementation | some i.~satisfiedBy }
38 -- Reason@fulfills
39 fact { all i: Implementation, s: i.~satisfiedBy | i.fulfills = s.~system }
40
41 -- Locate@FMF
```

Source Specification

Tarski Traceability View

Customer Requirements Specification.md

Specification.java

reqif10.ecore

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

Management
Refresh
Zoom In
Zoom Out
Zoom to Fit
Export to PNG or PDF

Change Type
Delete Atom
Map Atom

Specification

ContractRequirement1

ContractRequirement2

SystemRequirement0

SystemRequirement2

SystemRequirement1

Code

Model

fulfills

requires

refines

satisfiedBy

system

contract

contract

Management

Refresh

Zoom In

Zoom Out

Zoom to Fit

Export to PNG or PDF

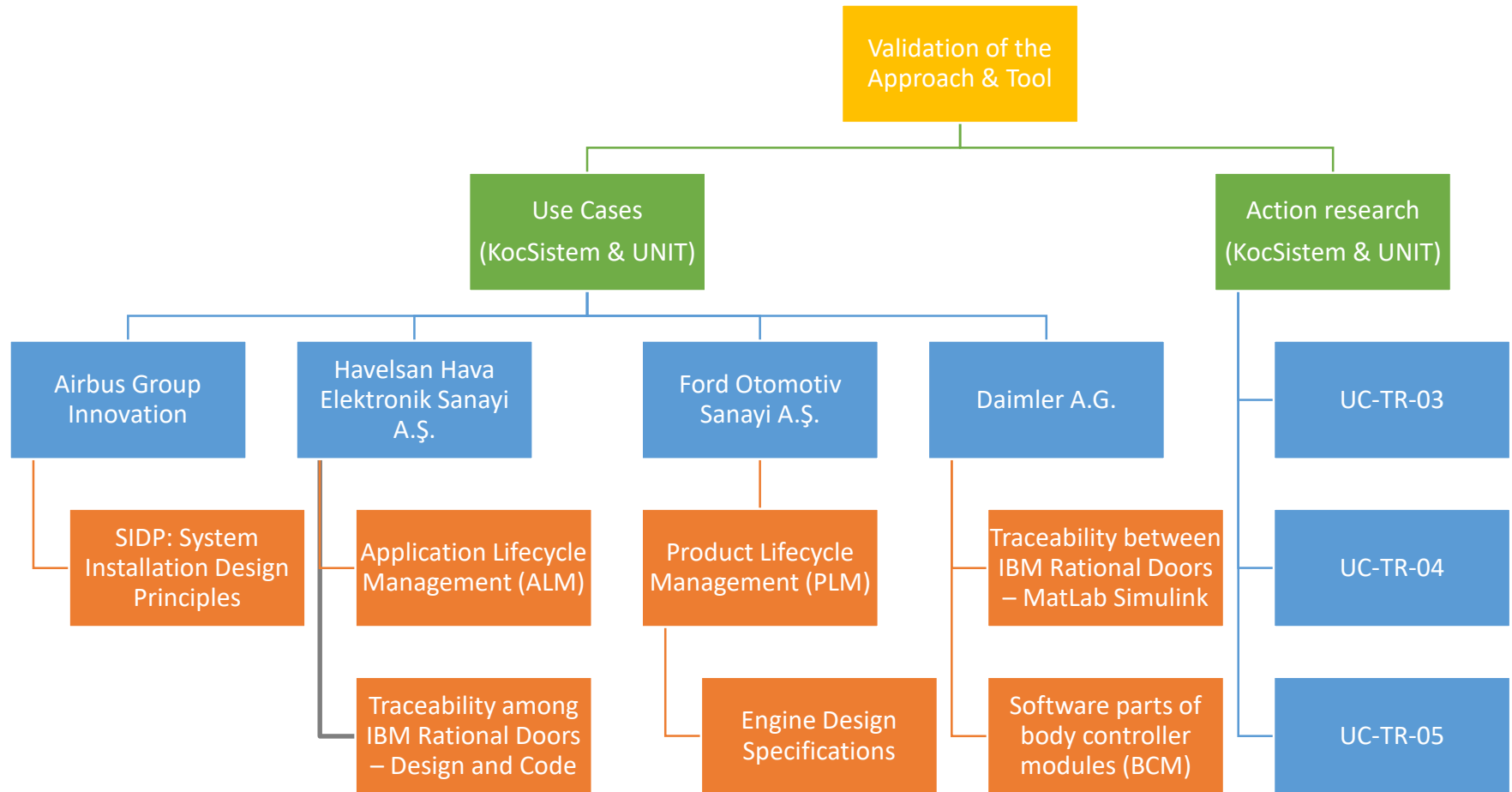
Change Type

Delete Atom

Map Atom

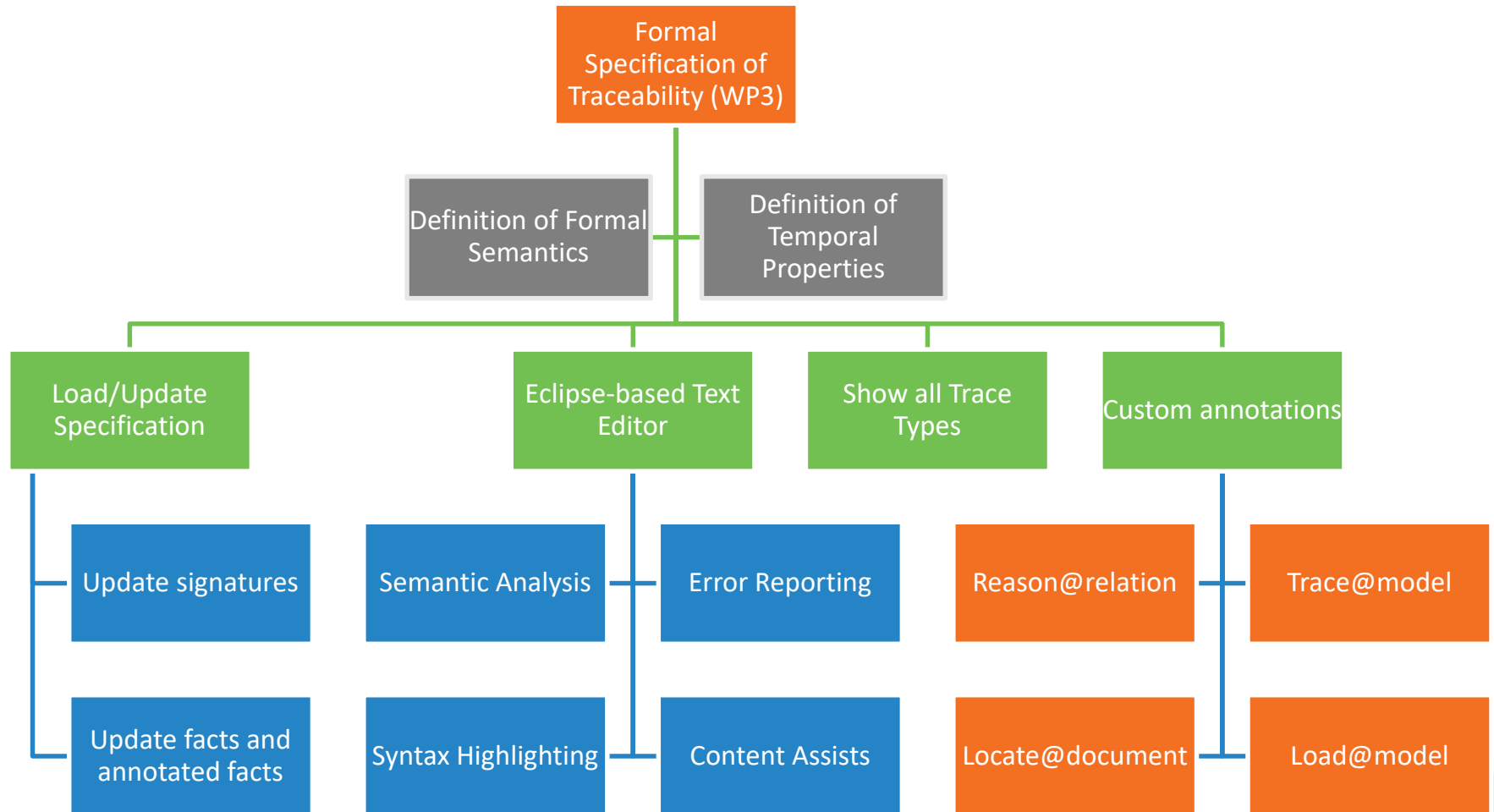
Running Platform

Validation of the Approach and Tool

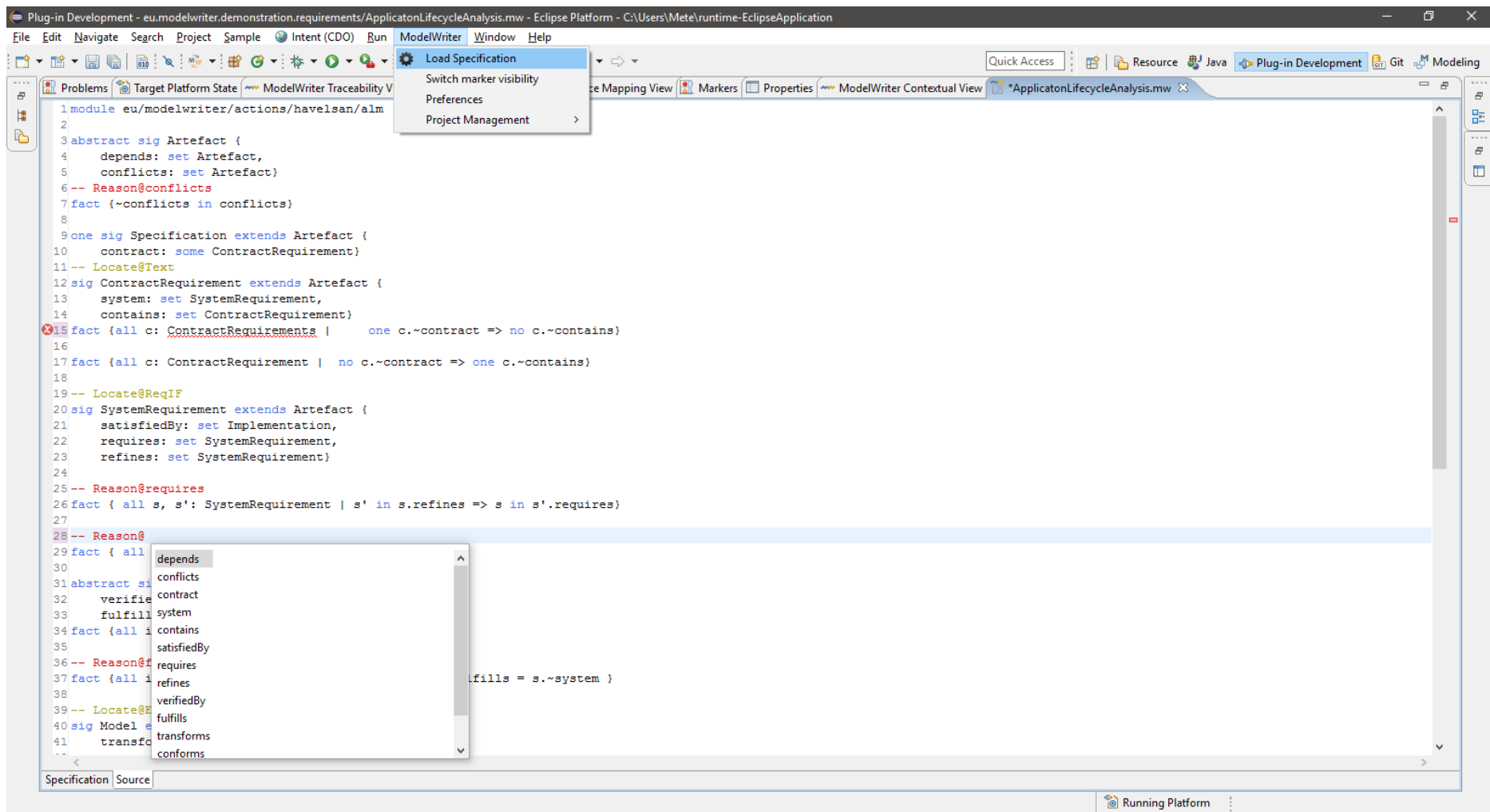


Formal Specification of Traceability (WP3)

[UNIT]

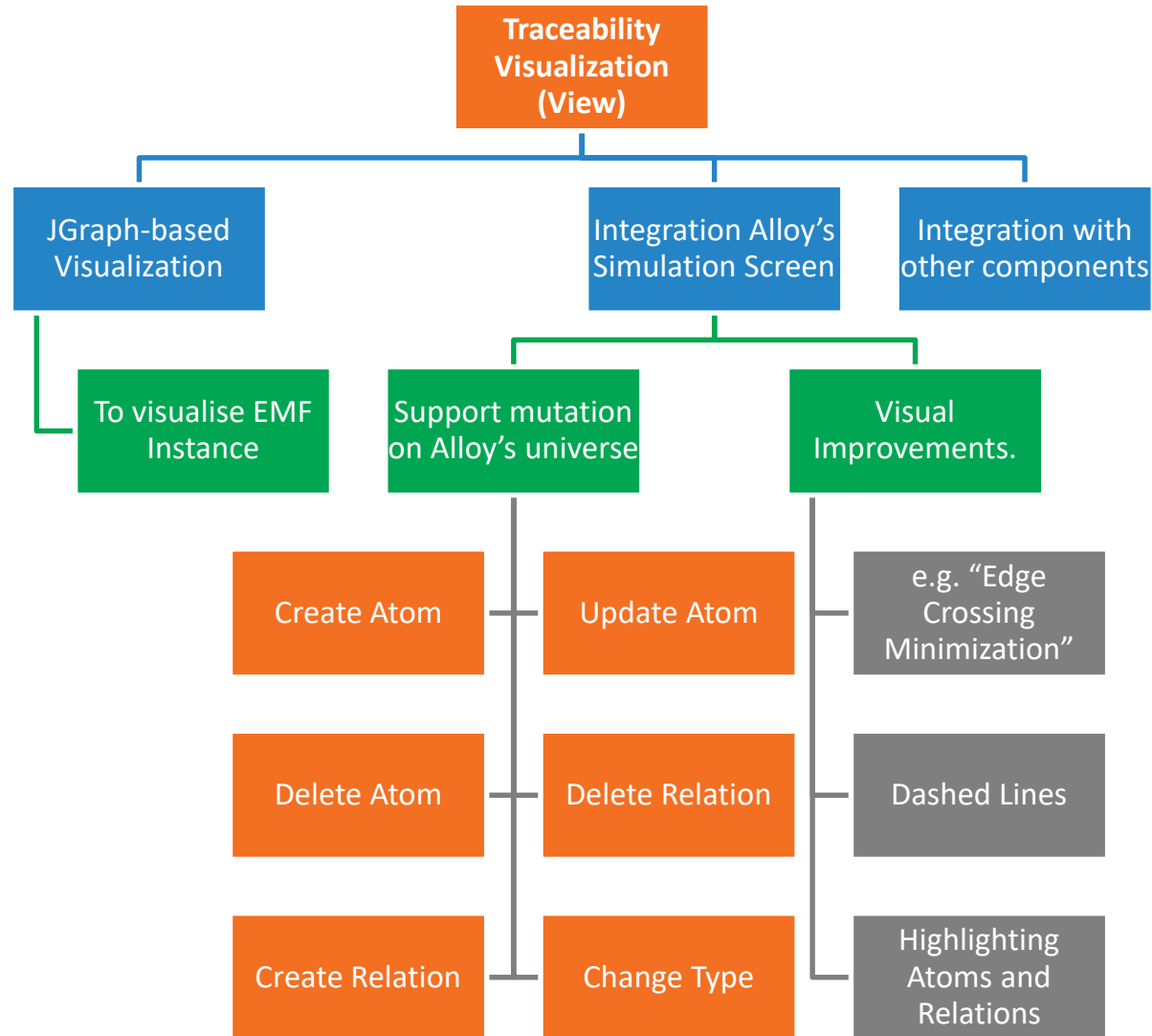


Demonstration Textual Editor in Action

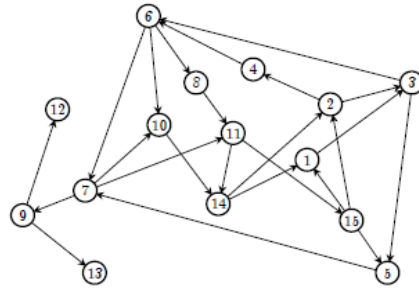


Traceability Visualization/View (WP3)

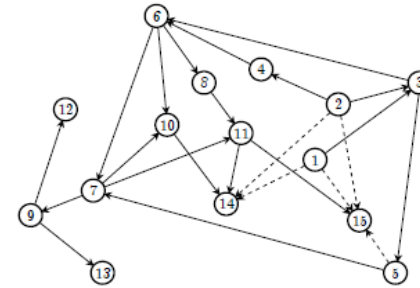
[KoçSistem]



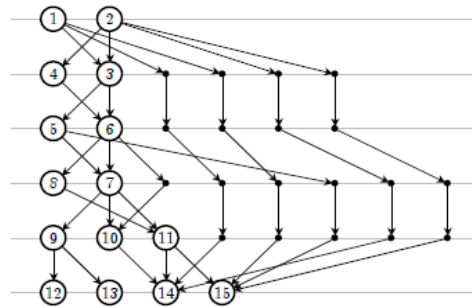
Layered/Hierarchical Graph Drawing Sugiyama Framework



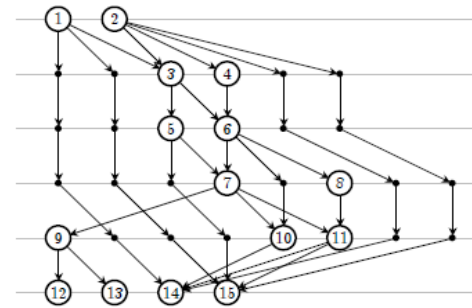
(a) Input digraph, G .



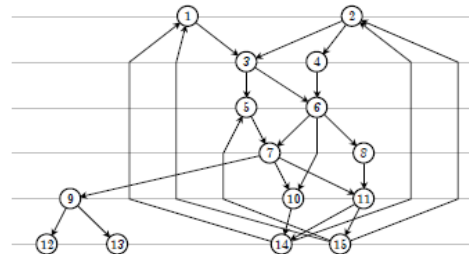
(b) Cycles removed.



(c) After leveling.

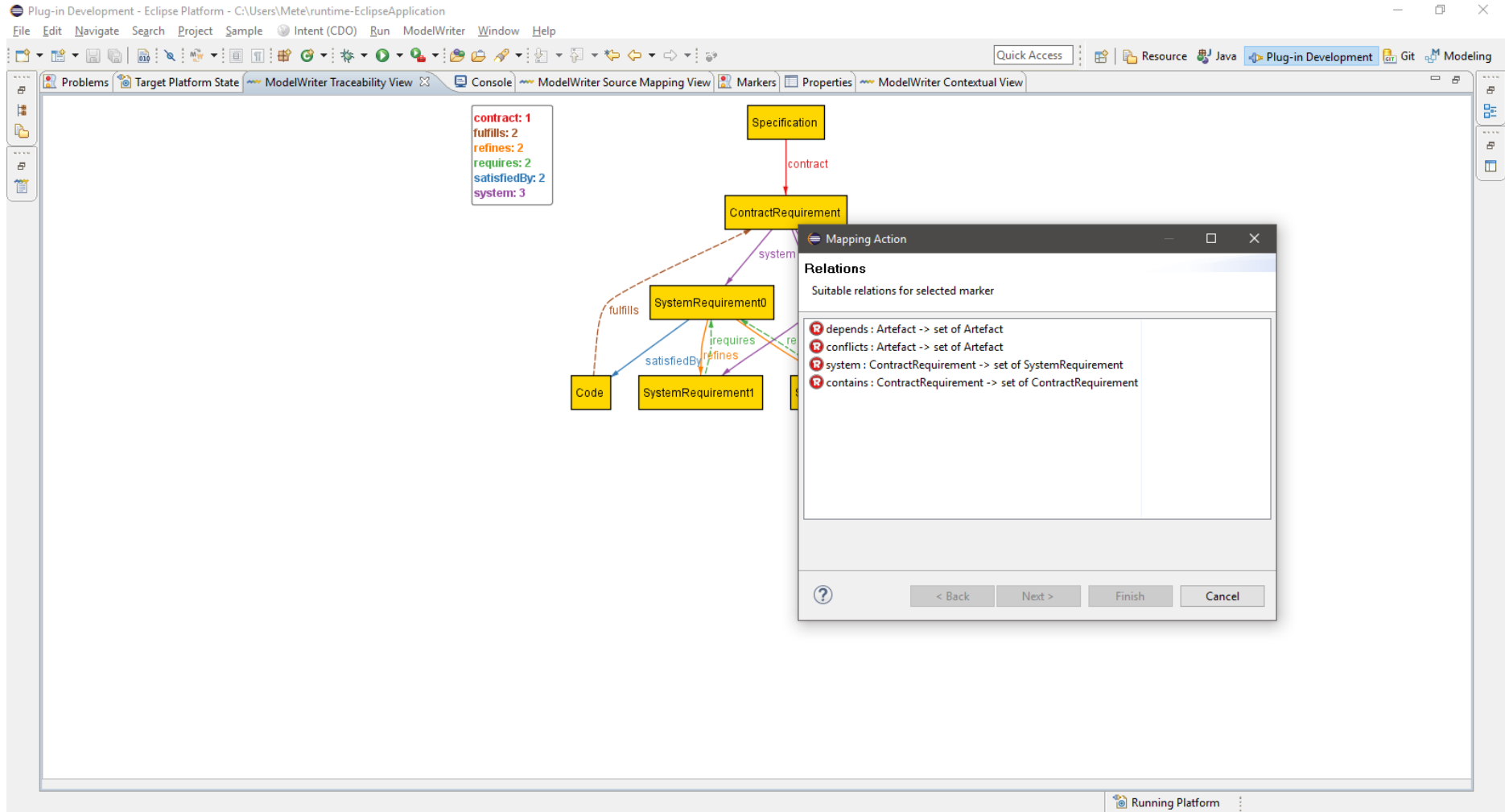


(d) Edge crossings minimized.



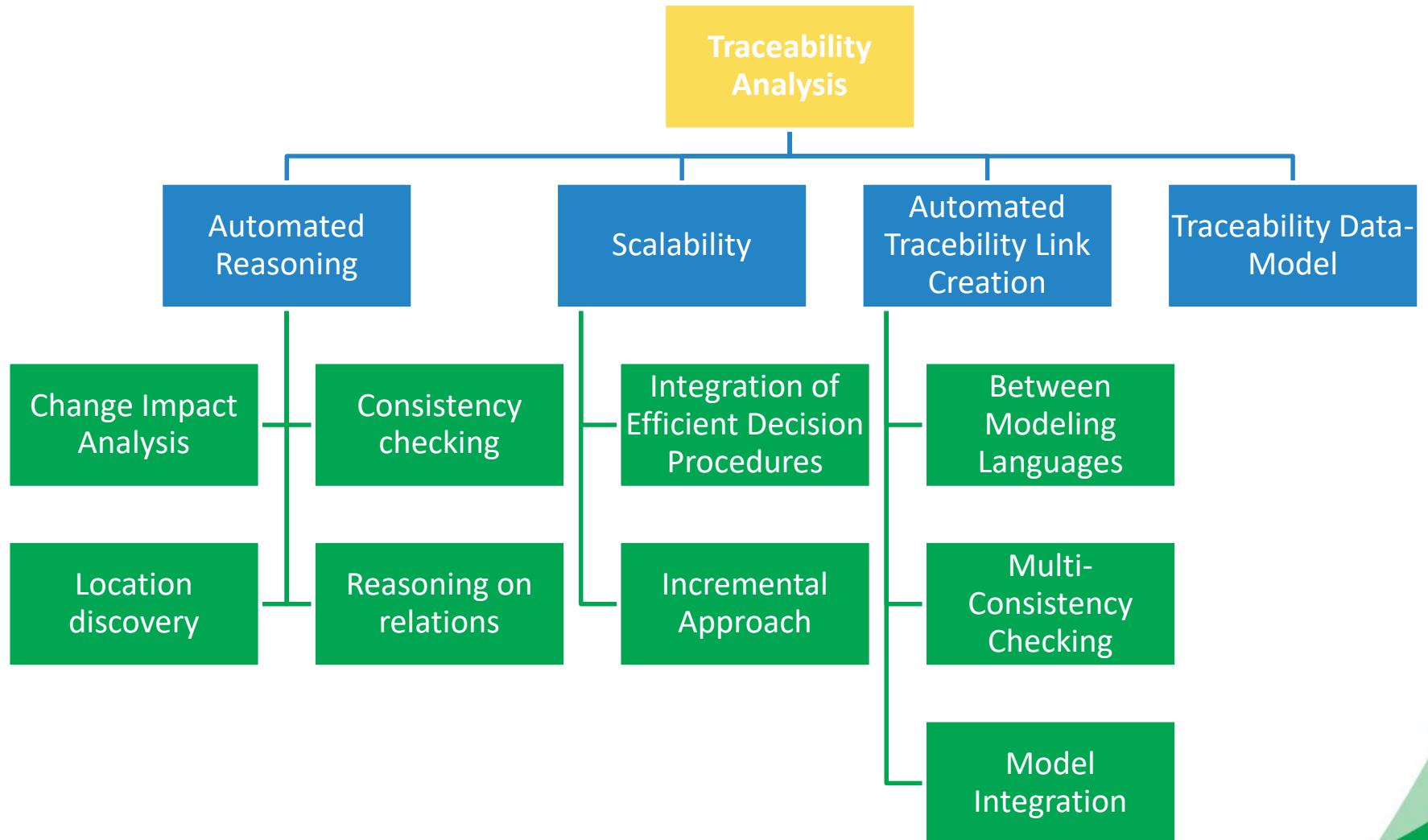
(e) Edges straightened.

Demonstration Visualization in Action

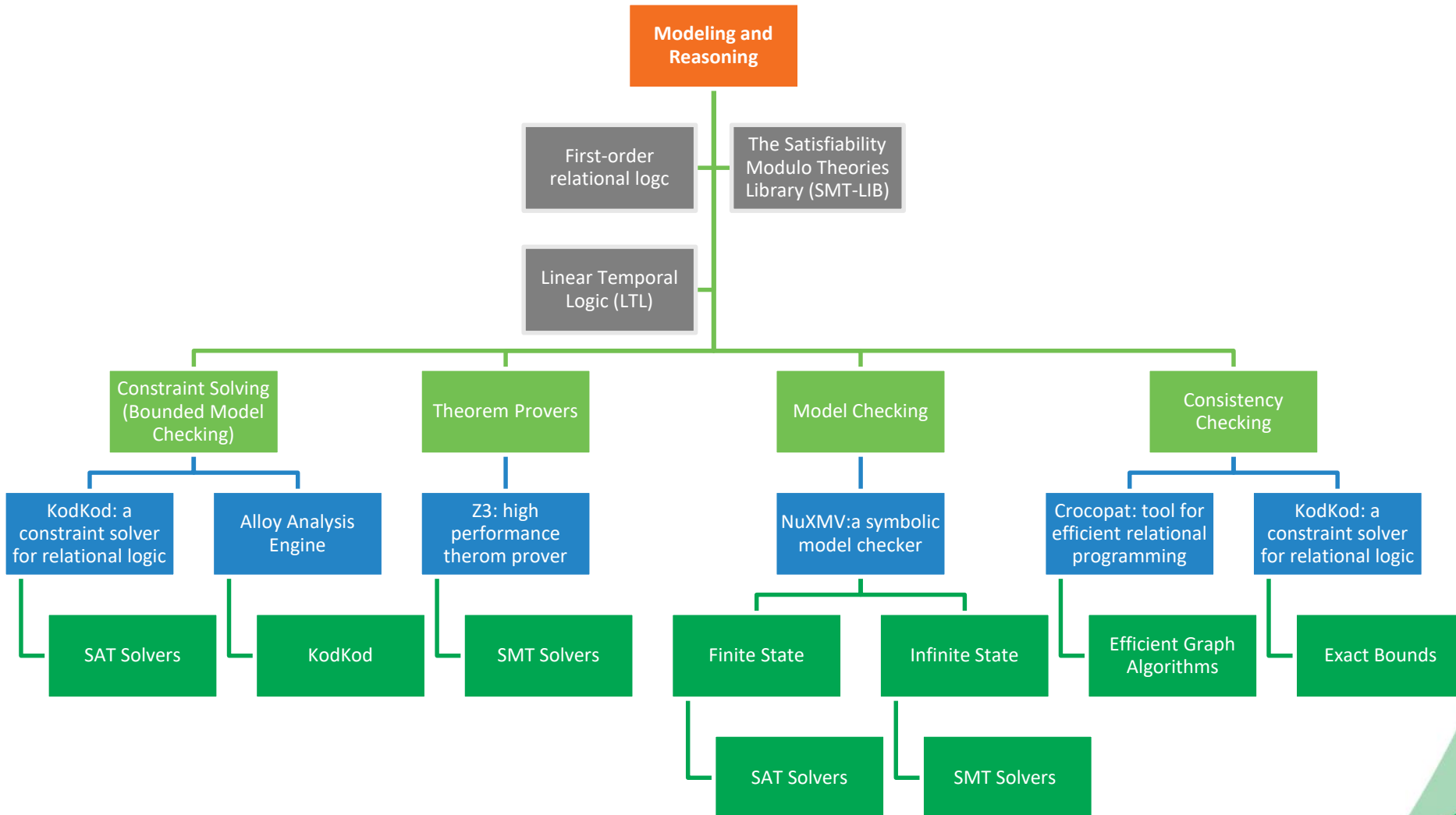


Traceability Analysis (WP4 & WP3)

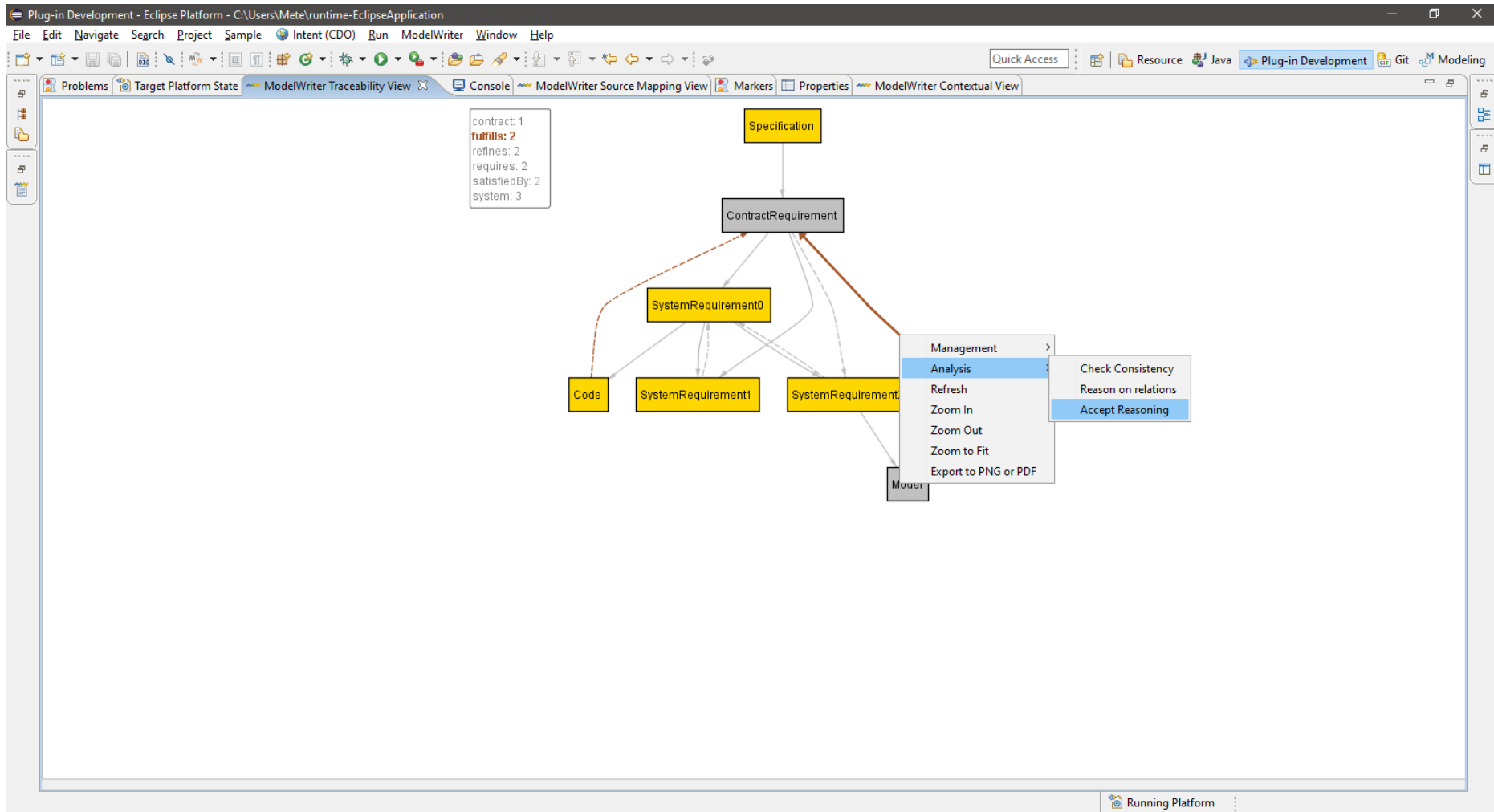
[UNIT]



Modeling and Reasoning Approaches (WP3) [UNIT]

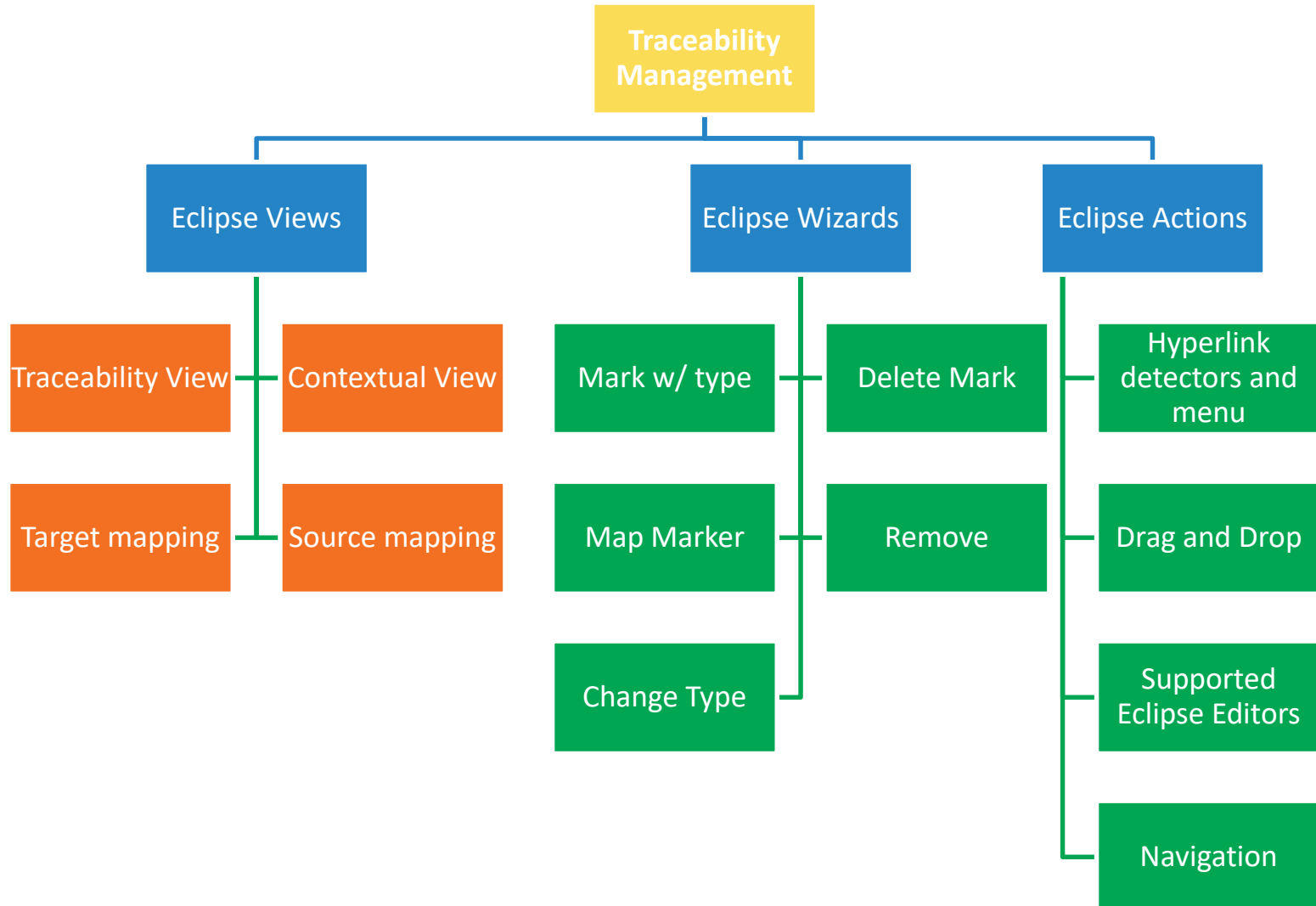


Demonstration Traceability Analysis in Action



Traceability Management (WP6)

[KoçSistem]



Demonstration Traceability Management in Action

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

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

ApplicationLifecycleAnalysis.mw

extends: 11
conflicts: 1
conforms: 1
contains: 1
contract: 1
depends: 1
fulfills: 1
generates: 1
generates: 1
refines: 1
requires: 1
satisfiedBy: 1
system: 1
transforms: 1
verifiedBy: 1

Artefact

Specification

ContractRequirement

SystemRequirement

extends

contract

contains

system

refines

requires

Customer Requirements Specification.md

```
1 # Customer Requirements Specification
2
3 ## UC-1 Create a new SpecObject
4
5 Note that the Specification
6 creating SpecObjects in the
7
8 ### Precondition
9
10 ReqIF model exists and is
11
12 ### Main Success Scenario
13
14 1. We assume that a Spec
15 scenario)
16
17 2. Open a row's context menu
18 3. Select the Child or Sibling
19 4. Select the desired Specification
20 5. This results in a new
21 created SpecObject with the
```

Undo Typing Ctrl+Z
Revert File
Save Ctrl+S
Open With
Show In Alt+Shift+W
Preview at "UC-1 Create a new SpecObject"
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Quick Fix Ctrl+1
Shift Right
Shift Left
Add to Snippets...
Run As
Debug As
Validate
GitHub
ModelWriter
Team
Compare With
Replace With
WikiText
Markup Language
Preferences...
Remove from Context Ctrl+Alt+Shift+Down

Specification

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

Code

SystemRequirement1

SystemRequirement2

fulfills

requires

refines

satisfiedBy

ModelWriter Traceability View

ModelWriter Source Mapping View

Markers

Properties

ModelWriter Contextual

Running Platform Writable Insert 3 : 24

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