

5 Demonstrations

Ferhat Erata (UNIT, WP3 Leader)

Emre Kirmizi (UNIT, Technical Contact)

Dr. Claire Gardent (CNRS/LORIA, WP2 Leader)

Dr. Samuel Cruz Lara (CNRS/LORIA, Technical Contact)

Dr. Bikash Gyawali (CNRS/LORIA, Technical Contact)

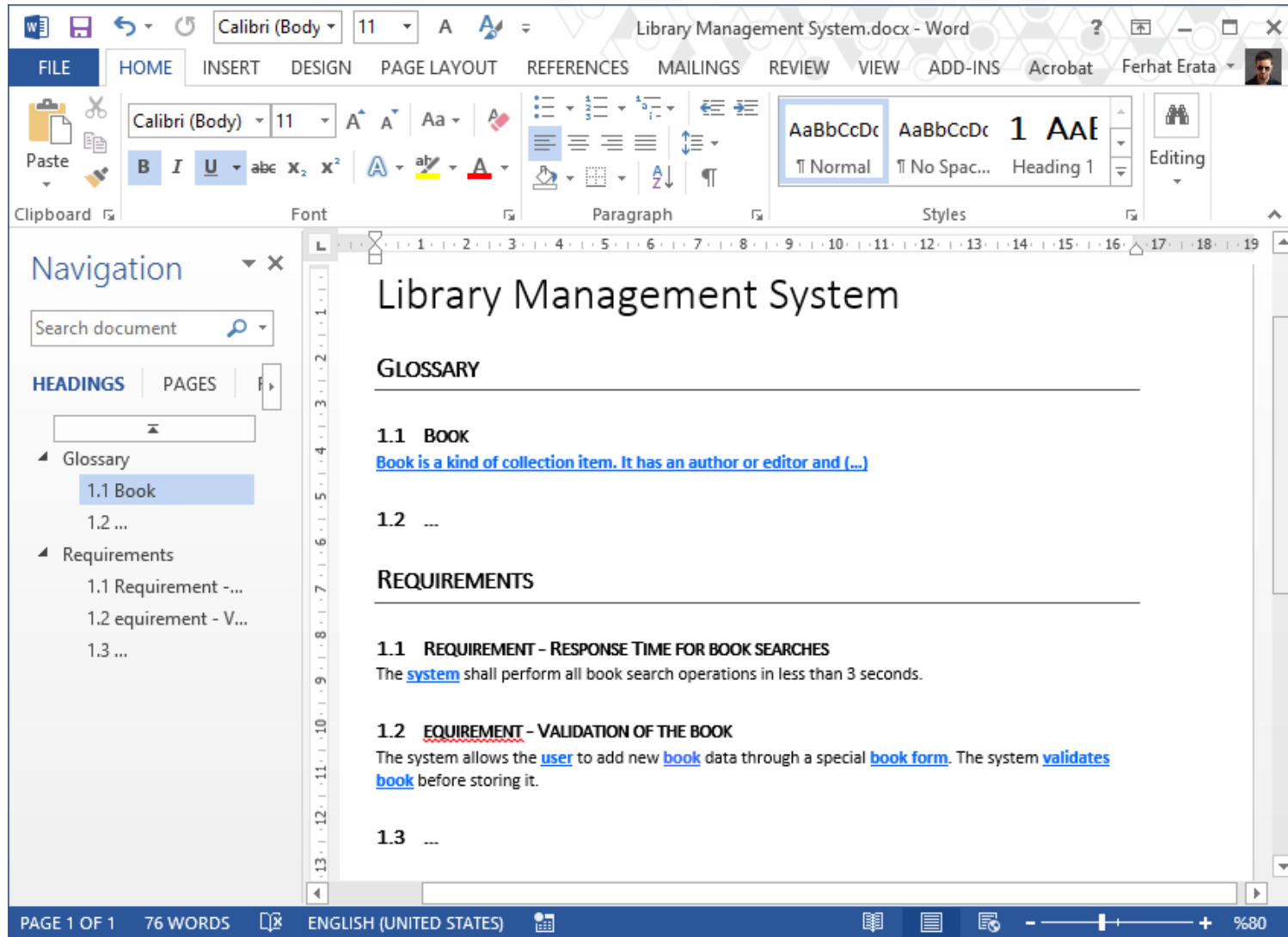
Anastasia Shimorina (CNRS/LORIA, Technical Contact)

Dr. Geylani Kardas (KOCSISTEM, Consultant)

What is a text?

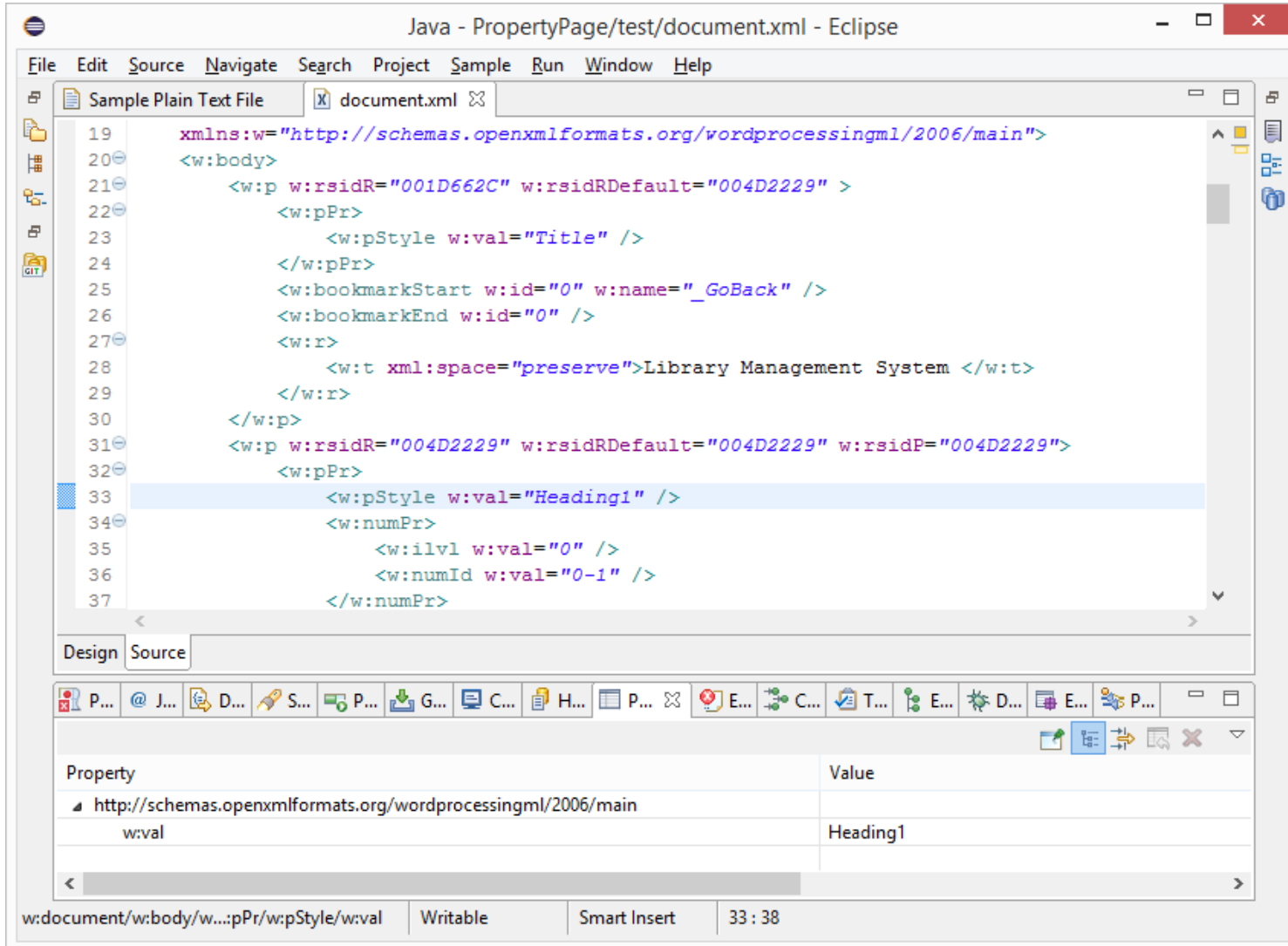
What is a text? (document file formats)

Office Open XML (.docx) (ISO/IEC 29500)



What is a text? (document file formats)

Office Open XML (.docx) (ISO/IEC 29500)



```
19  xmlns:w="http://schemas.openxmlformats.org/wordprocessingml/2006/main">
20  <w:body>
21    <w:p w:rsidR="001D662C" w:rsidRDefault="004D2229" >
22      <w:pPr>
23        <w:pStyle w:val="Title" />
24      </w:pPr>
25      <w:bookmarkStart w:id="0" w:name="_GoBack" />
26      <w:bookmarkEnd w:id="0" />
27      <w:r>
28        <w:t xml:space="preserve">Library Management System </w:t>
29      </w:r>
30    </w:p>
31    <w:p w:rsidR="004D2229" w:rsidRDefault="004D2229" w:rsidP="004D2229">
32      <w:pPr>
33        <w:pStyle w:val="Heading1" />
34        <w:numPr>
35          <w:ilvl w:val="0" />
36          <w:numId w:val="0-1" />
37        </w:numPr>

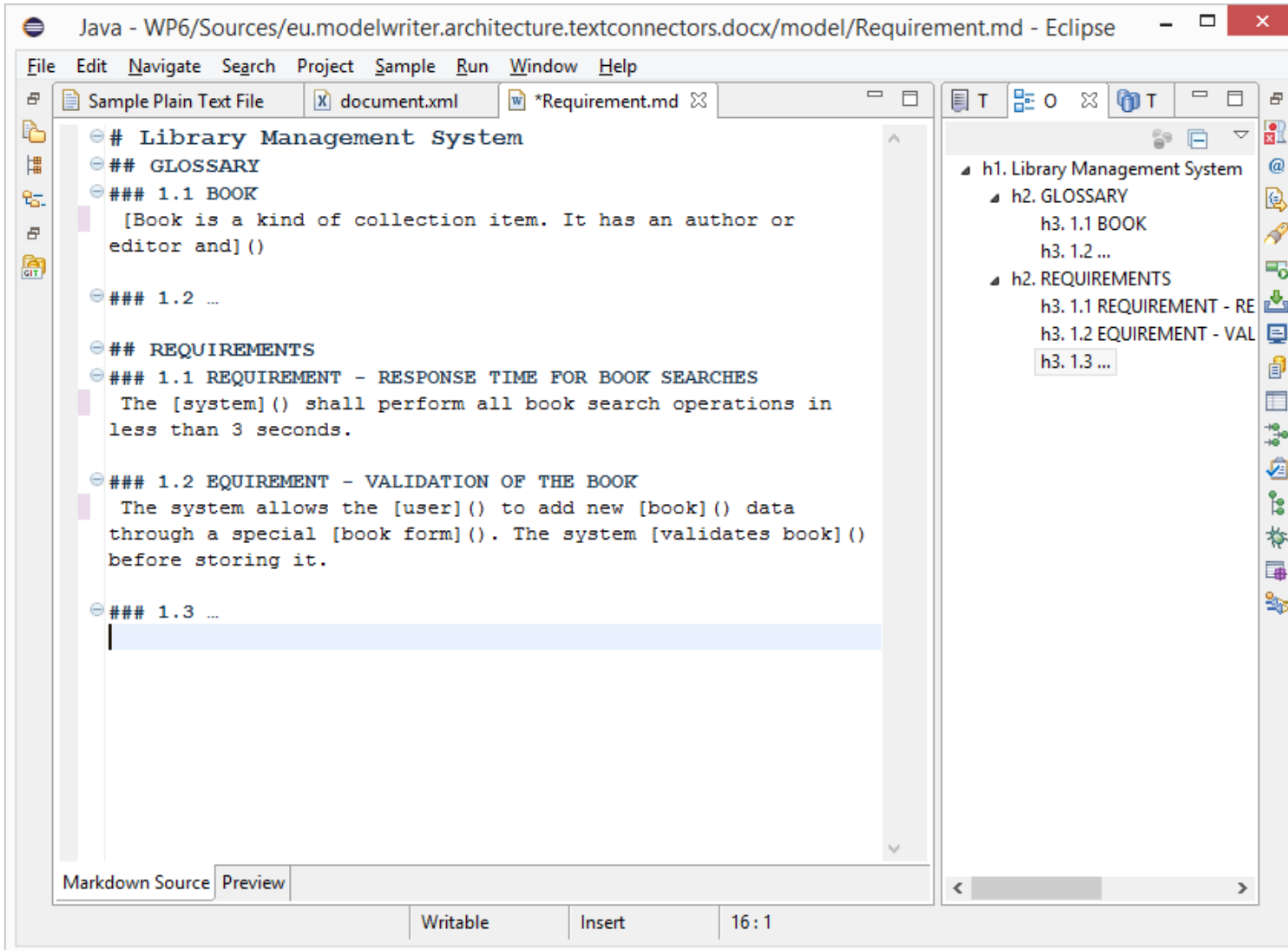
```

Property	Value
http://schemas.openxmlformats.org/wordprocessingml/2006/main	
w:val	Heading1

w:document/w:body/w:wp/w:pPr/w:pStyle/w:val Writable Smart Insert 33 : 38

What is a text? (.md source file)

text/markdown (ICANN Standard)



The screenshot shows the Eclipse IDE interface. The main editor window displays a markdown file named `*Requirement.md`. The content of the file is as follows:

```
# Library Management System
## GLOSSARY
### 1.1 BOOK
[Book is a kind of collection item. It has an author or
editor and]()

### 1.2 ...

## REQUIREMENTS
### 1.1 REQUIREMENT - RESPONSE TIME FOR BOOK SEARCHES
The [system]() shall perform all book search operations in
less than 3 seconds.

### 1.2 EQUIREMENT - VALIDATION OF THE BOOK
The system allows the [user]() to add new [book]() data
through a special [book form]() . The system [validates book]()
before storing it.

### 1.3 ...
```

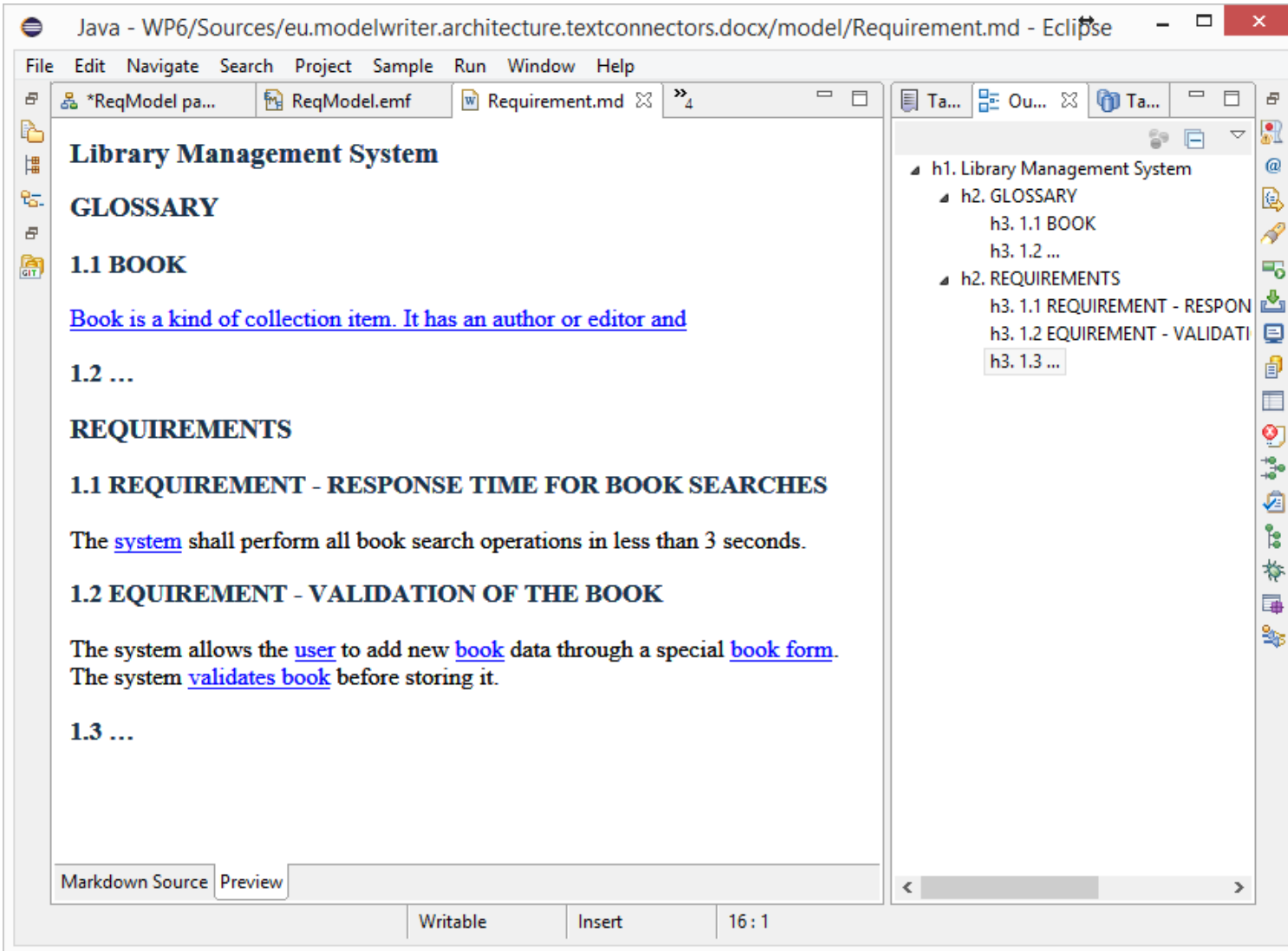
On the right side, there is a preview view showing the rendered markdown structure:

- h1. Library Management System
 - h2. GLOSSARY
 - h3. 1.1 BOOK
 - h3. 1.2 ...
 - h2. REQUIREMENTS
 - h3. 1.1 REQUIREMENT - RE
 - h3. 1.2 EQUIREMENT - VAL
 - h3. 1.3 ...

At the bottom of the editor, there are tabs for 'Markdown Source' and 'Preview'. The status bar at the very bottom indicates 'Writable', 'Insert' mode, and line '16:1'.

What is a text? (HTML Preview)

text/markdown (ICANN Standard)



The screenshot shows the Eclipse IDE interface. The main editor window displays the HTML preview of a Markdown document titled "Requirement.md". The document content is as follows:

Library Management System

GLOSSARY

1.1 BOOK

Book is a kind of collection item. It has an author or editor and

1.2 ...

REQUIREMENTS

1.1 REQUIREMENT - RESPONSE TIME FOR BOOK SEARCHES

The system shall perform all book search operations in less than 3 seconds.

1.2 EQUIREMENT - VALIDATION OF THE BOOK

The system allows the user to add new book data through a special book form.
The system validates book before storing it.

1.3 ...

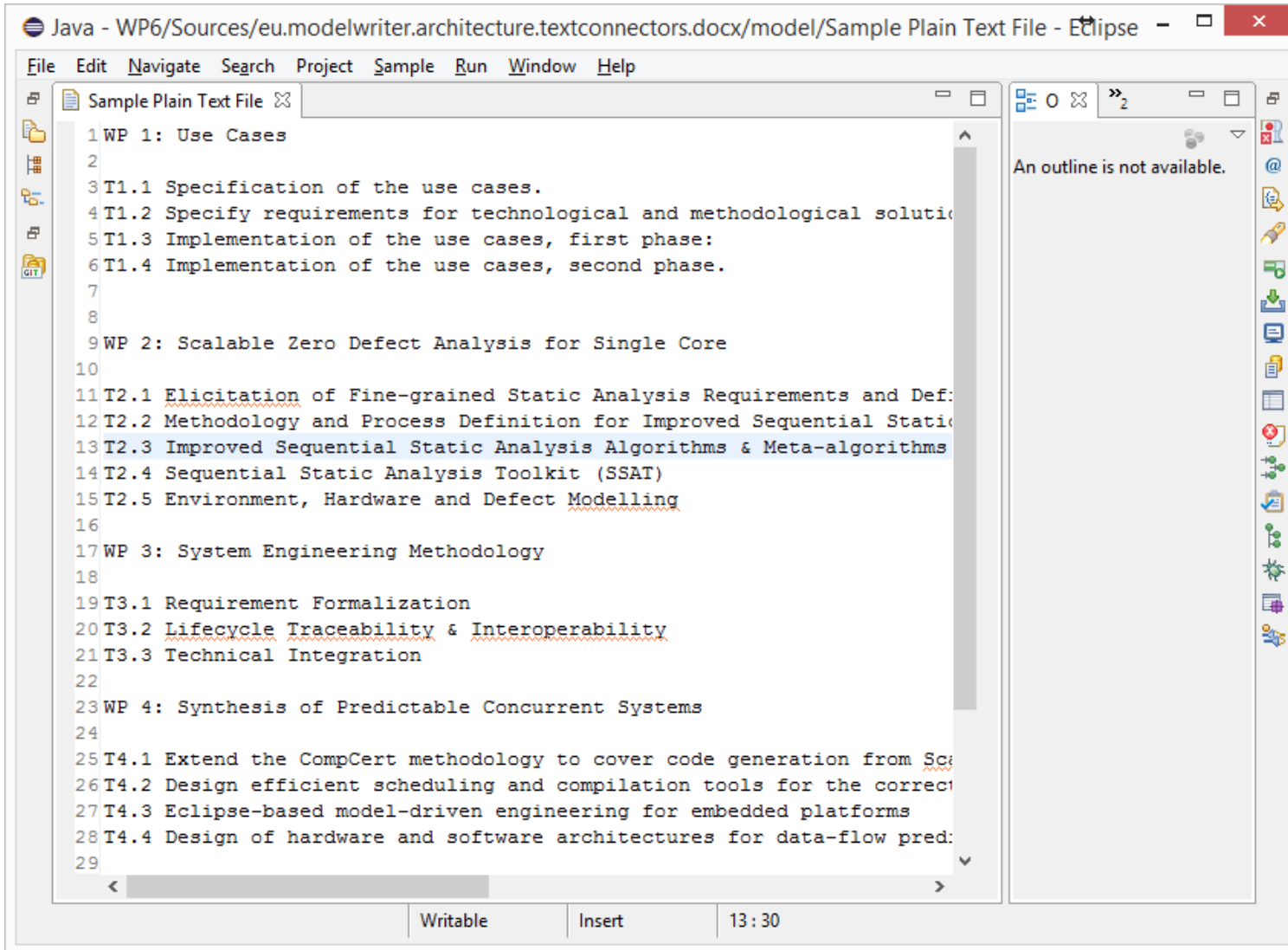
The right-hand side of the IDE shows the Outline view, which lists the document's structure:

- h1. Library Management System
 - h2. GLOSSARY
 - h3. 1.1 BOOK
 - h3. 1.2 ...
 - h2. REQUIREMENTS
 - h3. 1.1 REQUIREMENT - RESPON
 - h3. 1.2 EQUIREMENT - VALIDATI
 - h3. 1.3 ...

At the bottom of the editor, there are tabs for "Markdown Source" and "Preview", with "Preview" currently selected. The status bar at the very bottom indicates "Writable", "Insert" mode, and line "16 : 1".

What is a text? (unformatted text)

text/plain (ICANN Standard)



Java - WP6/Sources/eu.modelwriter.architecture.textconnectors.docx/model/Sample Plain Text File - Eclipse

File Edit Navigate Search Project Sample Run Window Help

Sample Plain Text File

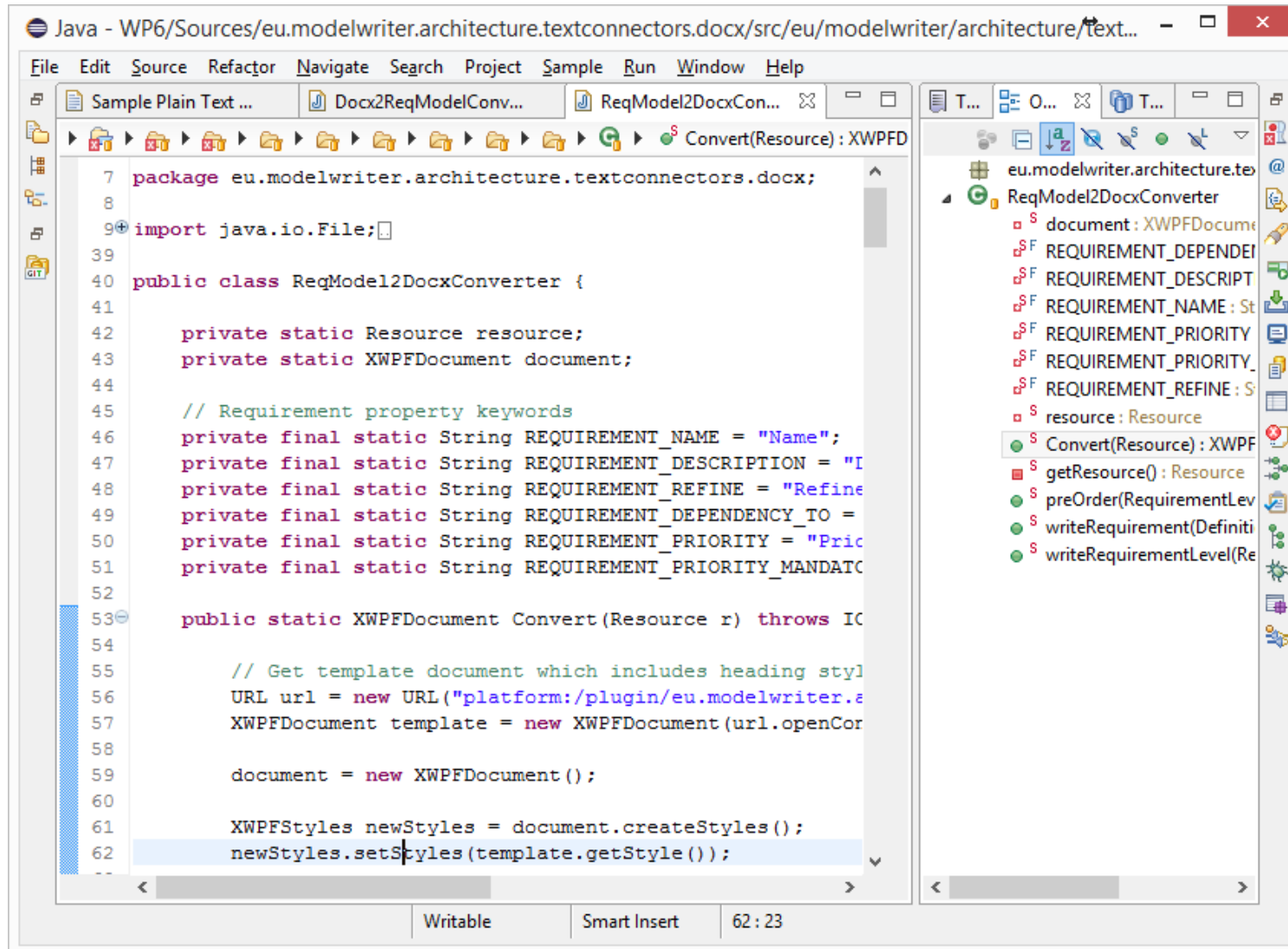
```
1 WP 1: Use Cases
2
3 T1.1 Specification of the use cases.
4 T1.2 Specify requirements for technological and methodological solutions
5 T1.3 Implementation of the use cases, first phase:
6 T1.4 Implementation of the use cases, second phase.
7
8
9 WP 2: Scalable Zero Defect Analysis for Single Core
10
11 T2.1 Elicitation of Fine-grained Static Analysis Requirements and Definitions
12 T2.2 Methodology and Process Definition for Improved Sequential Static Analysis
13 T2.3 Improved Sequential Static Analysis Algorithms & Meta-algorithms
14 T2.4 Sequential Static Analysis Toolkit (SSAT)
15 T2.5 Environment, Hardware and Defect Modelling
16
17 WP 3: System Engineering Methodology
18
19 T3.1 Requirement Formalization
20 T3.2 Lifecycle Traceability & Interoperability
21 T3.3 Technical Integration
22
23 WP 4: Synthesis of Predictable Concurrent Systems
24
25 T4.1 Extend the CompCert methodology to cover code generation from Specifications
26 T4.2 Design efficient scheduling and compilation tools for the correct compilation
27 T4.3 Eclipse-based model-driven engineering for embedded platforms
28 T4.4 Design of hardware and software architectures for data-flow prediction
29
```

An outline is not available.

Writable Insert 13:30

What is a text? (code files)

Java, C++ ... Programing Languages



The screenshot shows an IDE window titled "Java - WP6/Sources/eu.modelwriter.architecture.textconnectors.docx/src/eu/modelwriter/architecture/text...". The main editor displays the following Java code:

```
7 package eu.modelwriter.architecture.textconnectors.docx;
8
9 import java.io.File;
10
39
40 public class ReqModel2DocxConverter {
41
42     private static Resource resource;
43     private static XWPFDocument document;
44
45     // Requirement property keywords
46     private final static String REQUIREMENT_NAME = "Name";
47     private final static String REQUIREMENT_DESCRIPTION = "Description";
48     private final static String REQUIREMENT_REFINE = "Refine";
49     private final static String REQUIREMENT_DEPENDENCY_TO = "Dependency To";
50     private final static String REQUIREMENT_PRIORITY = "Priority";
51     private final static String REQUIREMENT_PRIORITY_MANDATORY = "Mandatory";
52
53     public static XWPFDocument Convert(Resource r) throws IOException {
54
55         // Get template document which includes heading styles
56         URL url = new URL("platform:/plugin/eu.modelwriter.architecture.textconnectors.docx/ReqModel2DocxConverterTemplate.docx");
57         XWPFDocument template = new XWPFDocument(url.openStream());
58
59         document = new XWPFDocument();
60
61         XWPFStyles newStyles = document.createStyles();
62         newStyles.setStyles(template.getStyles());
63     }
64 }
```

The right sidebar shows a project explorer with the following structure:

- eu.modelwriter.architecture.textconnectors.docx
 - ReqModel2DocxConverter
 - document: XWPFDocument
 - REQUIREMENT_DEPENDENCY_TO
 - REQUIREMENT_DESCRIPTION
 - REQUIREMENT_NAME: String
 - REQUIREMENT_PRIORITY
 - REQUIREMENT_PRIORITY_MANDATORY
 - REQUIREMENT_REFINE: String
 - resource: Resource
 - Convert(Resource): XWPFDocument
 - getResource(): Resource
 - preOrder(RequirementLevel)
 - writeRequirement(Definition)
 - writeRequirementLevel(RequirementLevel)

What is a model?

Everything is a model! (ReqIF Standard)

Requirements Interchange Format

ProR - platform:/resource/LibraryManagementSystem/My.reqif - formalmind Studio

File Edit Search Requirements fmStudio Window Help

Quick Access ProR

My.reqif Requirements Document

ID	Name	Description
1		
1.1	Librarian	Librarian
1.2	Response Time for Book Searches	The system shall perform all book search operations in less than 3 second
1.3	Add new Book	A user form to add new book
1.4	Validation of the Book	Validation of the Book

Outline

- Spec Hierarchy
 - Librarian
 - The system shall
 - A user form to
 - Validation of th

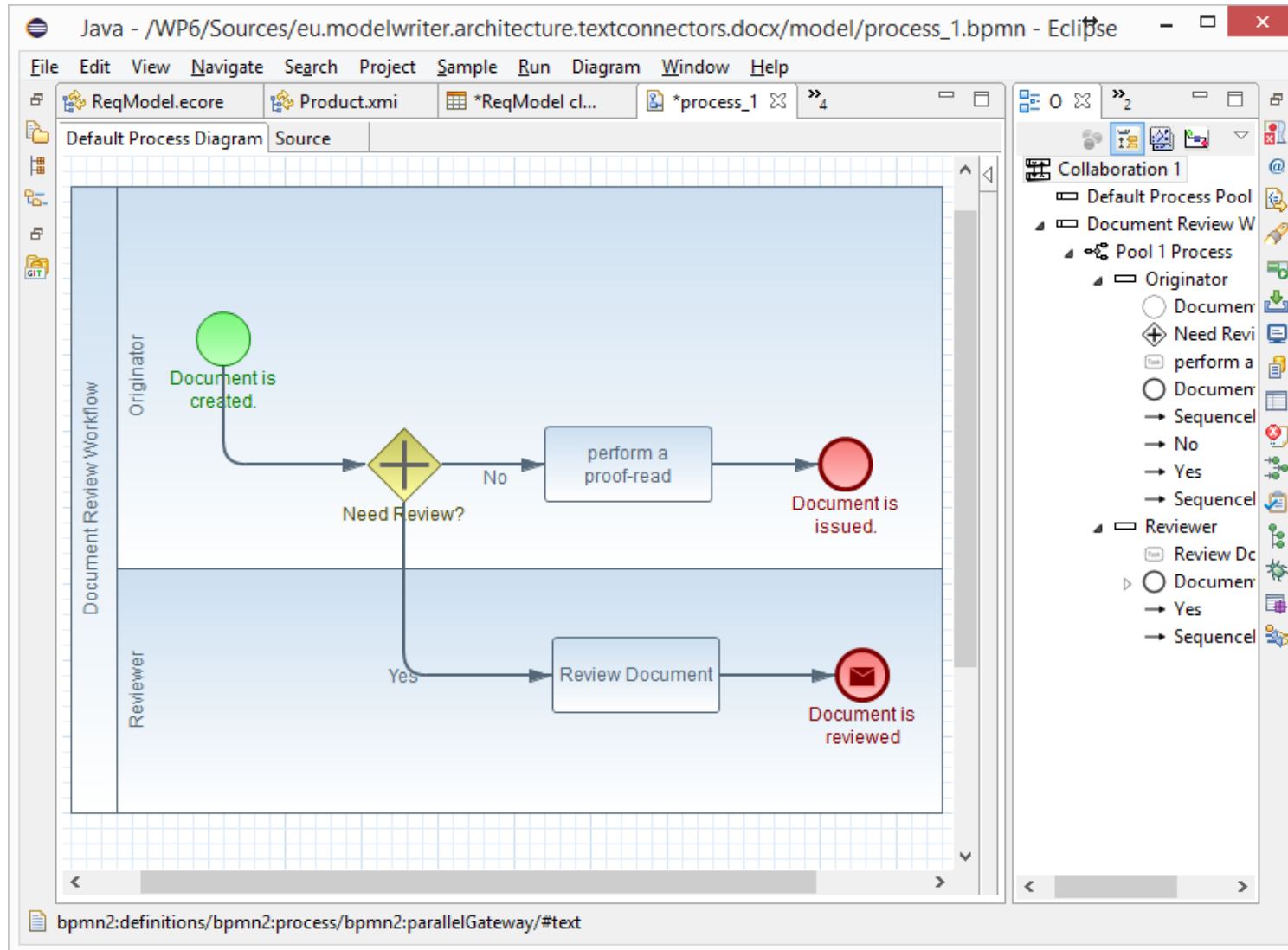
Properties

Property	Value
Requirement Type	
Description	The system shall perform all book search operations in less than 3 second
ID	R123
Name	Response Time for Book Searches
Responsible	Ferhat
Version	1
Spec Object	
Type	Requirement Type (Spec Object)

Standard Attributes All Attributes

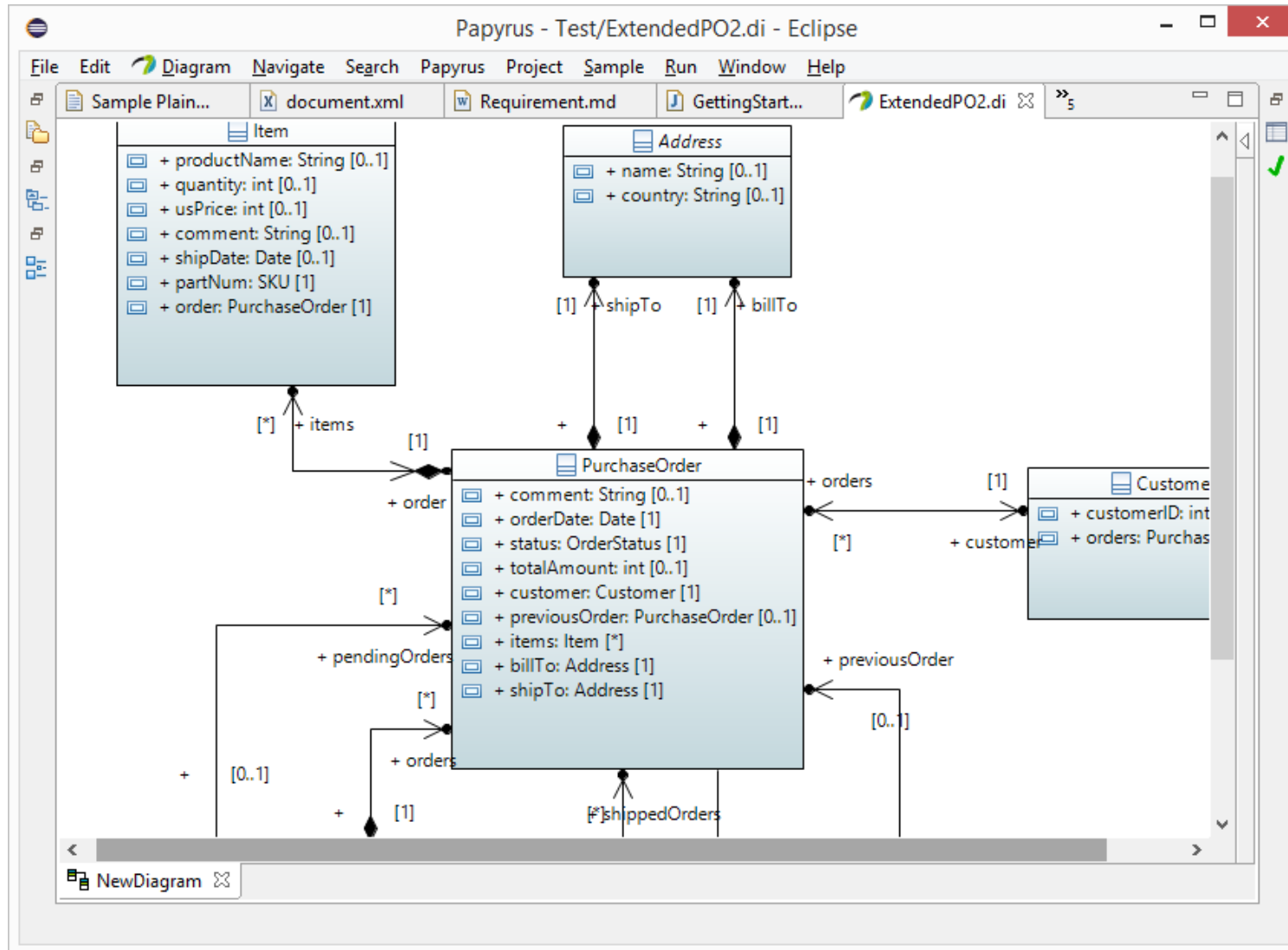
Everything is a model! (BPMN Standard)

Business Process Model & Notation



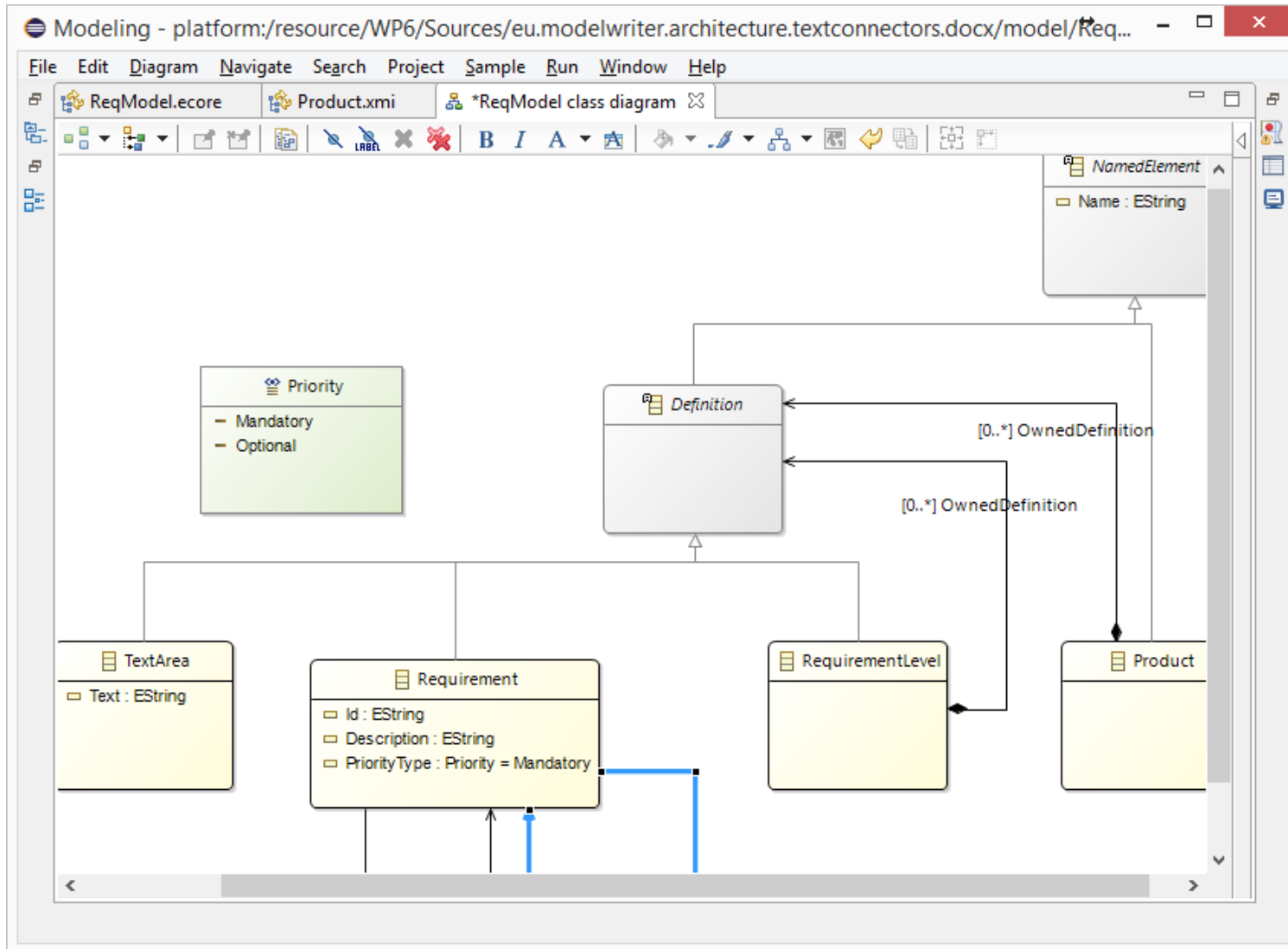
Everything is a model! (UML Standard)

UML Modeling Languages



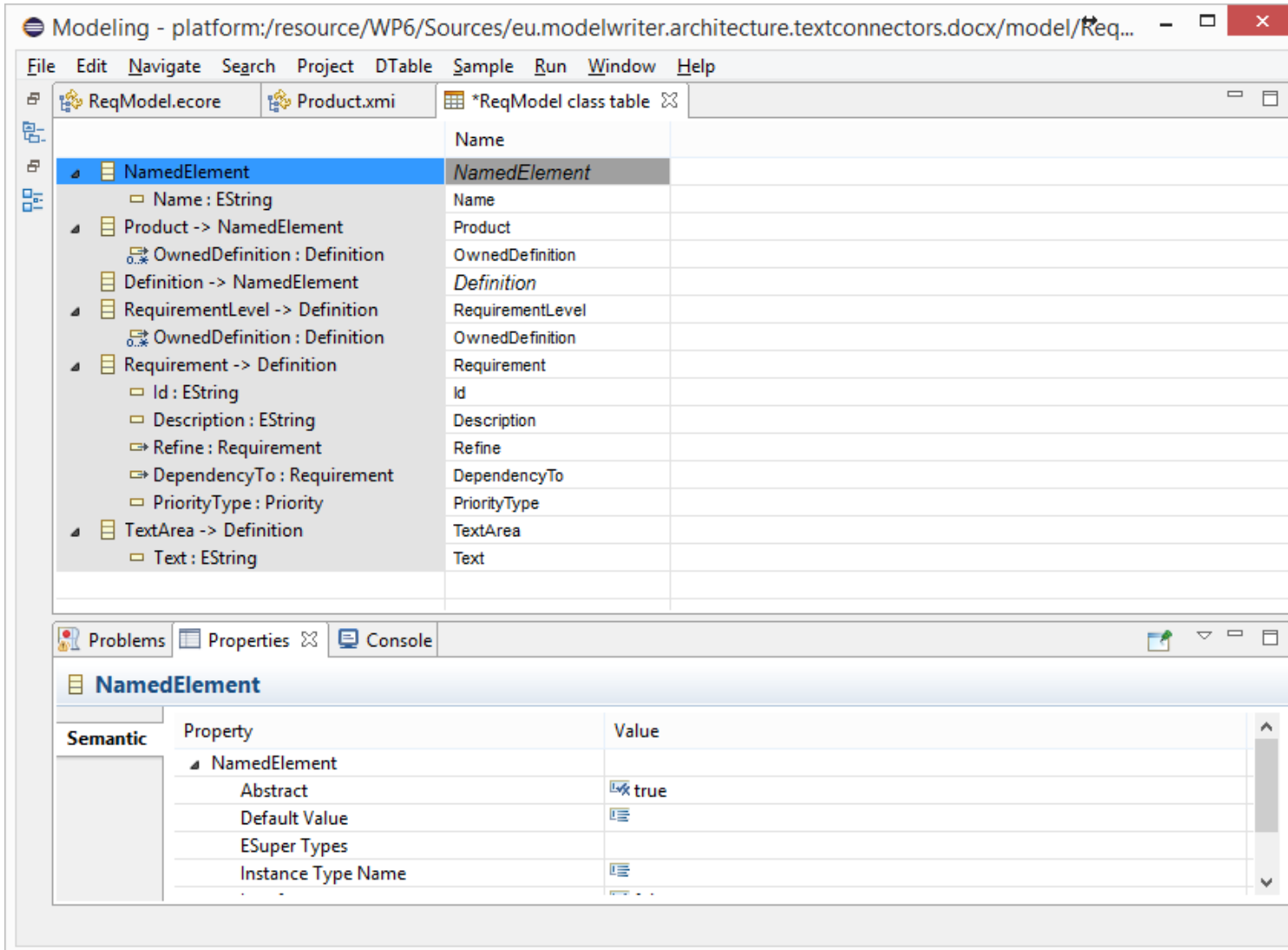
Everything is a model!

Eclipse Modeling Framework (EMF)



Everything is a model!

Tree-based or Tabular Representations



The screenshot shows the Modeling IDE interface. The top menu bar includes File, Edit, Navigate, Search, Project, DTable, Sample, Run, Window, and Help. The main workspace is divided into two panes. The left pane shows a tree-based representation of the model, with the following structure:

- NamedElement
 - Name : EString
- Product -> NamedElement
 - OwnedDefinition : Definition
- Definition -> NamedElement
 - RequirementLevel -> Definition
 - OwnedDefinition : Definition
- Requirement -> Definition
 - Id : EString
 - Description : EString
 - Refine : Requirement
 - DependencyTo : Requirement
 - PriorityType : Priority
- TextArea -> Definition
 - Text : EString

The right pane shows a tabular representation of the model, with the following columns: Name, Value, and Description. The table contains the following rows:

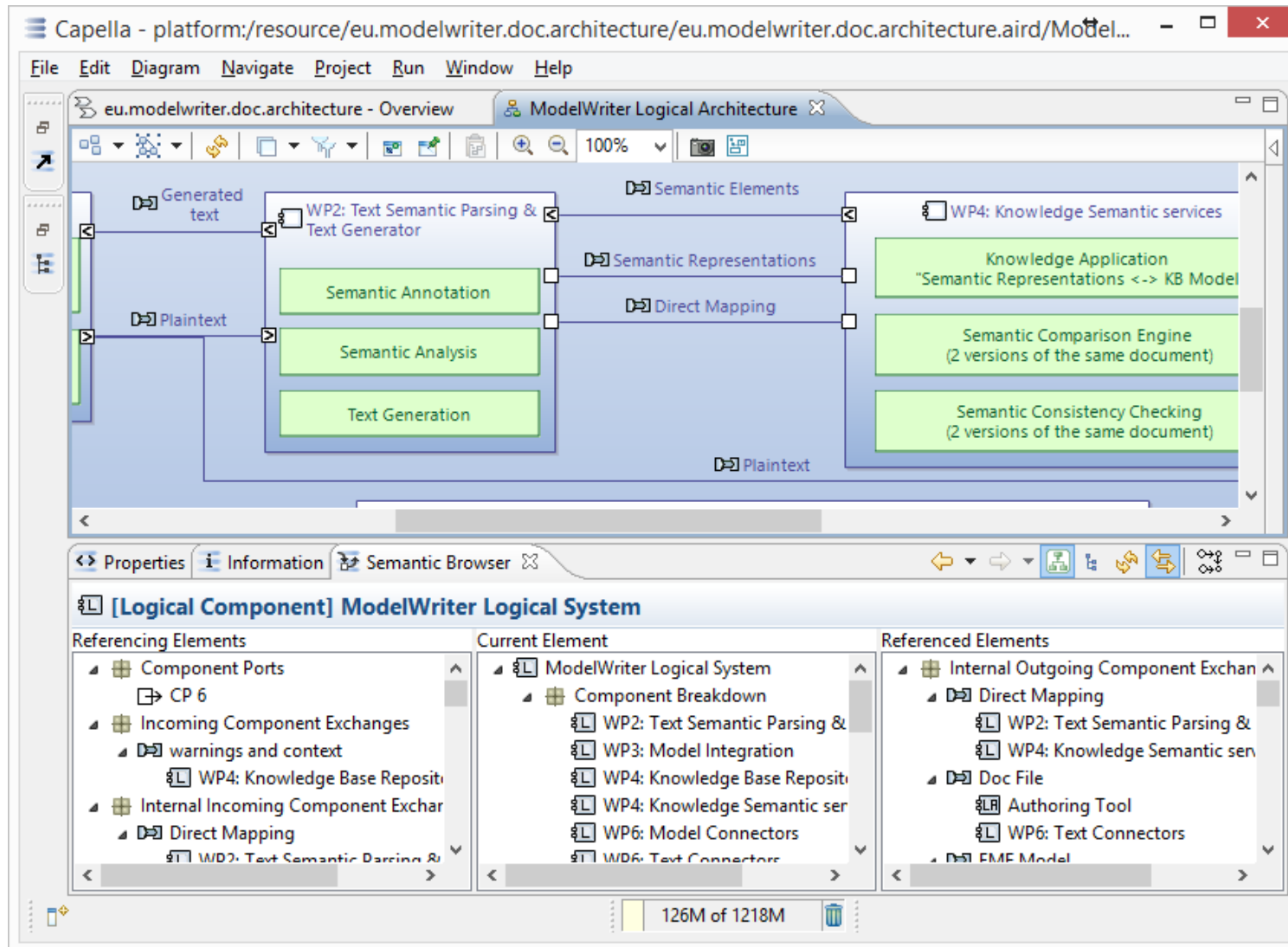
Name	Value	Description
NamedElement		
Name		
Product		
OwnedDefinition		
Definition		
RequirementLevel		
OwnedDefinition		
Requirement		
Id		
Description		
Refine		
DependencyTo		
PriorityType		
TextArea		
Text		

The bottom pane shows the Properties view for the selected NamedElement. It has a tab labeled "Semantic" and a table with the following columns: Property, Value, and Description. The table contains the following rows:

Property	Value	Description
NamedElement		
Abstract	true	
Default Value		
ESuper Types		
Instance Type Name		

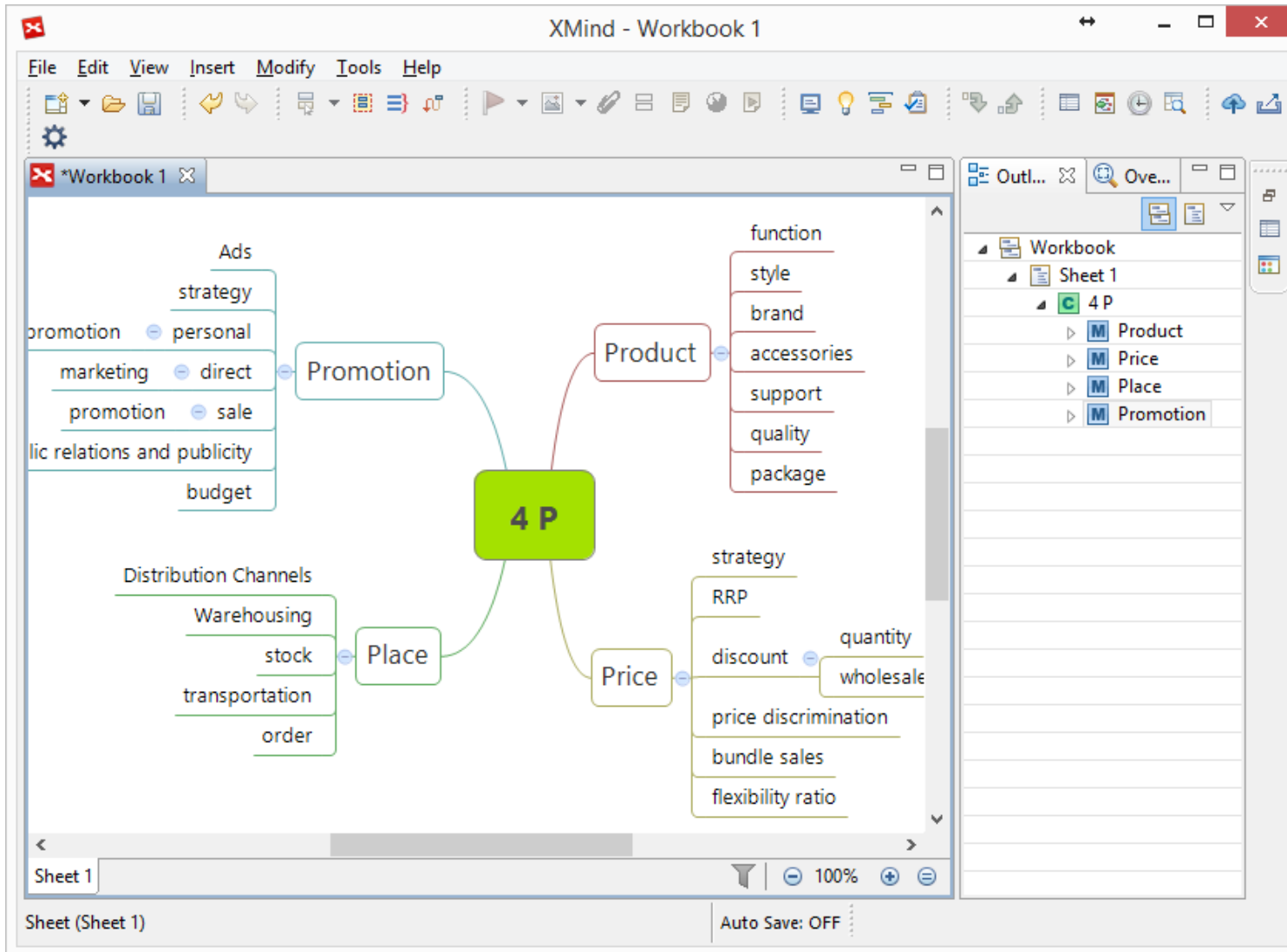
Everything is a model!

Software/System Architecture Design



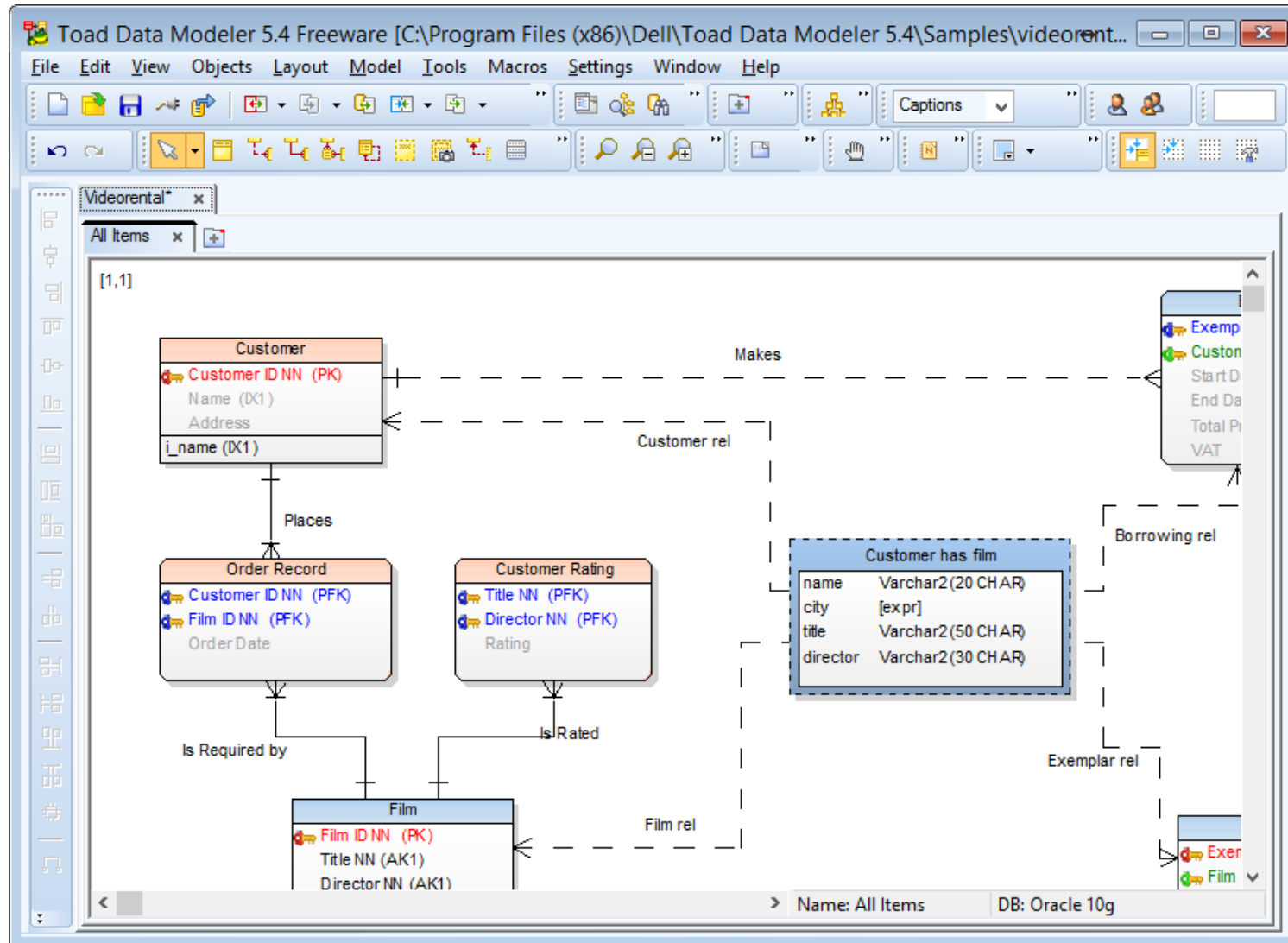
Everything is a model!

Topic Maps, Mind Maps, Vocabularies ...



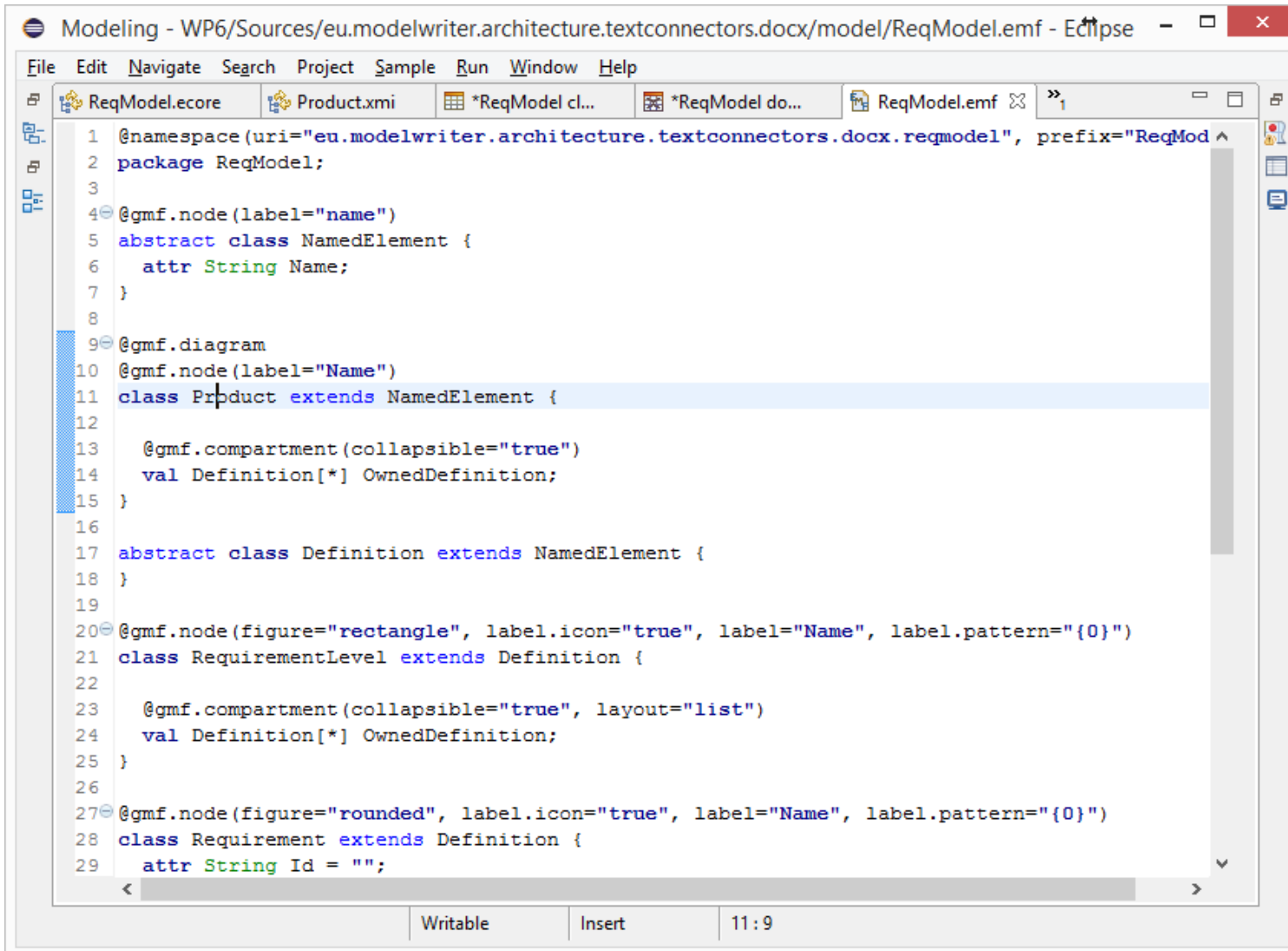
Everything is a model!

Databases (ER, IDEF1.x)



Everything is a model! (Textual Lang.)

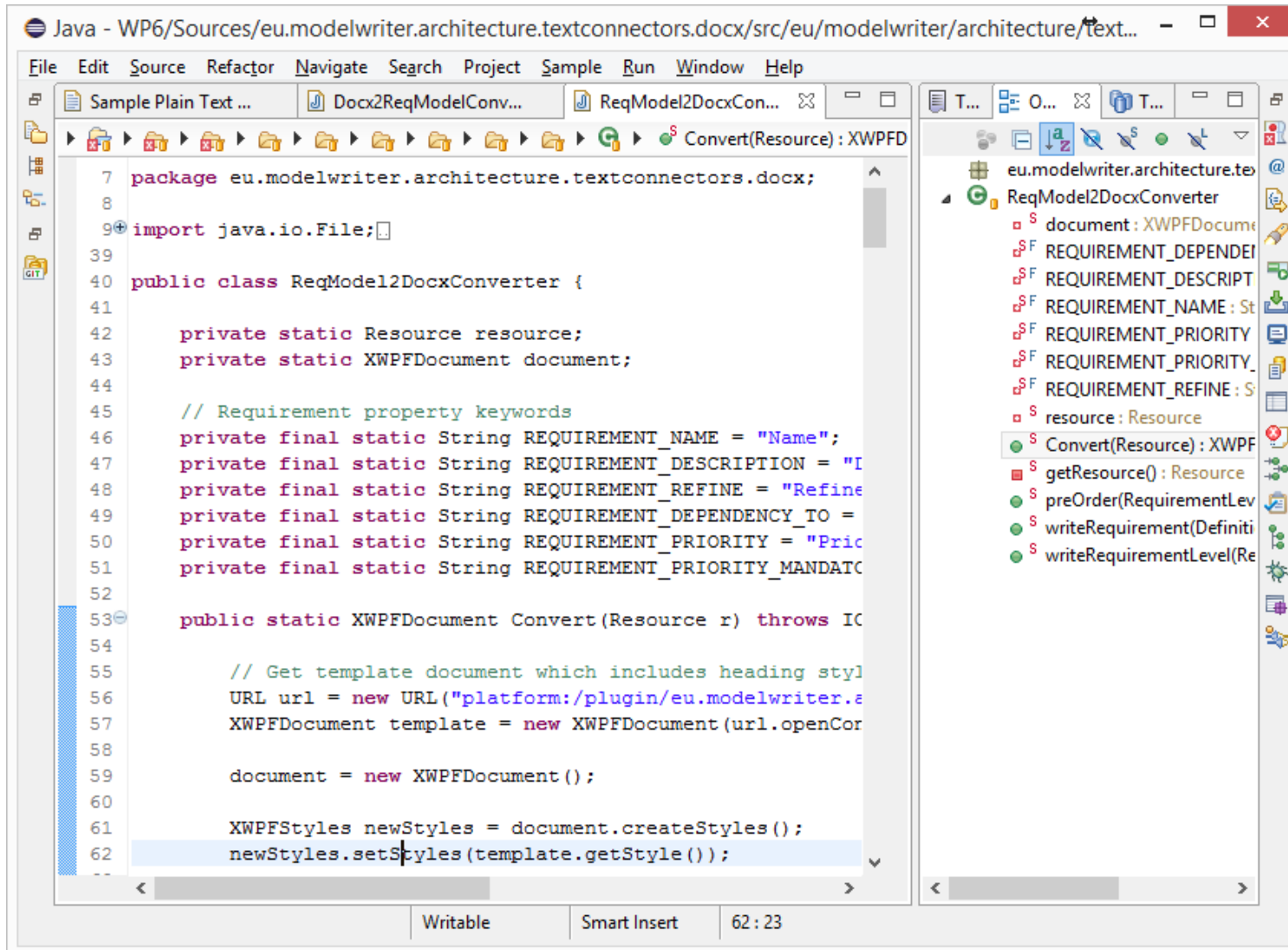
Domain Specific Languages



```
1 @namespace(uri="eu.modelwriter.architecture.textconnectors.docx.reqmodel", prefix="ReqMod
2 package ReqModel;
3
4 @gmf.node(label="name")
5 abstract class NamedElement {
6     attr String Name;
7 }
8
9 @gmf.diagram
10 @gmf.node(label="Name")
11 class Product extends NamedElement {
12
13     @gmf.compartment(collapsible="true")
14     val Definition[*] OwnedDefinition;
15 }
16
17 abstract class Definition extends NamedElement {
18 }
19
20 @gmf.node(figure="rectangle", label.icon="true", label="Name", label.pattern="{0}")
21 class RequirementLevel extends Definition {
22
23     @gmf.compartment(collapsible="true", layout="list")
24     val Definition[*] OwnedDefinition;
25 }
26
27 @gmf.node(figure="rounded", label.icon="true", label="Name", label.pattern="{0}")
28 class Requirement extends Definition {
29     attr String Id = "";
```

Everything is a model! (Java, C++, etc.)

Even Programming Languages (ASTs)



The screenshot shows an IDE window titled "Java - WP6/Sources/eu.modelwriter.architecture.textconnectors.docx/src/eu/modelwriter/architecture/text...". The main editor displays the source code for the `ReqModel2DocxConverter` class. The code includes package declarations, imports, and a `Convert` method that processes a resource into an XWPF document. The project structure on the right shows the package hierarchy and the class file.

```
package eu.modelwriter.architecture.textconnectors.docx;

import java.io.File;

public class ReqModel2DocxConverter {

    private static Resource resource;
    private static XWPFDocument document;

    // Requirement property keywords
    private final static String REQUIREMENT_NAME = "Name";
    private final static String REQUIREMENT_DESCRIPTION = "Description";
    private final static String REQUIREMENT_REFINE = "Refine";
    private final static String REQUIREMENT_DEPENDENCY_TO = "Dependency To";
    private final static String REQUIREMENT_PRIORITY = "Priority";
    private final static String REQUIREMENT_PRIORITY_MANDATORY = "Mandatory";

    public static XWPFDocument Convert(Resource r) throws IOException {

        // Get template document which includes heading styles
        URL url = new URL("platform:/plugin/eu.modelwriter.architecture.textconnectors.docx/ReqModel2DocxConverter.template.docx");
        XWPFDocument template = new XWPFDocument(url.openStream());

        document = new XWPFDocument();

        XWPFStyles newStyles = document.createStyles();
        newStyles.setStyles(template.getStyle());
    }
}
```

Project Structure:

- eu.modelwriter.architecture.textconnectors.docx
 - ReqModel2DocxConverter
 - document: XWPFDocument
 - REQUIREMENT_DEPENDENCY_TO
 - REQUIREMENT_DESCRIPTION
 - REQUIREMENT_NAME: String
 - REQUIREMENT_PRIORITY
 - REQUIREMENT_PRIORITY_MANDATORY
 - REQUIREMENT_REFINE: String
 - resource: Resource
 - Convert(Resource): XWPFDocument
 - getResource(): Resource
 - preOrder(RequirementLevel)
 - writeRequirement(Definition)
 - writeRequirementLevel(RequirementLevel)

**Is it possible to connect and
keep arbitrary software/system
engineering artifacts
synchronized ?**

Solution – Knowledge Capture

The screenshot displays the Eclipse IDE interface for a project named "Plug-in Development - eu.modelwriter.demonstration.requirements/". The main editor shows a file named "Customer Requirements Specification.md" with the following content:

```
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, creating
6 SpecObjects in this editor is the main success scenario.
7
8 ### Precondition
```

Below the editor, a diagram view shows a hierarchical structure of elements: "Specification" (yellow), "ContractRequirement1" (yellow), "ContractRequirement0" (grey), "SystemRequirement2" (yellow), "Code" (yellow), "SystemRequirement0" (yellow), "Model" (yellow), and "SystemRequirement1" (yellow). Relationships are indicated by colored arrows: "contract" (red), "system" (purple), "fulfills" (blue), "satisfies" (green), "requires" (orange), and "refines" (pink).

Three red circles highlight specific features:

- 1**: A toolbar icon for "Mapping Action".
- 2**: A "Mapping Action" dialog box titled "Relations" showing suitable relations for the selected marker. The relations listed are: "depends: Artifact -> set of Artifact", "conflicts: Artifact -> set of Artifact", "satisfies: SystemRequirement -> set of Implementation", "requires: SystemRequirement -> set of SystemRequirement", and "refines: SystemRequirement -> set of SystemRequirement".
- 3**: A context menu for the "Management" tab, showing options like "Check Consistency", "Reason on relations", "Zoom In", "Zoom Out", "Zoom to Fit", and "Export to PNG or PDF".

The bottom status bar indicates the "Running Platform" is active.

Text & Model-Synchronized Document Engineering Platform

**Is it possible to extract
knowledge from texts
fragments (based on a given
ontology / model) ?**

Solution – Knowledge Extraction

The screenshot displays the ModelWriter Project application window, which is designed for knowledge extraction and model synchronization. The interface is divided into several key sections:

- File Menu:** Located at the top left, it includes options for **File**, **Link**, **Change**, and **Statistic**. The **Link** menu is currently open, showing sub-options: **Generate Links**, **Search Link**, **Add Link**, and **Remove Link**.
- The Model:** This section on the right displays the RDF model in a **Plain** view. It shows a series of namespace declarations for various standards, including `xmlns:acs="http://airbus-group/aircraft-system#"`, `xmlns:evt="http://airbus-group.installsys/event#"`, `xmlns:rdp="http://www.w3.org/1999/02/22-rdf-syntax-ns#"`, `xmlns:spin="http://spinrdf.org/spin#"`, `xmlns:qudt="http://qudt.org/schema/qudt#"`, `xmlns:dct="http://purl.org/dc/terms/"`, `xmlns:arg="http://spinrdf.org/arg#"`, `xmlns:xsd="http://www.w3.org/2001/XMLSchema#"`, `xmlns:vaem="http://www.linkedmodel.org/schema/vaem#"`, `xmlns:skos="http://www.w3.org/2004/02/skos/core#"`, `xmlns:voag="http://voag.linkedmodel.org/voag/"`, `xmlns:comp="http://airbus-group.installsys/component#"`, `xmlns:qudt-dimension="http://qudt.org/vocab/dimension#"`, `xmlns:dc="http://purl.org/dc/elements/1.1/"`, `xmlns:iems="http://airbus-group/installationMeasure#"`, `xmlns:dtype="http://www.linkedmodel.org/schema/dtype#"`, and `xmlns:mat="http://airbus-group/material#"`.
- The links between text and model:** This section at the bottom provides a detailed view of the synchronization links. It includes tabs for **T2M**, **M2T**, and **Link**. The **Link** tab is active, showing a list of RDF descriptions and their corresponding model identifiers. For example, it lists `<rdf:Description rdf:about="http://ModelWriter/TxtDocument/id270">` with properties like `<j.0:hasOffset>270</j.0:hasOffset>` and `<j.0:isSameAs>http://www.linkedmodel.org/schema/vaem#id</j.0:isSameAs>`. Other entries include `<rdf:Description rdf:about="http://ModelWriter/TxtDocument/attach818">` and `<rdf:Description rdf:about="http://ModelWriter/TxtDocument/attached709">`.

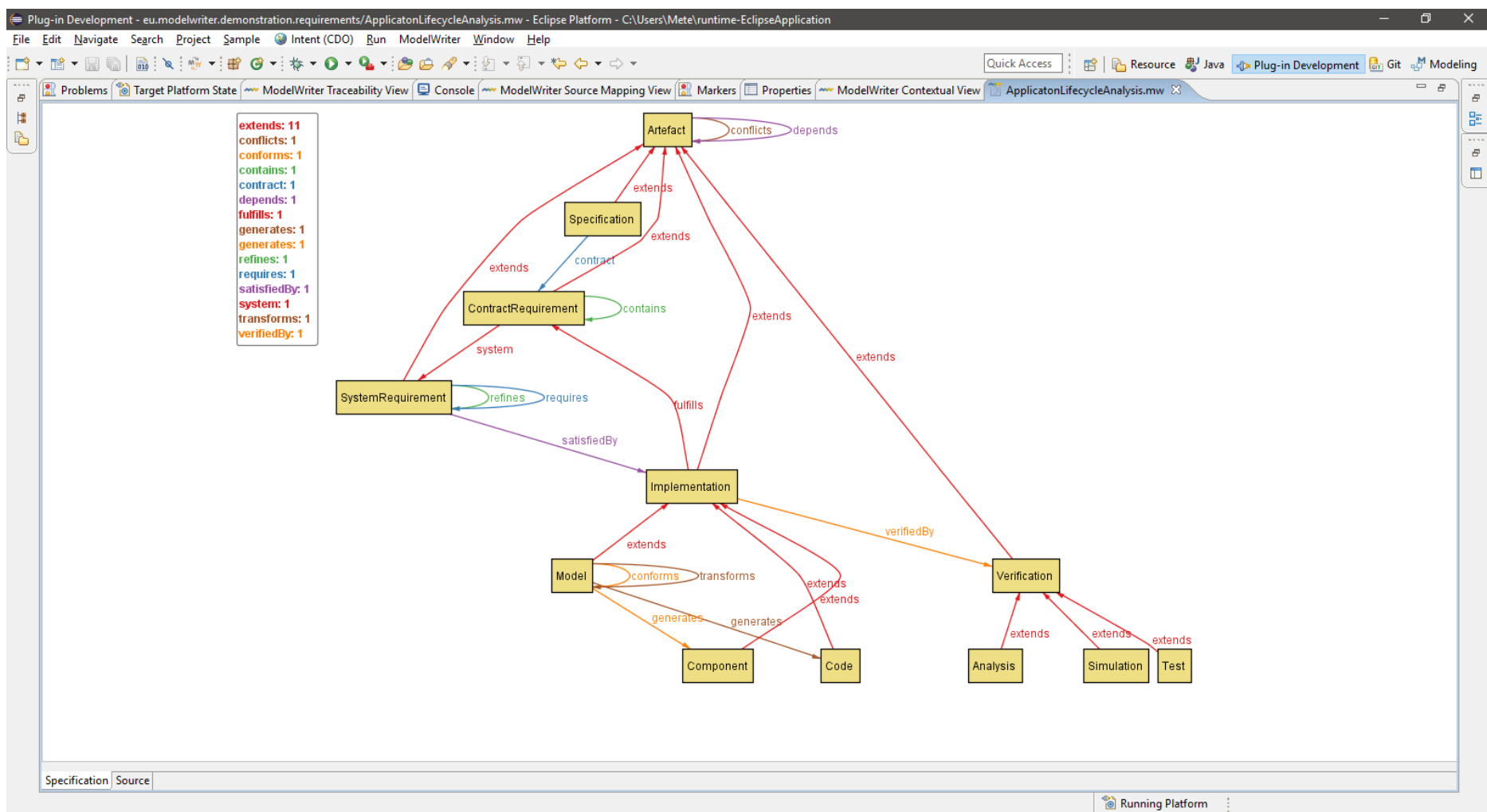
Text & Model-Synchronized Document Engineering Platform



Synchronization is maintained!

What about configuration/formalization of the platform?

Configuration: Havelsan example



A Formal Specification Model to configure the ModelWriter

Is it possible to visualize the trace links?

Traceability: Havelsan example

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

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

Quick Access Resource Java Plug-in Development Git Modeling

ApplicationLifecycleAnalysis.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

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

Req18 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.

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

ContractRequirement2

Code

SystemRequirement2

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

A Formal Specification Model to configure the ModelWriter

**Can we reason about trace
locations and links to repair
broken or identify missing links?**

Traceability: Havelsan example

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

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

Quick Access Resource Java Plug-in Development Git Modeling

ApplicationLifecycleAnalysis.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

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

Req18 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.

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

ContractRequirement2

Code

SystemRequirement2

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

A Formal Specification Model to configure the ModelWriter

Airbus Use Cases

Ford-Otosan Use Case

Havelsan Use Case