

# 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)*

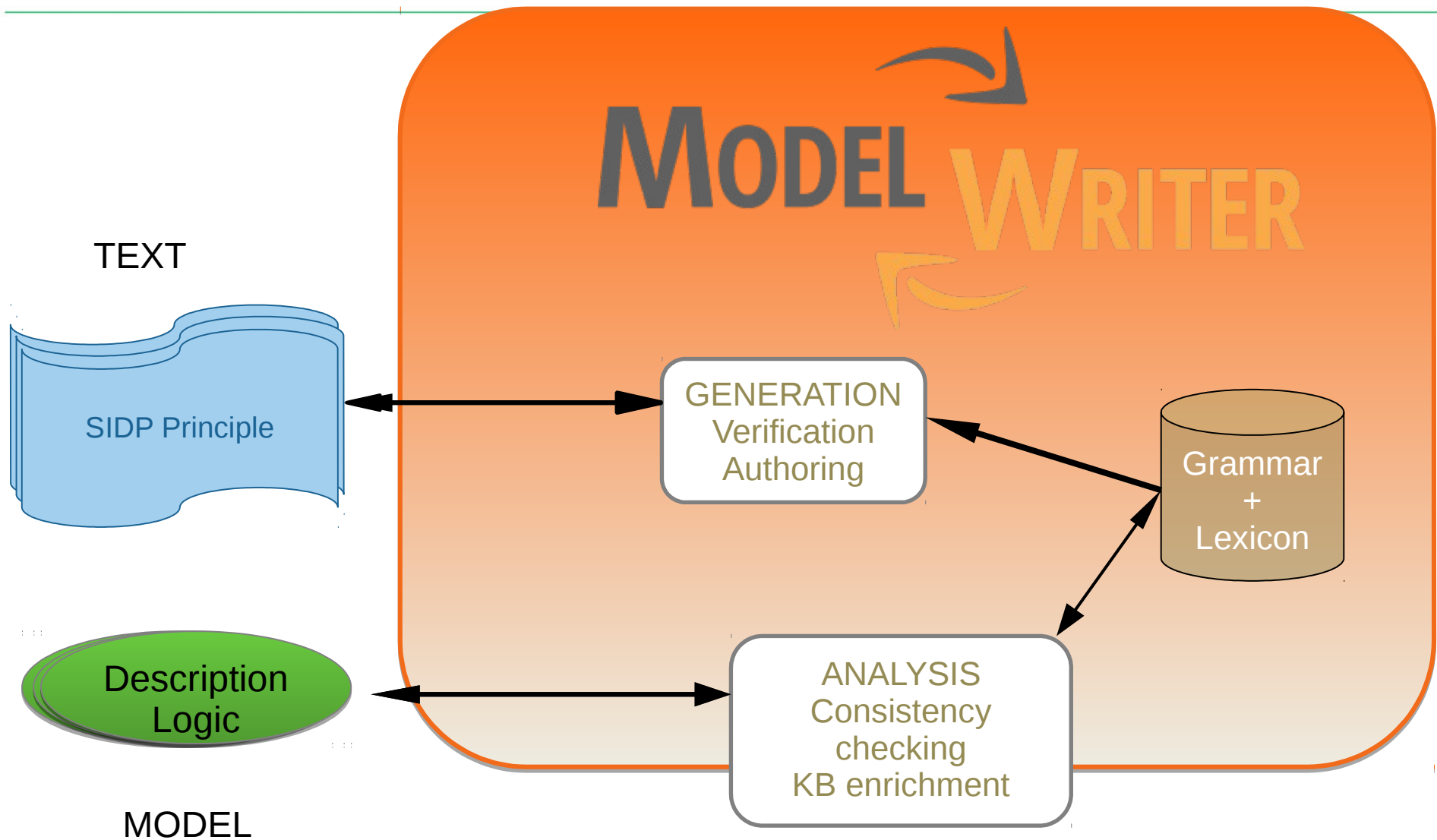
*Dr. Bikash Gyawali (CNRS/LORIA, Technical Contact)*

*Anastasia Shimorina (CNRS/LORIA, Technical Contact)*

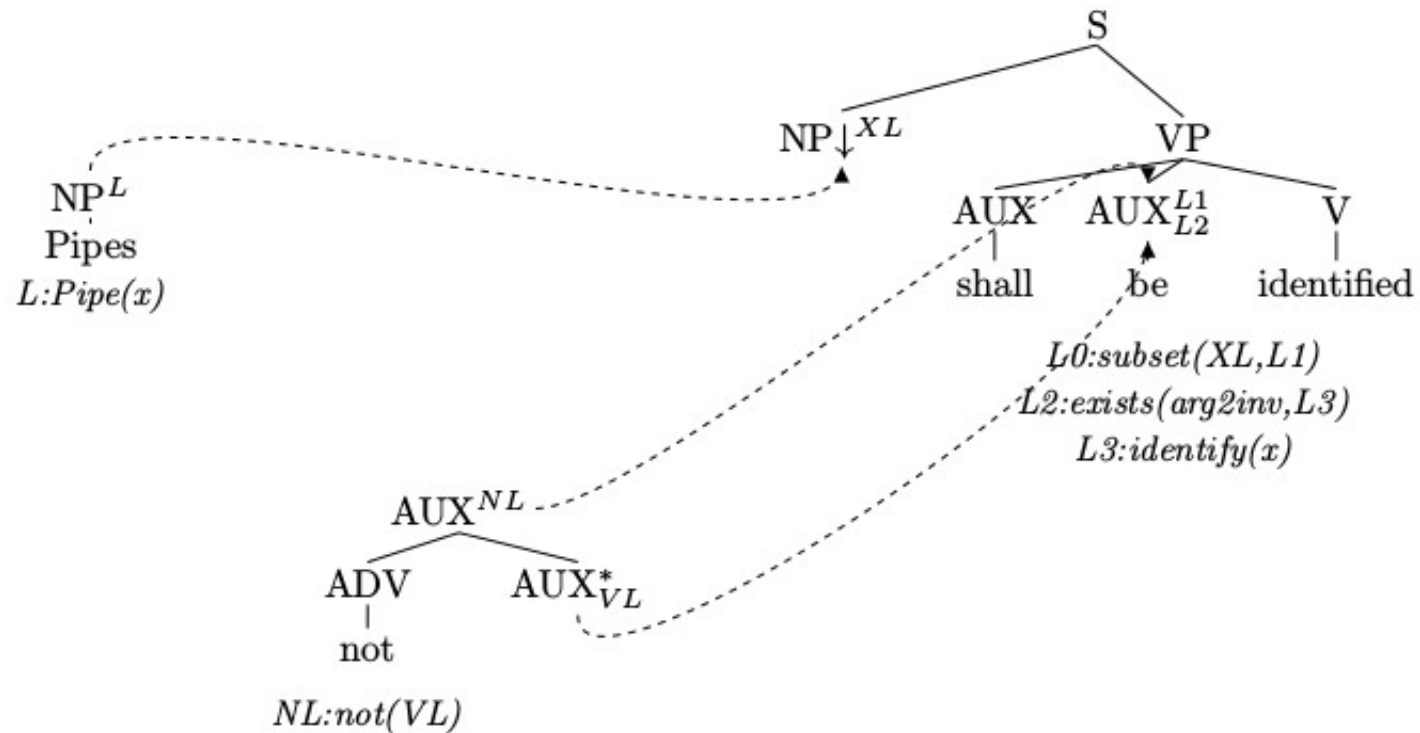
*Dr. Geylani Kardas (KOCSISTEM, Consultant)*

# Airbus Use Cases

# Synchronising Text and Model



# Grammar-Based Parsing and Generation



# Grammar and Lexicon

Semantics:

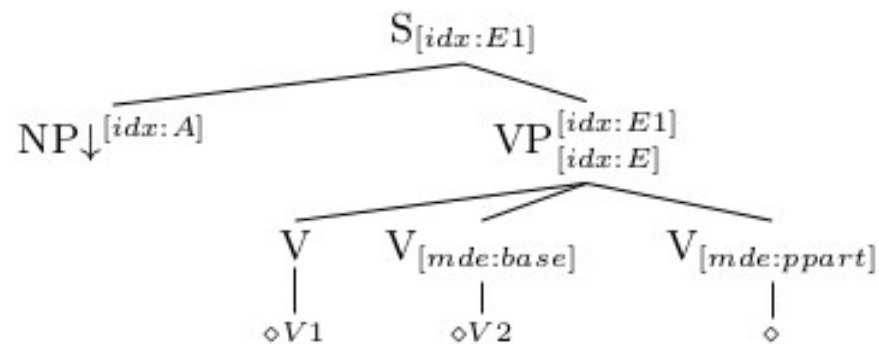
$V = Identify, A2 = A2_{inv}$

Tree: nx0V

Anchor: *identified*

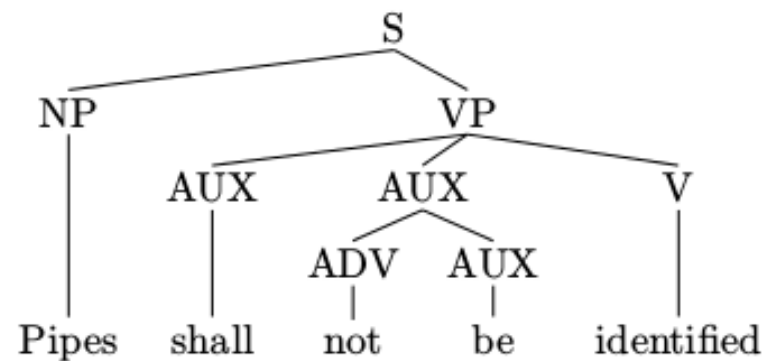
Coanchor:  $V1 \rightarrow shall/V$

Coanchor:  $V2 \rightarrow be/V$



$L0:subset(PL, NL) \quad VPL:exists(A2, VL) \quad VL:V(x)$

# Grammar-Based Parsing and Generation



$PL:Pipe(x)$   $L0:subset(PL,NL)$   $NL:not(VPL)$   
 $VPL:exists(identifyA2inv,VL)$   $VL:Identify(x)$

$Pipe \sqsubseteq \exists \neg identifyA2^{-}.(Identify)$



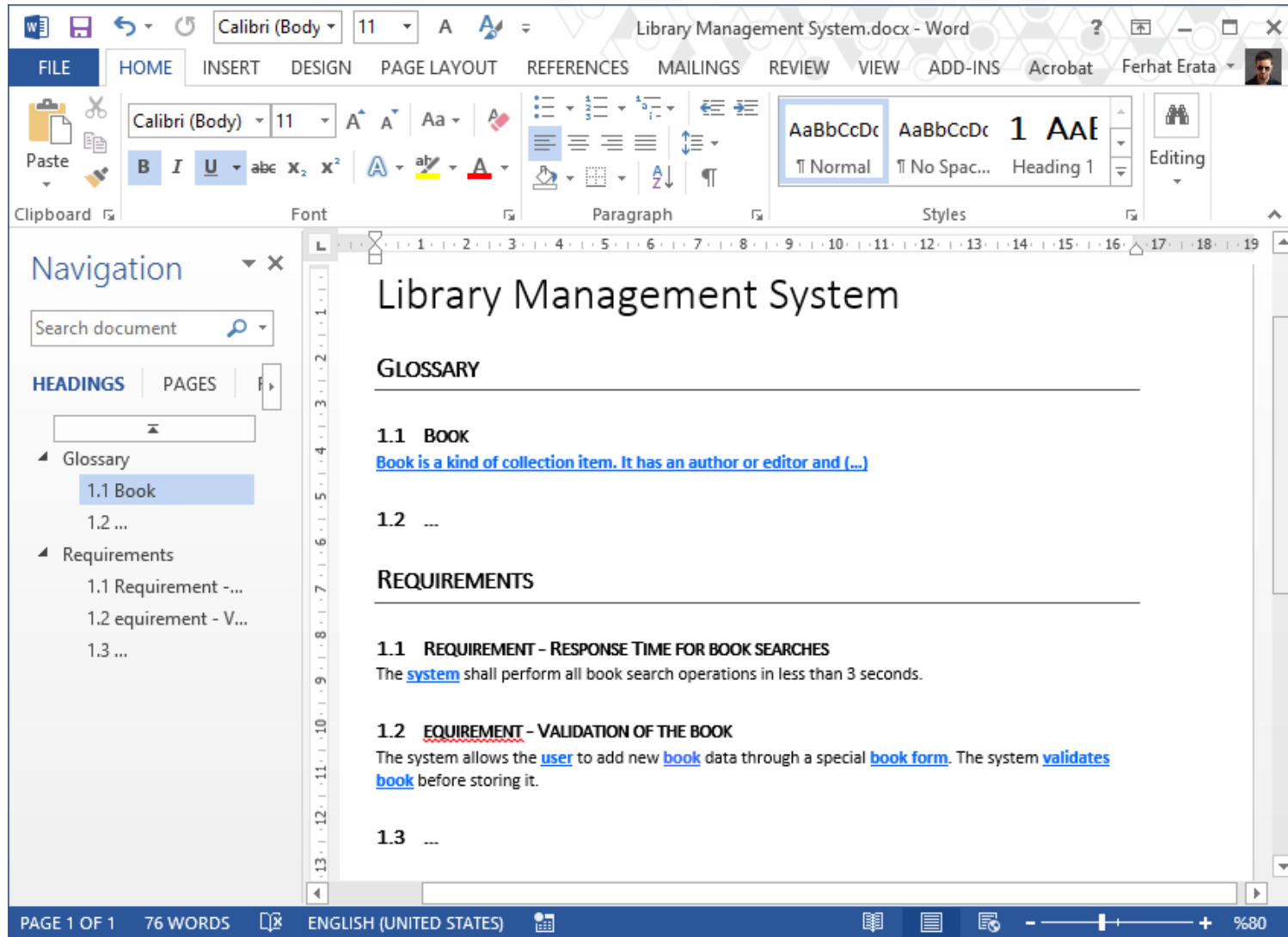
# Ford-Otosan Use Case

# What is a text?



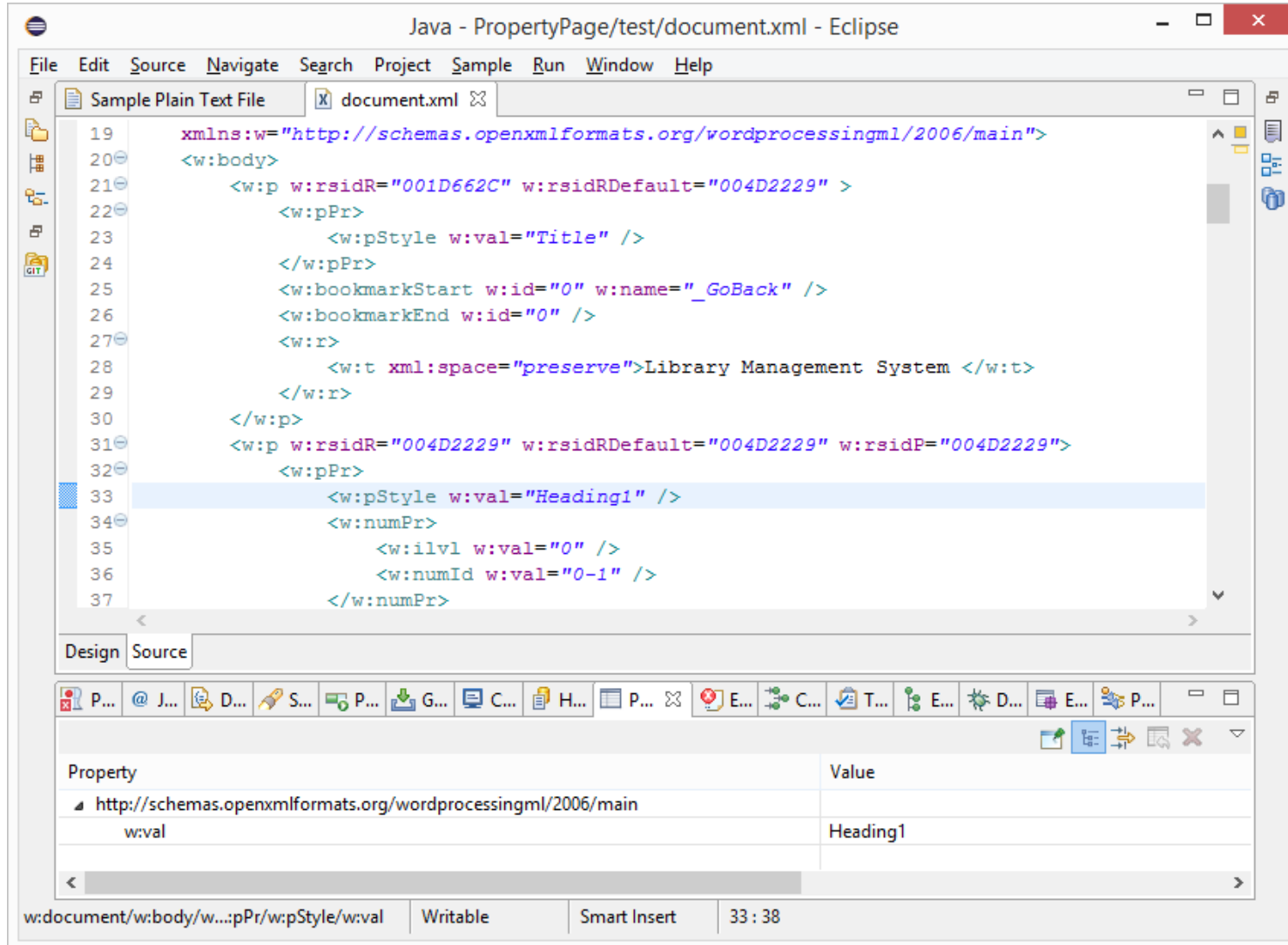
# What is a text? (document file formats)

## 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>

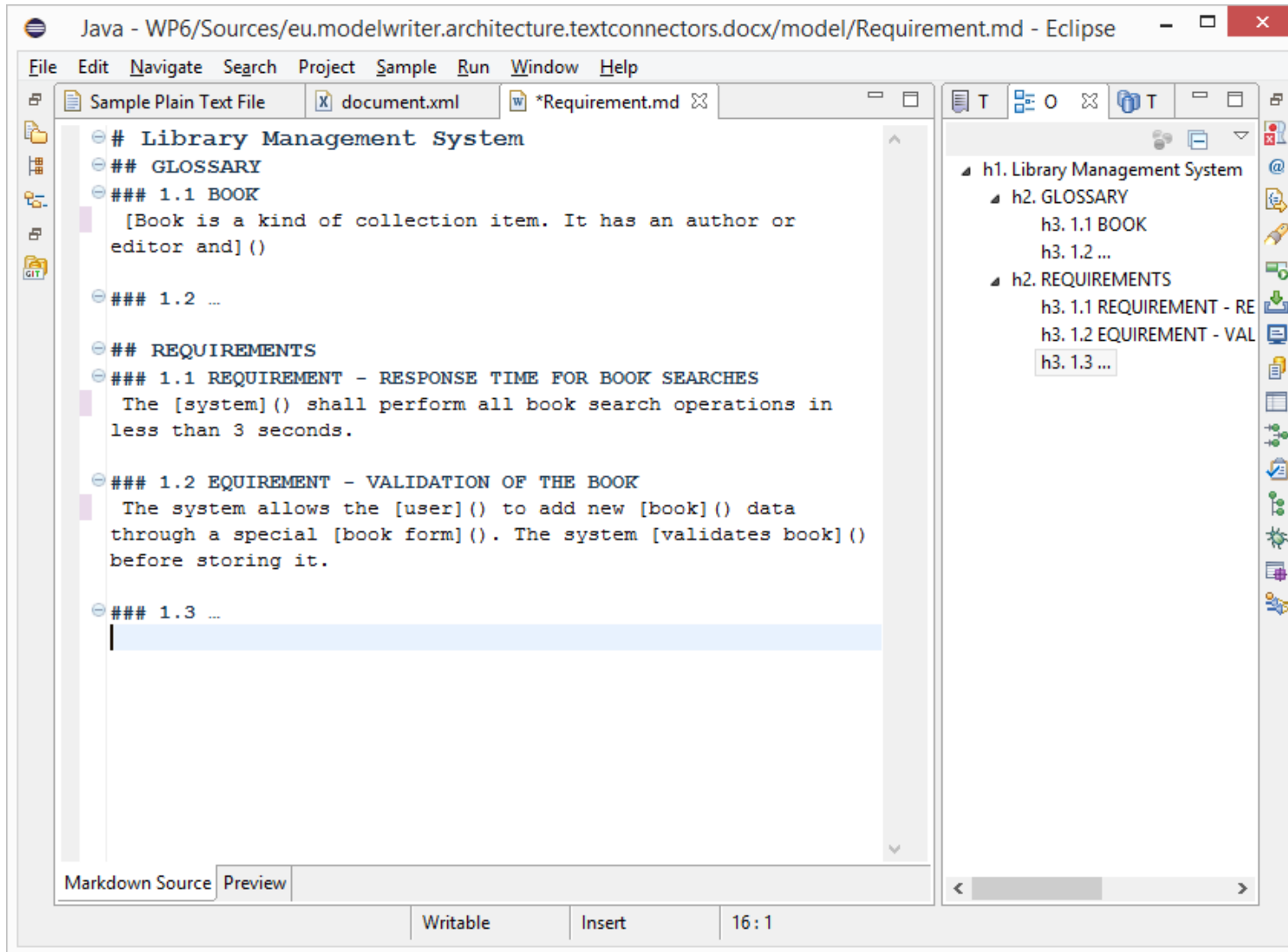
```

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 ...
```

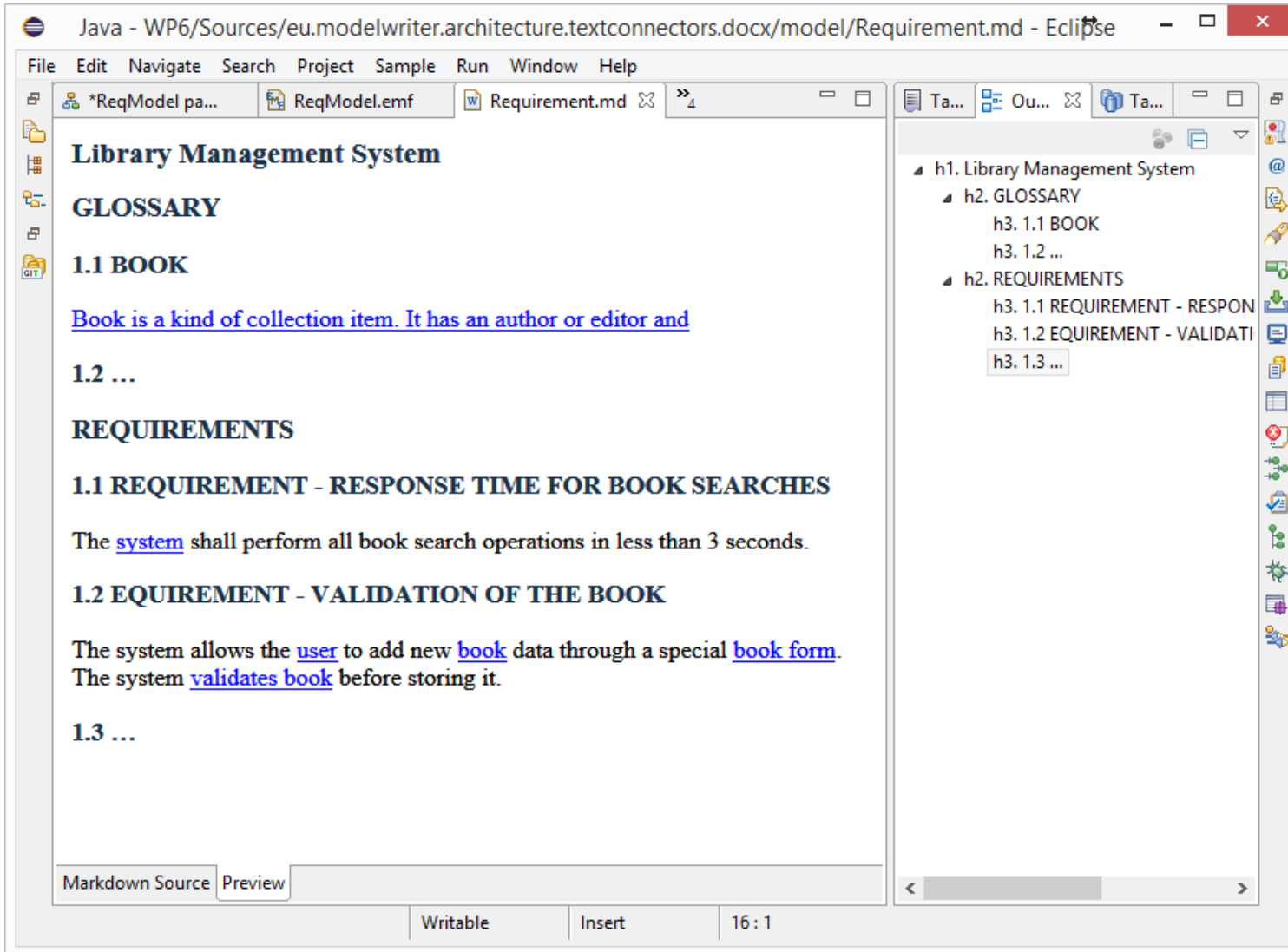
On the right side, there is a preview view showing the rendered markdown structure:

- 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:

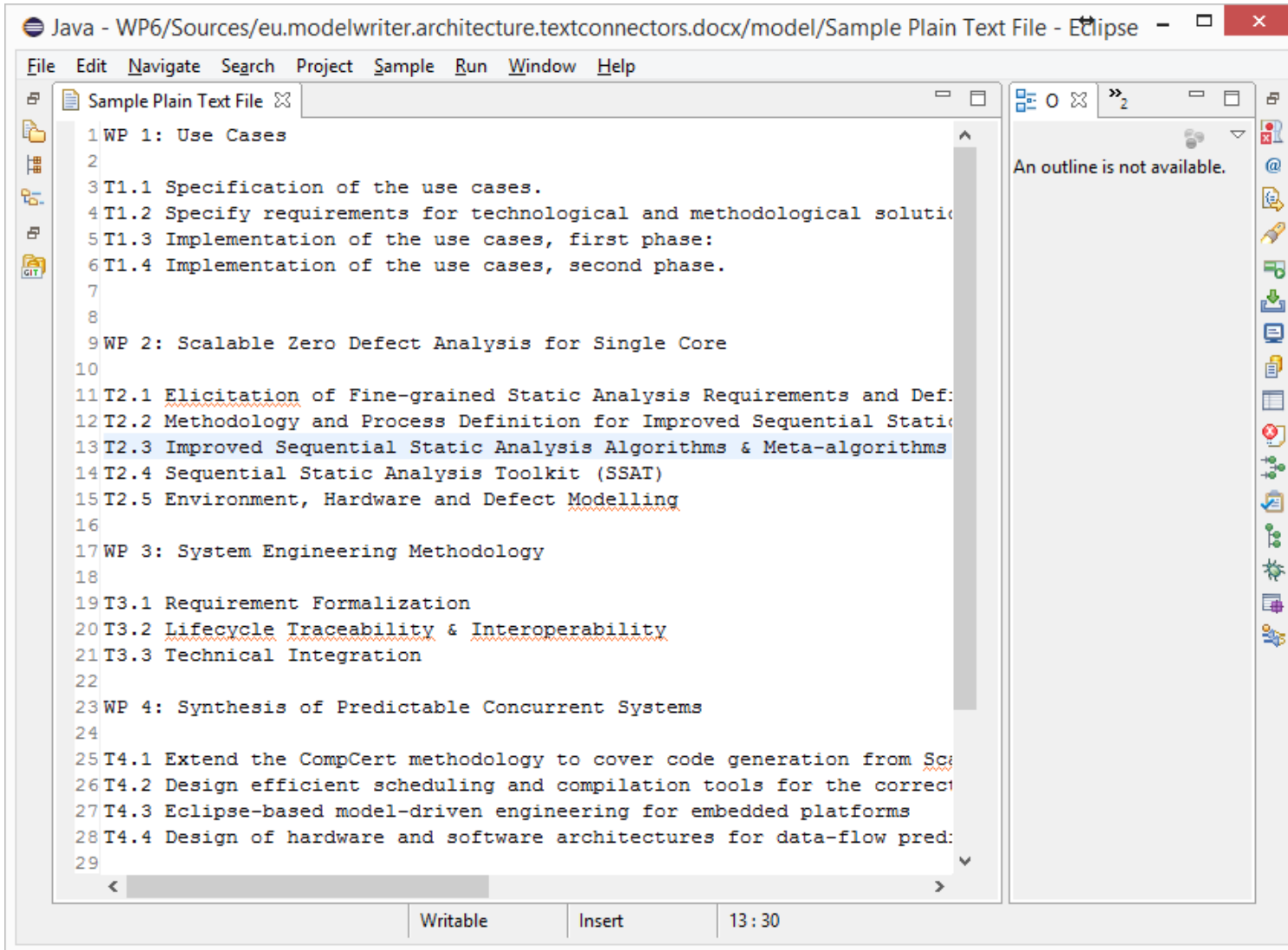
- h1. Library Management System
  - h2. GLOSSARY
    - h3. 1.1 BOOK
    - h3. 1.2 ...
  - h2. REQUIREMENTS
    - h3. 1.1 REQUIREMENT - RESPON
    - h3. 1.2 EQUIREMENT - VALIDATI
    - h3. 1.3 ...

At the bottom of the editor, there are tabs for "Markdown Source" and "Preview", with "Preview" currently selected. The status bar at the 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

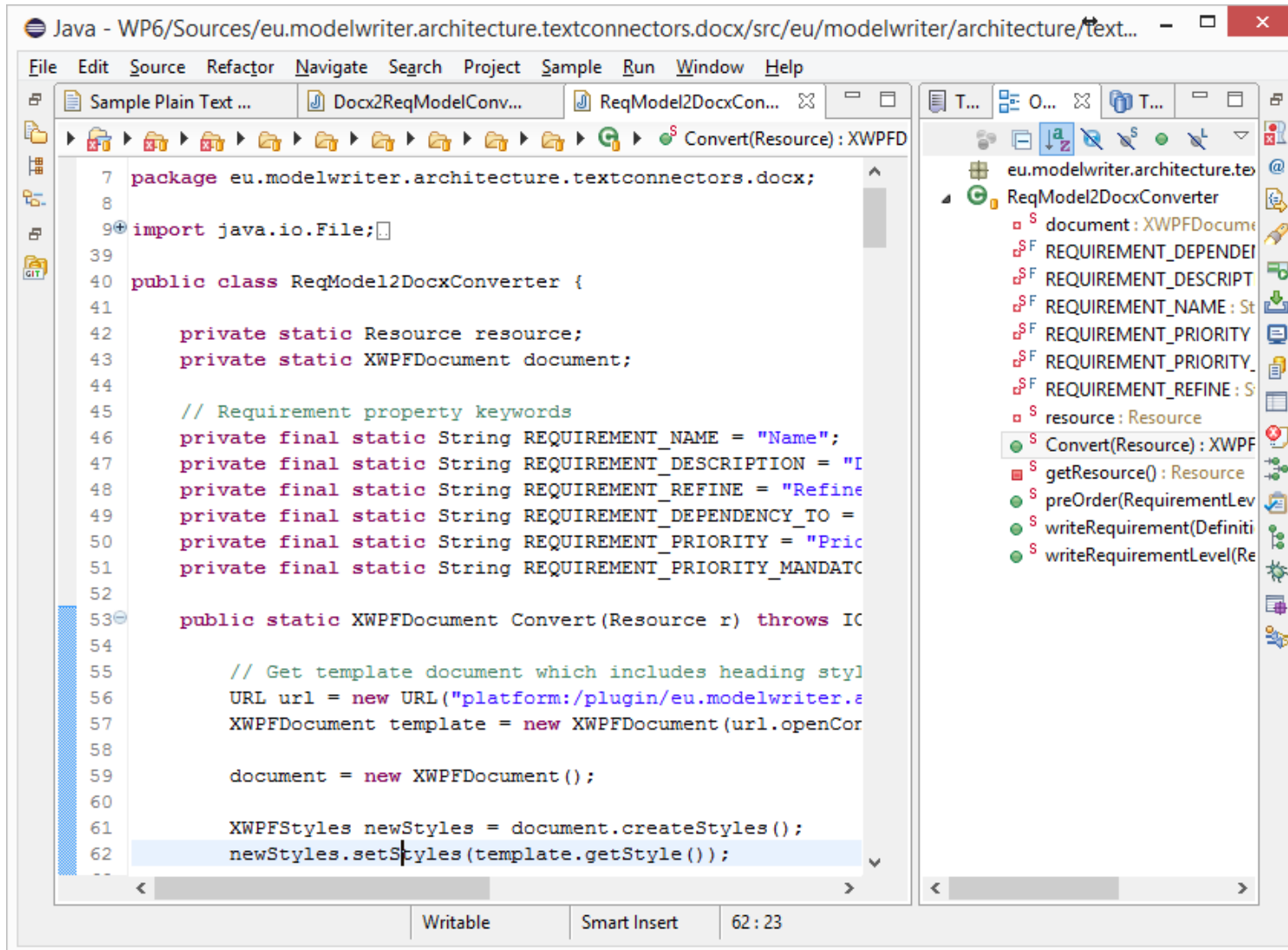
```
1 WP 1: Use Cases
2
3 T1.1 Specification of the use cases.
4 T1.2 Specify requirements for technological and methodological solutions
5 T1.3 Implementation of the use cases, first phase:
6 T1.4 Implementation of the use cases, second phase.
7
8
9 WP 2: Scalable Zero Defect Analysis for Single Core
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 a Java code file named "ReqModel2DocxConverter.java". The code defines a package, imports java.io.File, and defines a public class ReqModel2DocxConverter with static fields for resource and document, and a public static method Convert. The method Convert uses a URL to load a template document and creates styles for it. The project explorer on the right shows the project structure, including the package eu.modelwriter.architecture.textconnectors.docx and the class ReqModel2DocxConverter.

```
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.openConnection().getInputStream());

        document = new XWPFDocument();

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

# 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	Ⓡ R123 Response Time for Book Searches	The system shall perform all book search operations in less than 3 second
1.3	Ⓡ UC071 Add new Book	A user form to add new book
1.4	Ⓡ R124 Validation of the Book	Validation of the Book

Outline

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

Properties

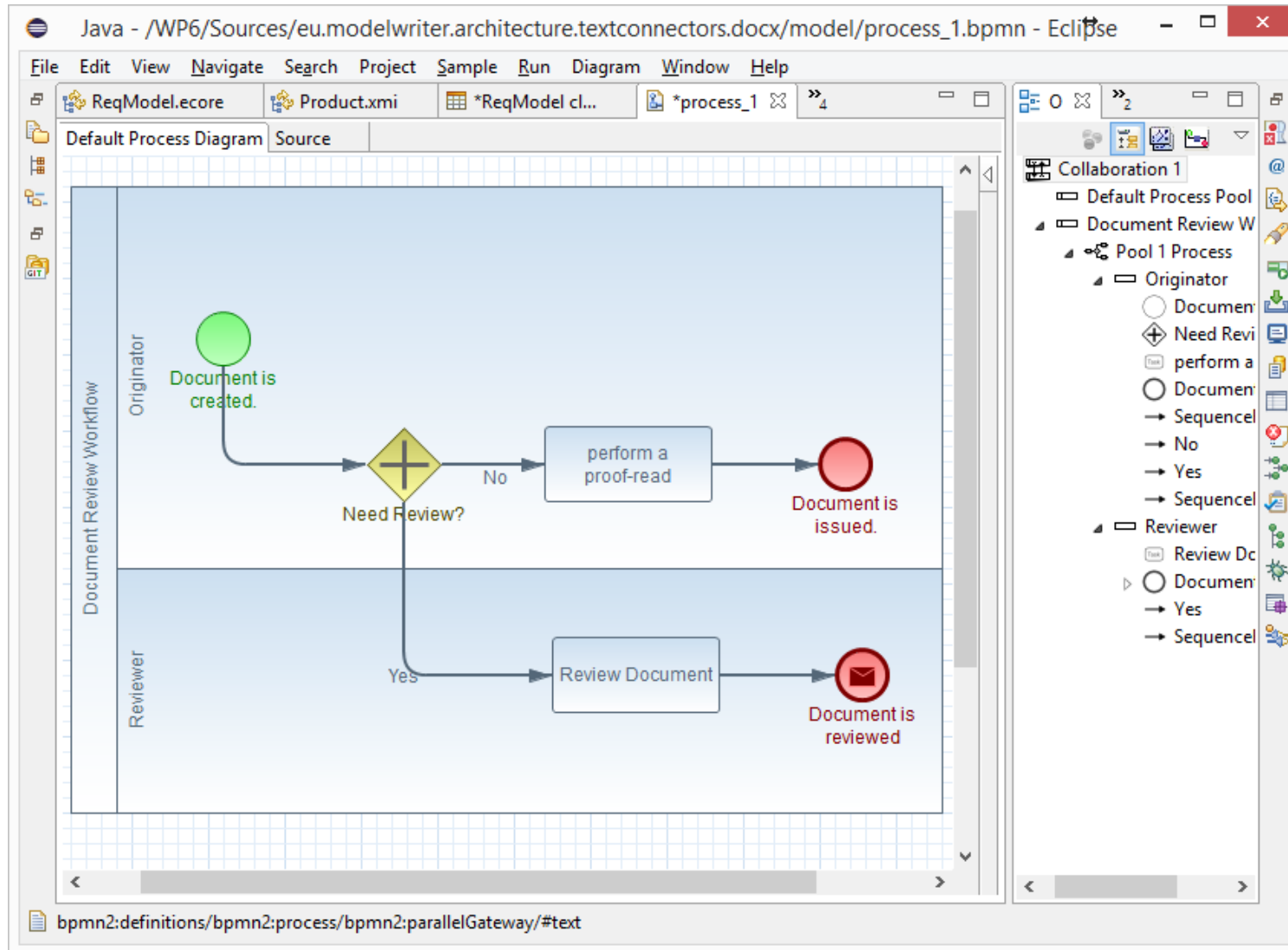
Property	Value
<b>Requirement Type</b>	
Description	The system shall perform all book search operations in less than 3 second
ID	R123
Name	Response Time for Book Searches
Responsible	Ferhat
Version	1
<b>Spec Object</b>	
Type	Ⓡ Requirement Type (Spec Object)

Standard Attributes All Attributes



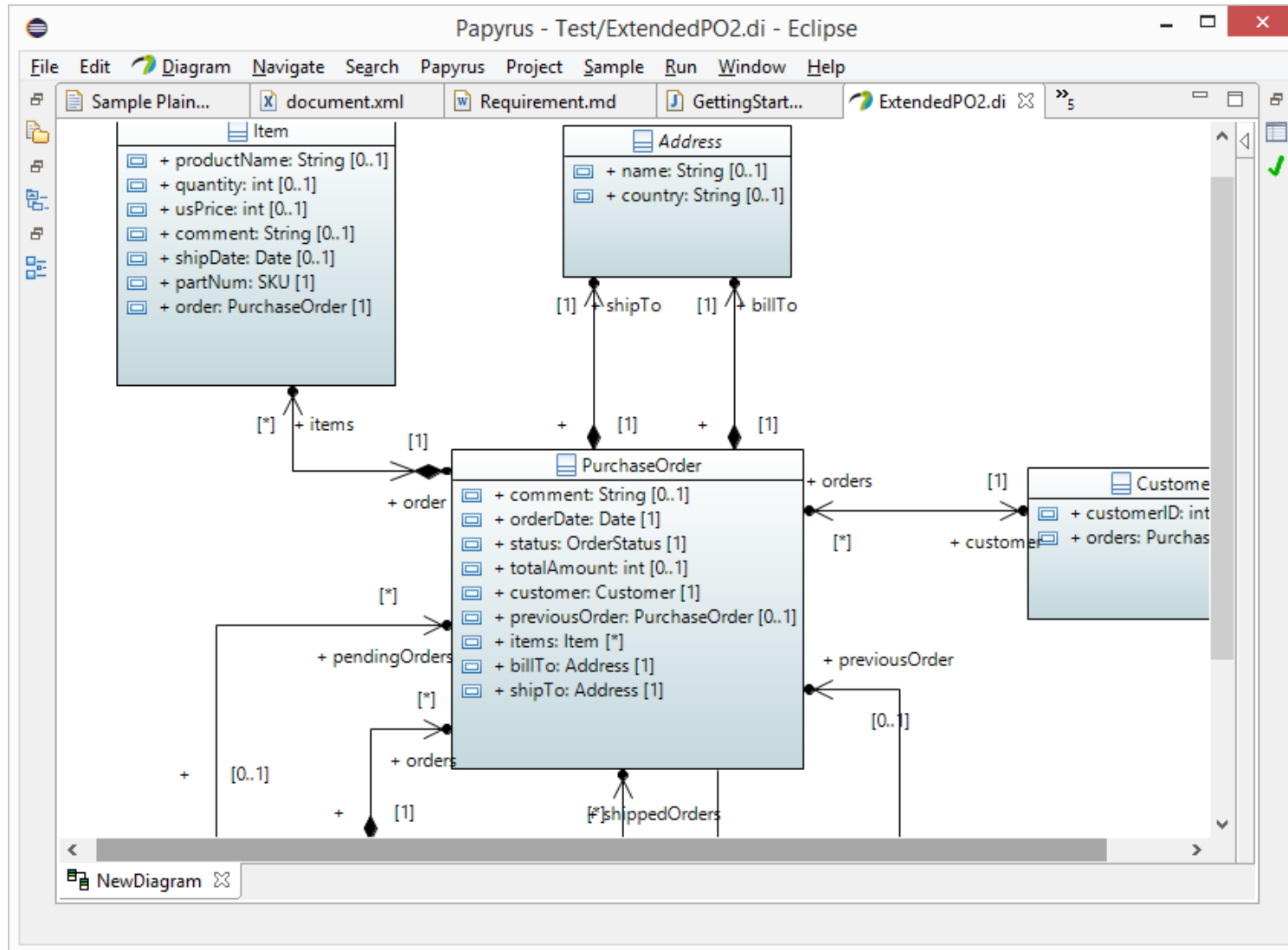
# Everything is a model! (BPMN Standard)

## Business Process Model & Notation



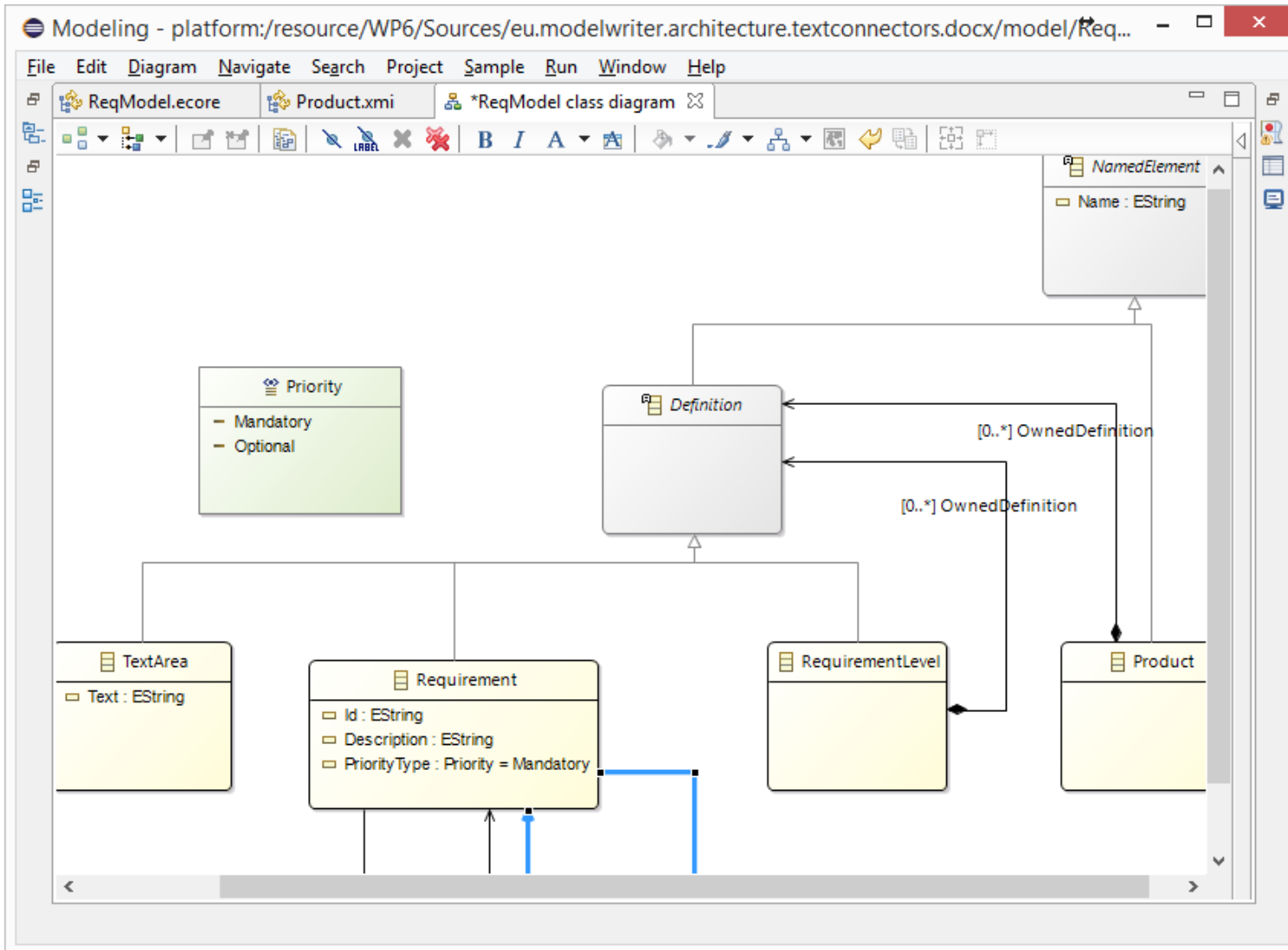
# Everything is a model! (UML Standard)

## UML Modeling Languages



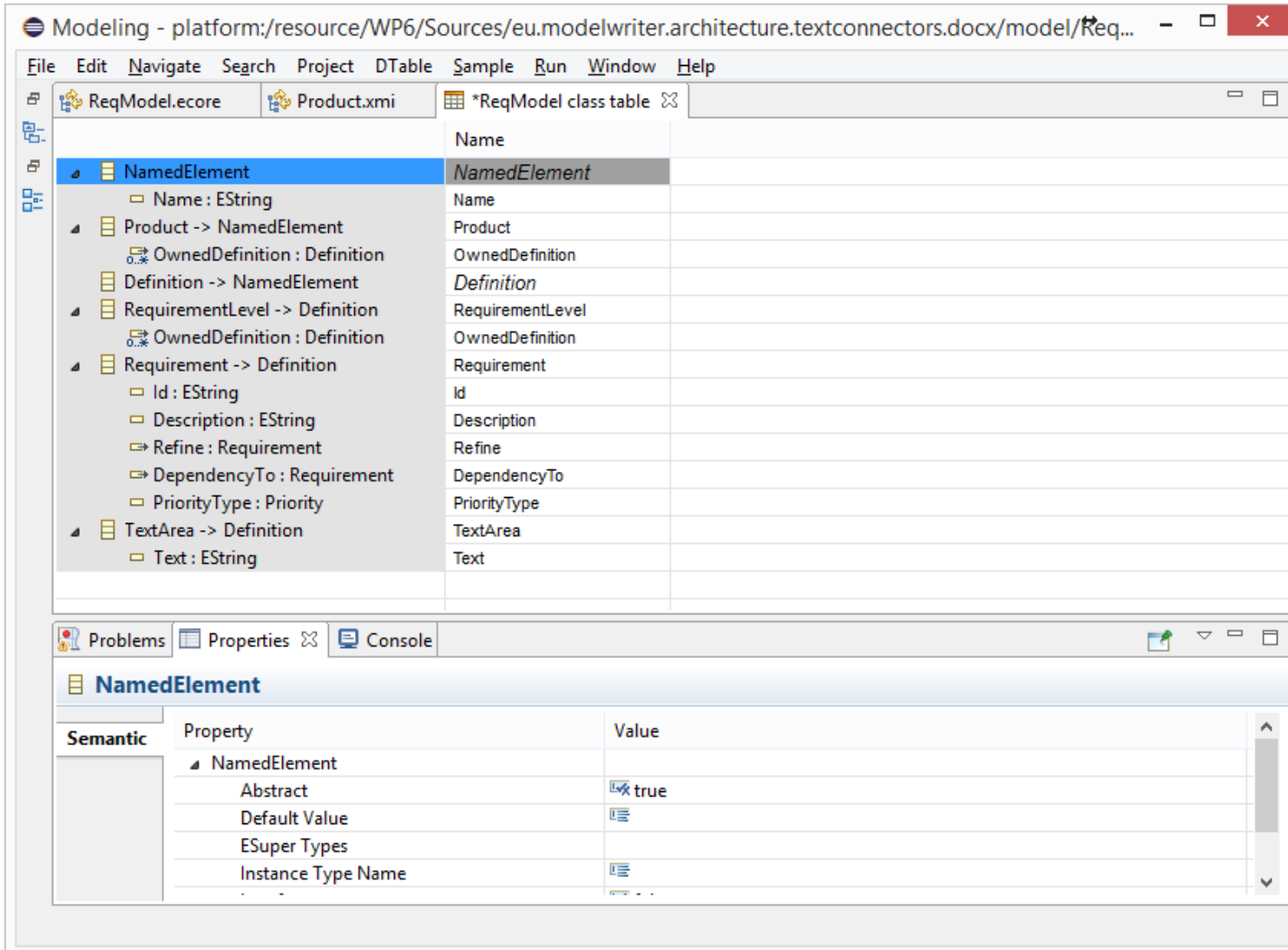
# Everything is a model!

## 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		

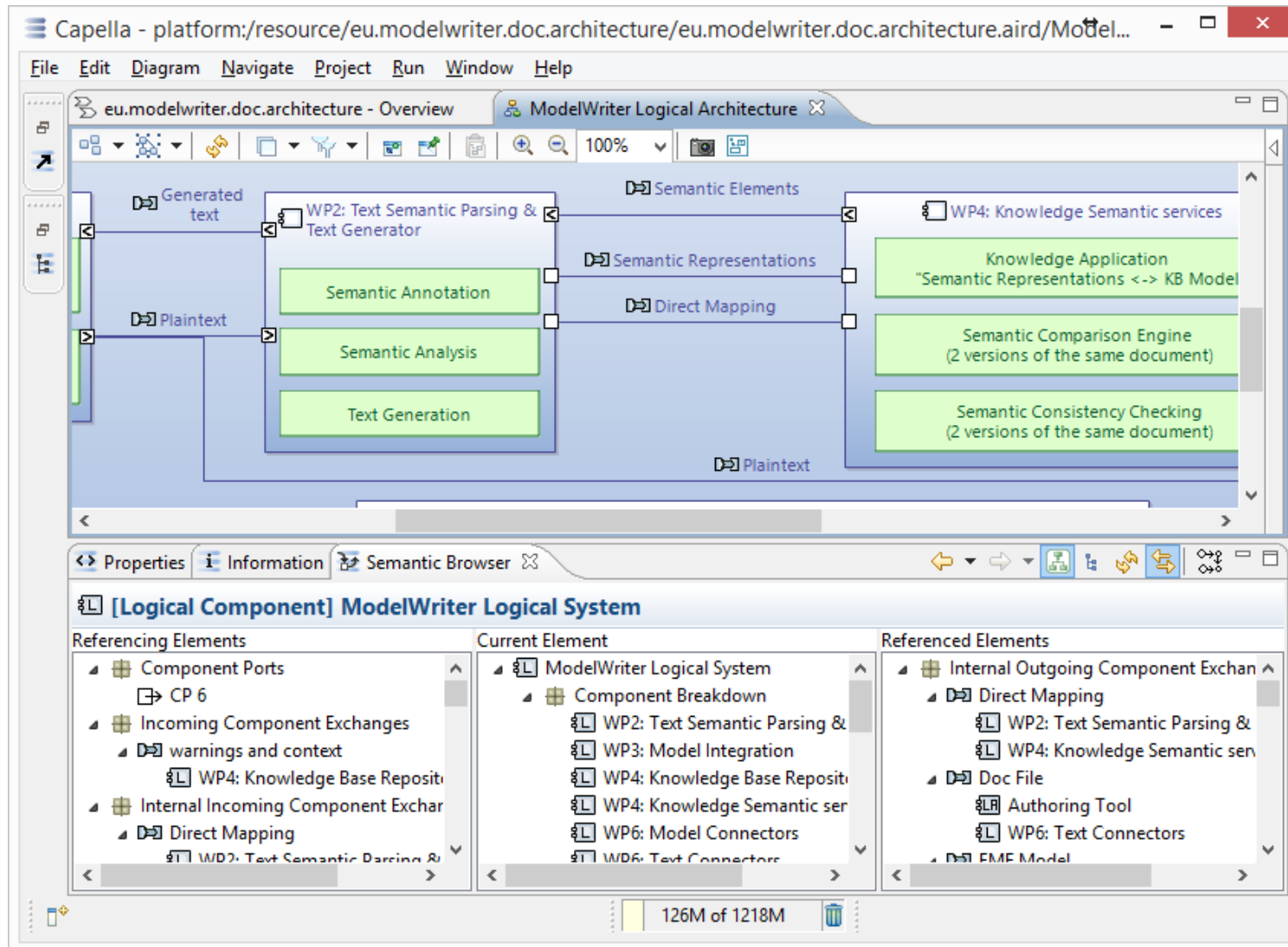
The bottom pane shows the Properties view for the selected NamedElement. It has a tab labeled "Semantic" and a table with the following columns: Property and Value.

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



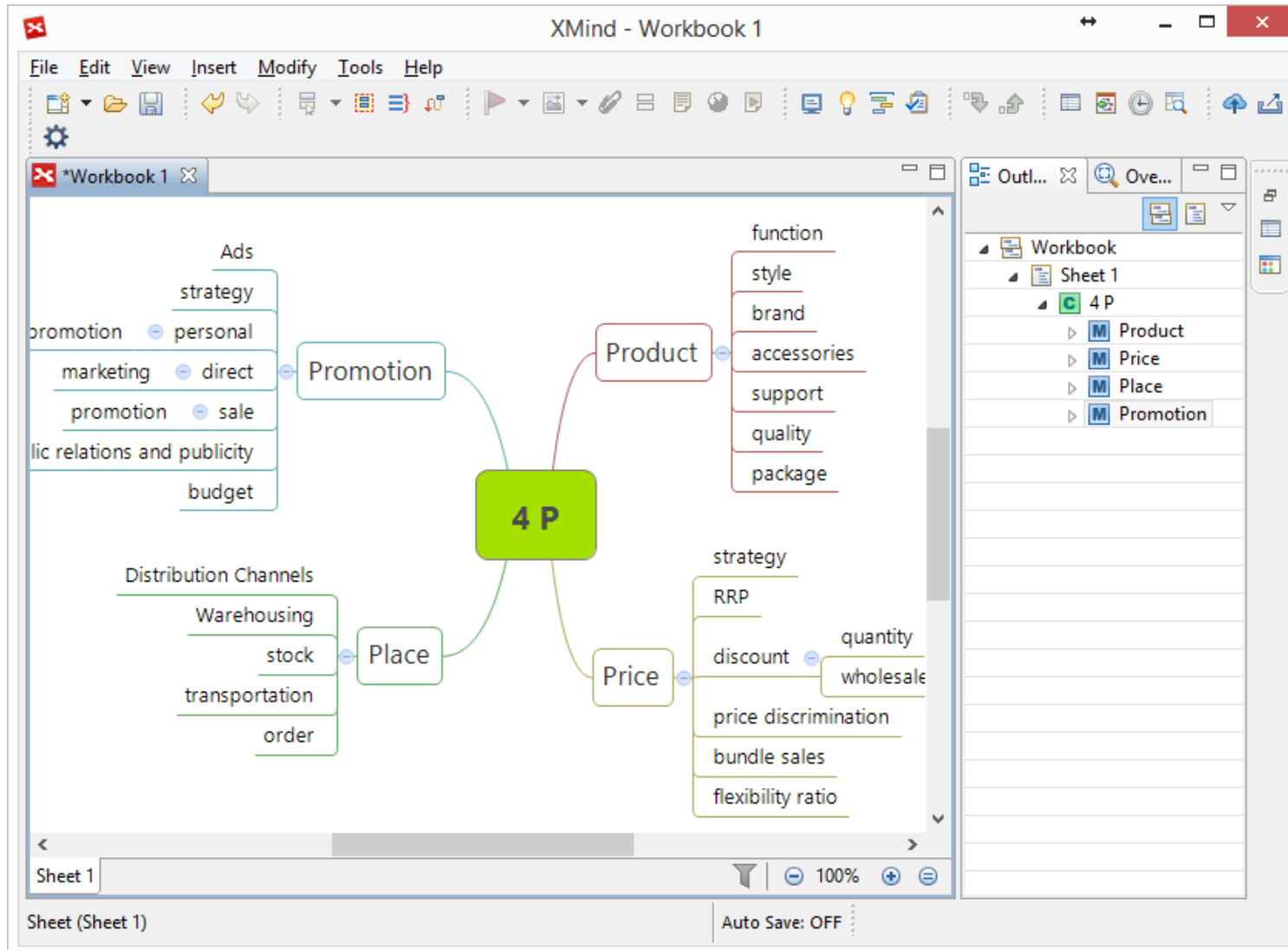
# Everything is a model!

## Software/System Architecture Design



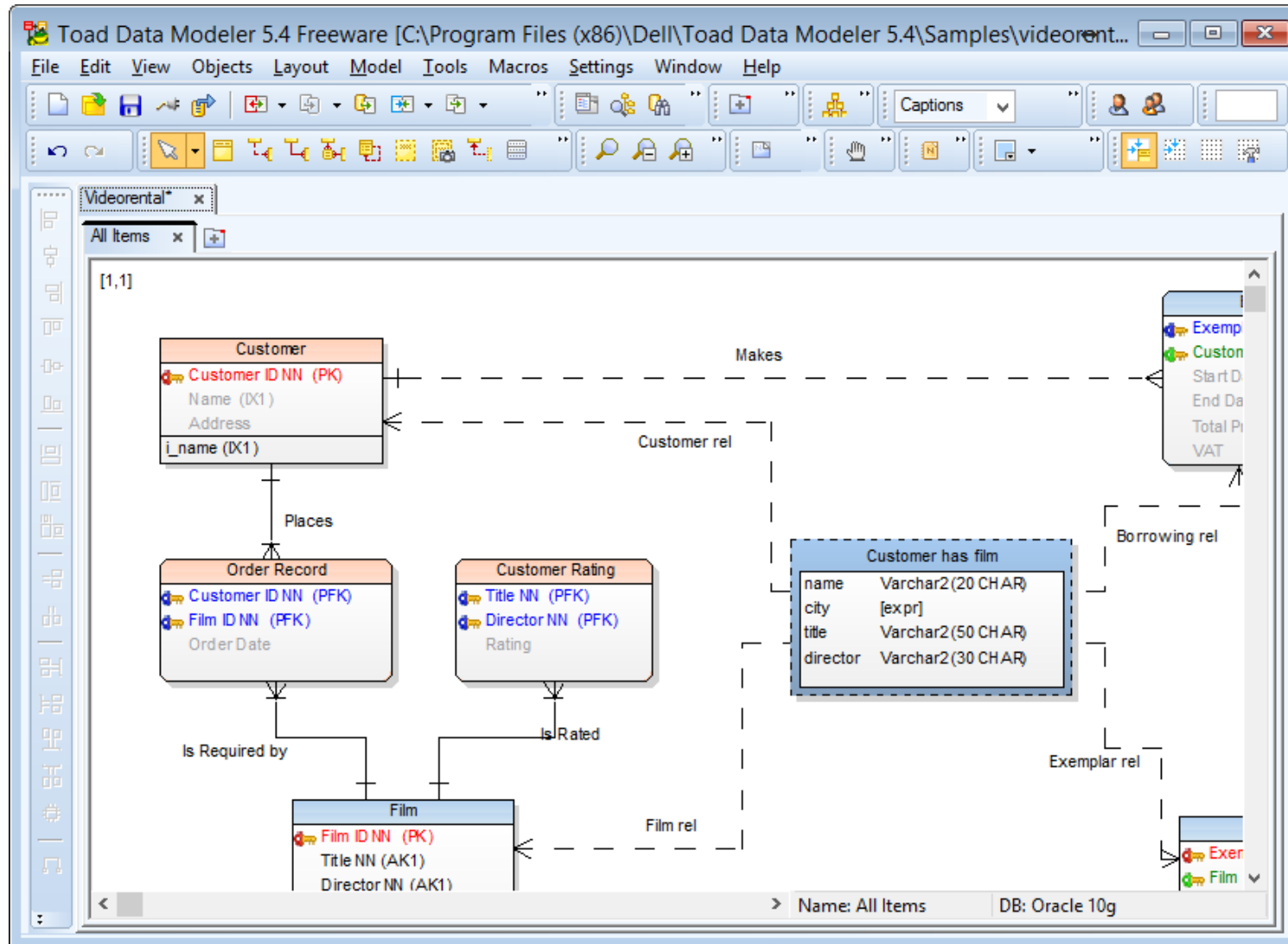
# Everything is a model!

## Topic Maps, Mind Maps, Vocabularies ...



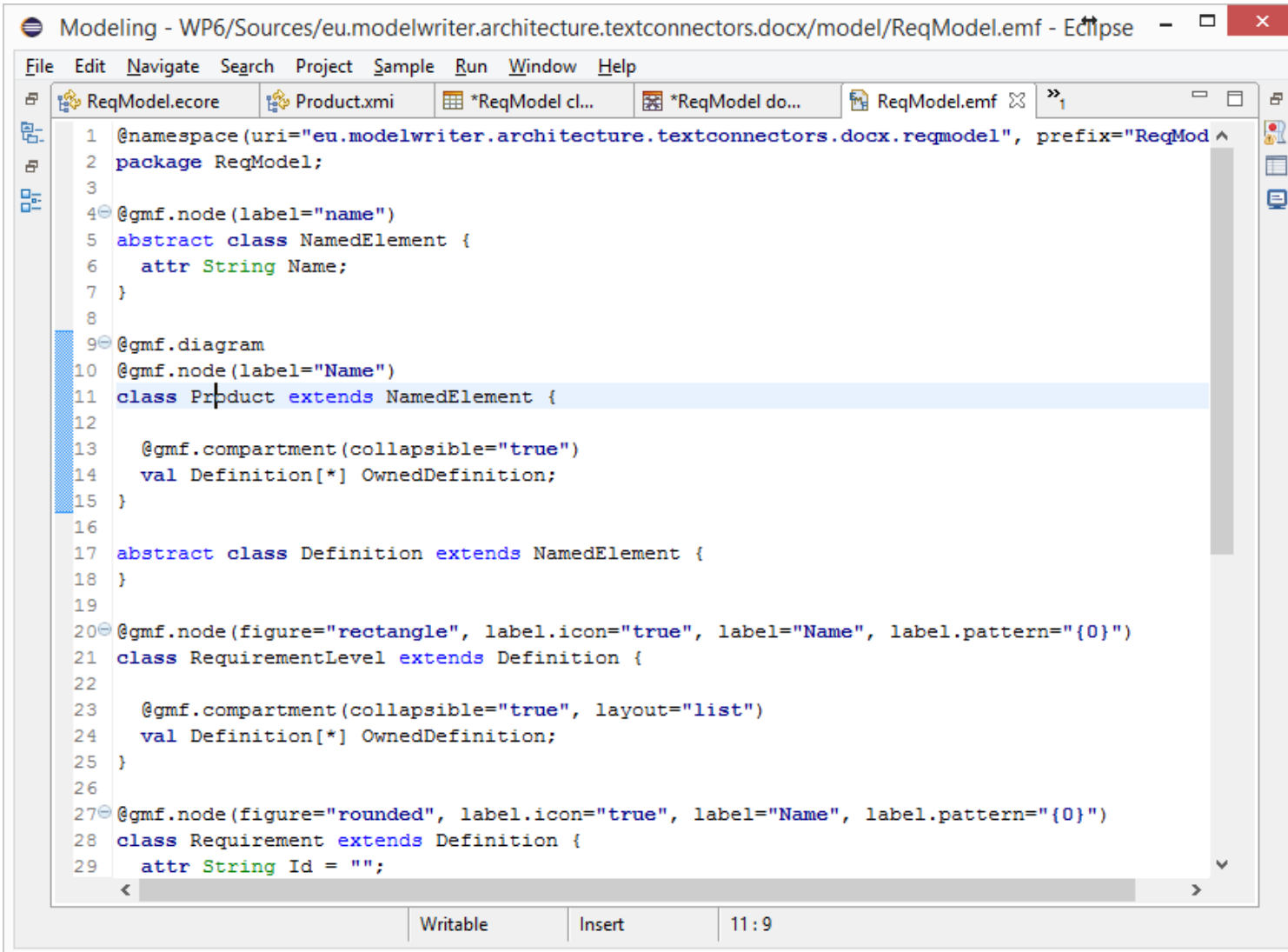
# Everything is a model!

## Databases (ER, IDEF1.x)



# Everything is a model! (Textual Lang.)

## Domain Specific Languages



The screenshot shows the Eclipse IDE with a project named "Modeling - WP6/Sources/eu.modelwriter.architecture.textconnectors.docx/model/ReqModel.emf". The editor displays the GMF diagram model file "ReqModel.emf" with the following code:

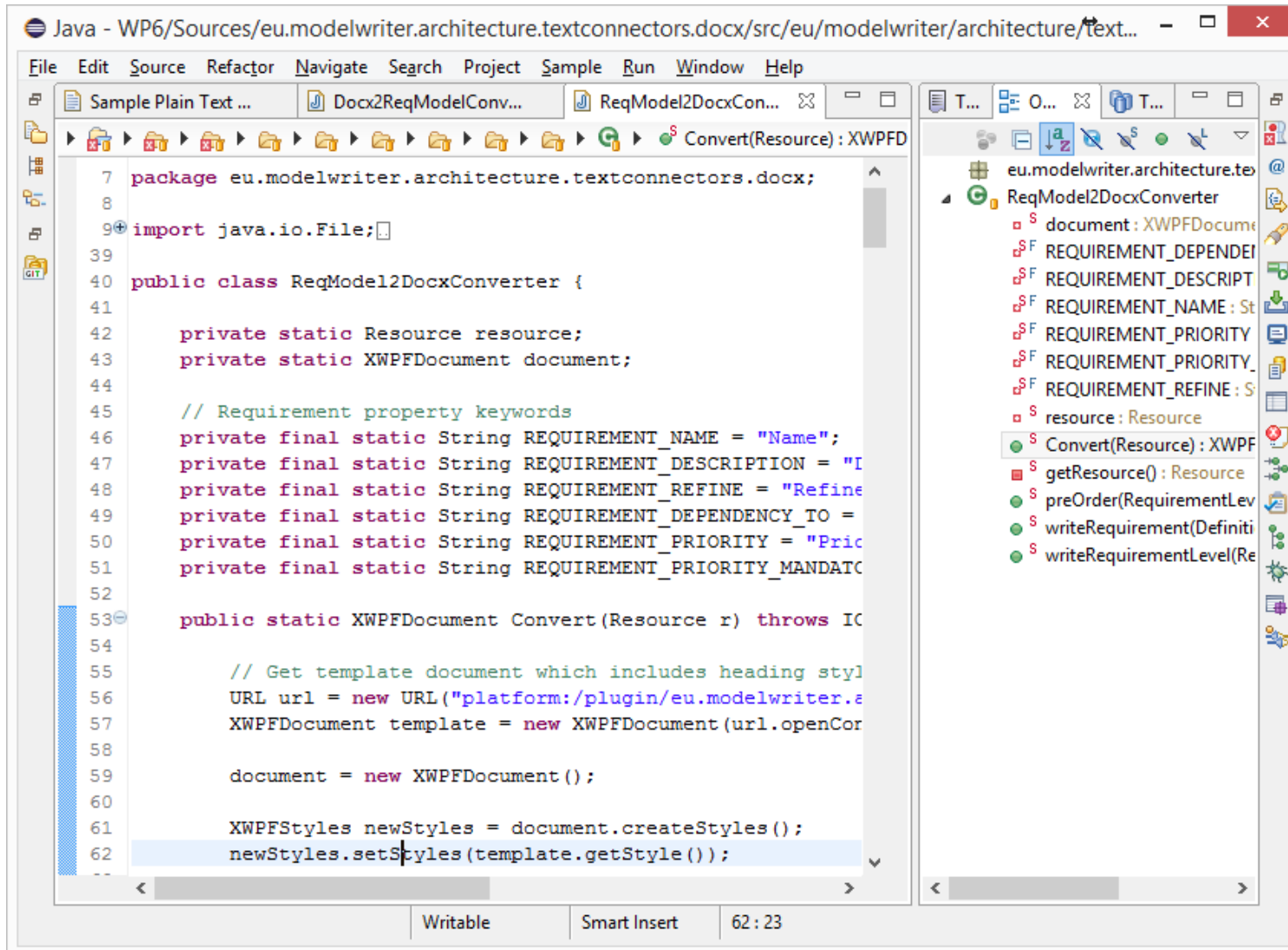
```
1 @namespace(uri="eu.modelwriter.architecture.textconnectors.docx.reqmodel", prefix="ReqMod
2 package ReqModel;
3
4 @gmf.node(label="name")
5 abstract class NamedElement {
6     attr String Name;
7 }
8
9 @gmf.diagram
10 @gmf.node(label="Name")
11 class Product extends NamedElement {
12
13     @gmf.compartment(collapsible="true")
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 = "";
```

The status bar at the bottom indicates the file is "Writable", in "Insert" mode, and the time is "11:9".



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

## Even Programming Languages (ASTs)



**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
```

Red circle 1 highlights the "Mapping Action" dialog box, which shows relations for the 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

Red circle 2 highlights the "Management" menu, which includes options like Analysis, Refresh, Zoom In, Zoom Out, Zoom to Fit, and Export to PNG or PDF.

Red circle 3 highlights the "Check Consistency" and "Reason on relations" buttons in the "Management" menu.

The diagram view shows a graph of elements and their relationships:

- Specification** (yellow box) is connected to **ContractRequirement1** and **ContractRequirement0** via red arrows labeled "contract".
- ContractRequirement1** is connected to **SystemRequirement2** via a dashed blue arrow labeled "fulfills".
- ContractRequirement0** is connected to **SystemRequirement2** via a dashed blue arrow labeled "satisfiedBy".
- SystemRequirement2** is connected to **Code** (yellow box) via a dashed blue arrow labeled "fulfills".
- SystemRequirement2** is connected to **SystemRequirement0** via a dashed blue arrow labeled "requires".
- SystemRequirement0** is connected to **SystemRequirement1** via a dashed blue arrow labeled "requires".
- SystemRequirement0** is connected to **Model** (yellow box) via a dashed blue arrow labeled "satisfiedBy".
- SystemRequirement1** is connected to **Model** via a dashed blue arrow labeled "refines".

The "Management" menu is open, showing options like "Change Type", "Delete Atom", and "Map Atom".

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 interface, which is designed for knowledge extraction and synchronization between text documents and an RDF model.

**ModelWriter Project**

**File Link Change Statistic**

**The**

- Generate Links
- Search Link
- Add Link
- Remove Link

2 Pipes shall be defined with a maximum length of 500 mm regardless of  
ABS2195 -LRB- preferred for weight saving -RRB- or NSA5516J P-Clamp Shall be u  
Rigid Pipes Shall be segregated to fixed Structure by not less than 10 mm as sh  
Flexible Hoses Shall be segregated to rigid Component/Item/Object by not less t  
Rigid Pipes Shall be segregated to movable Component/Item/Object by not less  
Flexible Hoses Shall be segregated to movable Component/Item/Object by not less  
Pipes Shall be fixed  
Unions Shall be fixed on Pipes at alternating positions as shown in the attached  
Unions Shall be positioned close to one fixation point .

**The Model**

Plain Tree

```
<?xml version="1.0"?>
<rdf:RDF
  xmlns:acs="http://airbus-group/aircraft-system#"
  xmlns:evt="http://airbus-group.installsys/event#"
  xmlns:rdf="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#"
  xmlns:mat="http://airbus-group/material#"
  >
```

**The links between text and model**

T2M M2T Link

```
</rdf:Description>
<rdf:Description rdf:about="http://ModelWriter/TxtDocument/id270">
  <j.0:hasOffset>270</j.0:hasOffset>
  <j.0:isSameAs>http://www.linkedmodel.org/schema/vaem#id</j.0:isSameAs>
  <j.0:hasValue>id</j.0:hasValue>
</rdf:Description>
<rdf:Description rdf:about="http://ModelWriter/TxtDocument/attach818">
  <j.0:hasOffset>818</j.0:hasOffset>
  <j.0:isSameAs>http://airbus-group/opd-function#Attach</j.0:isSameAs>
  <j.0:hasValue>attach</j.0:hasValue>
</rdf:Description>
<rdf:Description rdf:about="http://ModelWriter/TxtDocument/attached709">
  <j.0:hasOffset>709</j.0:hasOffset>
  <j.0:isMorphologySimilarTo>http://airbus-group/opd-function#Attach</j.0:isMorphologySim
  <j.0:hasValue>attached</j.0:hasValue>
</rdf:Description>
</rdf:RDF>
```

Text & Model-Synchronized Document Engineering Platform

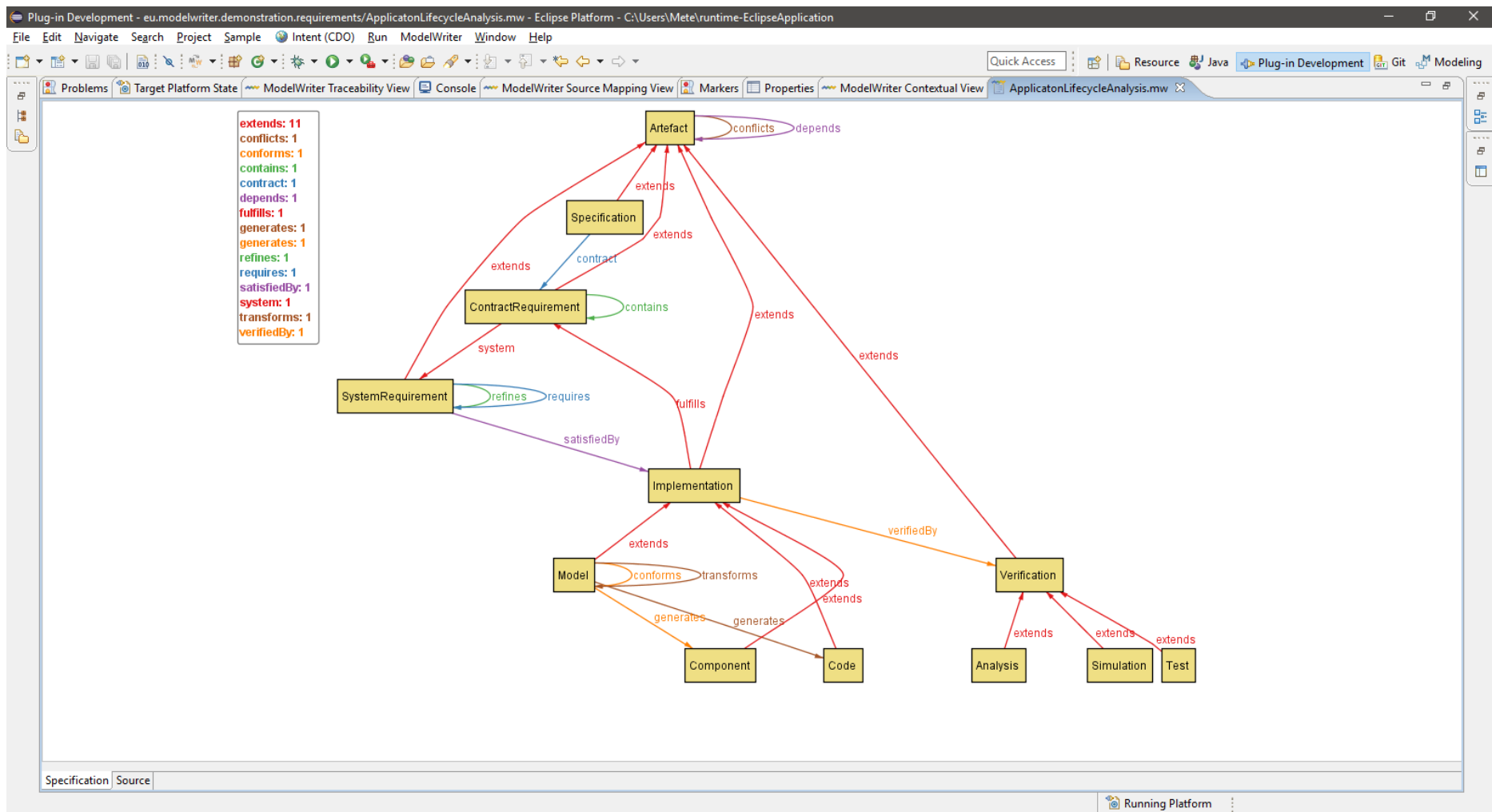




**Synchronization is maintained!**

# 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

10 ## Customer Requirements Specification

11

12 ## UC-1 Create a new SpecObject

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

14

15 ### Precondition

16 Req10 model exists and is open.

17

18 ### Main Success Scenario

19

20 1. We assume that a Specification exists and is open (not required for alternative scenario)

21

22 2. Open a row's context menu (or in the empty editor space)

23

24 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

# Havelsan Use Case