

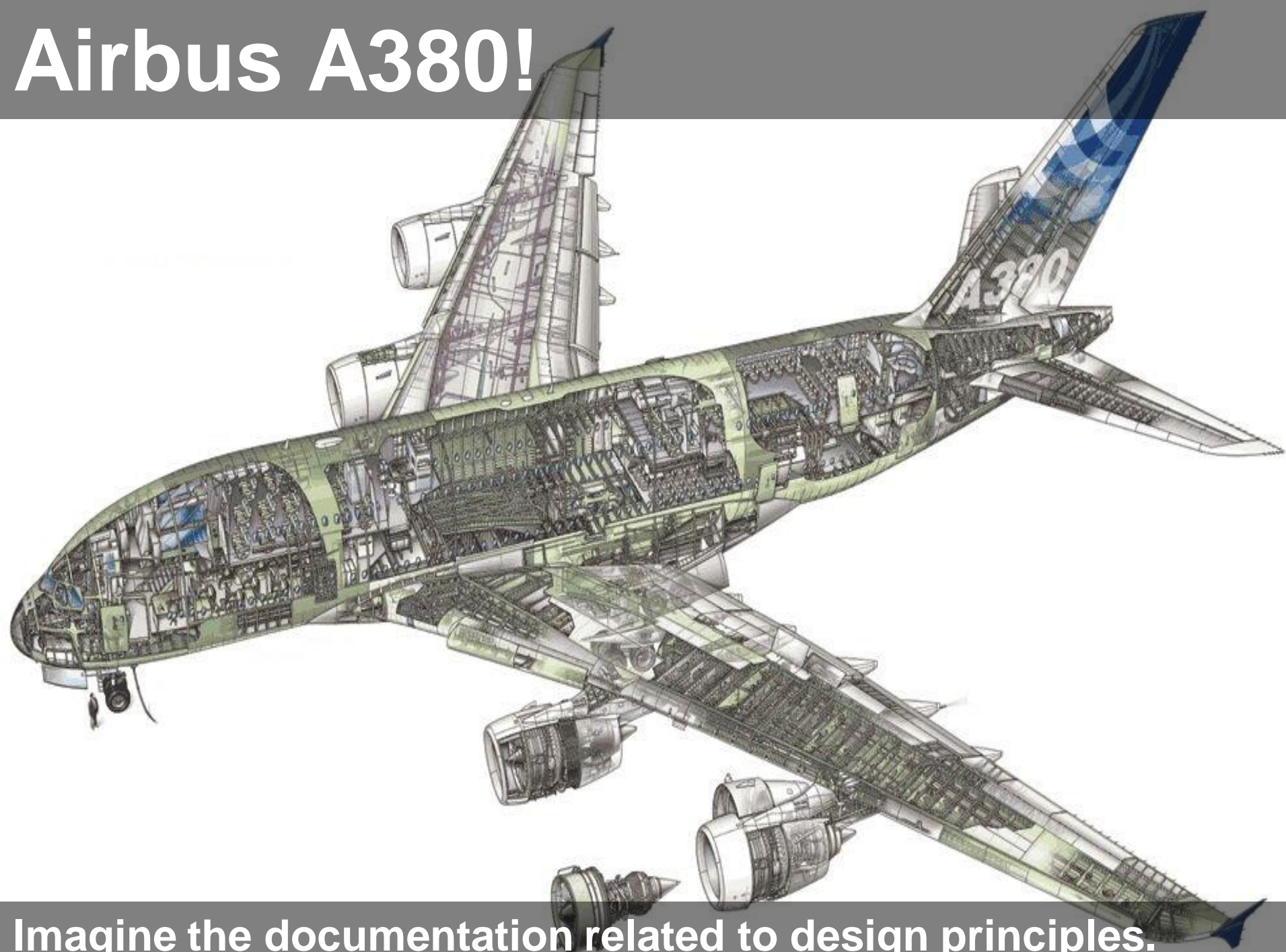
5 The Project Idea with Demonstrations

Ferhat Erata (UNIT, ModelWriter Project Leader)

Dr. Mariem Mahfoudh (CNRS/LORIA)

What is the problem?

Airbus A380!

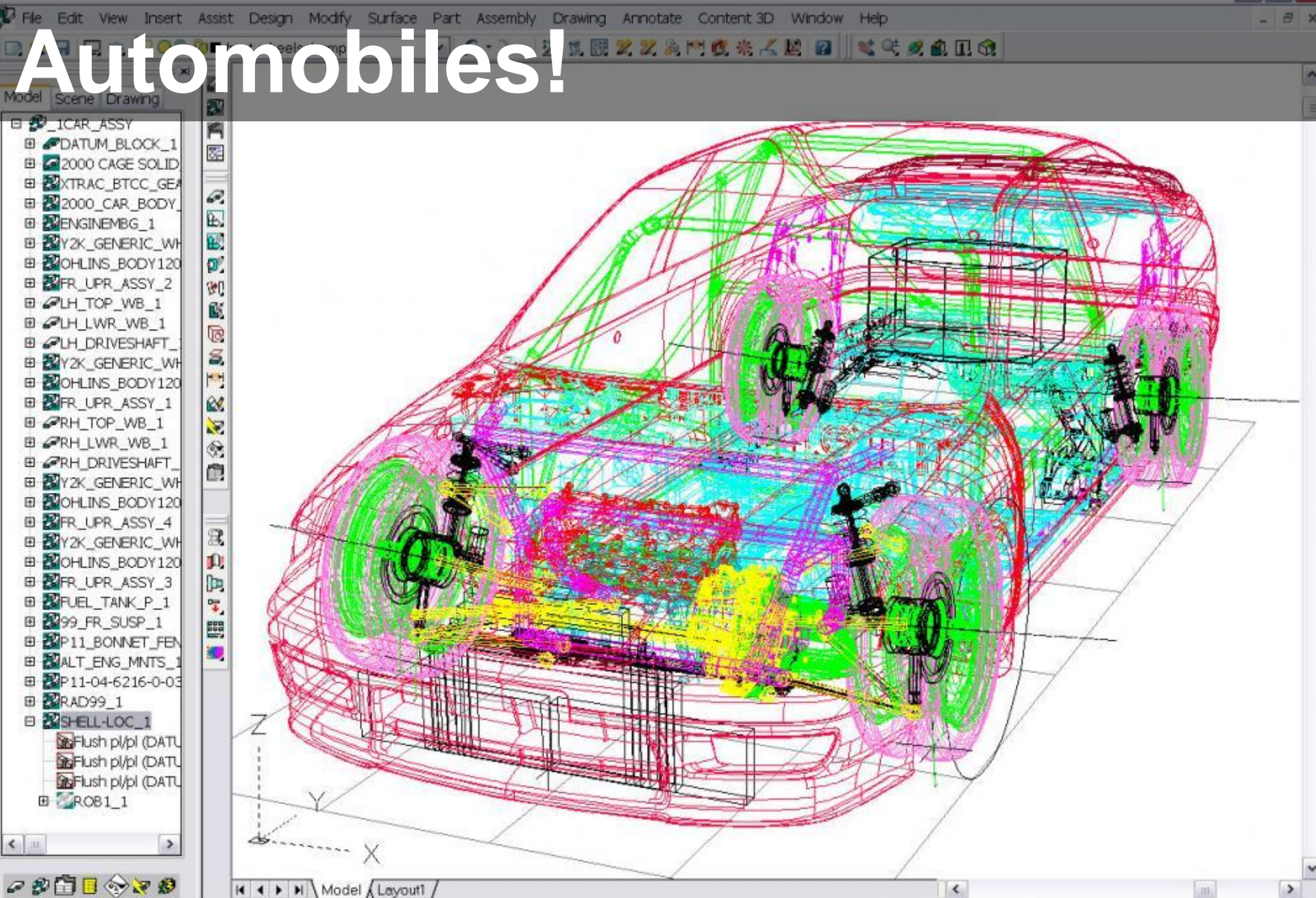


Imagine the documentation related to design principles.

Plants!



Imagine the documentation of a construction site.



Imagine the technical specifications

Documentation!



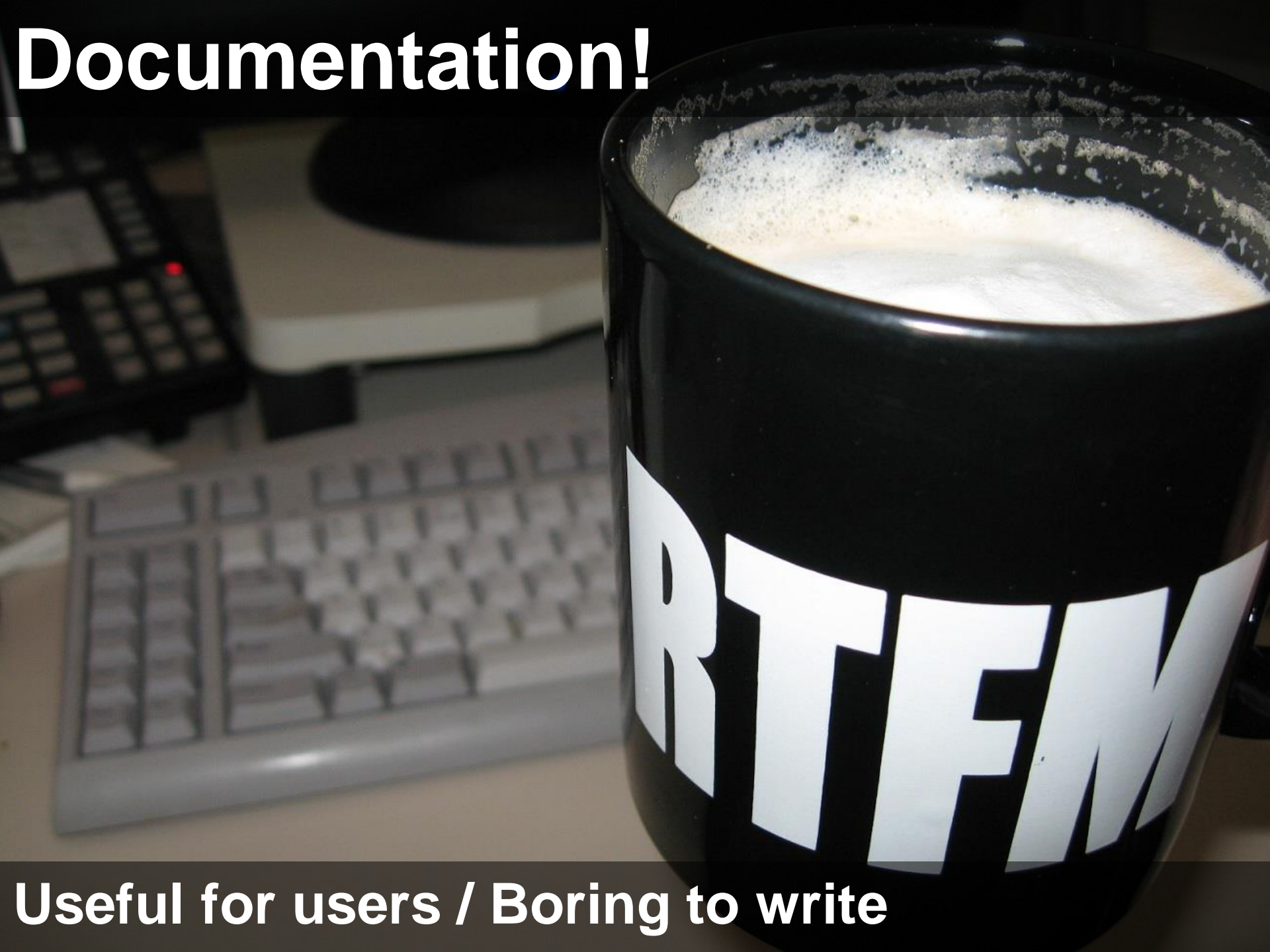
Write once ... and never look at after

Documentation!



Hard to keep it up-to-date

Documentation!



Useful for users / Boring to write

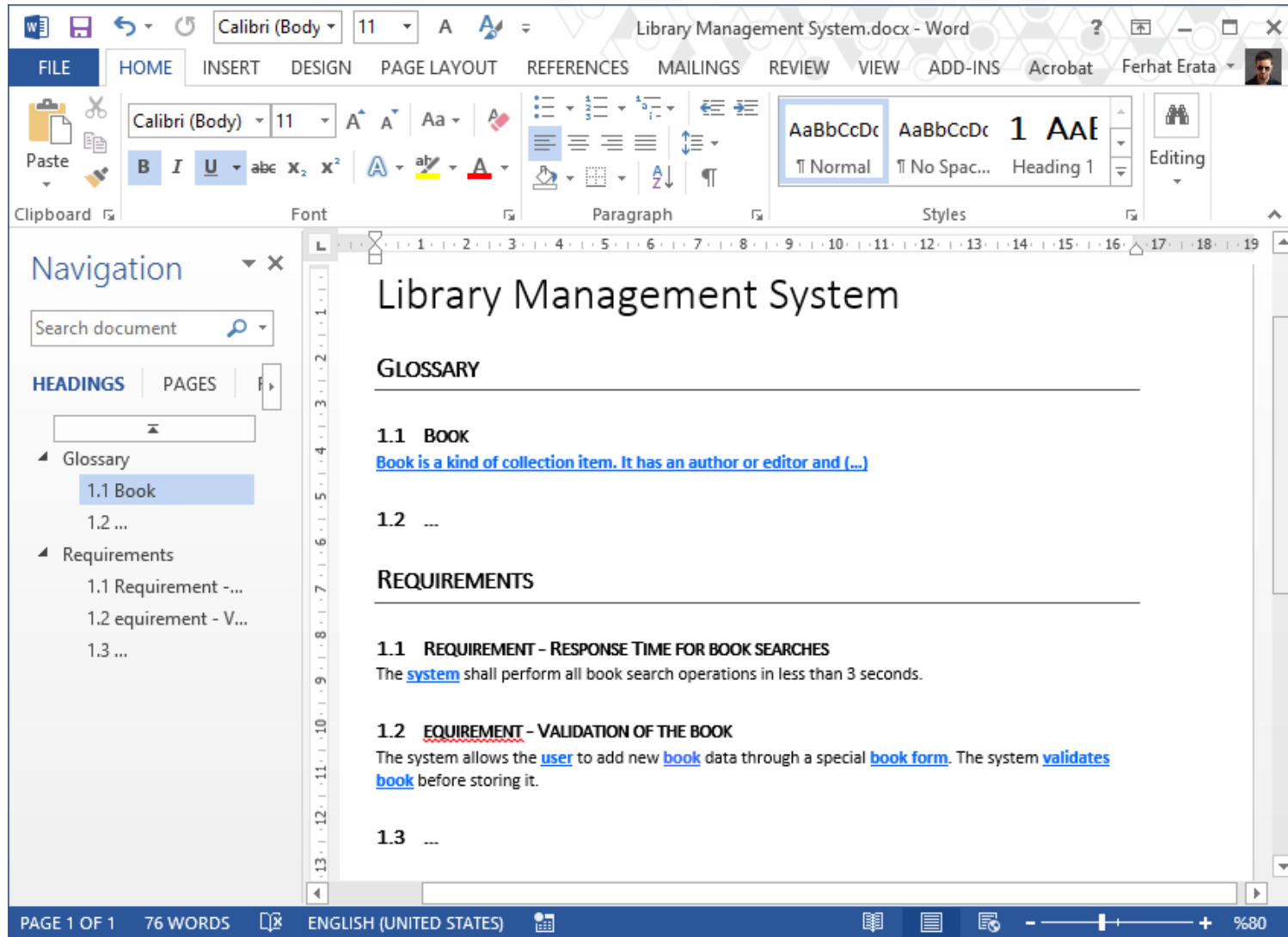
Semantics!

No meaning of syntax (words, sentences ...)

What is a text?

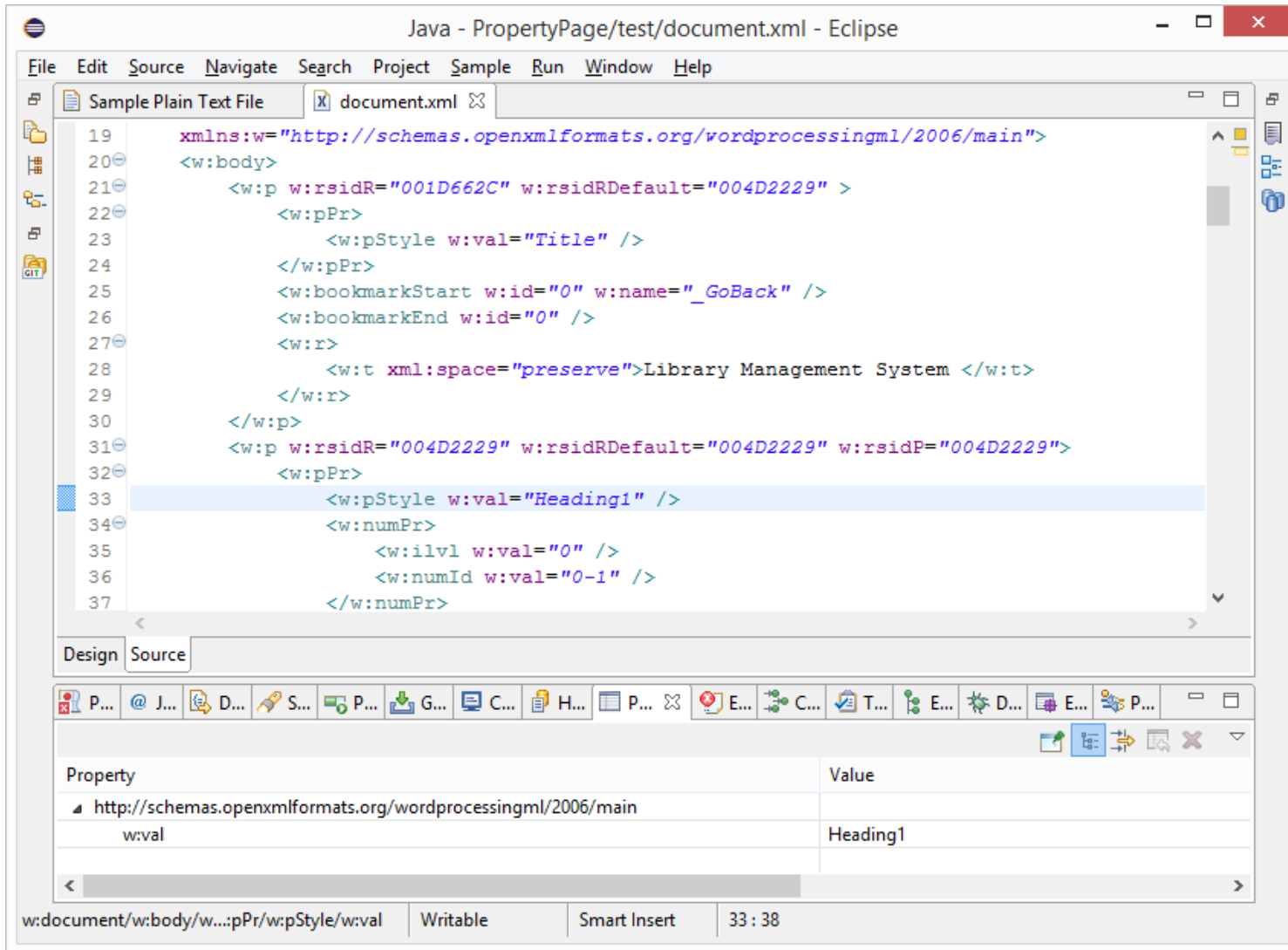
What is a text? (document file formats)

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>

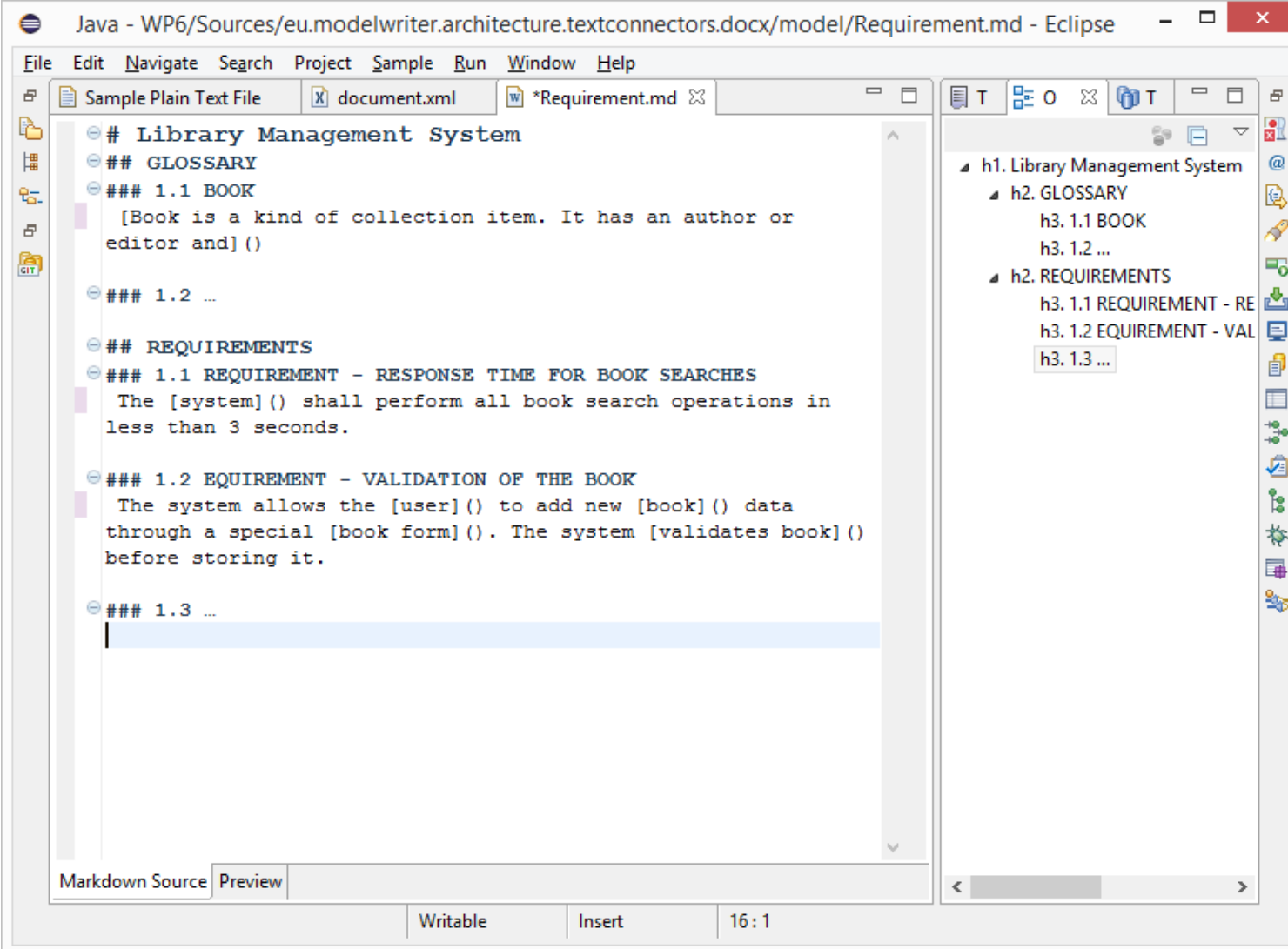
```

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` with the following content:

```
# 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 ...
```

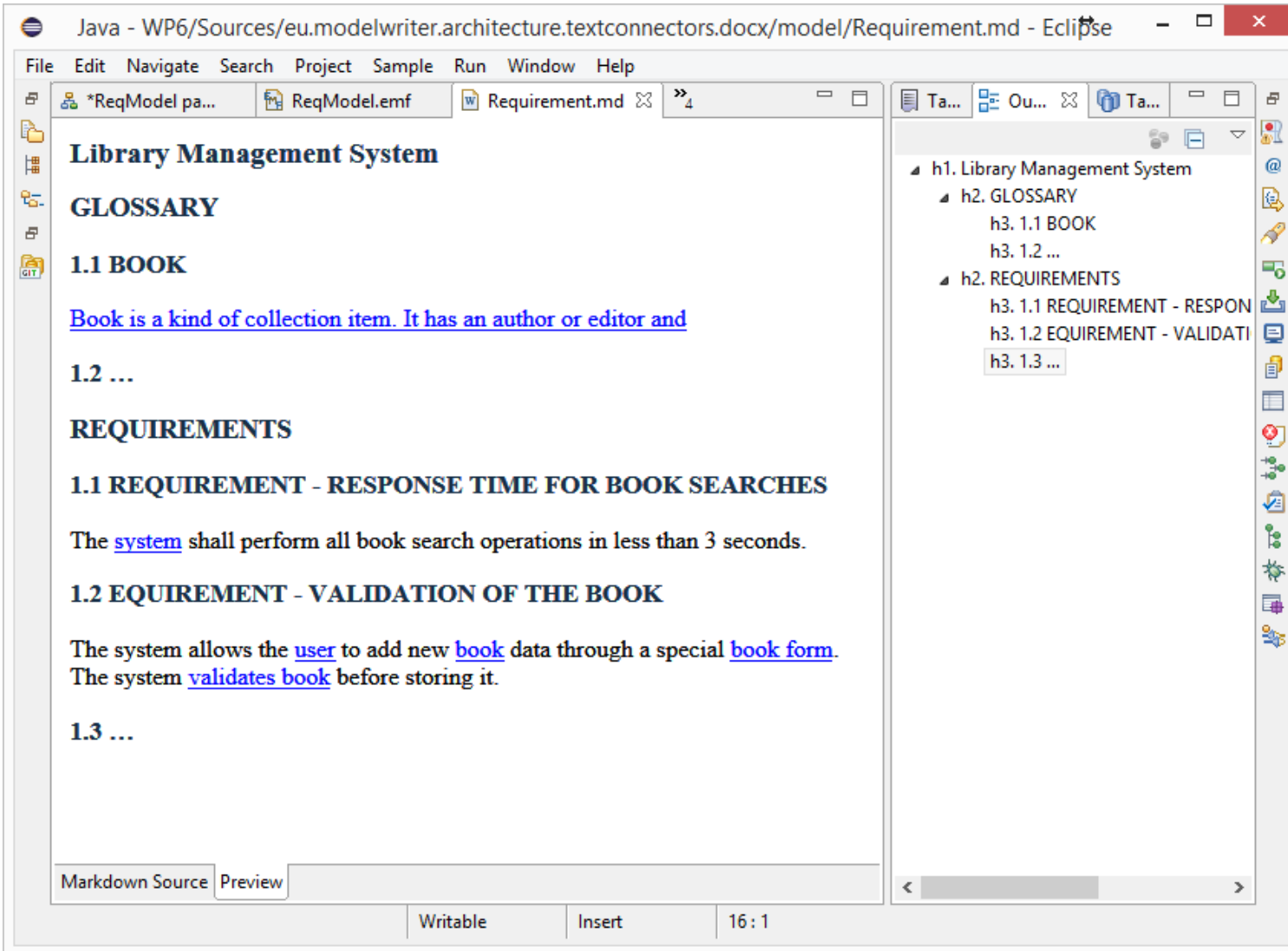
On the right side, there is a preview view showing the rendered markdown structure:

- 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', and a status bar showing 'Writable', 'Insert', and '16:1'.

What is a text? (HTML Preview)

text/markdown (ICANN Standard)



Java - WP6/Sources/eu.modelwriter.architecture.textconnectors.docx/model/Requirement.md - Eclipse

File Edit Navigate Search Project Sample Run Window Help

*ReqModel pa... ReqModel.emf Requirement.md »4

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 ...

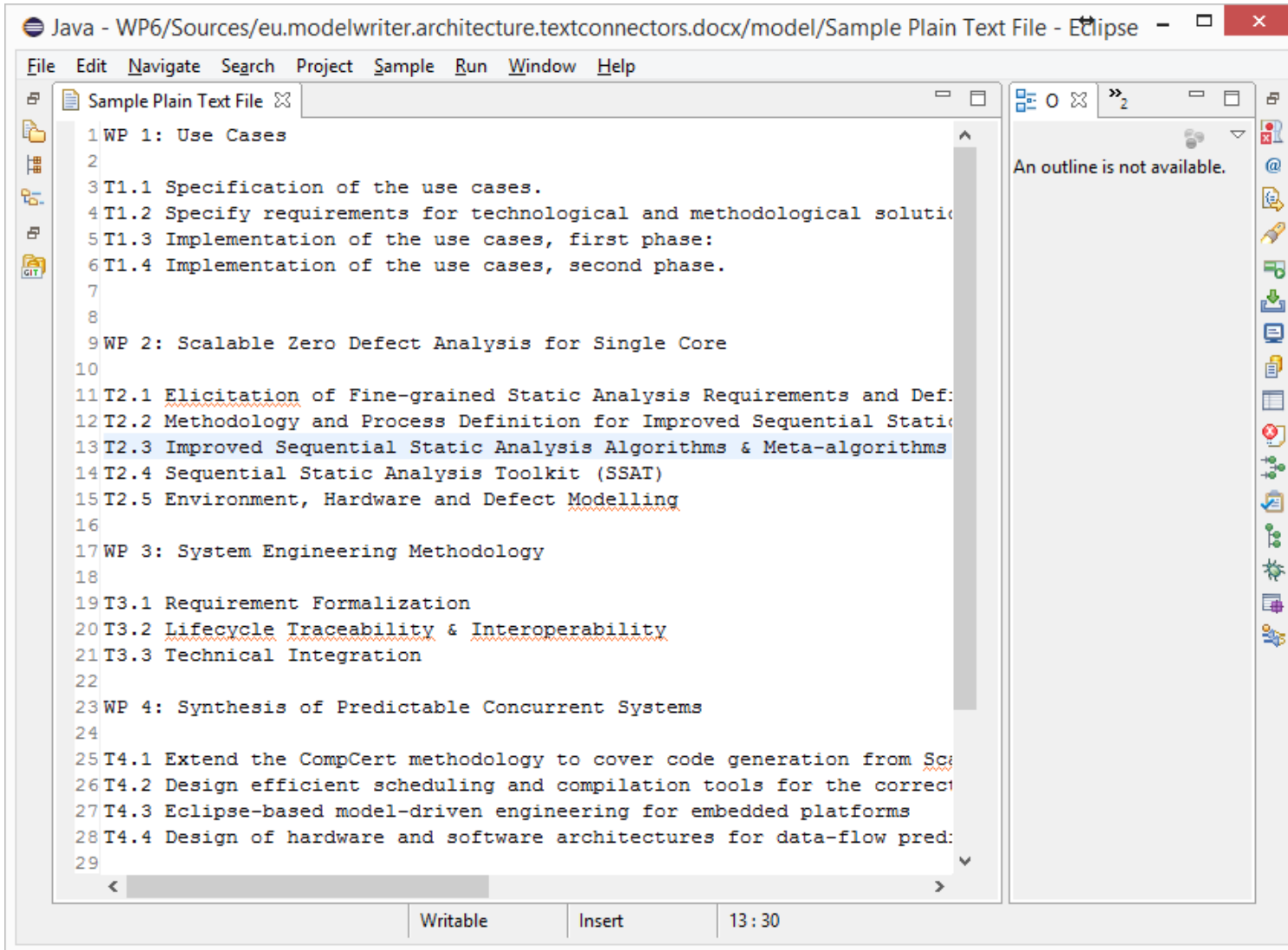
- h1. Library Management System
 - h2. GLOSSARY
 - h3. 1.1 BOOK
 - h3. 1.2 ...
 - h2. REQUIREMENTS
 - h3. 1.1 REQUIREMENT - RESPON
 - h3. 1.2 EQUIREMENT - VALIDATI
 - h3. 1.3 ...

Markdown Source Preview

Writable Insert 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

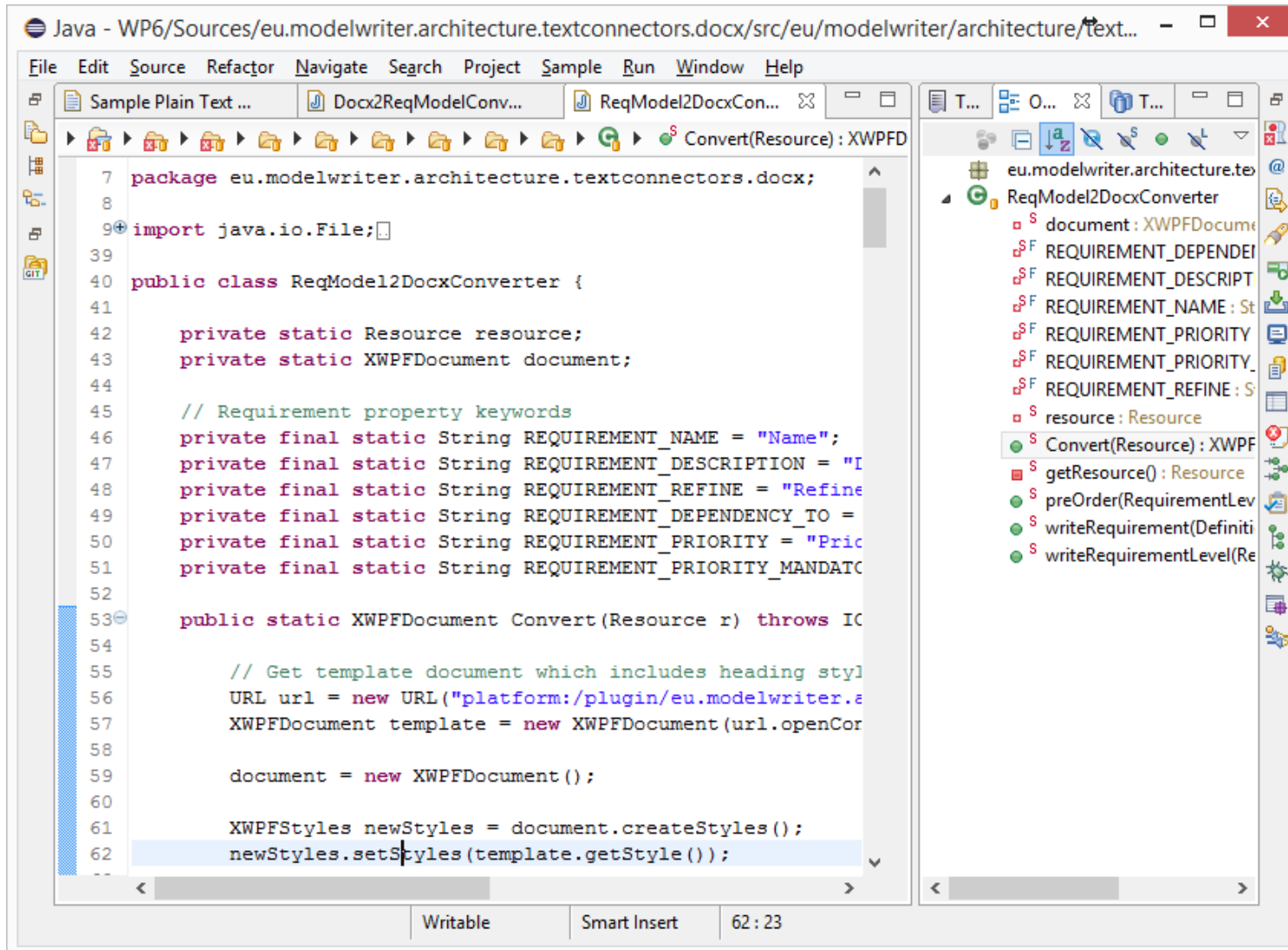
```
1 WP 1: Use Cases
2
3 T1.1 Specification of the use cases.
4 T1.2 Specify requirements for technological and methodological solutions
5 T1.3 Implementation of the use cases, first phase:
6 T1.4 Implementation of the use cases, second phase.
7
8
9 WP 2: Scalable Zero Defect Analysis for Single Core
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/ReqModel2DocxConverter.template.docx");
57         XWPFDocument template = new XWPFDocument(url.openConnection().getInputStream());
58
59         document = new XWPFDocument();
60
61         XWPFDStyles newStyles = document.createStyles();
62         newStyles.setStyles(template.getStyle());
63     }
64 }
```

The right-hand side of the IDE 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

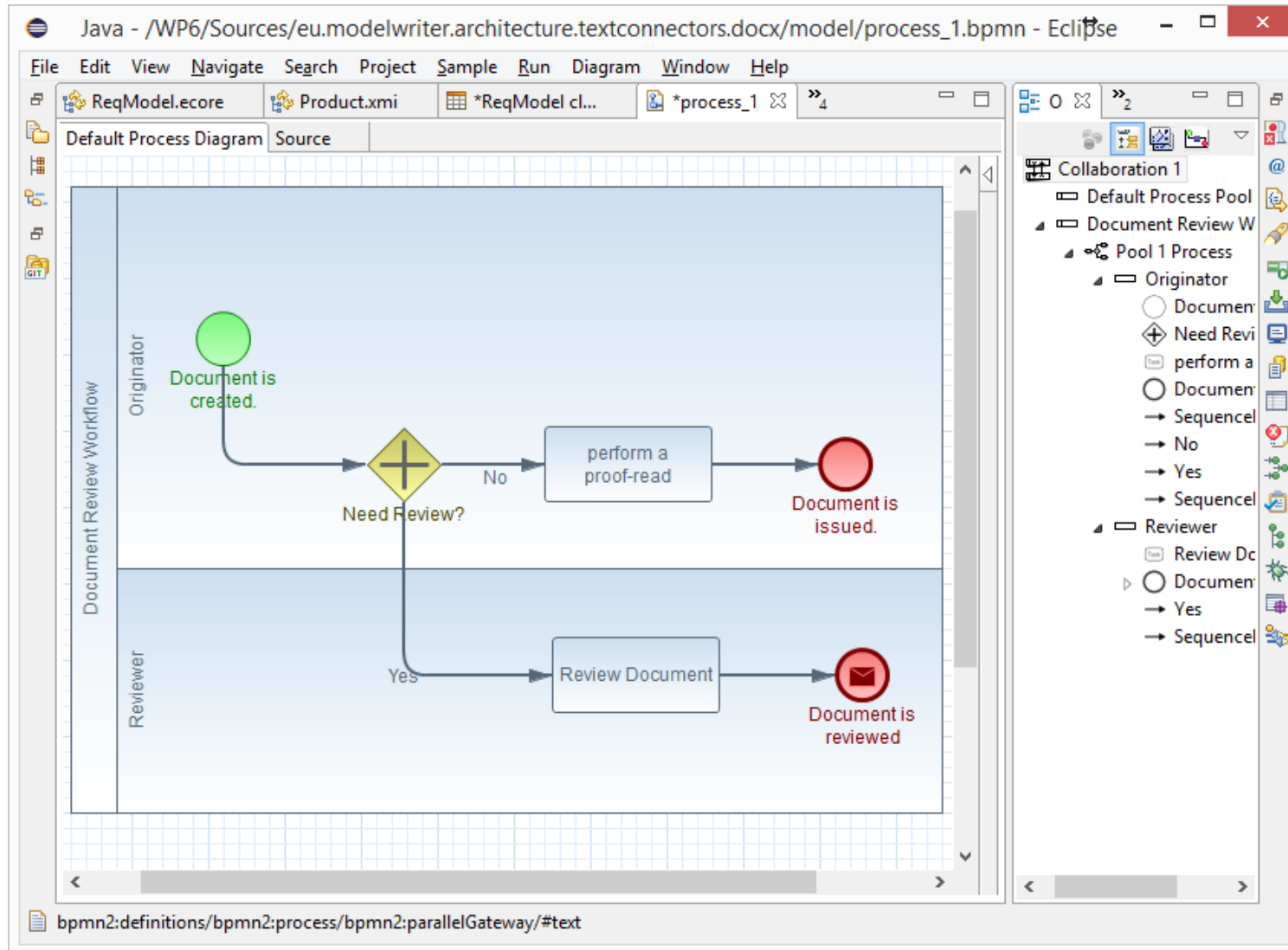
Properties

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

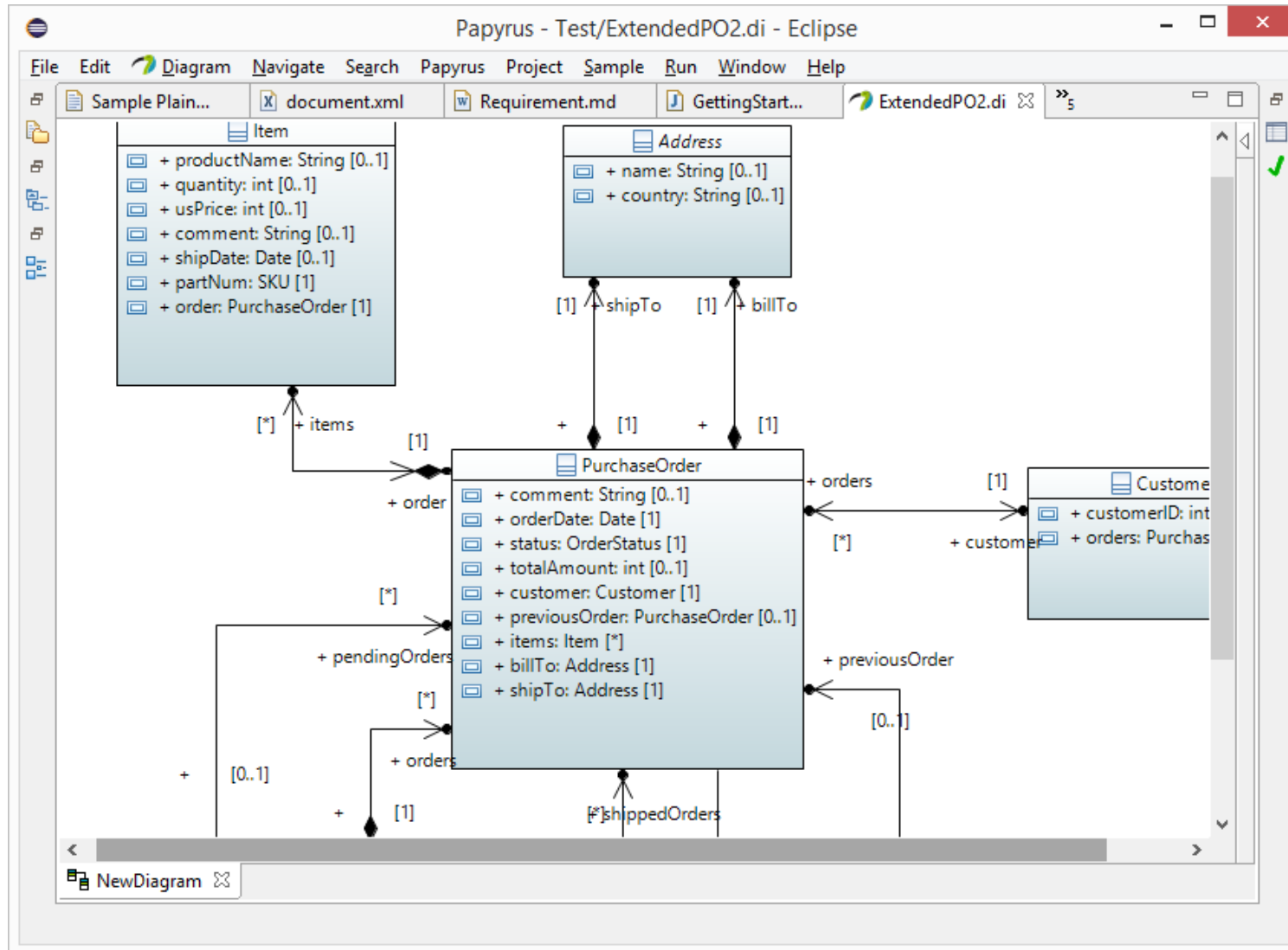
Everything is a model! (BPMN Standard)

Business Process Model & Notation



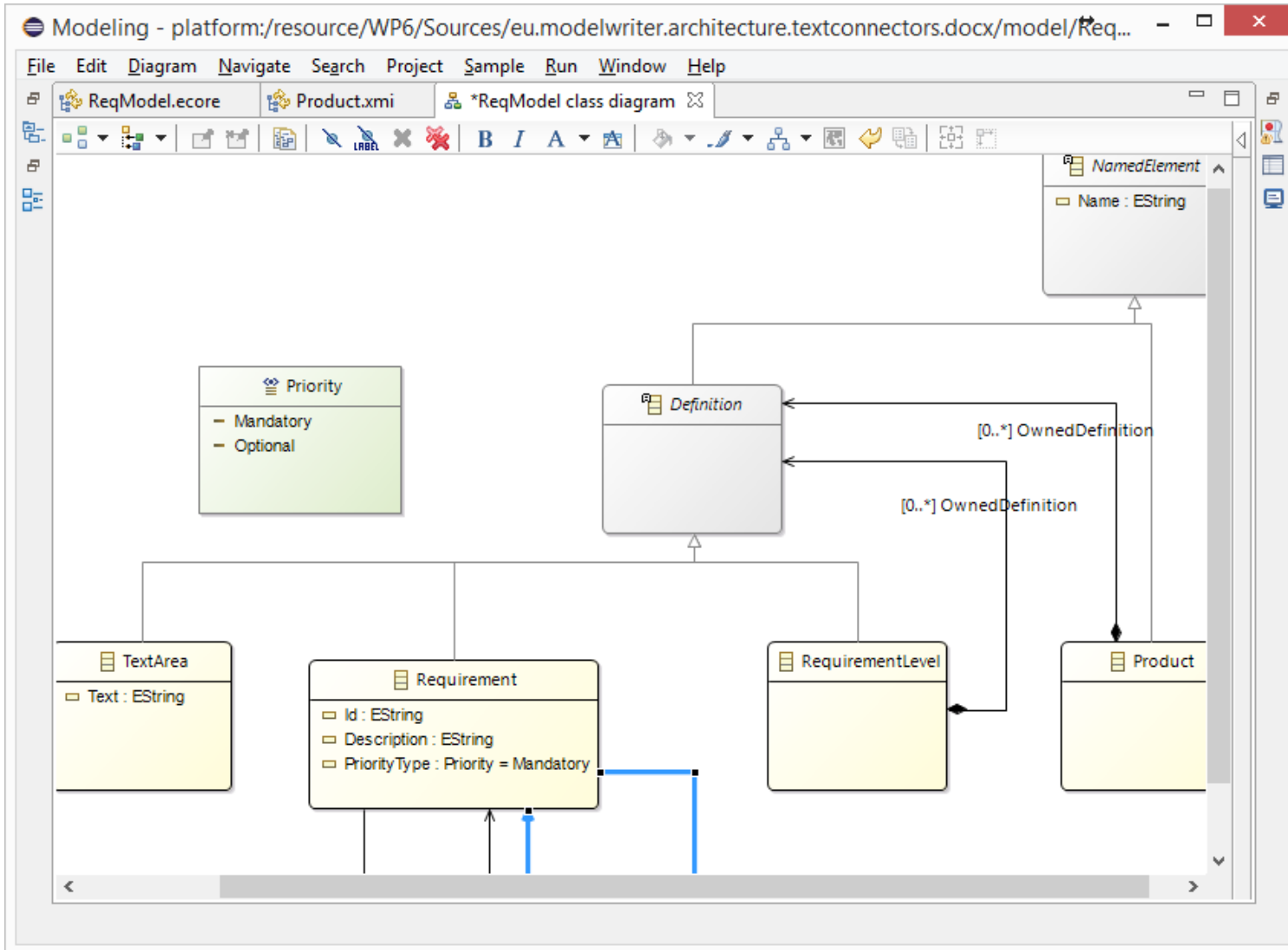
Everything is a model! (UML Standard)

UML Modeling Languages



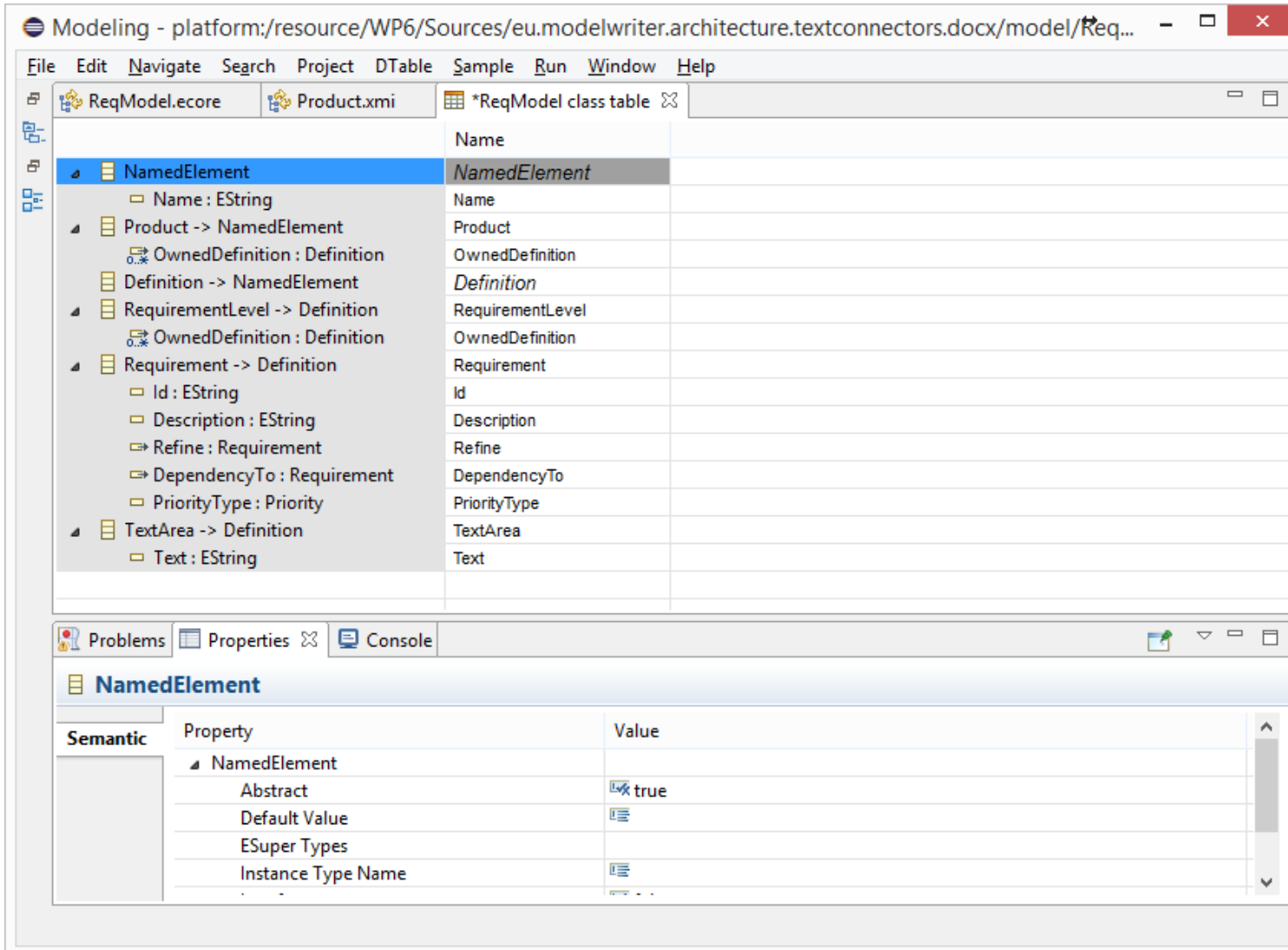
Everything is a model!

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:

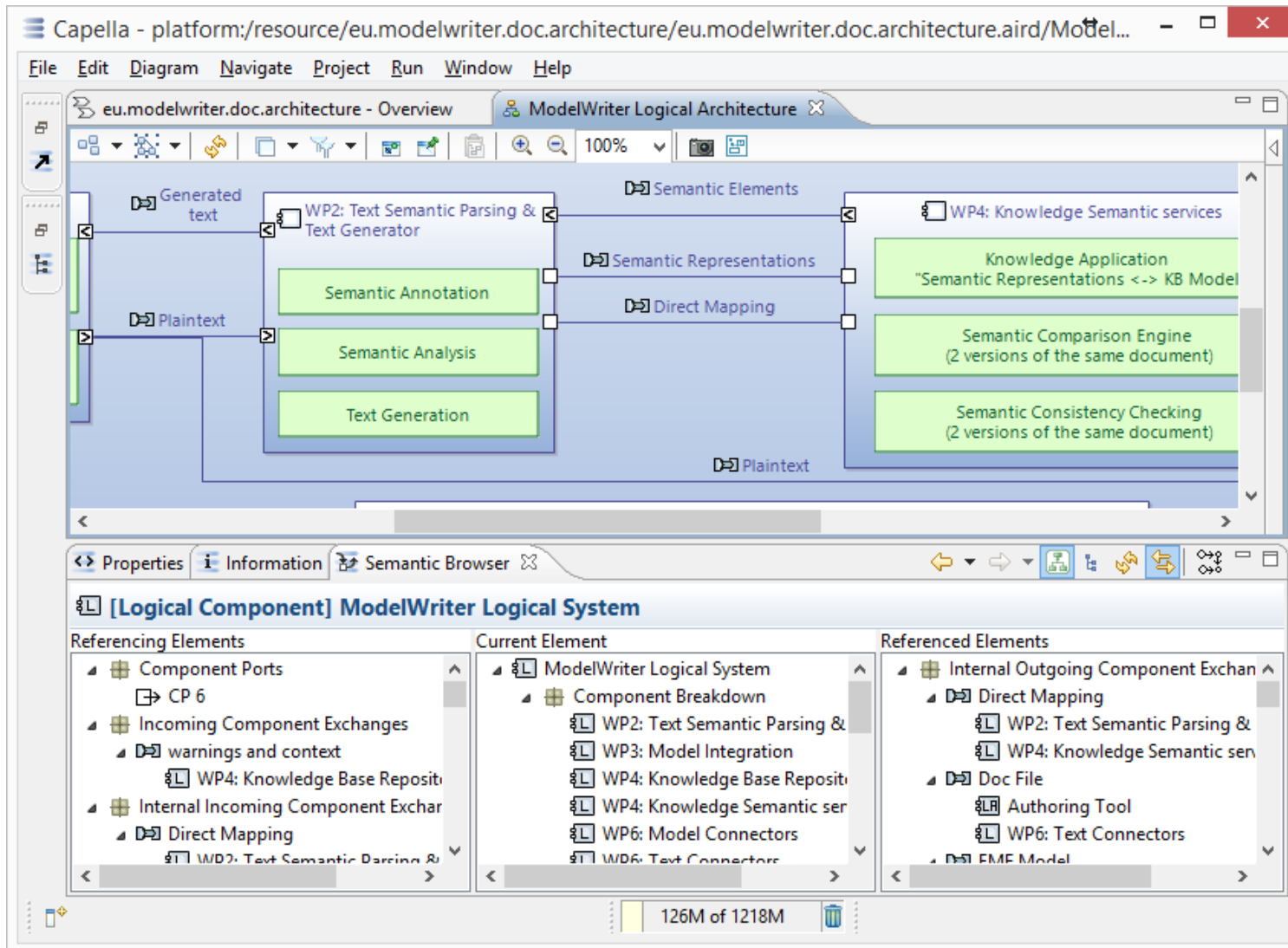
Name
NamedElement
Name
Product
OwnedDefinition
Definition
RequirementLevel
OwnedDefinition
Requirement
Id
Description
Refine
DependencyTo
PriorityType
TextArea
Text

The bottom pane shows the Properties view for the selected element, NamedElement. It displays the following properties:

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

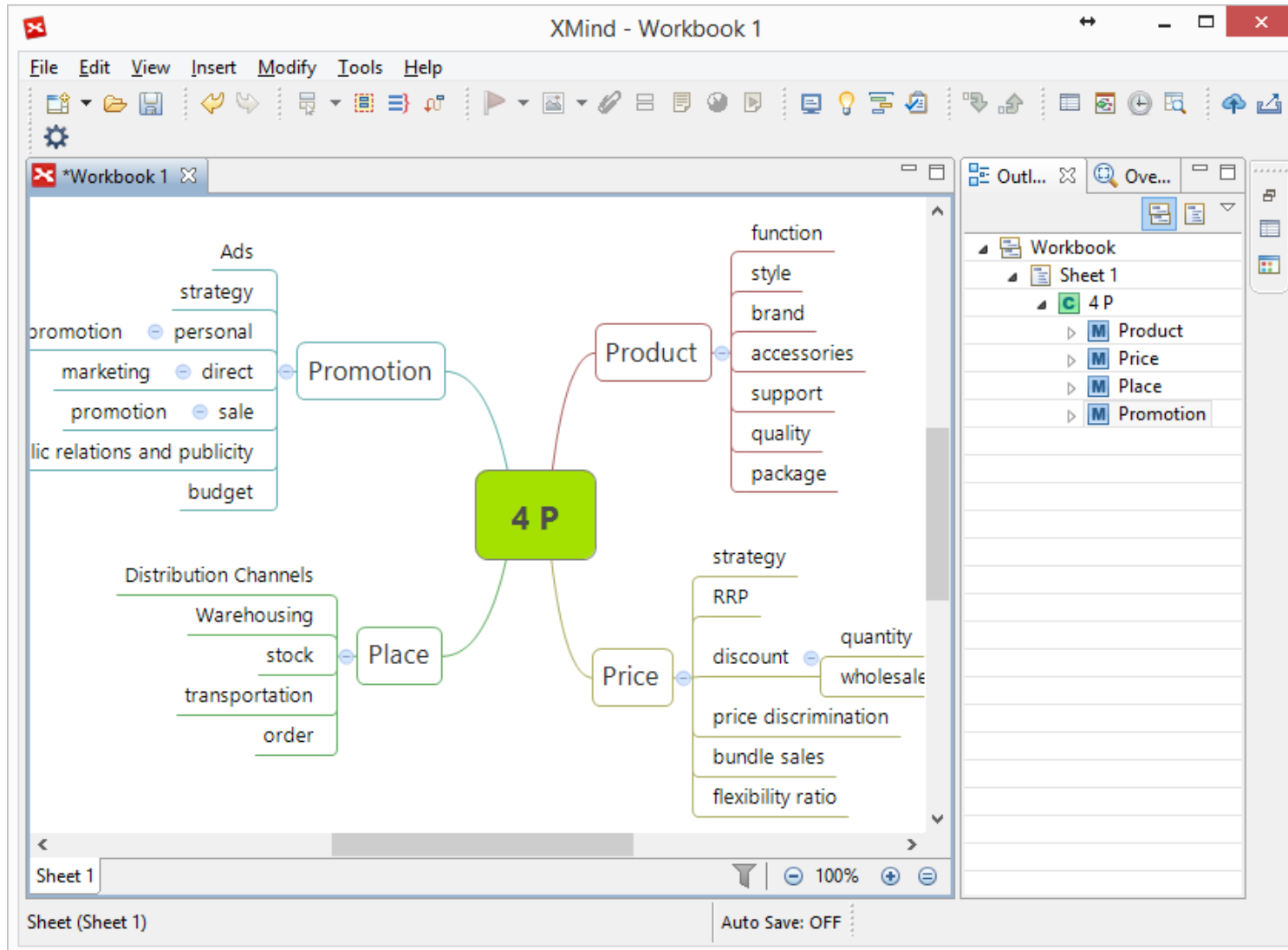
Everything is a model!

Software/System Architecture Design



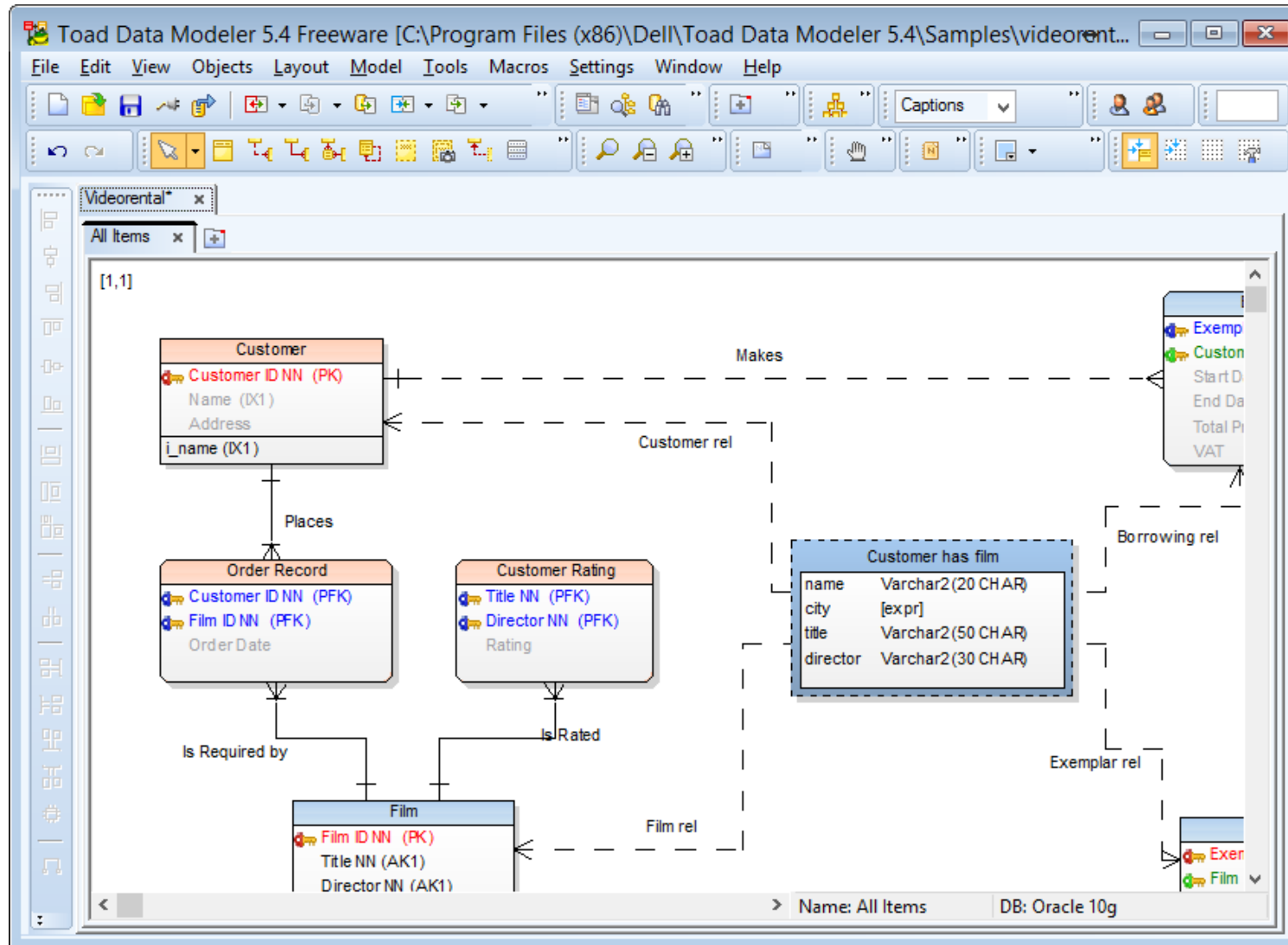
Everything is a model!

Topic Maps, Mind Maps, Vocabularies ...



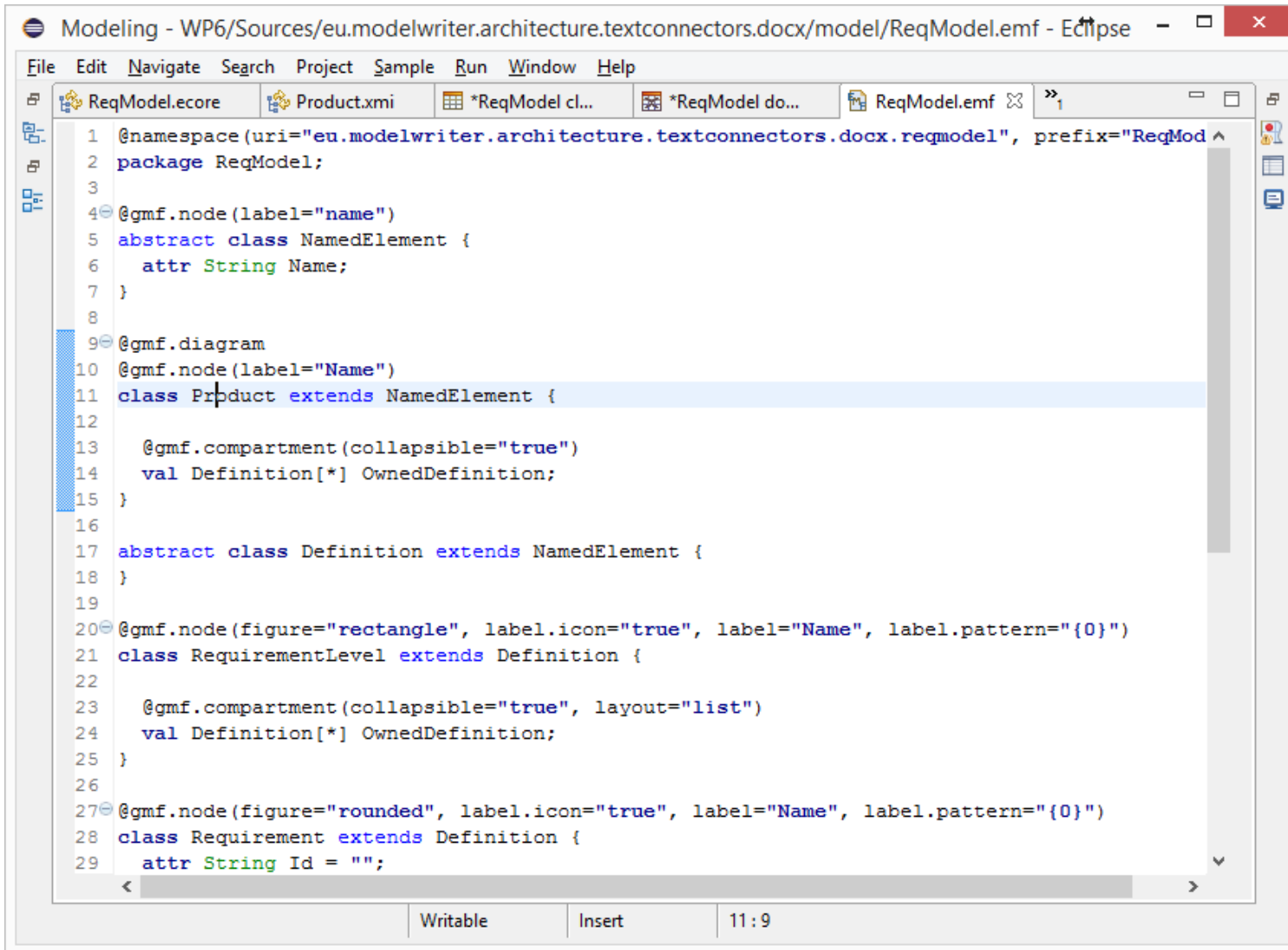
Everything is a model!

Databases (ER, IDEF1.x)



Everything is a model! (Textual Lang.)

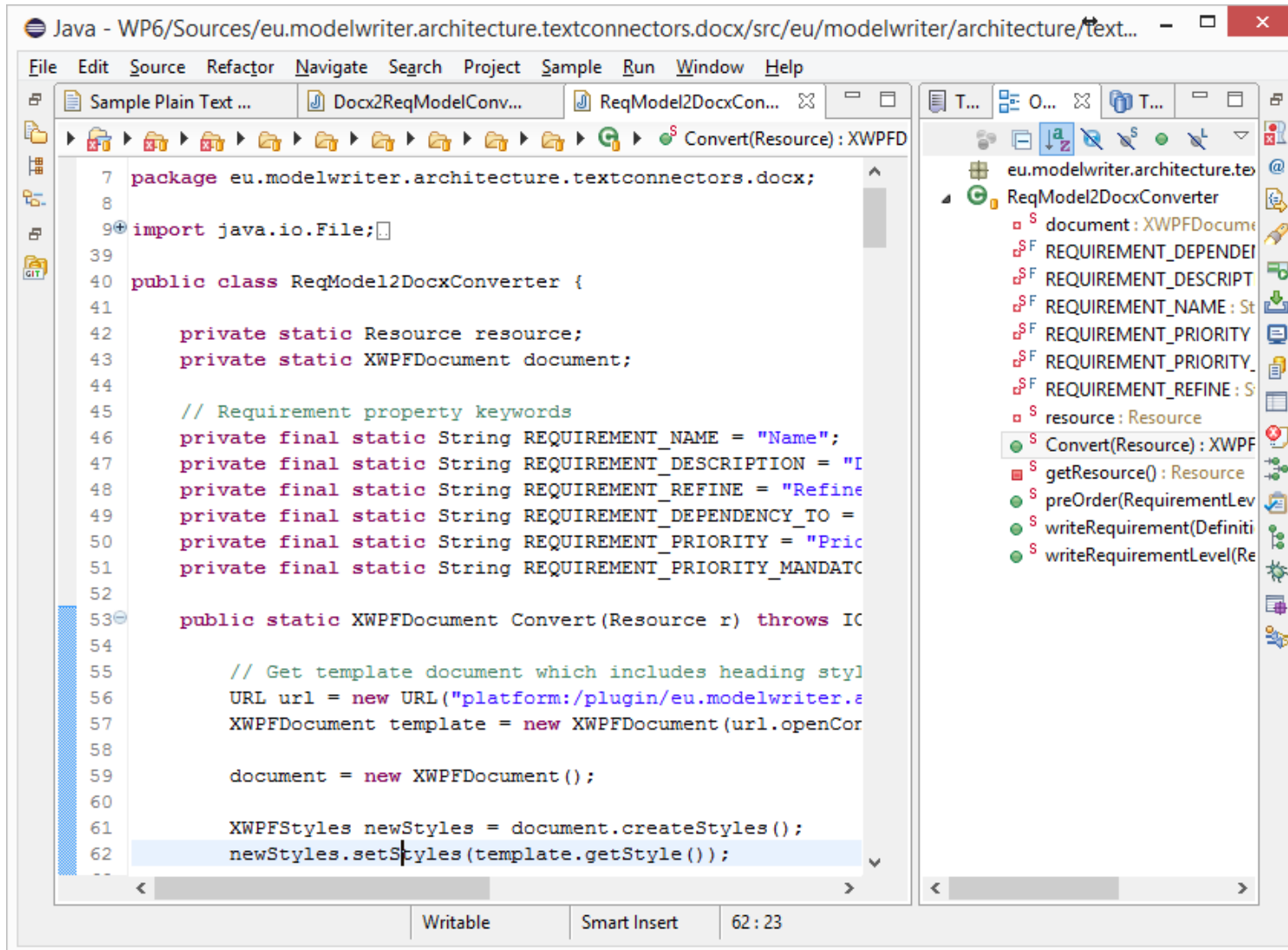
Domain Specific Languages



```
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 right-hand side of the IDE shows the "Project Explorer" with the project structure, including the `eu.modelwriter.architecture.textconnectors.docx` package and the `ReqModel2DocxConverter` class. The "Outline" view on the far right lists the methods and fields within the class.

```
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 {

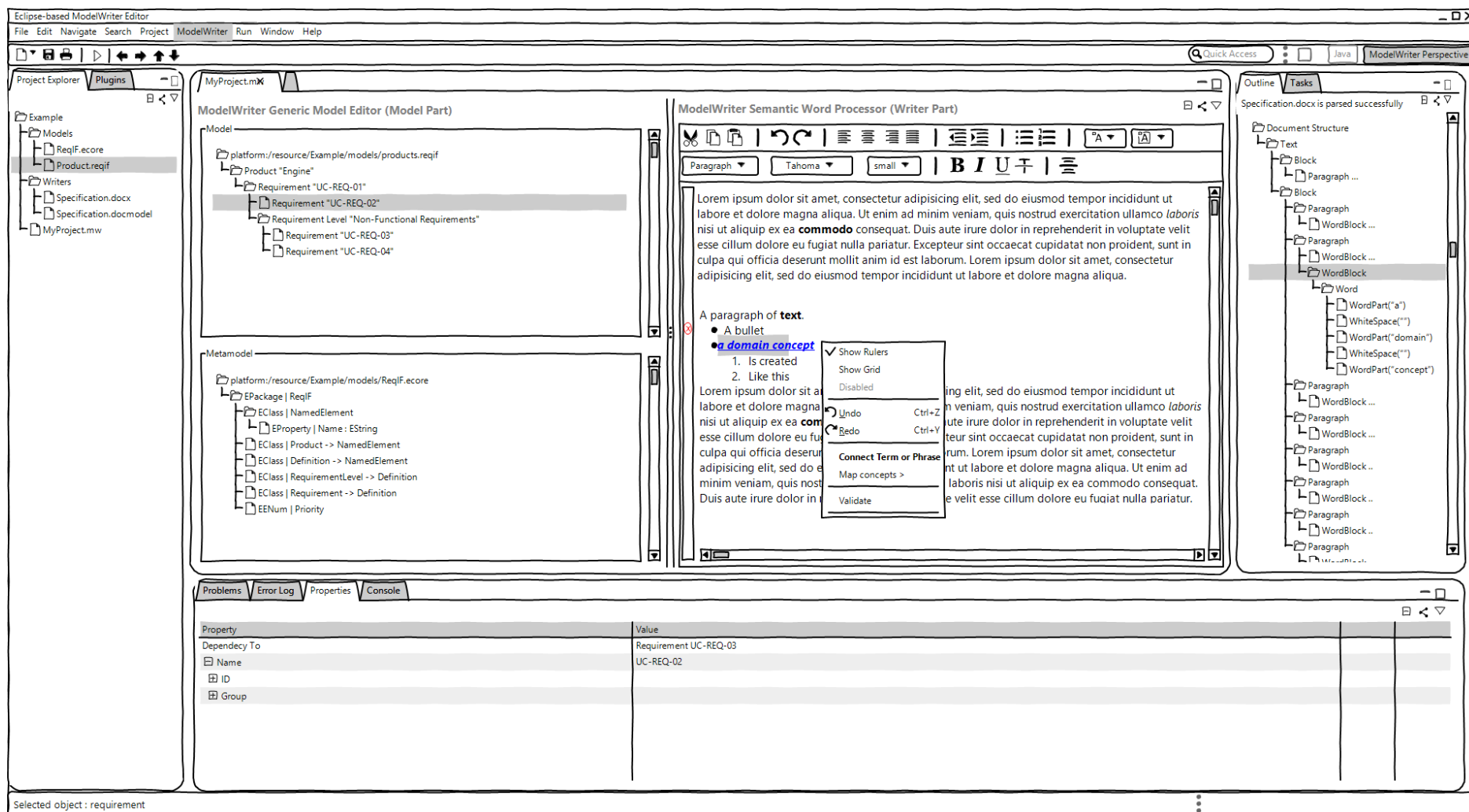
        // Get template document which includes heading styles
        URL url = new URL("platform:/plugin/eu.modelwriter.architecture.textconnectors.docx/ReqModel2DocxConverterTemplate.xwpx");
        XWPFDocument template = new XWPFDocument(url.openStream());

        document = new XWPFDocument();

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


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

ModelWriter – The Solution



Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Capture

The screenshot displays the Eclipse Platform interface for plug-in development. The main window shows the 'File.md' document in the 'ModelWriterTest/docs' directory. The document content is as follows:

```
# The largest heading

3. **SHOULD** This word, or the adjective "RECOMMENDED", mean that there
may exist valid reasons in particular circumstances to ignore a
particular item, but the full implications must be understood and
carefully weighed before a different course.

## The second largest heading

*This text will be italic*

### The 3th largest heading

1. **MUST** User This word, or the terms "REQUIRED" or " SHALL", mean that
the definition is an absolute requirement of the specification.

2. **MUST NOT** This phrase, or the phrase "SHALL NOT", mean that the
definition is an absolute prohibition of the specification.

.... is dependent on ...

# The largest heading

4. **SHOULD NOT** This phrase, or the phrase "NOT RECOMMENDED" mean that
```

The Package Explorer on the left shows the project structure, including the 'extlibary.ecore' project and its sub-projects. The Project Explorer on the right shows the 'File.md' document and its sub-projects. The Markers view at the bottom left shows 0 errors, 1 warning, and 8 others. The ModelWriter Source Mapping View at the bottom center shows a table of mappings:

Description	Resource	Path
EMF Problem (3 items)		
ModelWriter Marker (6 items)		
Bicycle	Bicycle.java	/Model
REQUIRED	File.md	/Model
SHOULD	File.md	/Model
Book	extlibary.ecore	/Model
books	extlibary.ecore	/Model
employees	extlibary.ecore	/Model

The ModelWriter Master View at the bottom right shows a tree structure of the model elements, including 'employees', 'Book', and 'books'. The ModelWriter Target Platform view at the bottom right shows a table of mappings:

ID	Text

Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Capture

The screenshot displays the Eclipse IDE interface for a project named 'ModelWriterTest'. The main editor shows the 'Bicycle.java' file, which contains the following code:

```
1 package ModelWriter;
2
3 public class Bicycle {
4
5     // the Bicycle class has
6     // three fields
7     public int cadence;
8     public int gear;
9     public int speed;
10
11     // the Bicycle class has
12     // one constructor
13     public Bicycle(int startCadence, int startSpeed, int startGear) {
14         gear = startGear;
15         cadence = startCadence;
16         speed = startSpeed;
17     }
18
19     // the Bicycle class has
20     // four methods
21     public void setCadence(int newValue) {
22         cadence = (int)newValue;
23     }
24
25     public void setGear(int newValue) {
26         gear = newValue;
27     }
28 }
```

The Package Explorer on the left shows the project structure, including the 'src' directory with 'Bicycle.java' and 'MountainBike.java'. The Outline on the right shows the 'Bicycle' class with its attributes and methods. The Task List on the right shows the 'Bicycle' class with its attributes and methods. The Source Mapping View at the bottom shows the mapping between the code and the model elements.

Description	Resource	Path
Bicycle	Bicycle.java	/Mc
cadence	Bicycle.java	/Mc
gear	Bicycle.java	/Mc
speed	Bicycle.java	/Mc
REQUIRED	File.md	/Mc
SHOULD	File.md	/Mc
Book	extlibrary.ecore	/Mc
books	extlibrary.ecore	/Mc
employees	extlibrary.ecore	/Mc

Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Capture

The screenshot displays the Eclipse IDE environment for developing a plug-in. The main editor shows the `extlibrary.ecore` file with XML content defining an EMF package. The Package Explorer on the left shows the project structure, including the `extlibrary` package and its classes. The bottom views provide additional context: the Markers view shows 0 errors and 11 warnings; the ModelWriter Source Mapping View shows a table of source mappings; the ModelWriter Master View shows a tree of classes; and the ModelWriter Target View shows a table of target mappings.

extlibrary.ecore

```
<?xml version="1.0" encoding="UTF-8"?>
<ecore:EPackage xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI" xmlns:xs="http://www.eclipse.org/emf/2002/Ecore" name="extlibrary" nsU
nsPrefix="extlib">
  <eClassifiers xsi:type="ecore:EClass" name="Book" eSuperTypes="#//Circulati
  <eStructuralFeatures xsi:type="ecore:EAttribute" name="title" eType="ecor
  <eStructuralFeatures xsi:type="ecore:EAttribute" name="pages" eType="ecor
    defaultLiteral="100"/>
  <eStructuralFeatures xsi:type="ecore:EAttribute" name="category" eType="#
    unsettable="true"/>
  <eStructuralFeatures xsi:type="ecore:EReference" name="author" lowerBound
    eType="#//Writer" eOpposite="#//Writer/books"/>
</eClassifiers>
  <eClassifiers xsi:type="ecore:EClass" name="Library" eSuperTypes="#//Addres
  <eStructuralFeatures xsi:type="ecore:EAttribute" name="name" eType="ecore
  <eStructuralFeatures xsi:type="ecore:EReference" name="writers" upperBound
    eType="#//Writer" volatile="true" transient="true" derived="true" con
    resolveProxies="false"/>
  <eAnnotations source="http://org.eclipse.org/emf/ecore/util/ExtendedMetaDa
    <details key="group" value="#people"/>
  </eAnnotations>
  <eStructuralFeatures>
  <eStructuralFeatures xsi:type="ecore:EReference" name="employees" upperBo
    eType="#//Employee" volatile="true" transient="true" derived="true" c
    resolveProxies="false"/>
</eStructuralFeatures>
</ecore:EPackage>
```

ModelWriter Source Mapping View

Description	Resource	Pat
EMF Problem (3 items)		
ModelWriter Marker (9 items)		
Bicycle	Bicycle.java	/Mc
cadence	Bicycle.java	/Mc
gear	Bicycle.java	/Mc
speed	Bicycle.java	/Mc
REQUIRED	File.md	/Mc
SHOULD	File.md	/Mc
Book	extlibrary.ecore	/Mc

ModelWriter Master View

- employees
- Book
- books

ModelWriter Target View

ID	Text
----	------

Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Extraction

The screenshot displays the ModelWriter Project application window, 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 used in function zones
ABS2195 -LRB- preferred for weight saving -RRB- or NSA5516J P-Clamp Shall be used
Flexible Hoses Shall be defined with a maximum length of 500 mm regardless of
Rigid Pipes Shall be segregated to fixed Structure by not less than 10 mm as shown
Flexible Hoses Shall be segregated to rigid Component/Item/Object by not less than
Rigid Pipes Shall be segregated to movable Component/Item/Object by not less than
Flexible Hoses Shall be segregated to movable Component/Item/Object by not less than
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>
```

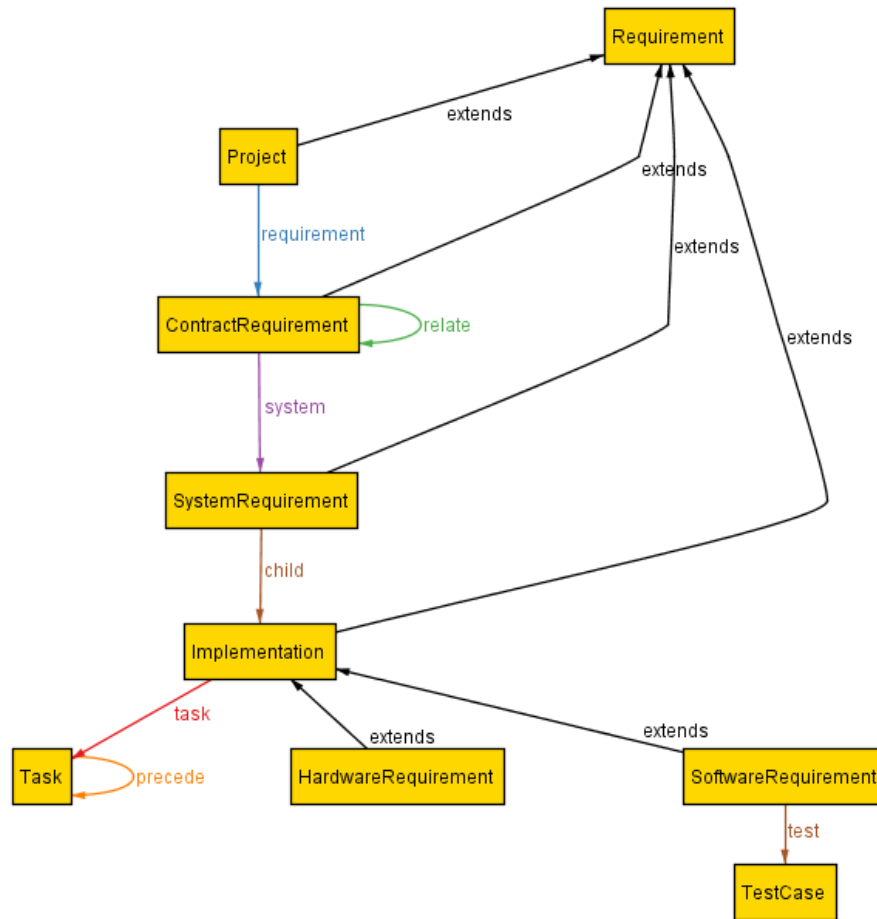
Text & Model-Synchronized Document Engineering Platform



Synchronization is maintained!

Configuration:HAVELSAN example

extends: 6
child: 1
precede: 1
relate: 1
requirement: 1
system: 1
task: 1
test: 1



```
module Haveksan/Requirement

abstract sig Requirement {}

sig Task {
  precede: lone Task,
}{ all t: Task | one t.~task}

one sig Project extends Requirement {
  requirement: some ContractRequirement }

sig ContractRequirement extends Requirement {
  system: set SystemRequirement,
  relate: set ContractRequirement
}{all c: ContractRequirement | one c.~requirement}

--@name: "System Requirement"
sig SystemRequirement extends Requirement {
  child: some Implementation
}{ all s: SystemRequirement | one s.~system}

abstract sig Implementation extends Requirement {
  task: set Task
}{ all i: Implementation | one i.~child}

--@context.editor: "ReqIFEditor"
sig SoftwareRequirement extends Implementation {
  test: some TestCase
}

sig HardwareRequirement extends Implementation {}

sig TestCase { }{ all t:TestCase | one t.~test}

fact noSelfRelation{
  no c: ContractRequirement | c in c.relate
  no t: Task | t in t.precede }

fact noCycles{no t:Task | t in t.^precede}

fact realismConstraint {
  some ContractRequirement
  some HardwareRequirement
  some SoftwareRequirement
  some precede}
```

A Formal Specification Model to configure the ModelWriter

Quantification of the expected benefits

- Improvement in quality and productivity of technical documentation.
- Quality increase of the product with consistent requirements and designs.
 - For instance, according to AIRBUS's claims in their use cases, the global saving would be 5 M€ to 7 M€ (A350 Recurring Cost)
- 50% reduction of costs for keeping the documentation up-to-date with the developed software

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