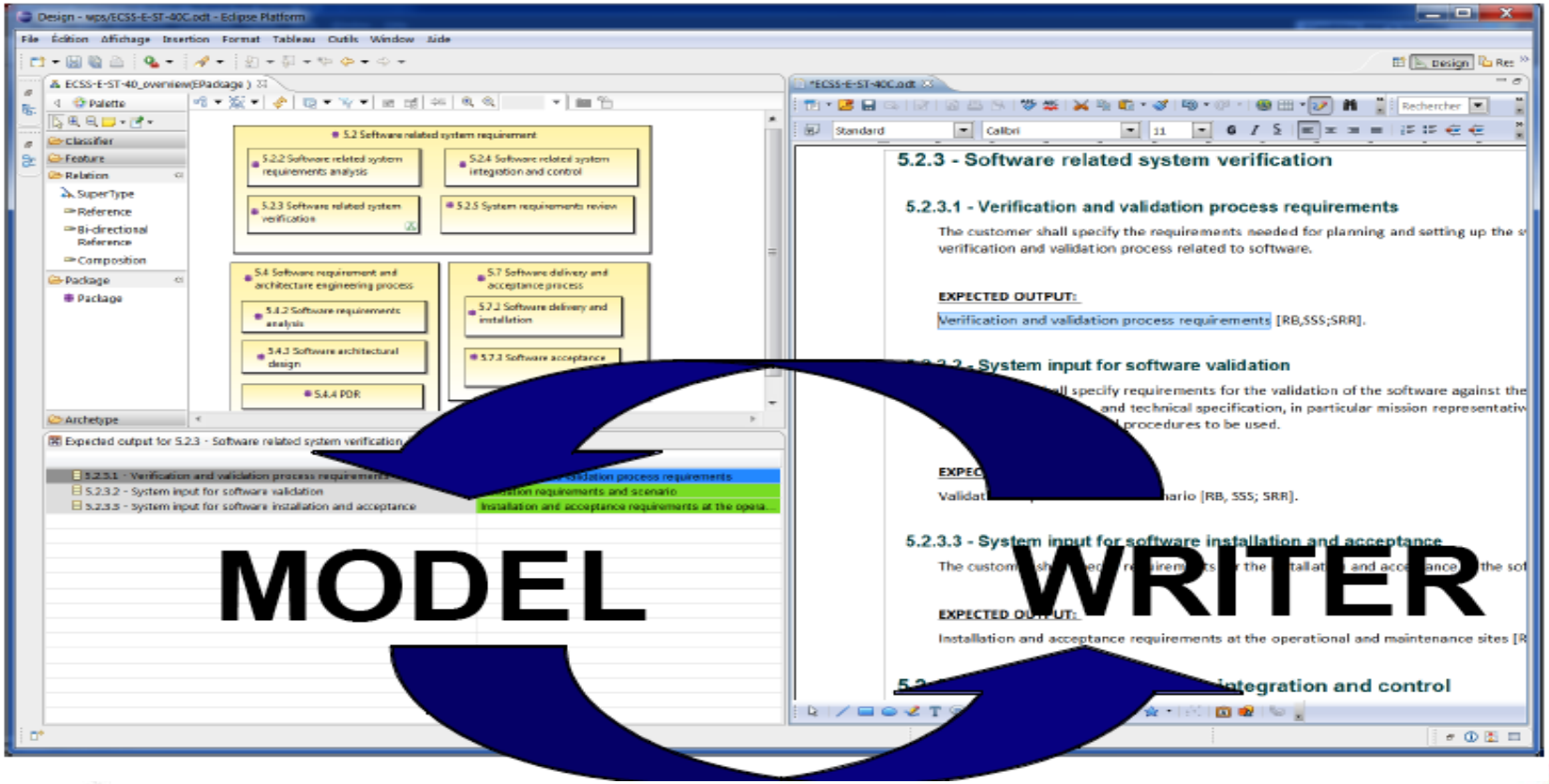
ModelWriter

Text & Model-Synchronized Document Engineering Platform



Project Leader: Ferhat Erata (ferhat@computer.org)
Project Email: project@modelwriter.eu

ITEA 3

1 Introduction

Ferhat Erata
UNIT, ModelWriter Project Leader

Participants
ModelWriter #1 Review

UNIT

- Ferhat Erata
- Dr. Moharram Challenger

cnrs

- Prof. Claire Gardent,
- Prof. Samuel Cruz-Lara
- Dr. Mariem Mahfoudh

AIRBUS GROUP

- Dr. Anne Monceaux

HAVELSAN

- Dr. Eray Tuzun

beo Model Driven Company

- Yvan Lussaud

KoçSistem

- Prof. Geylani Kardas
- Hale Gezgen

Mantis Software Company

- Prof. Erhan Mengüsoğlu

HiSBiM

- Ersan Gürdoğan
- Taskin Kızıl

Agenda [9:00 - 12:30]

1. Introduction (10 min) [09:00 - 09:10]
2. Overview of the project (20 min) [09:10 - 09:30]
3. Use-cases and exploitation prospects (45 min) [09:30 - 10:15]
4. Progress Status per WP (45 min) [10:15 - 11:00]

   - Break (10 mins)
5. Demonstrations (30 min): [11:10 - 11:40]
6. Summary (achievements and exploitations) (5 min) [11:40 - 11:45]
7. Reviewers Private Section (30 min) [11:45 - 12:15]
8. Feedback Session (15min) [12:15 - 12:30]
9. Lunch (30 min) [12:30 - 13:00]

2 Overview of the Project

Ferhat Erata
UNIT, ModelWriter Project Leader

Project 13028

ModelWriter

Text & Model-Synchronized Document Engineering Platform

The project envisions an integrated authoring environment called "ModelWriter" for Technical Authors (such as Software or Systems Engineers etc.) which will combine a Semantic Word Processor (= the "Writer" part) and a Knowledge Capture Tool (= the "Model" part).

Project information

Project name	13028 ModelWriter
Status	Running
Period	Oct 2014 - Sep 2017
Call	ITEA 2 Call 8
Challenge	Knowledge-based society
Website	www.modelwriter.eu
Partners	10
Countries	Belgium, France, Turkey

Project leader

Name
Ferhat Erata
Organisation
UNIT Information Technologies R&D Ltd.
Country
Turkey
Project involvement
13028 ModelWriter

Project documents

Project publications

- ITEA Annual report 2013 published online
- ModelWriter Posters Co-summit 2015
- ModelWriter Project Leaflet

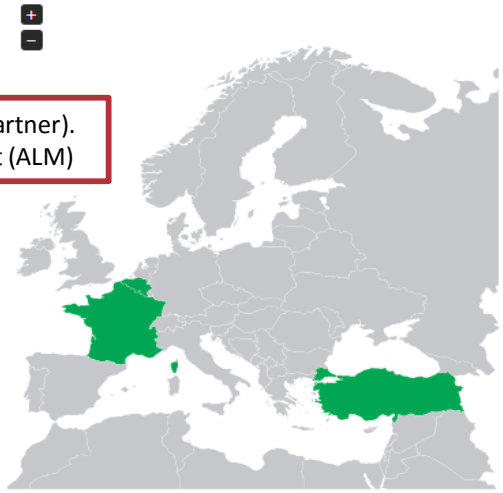
Project partners

Airbus Group SAS France	Centre National de la Recherche Scientifique France	Havelsan Turkey	Hisbim Bilgi ve İletişim Teknolojileri France
Katholieke Universiteit Leuven Belgium	KoçSistem Turkey	Mantis Turkey	beo France
Sogeti Belgium Belgium	UNIT Information Technologies R&D Ltd. Turkey	Ford Otosan Turkey	

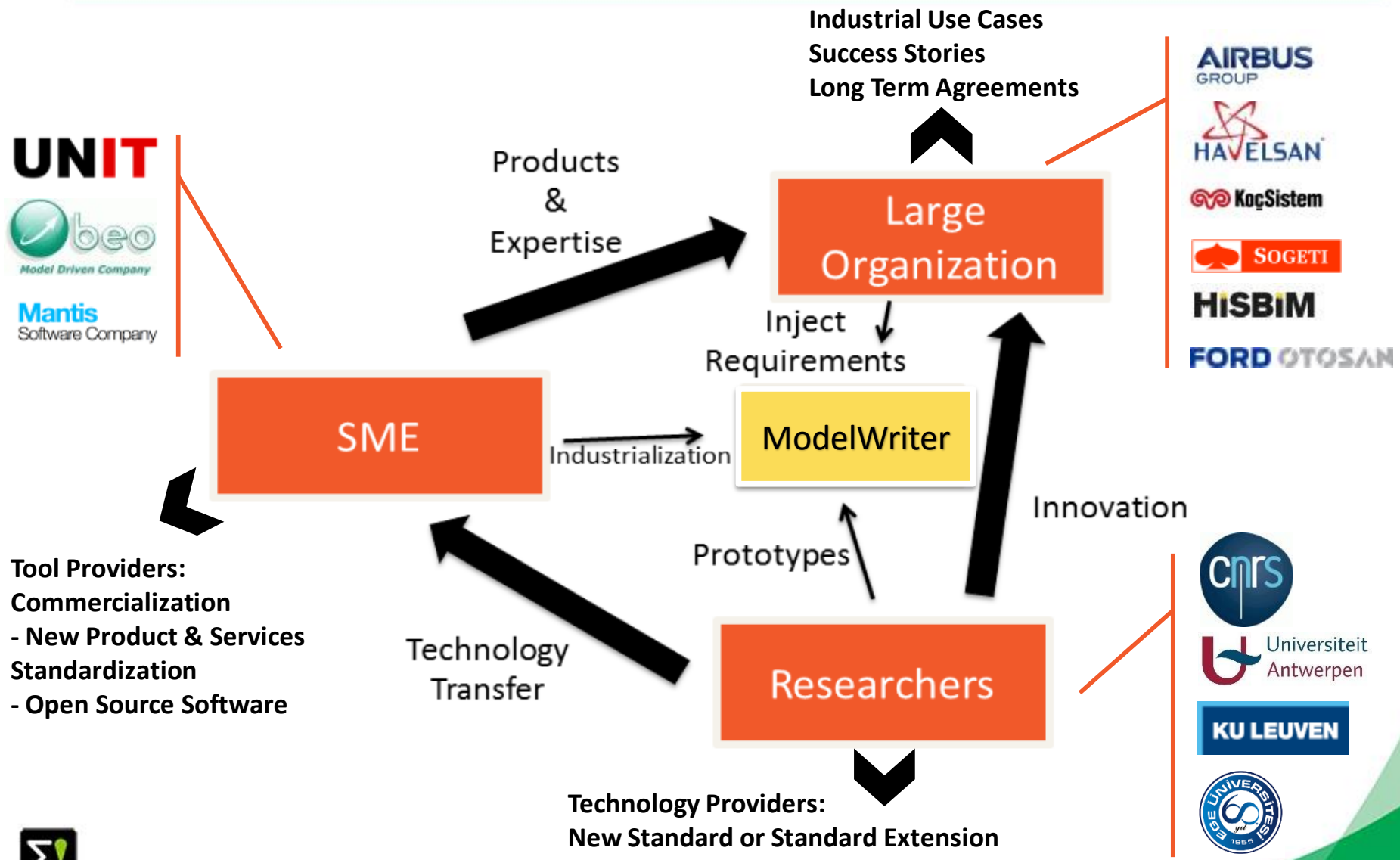
Industrial End User (self-funded partner).
Application Lifecycle Management (ALM)

Industrial End User.
Testing & Global Engineering Services

Industrial End User (self-funded partner).
Product Lifecycle Management (PLM)



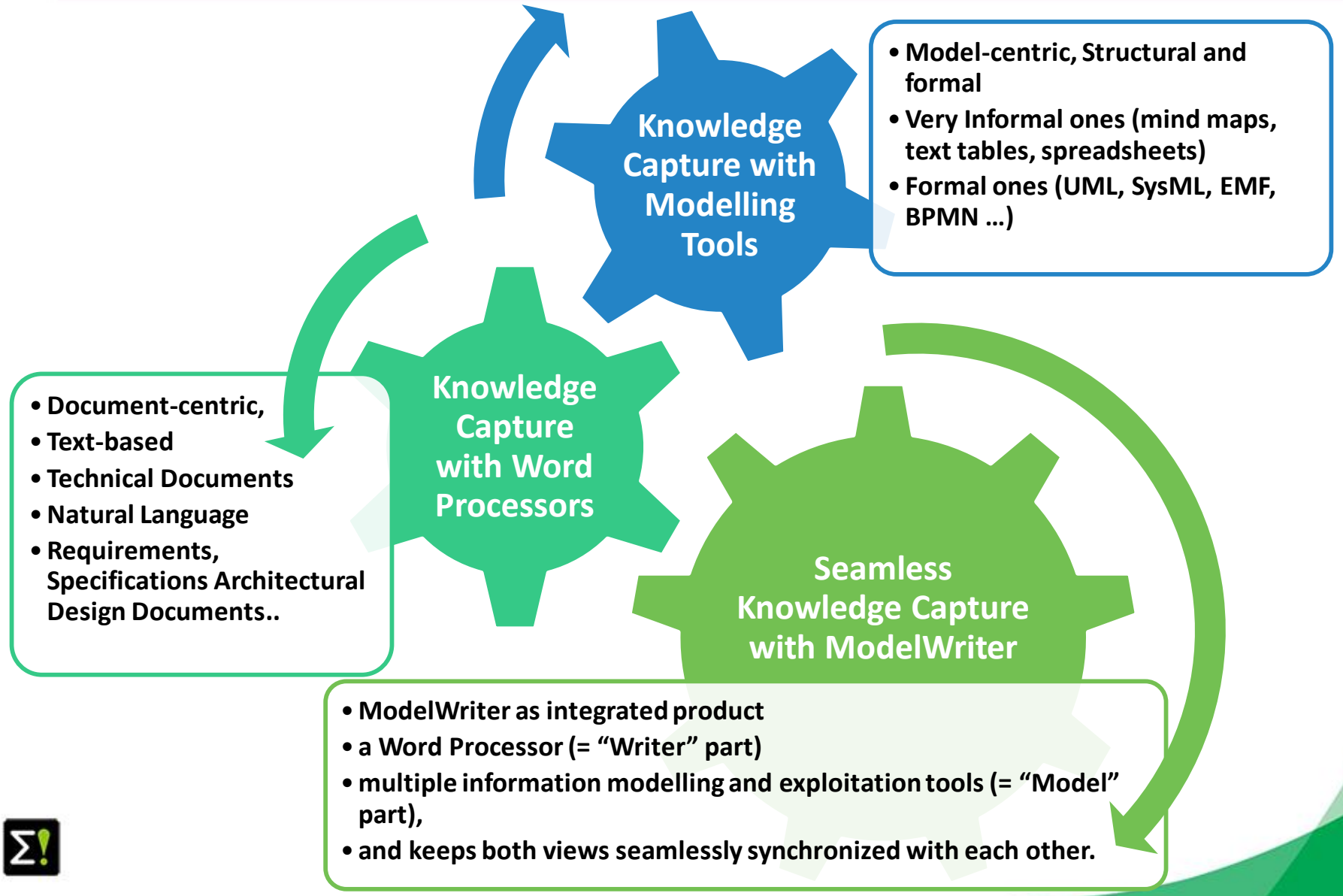
Industrialization Triangle in ModelWriter
Open Source Software



ModelWriter
Project Overview

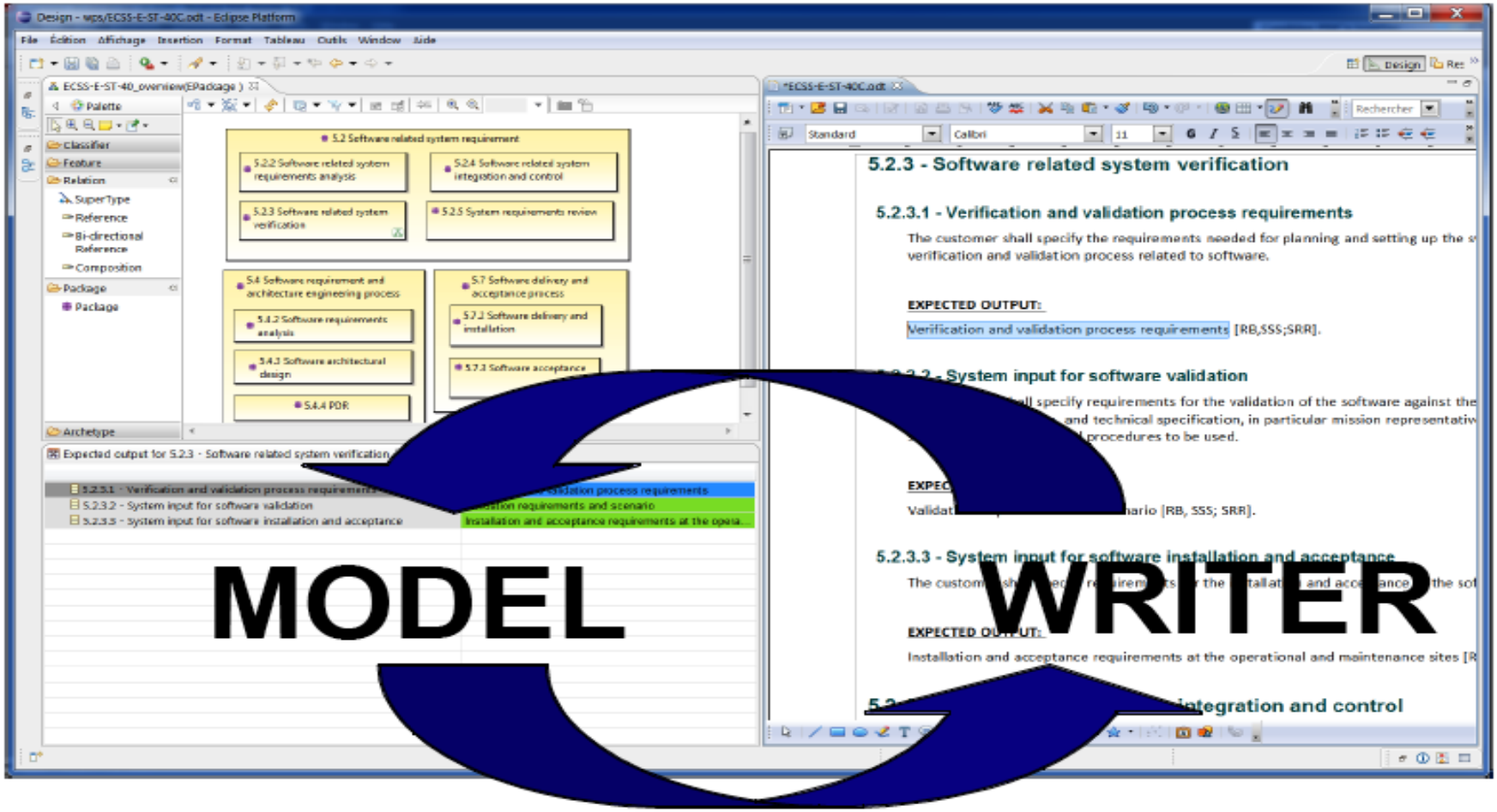
- Resource Allocation: 68,71 person year
- Project Duration: 36 months
- Planned Budget: 5,543,000 Euro
- Start and Finish Date: 01 Oct 2014 – 30 Sep 2017
- Open-Source Software Platform to be submitted to Eclipse Foundation

ModelWriter
Today's Knowledge Capture



ModelWriter

bi-directional Knowledge Capture tool



ModelWriter
Text-Based Knowledge Extraction

Semantic Word Processor
(Text-Based Knowledge Extractor)

Understands the various textual parts of a document expressed in Natural Language

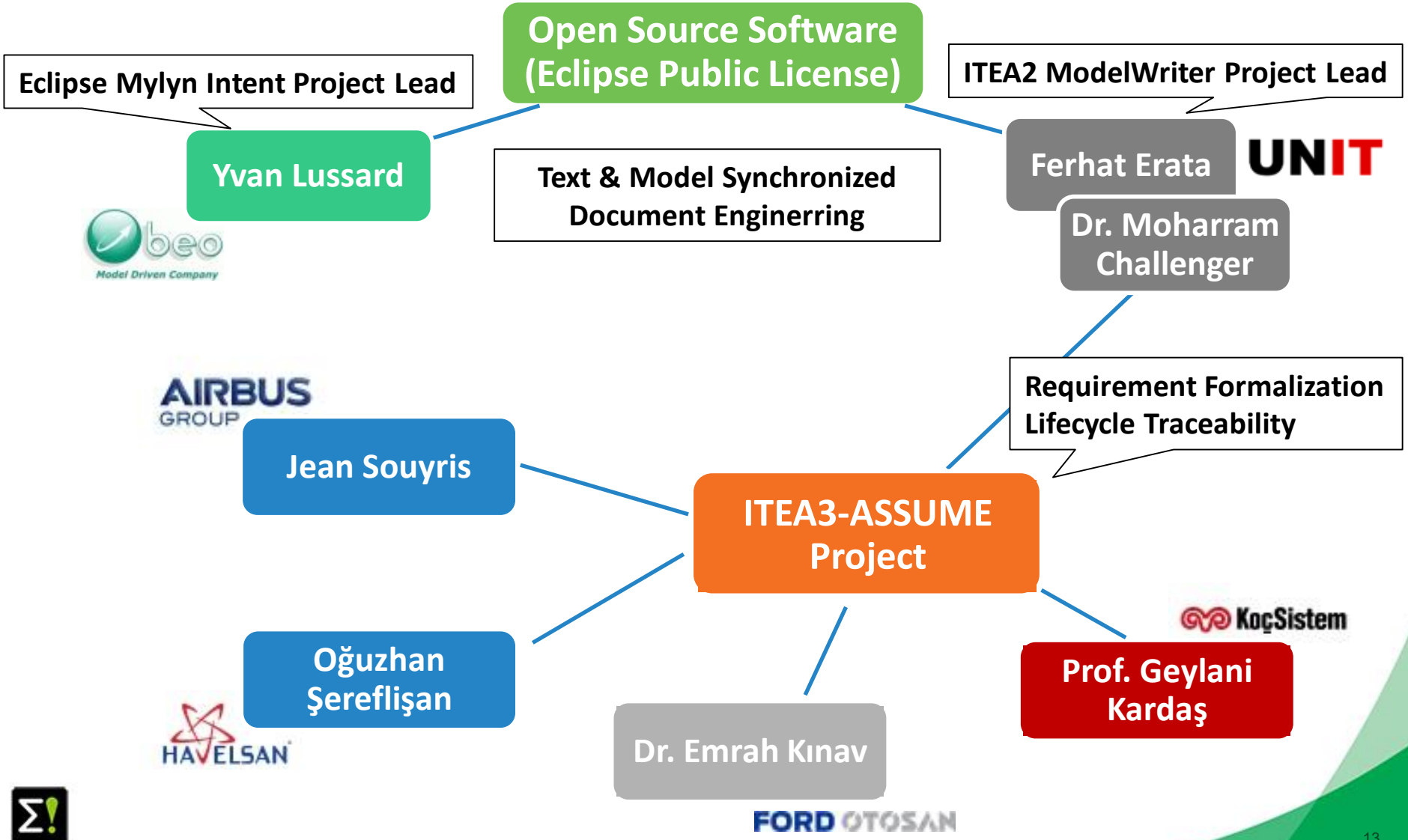
Reveals concepts and relationships between them (“Model”-part)

Consistency & Completeness Checking

ModelWriter
Further Knowledge Valorizations

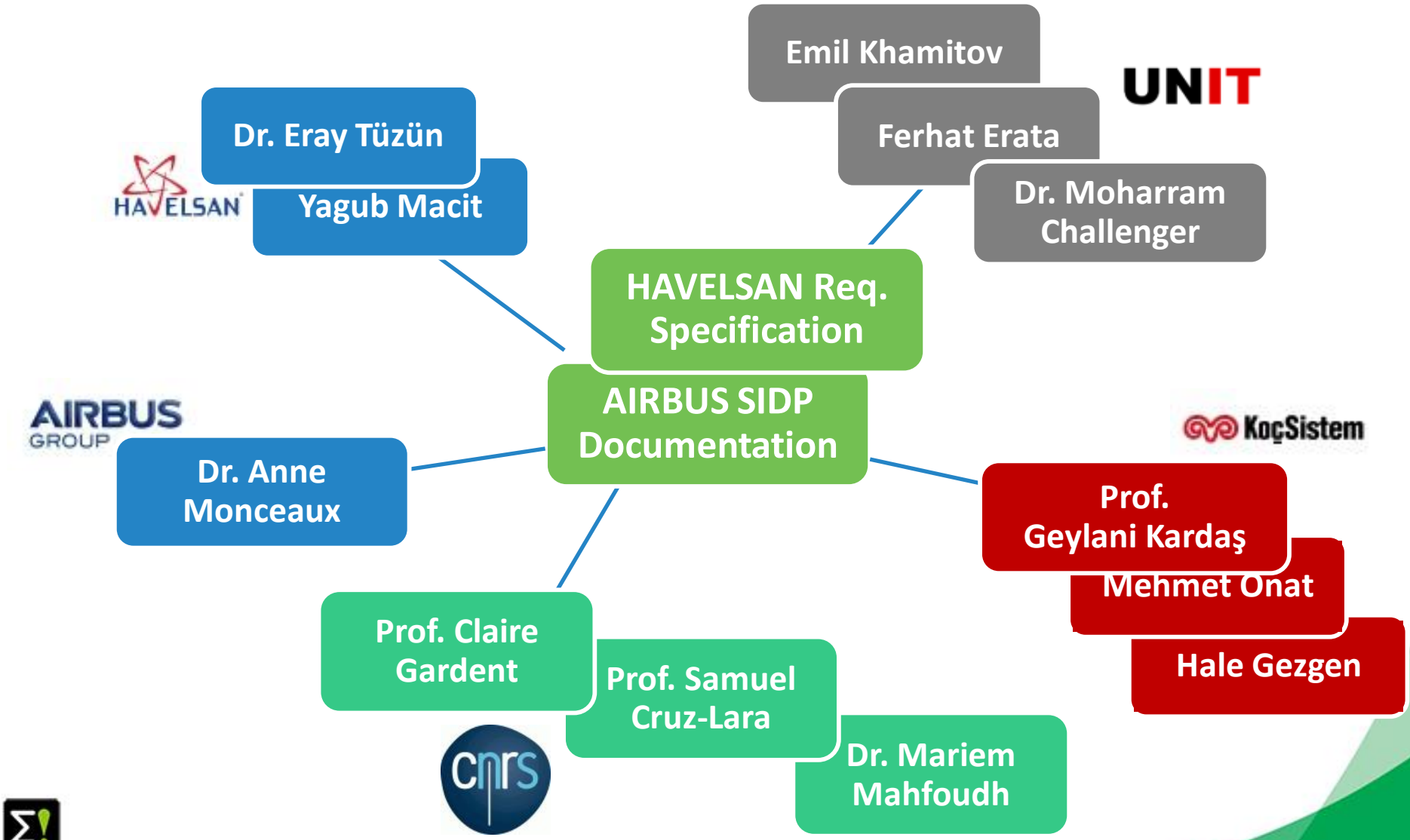
- “Everywhere” Document Regeneration: "tell once, show everywhere“: recycling knowledge from (1) the same document, or of (2) another related document
- Consistency Checking: the objective to provide a Consistency Checker that automates Quality Reviews of Requirements Engineering
- Open Source Software under Eclipse Foundation for Future Dissemination and Exploitation to further extend the Business Value Chain
- “MW” Knowledge Dissemination Standard (.mw ModelWriter exchange format)

Level of Collaboration within ModelWriter
Exploitation of MW in Eclipse & ASSUME



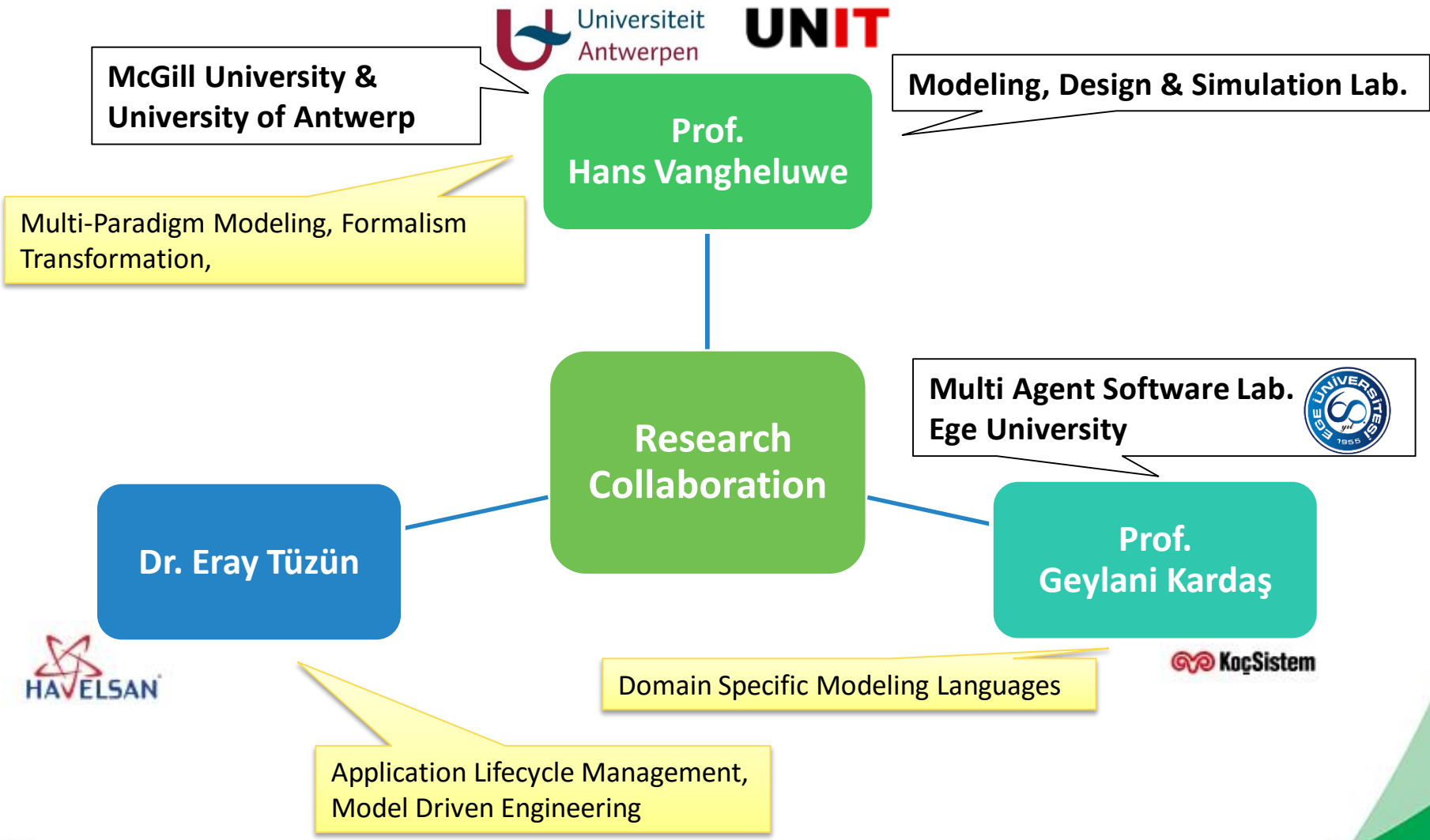
Level of Collaboration within ModelWriter

International Collaboration (through UCs)

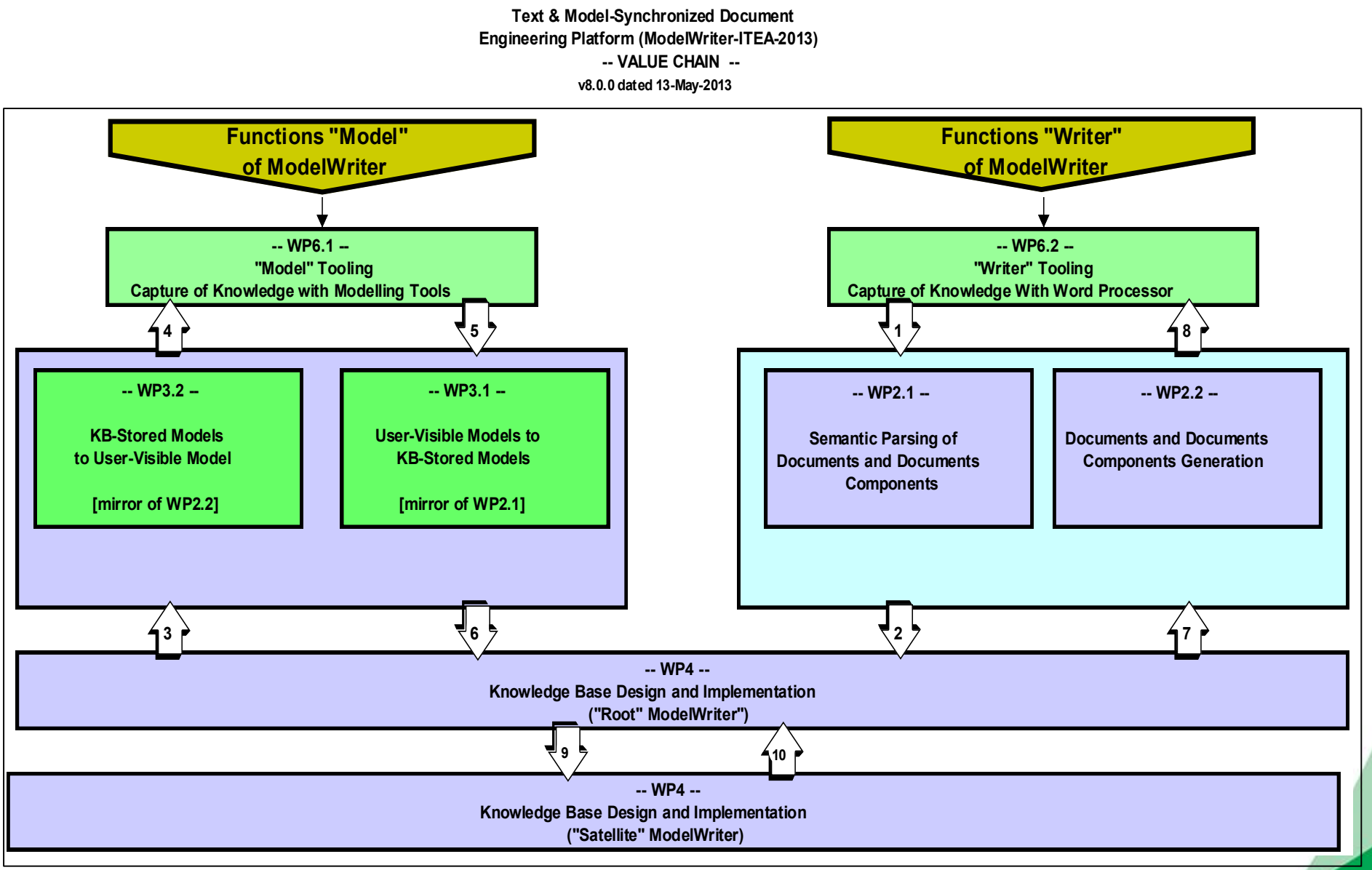


Level of Collaboration within ModelWriter

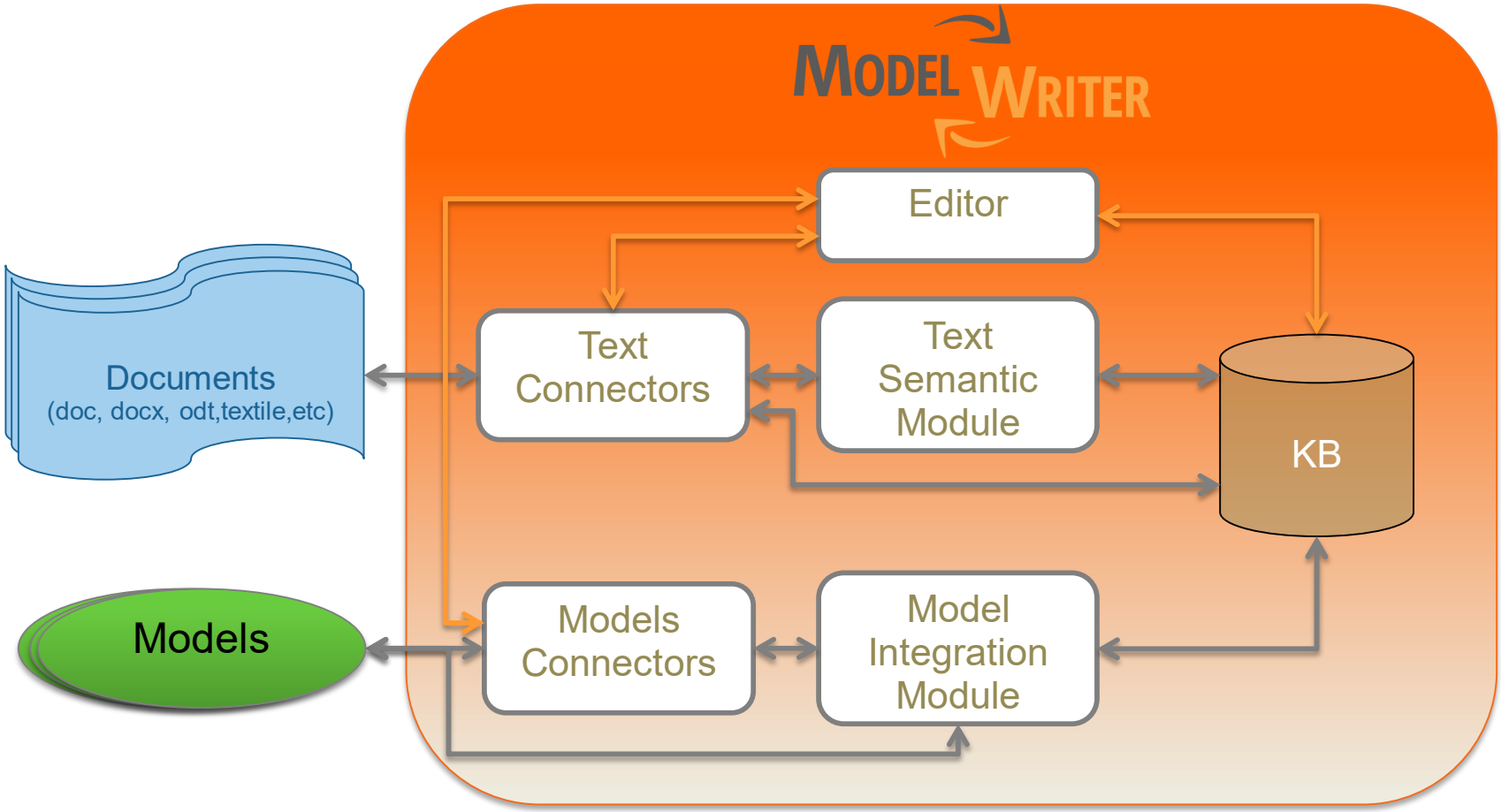
International Collaboration (research)



Technological components & interactions



ModelWriter
Conceptual Architecture



ModelWriter
Work Packages & Technical Innovations

ITEA 3

WP1 Industrial Use Cases and Requirements (AIRBUS)

WP2 (LORIA)

- Semantic Parser
- Document Generation
- bi-directional transformation between text and formal knowledge representation

WP3 (UNIT)

- Bi-directional synchronization mechanism between texts and models
- Configuration & Traceability Components
- Consistency checker plug-in for consistency

WP4 (MANTIS)

- A federated Knowledge Base and its API
- Synchronization mechanism between texts/models & knowledge base

WP6 (OBEO)

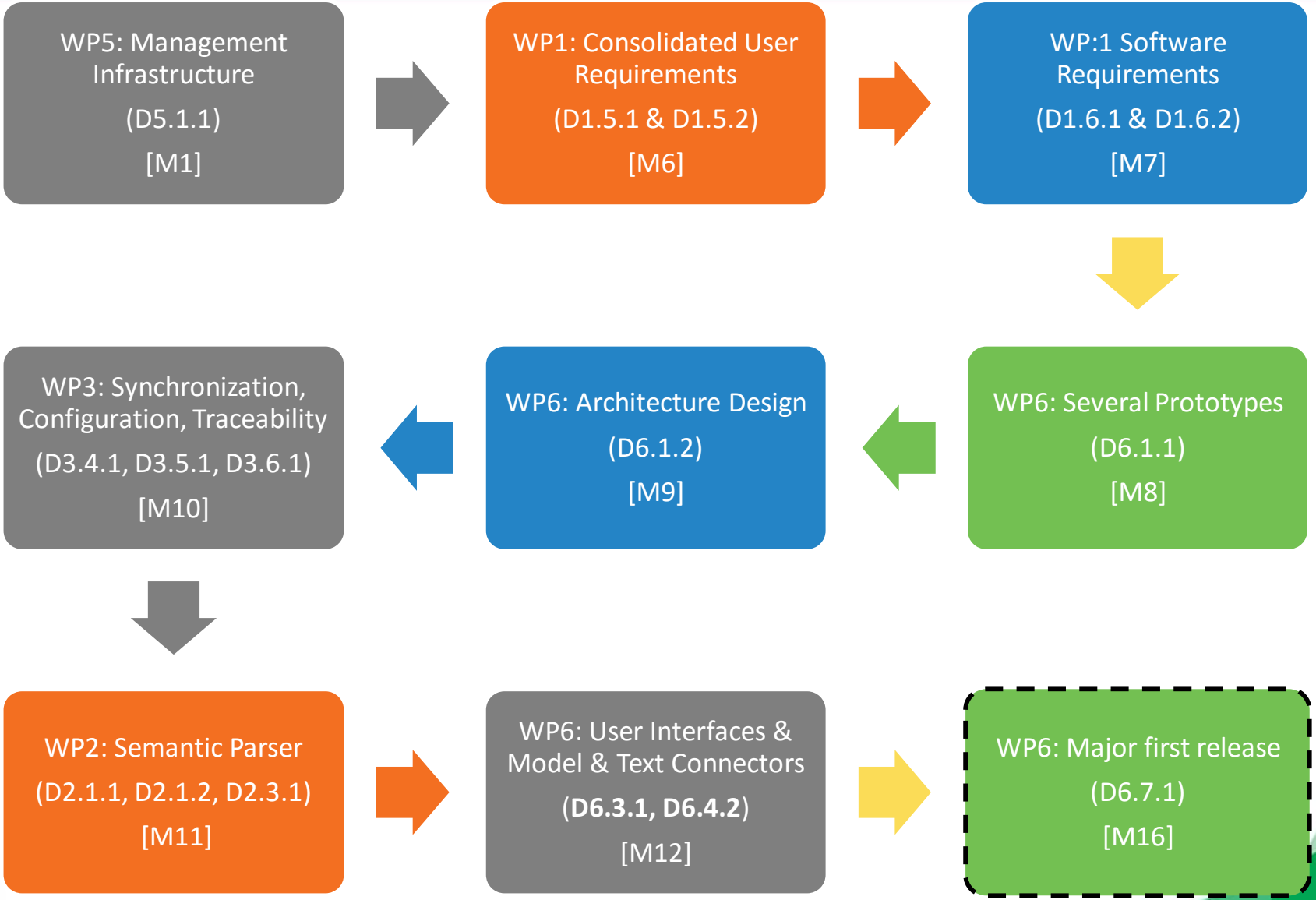
- A complete “ModelWriter” tool integrating of all these in a consistent way
- User Interfaces

WP5 Project Management (UNIT)

WP7 Standardization, Dissemination and Exploitation (OBEO)

ModelWriter

Overall Progress in the first year



ModelWriter Workshops in the First Year

https://github.com/modelwriter/workshops

Project Kick-off in Istanbul, Turkey (Nov 08, 2014) [M1]

Initial Architectural Design, Indutrial Use Cases, Technical WP discussions

Collaboration Infrastructure

The 1st International ModelWriter Workshop in Izmir, Turkey (Jan 15-17, 2015) [M4]

Exploitation: Havelsan's participation

The 1st International Eureka Project Exhibition in Berlin, Germany (Mar 10-11, 2015) [M6]

Consolidated User Requirements & Review

The 2nd International ModelWriter Workshop in Brussels, Belgium (Apr 30, 2015) [M7]

Software Requirements Review & Architecture

The 3rd International ModelWriter Workshop in Toulouse, France (Jun 22-23, 2015) [M10]

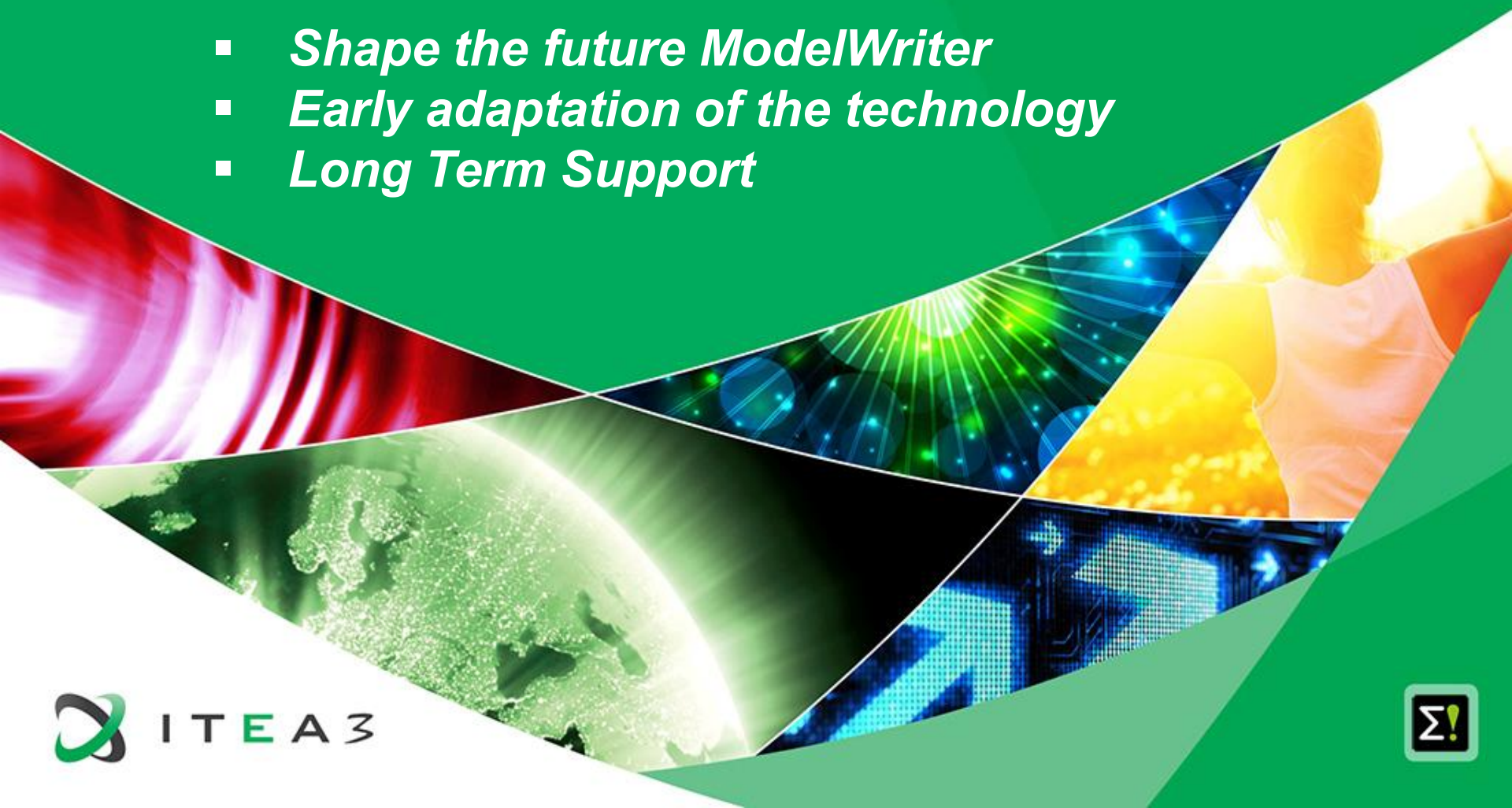
The 4th International ModelWriter Workshop in Brussels, Belgium (Sep 23-24, 2015) [M12]

Integration of software components

The 5th International ModelWriter Workshop in Ludwigsburg, Germany (Nov 2-5, 2015) [M16]

Open Source Campaign
Open Call for Industrial User Stories

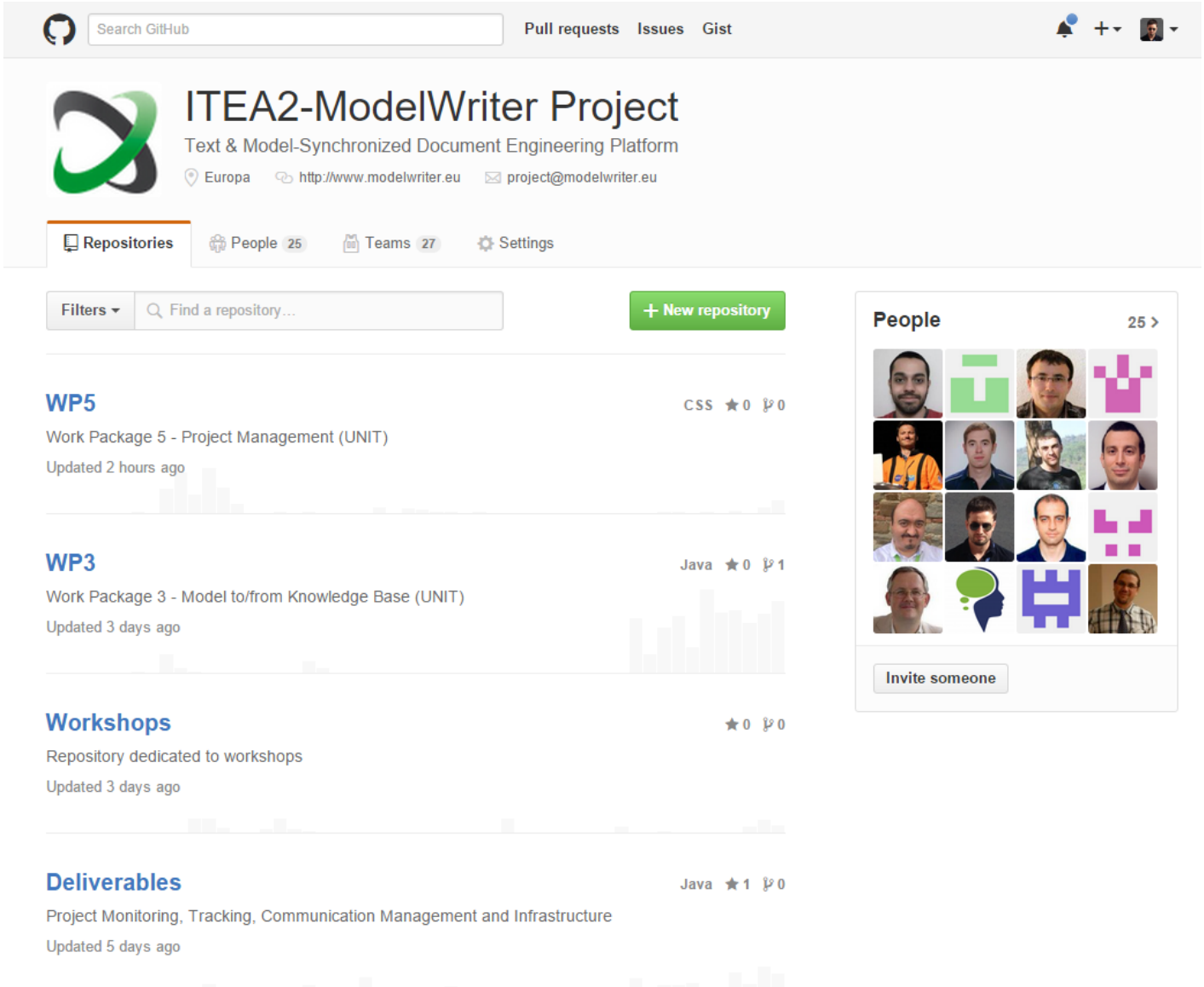
- *Shape the future ModelWriter*
- *Early adaptation of the technology*
- *Long Term Support*



ITEA 3

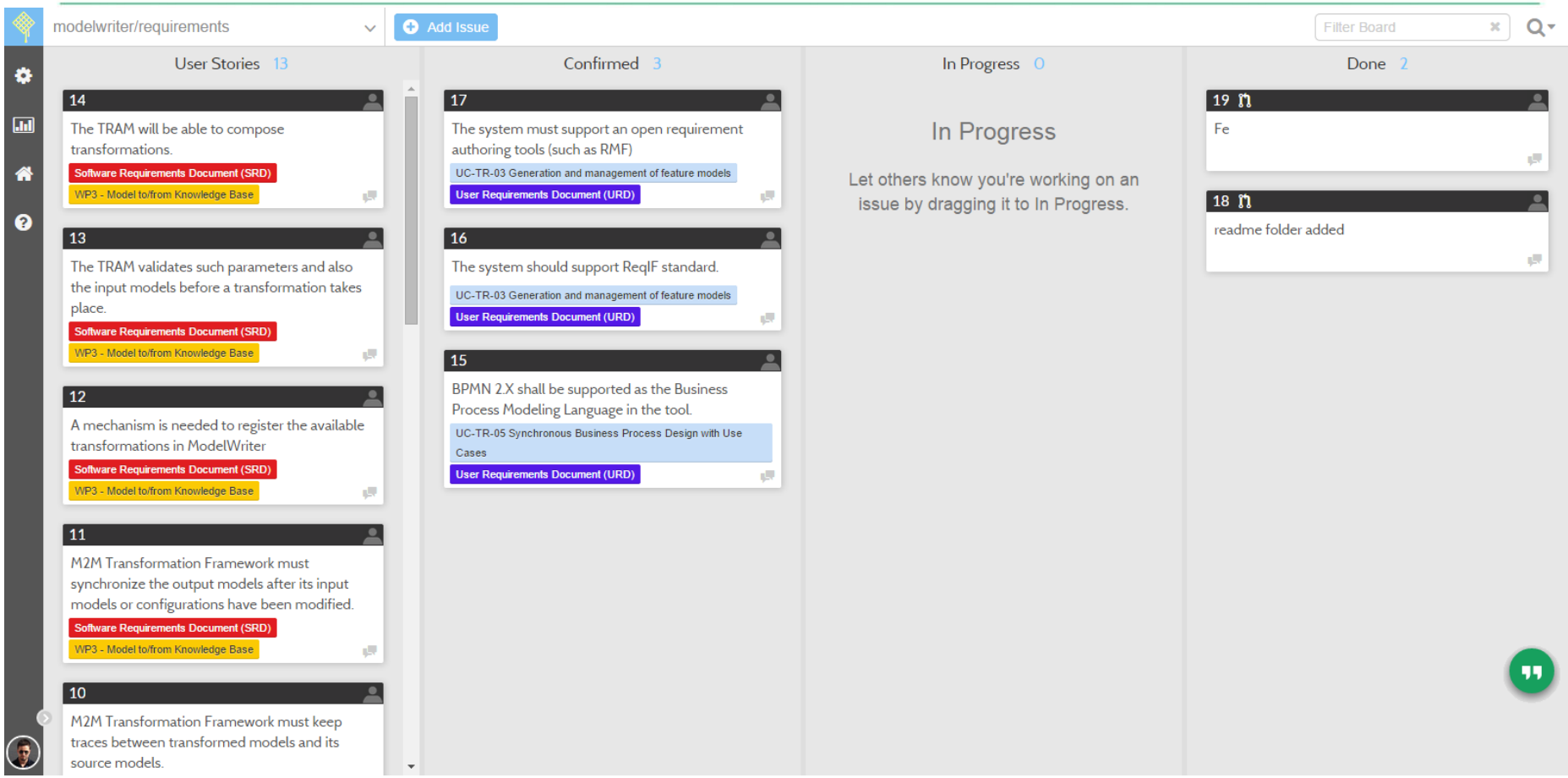
ModelWriter Open Source Campaign

https://github.com/modelwriter



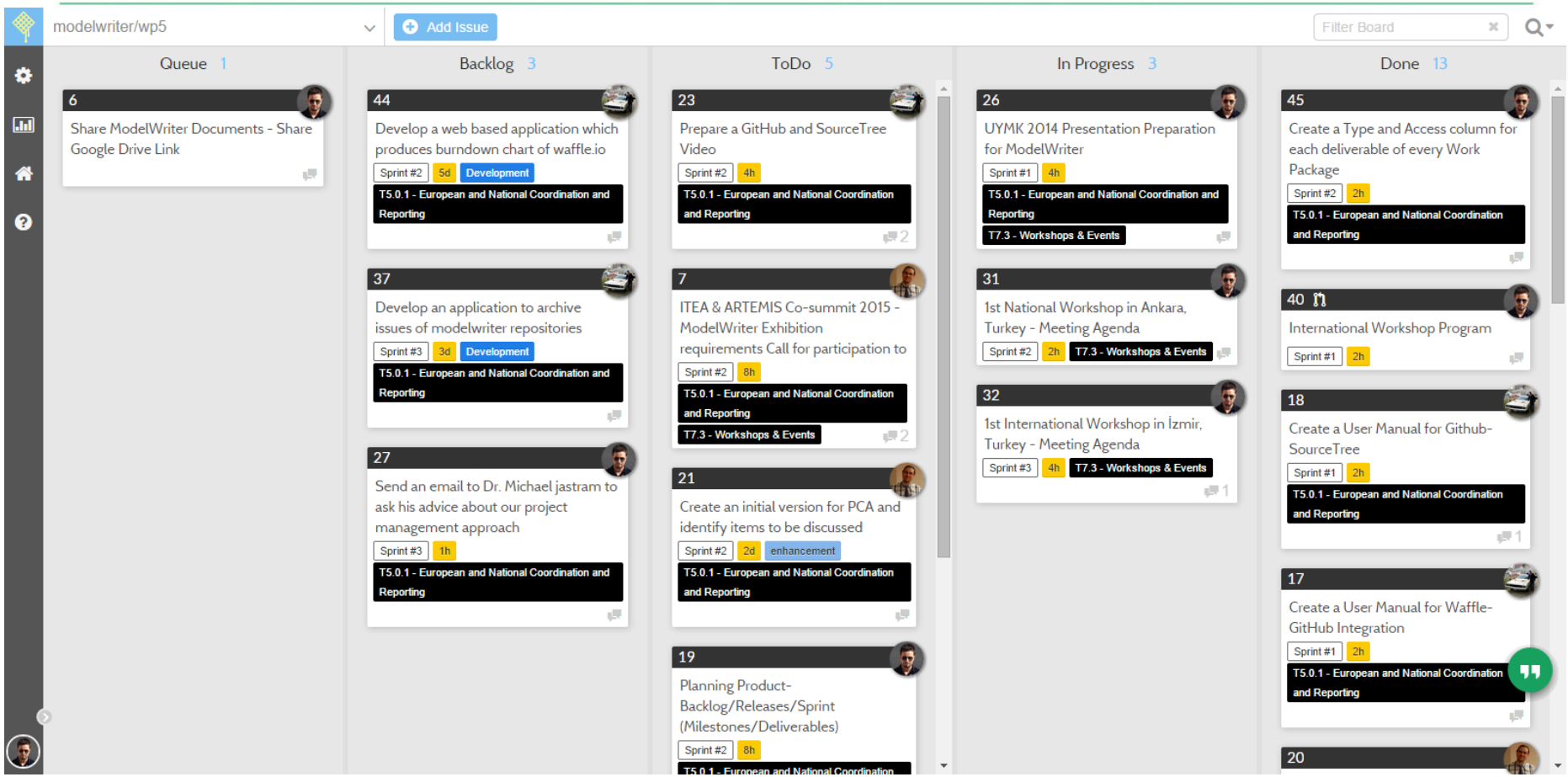
ModelWriter Requirements & User Stories

https://waffle.io/modelwriter/requirements



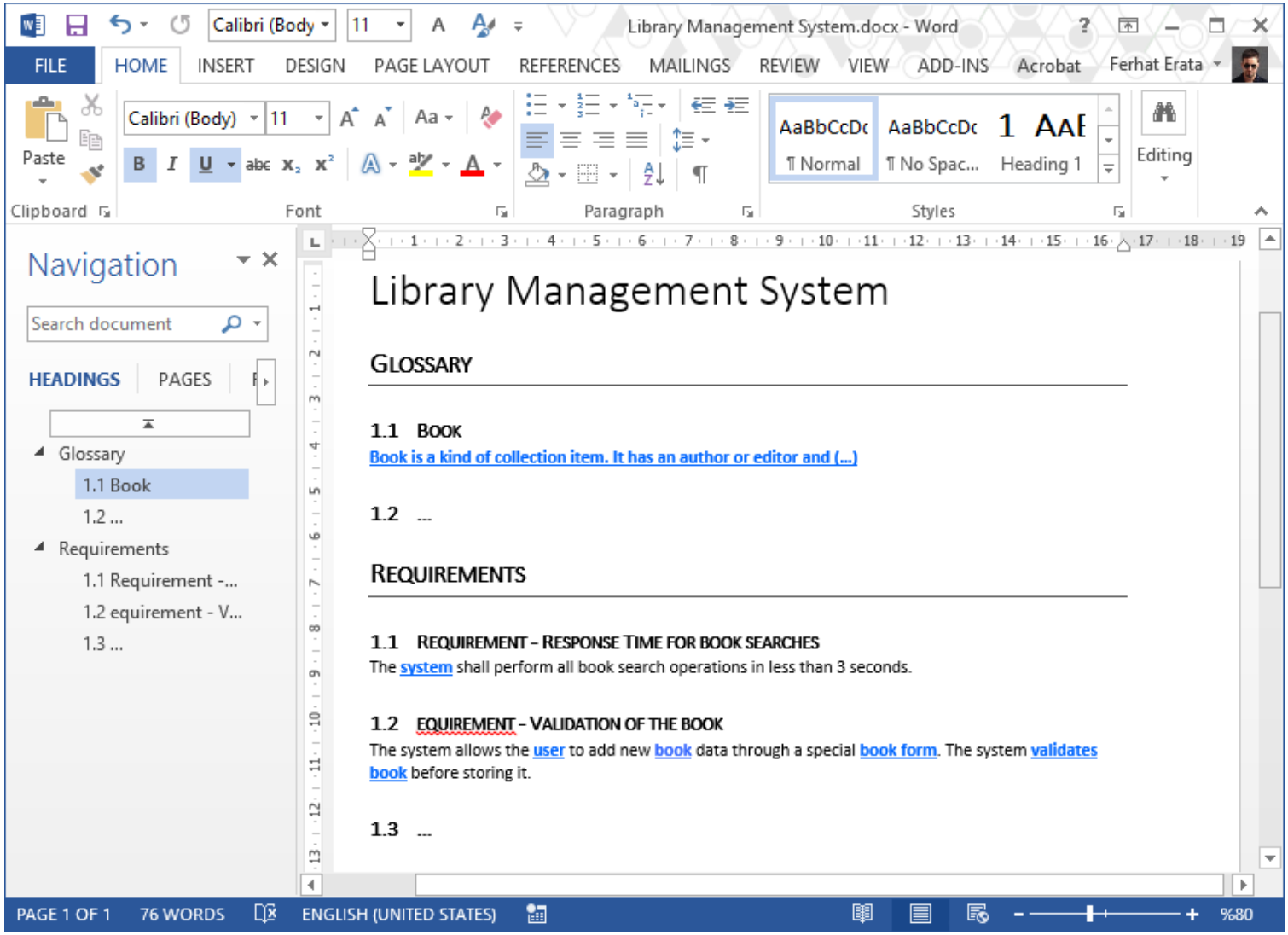
ModelWriter Requirements & User Stories

https://waffle.io/modelwriter/wp5

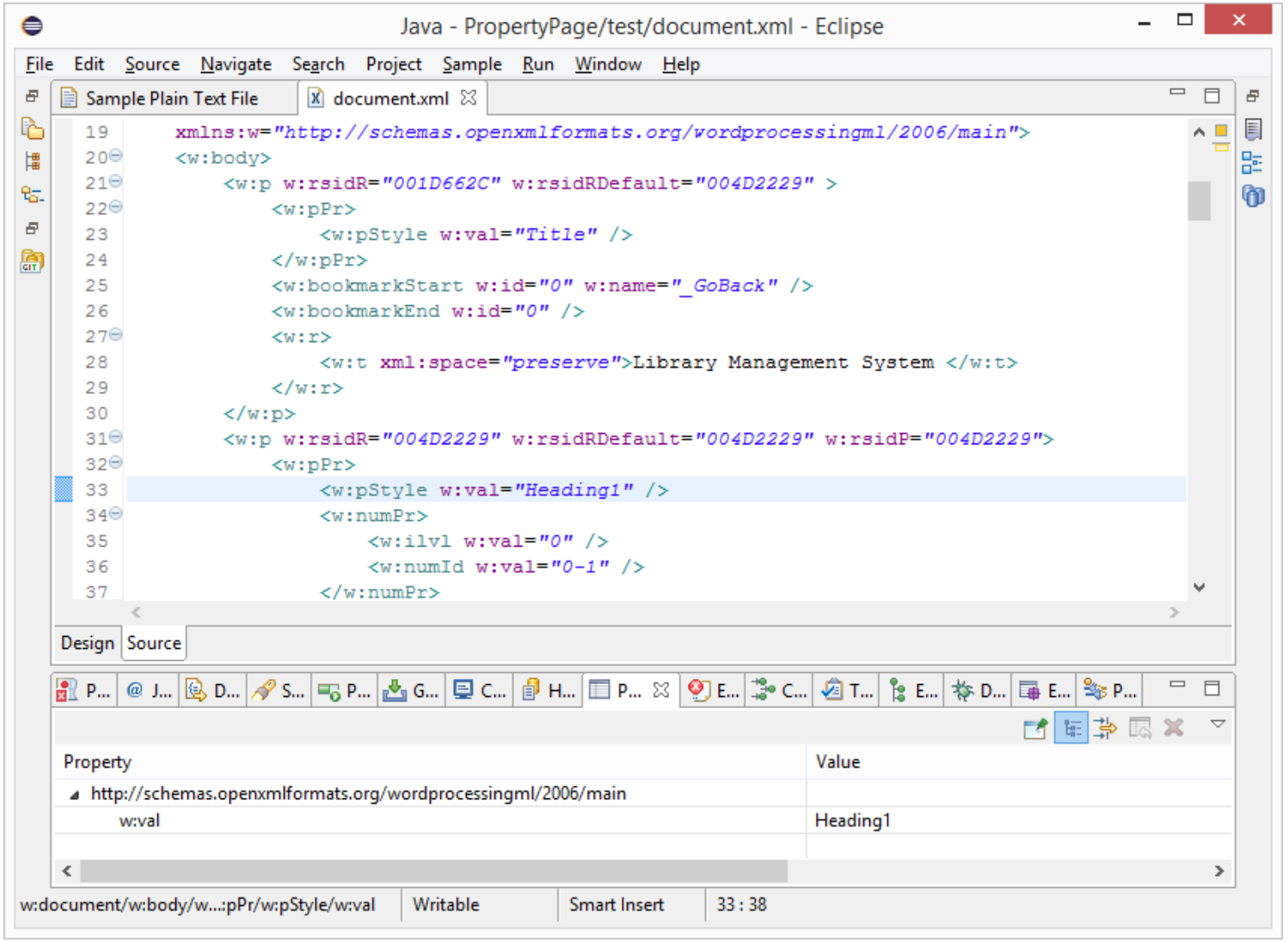


What is a text?

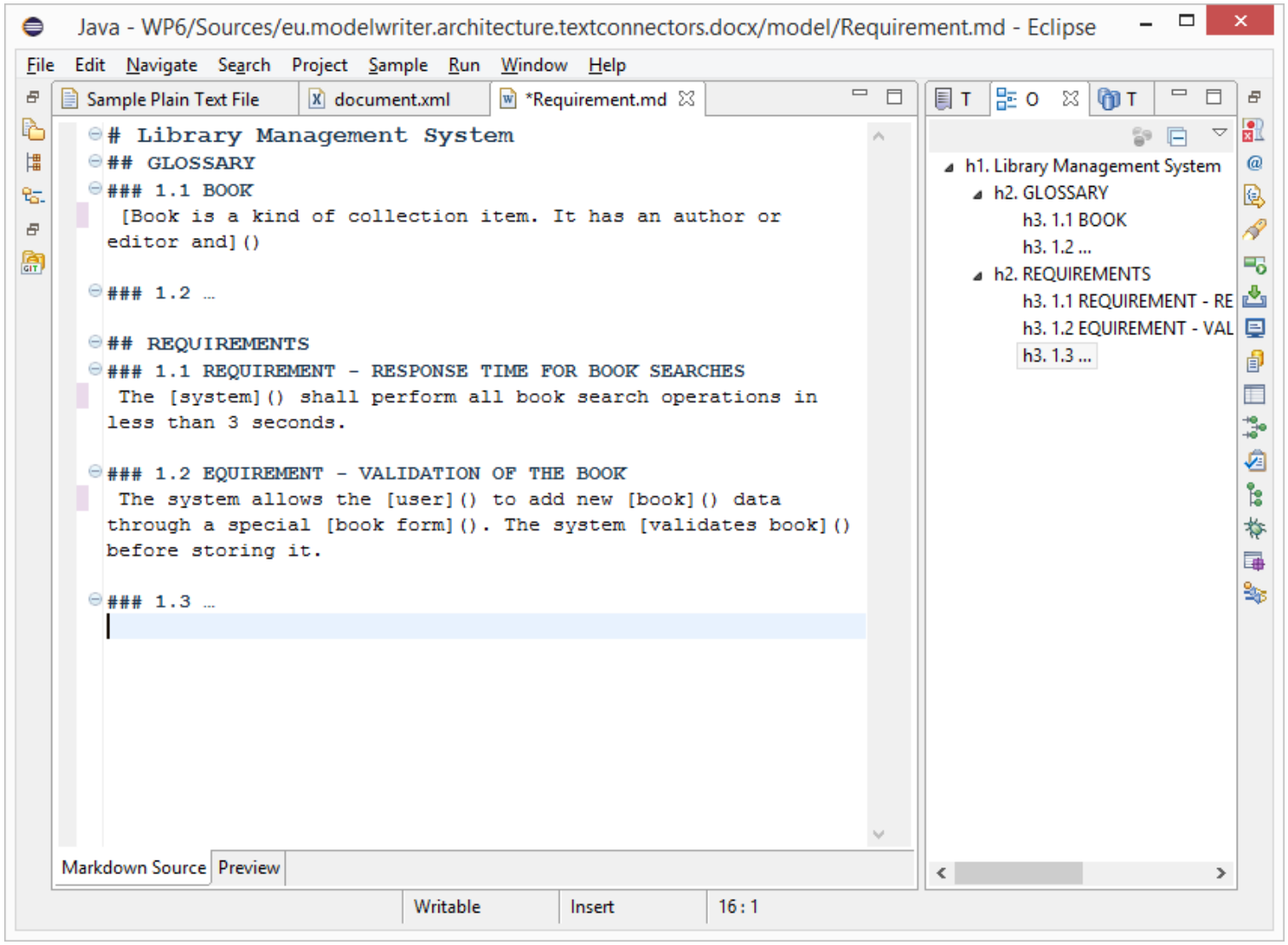
What is a text? (document file formats)
Office Open XML (.docx) (ISO/IEC 29500)



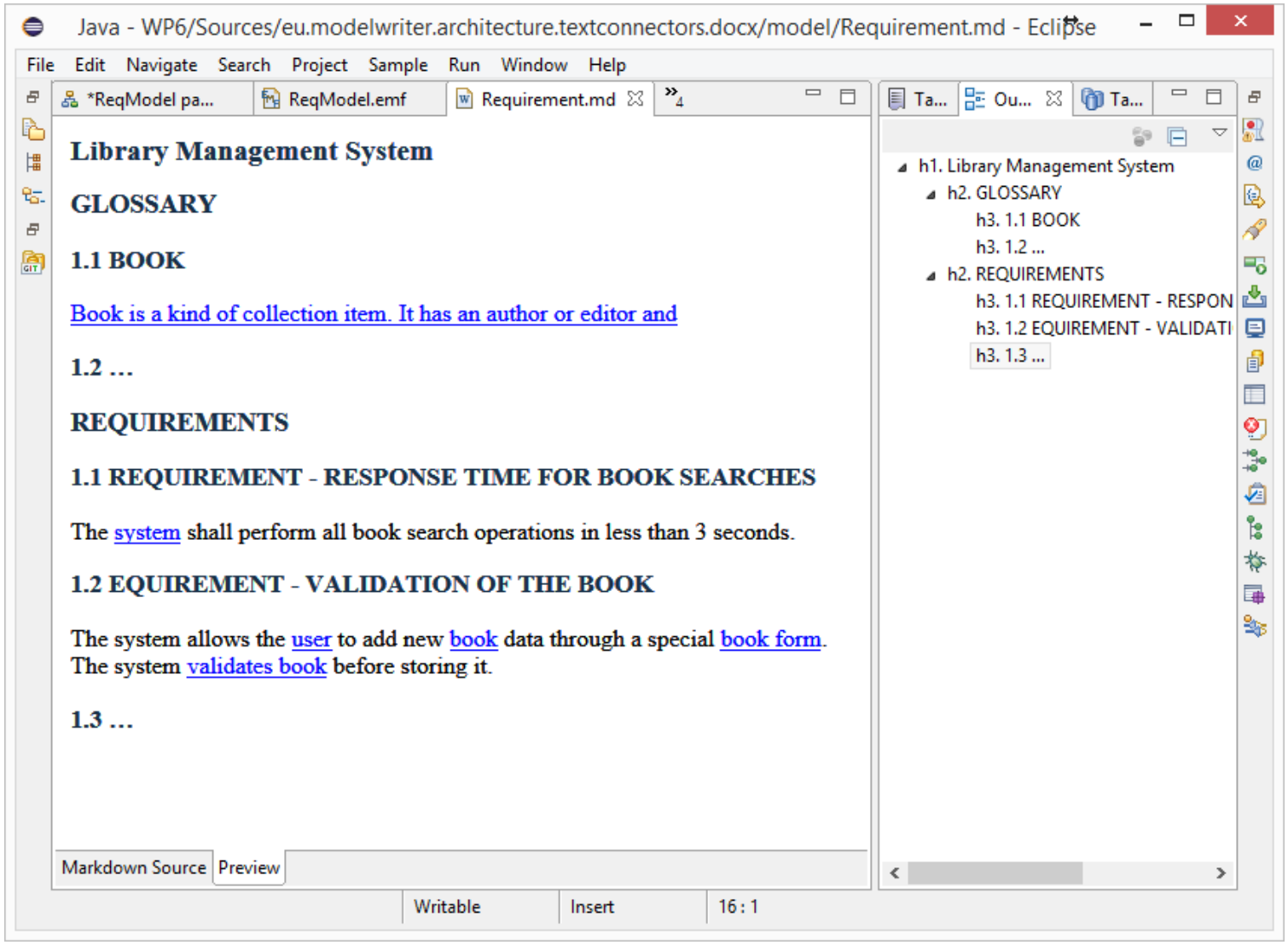
What is a text? (document file formats)
Office Open XML (.docx) (ISO/IEC 29500)



What is a text? (.md source file)
text/markdown (ICANN Standard)

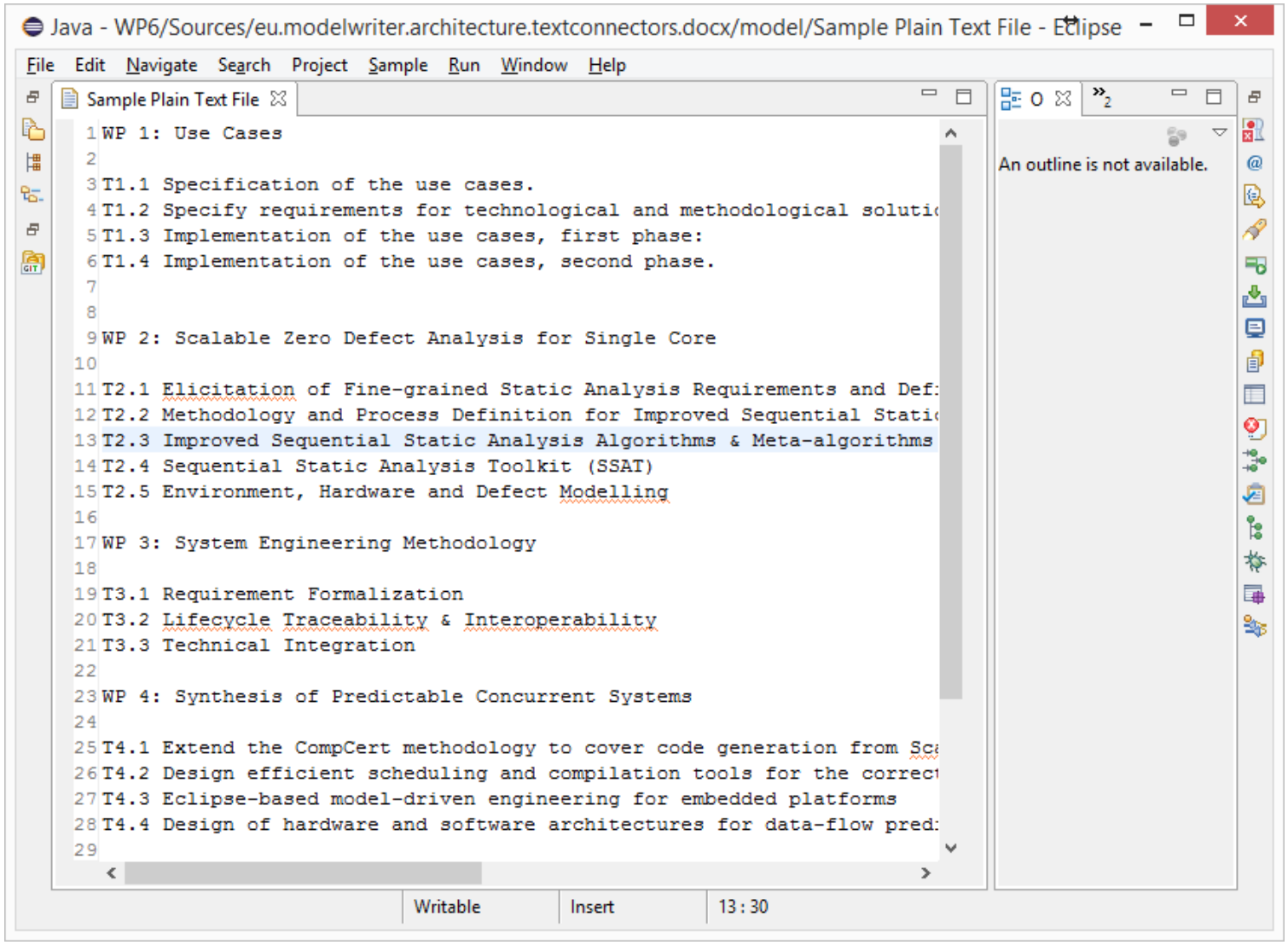


What is a text? (HTML Preview)
text/markdown (ICANN Standard)



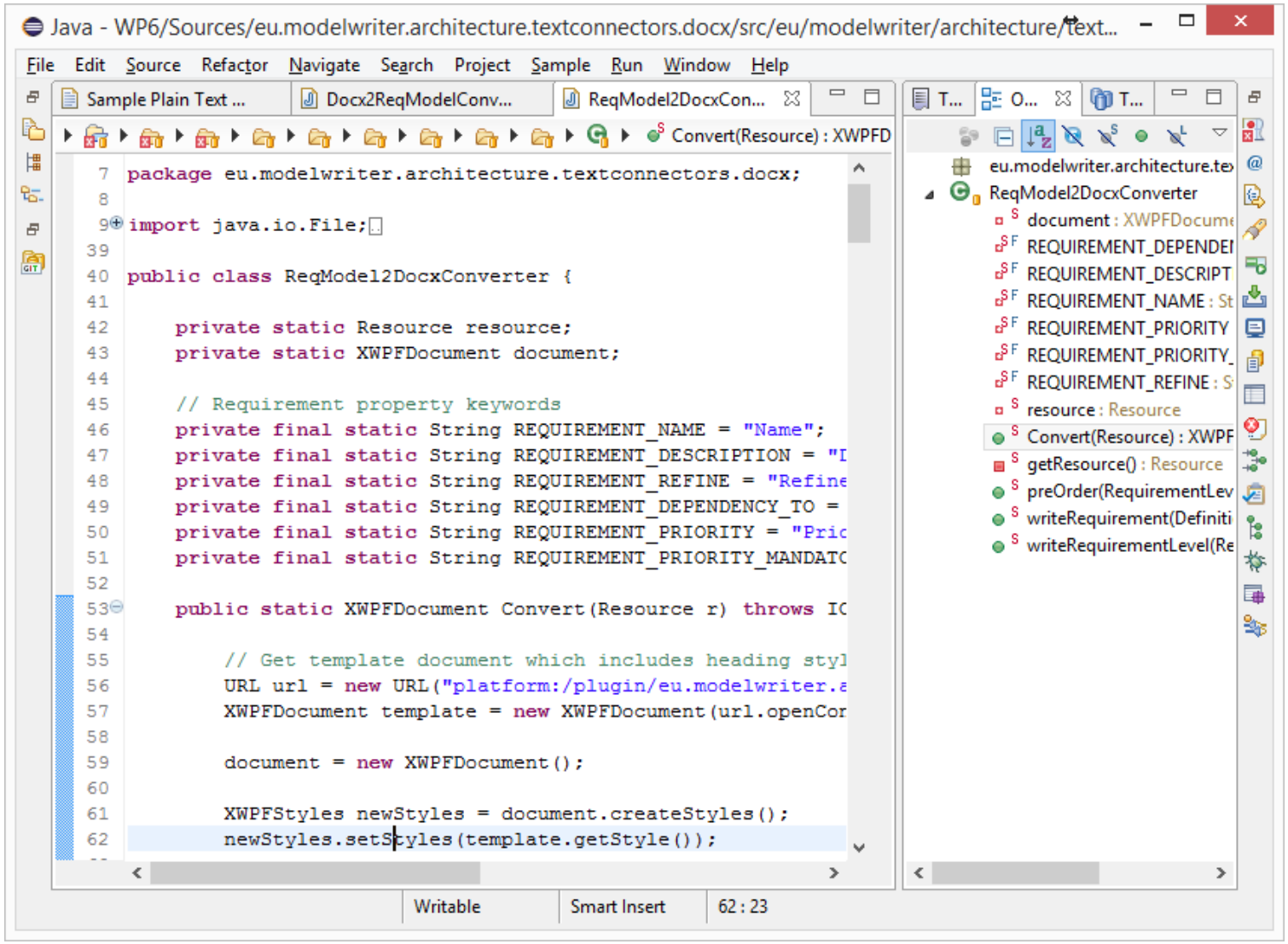
What is a text? (unformatted text)

text/plain (ICANN Standard)

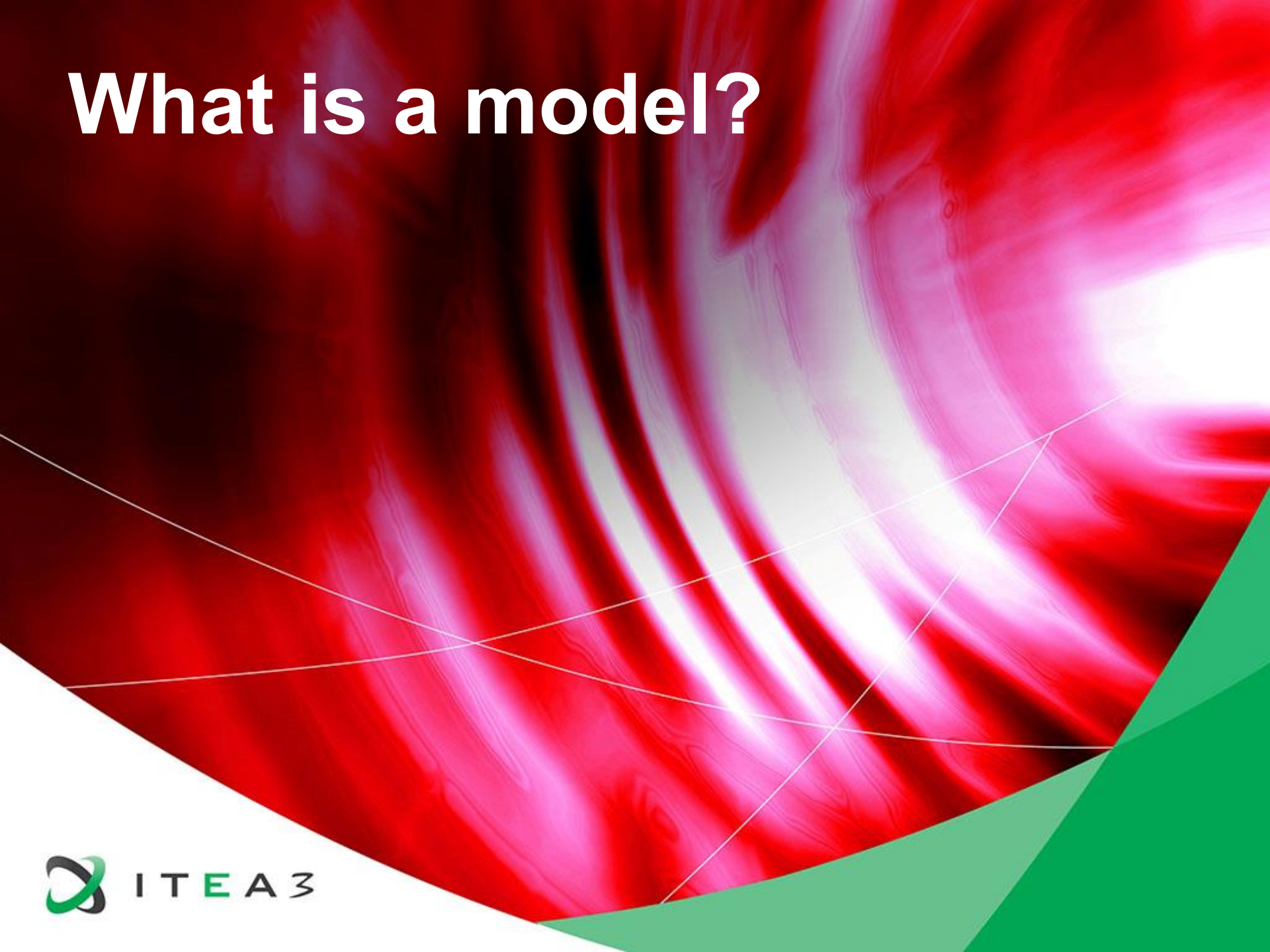


What is a text? (code files)

Java, C++ … Programing Languages

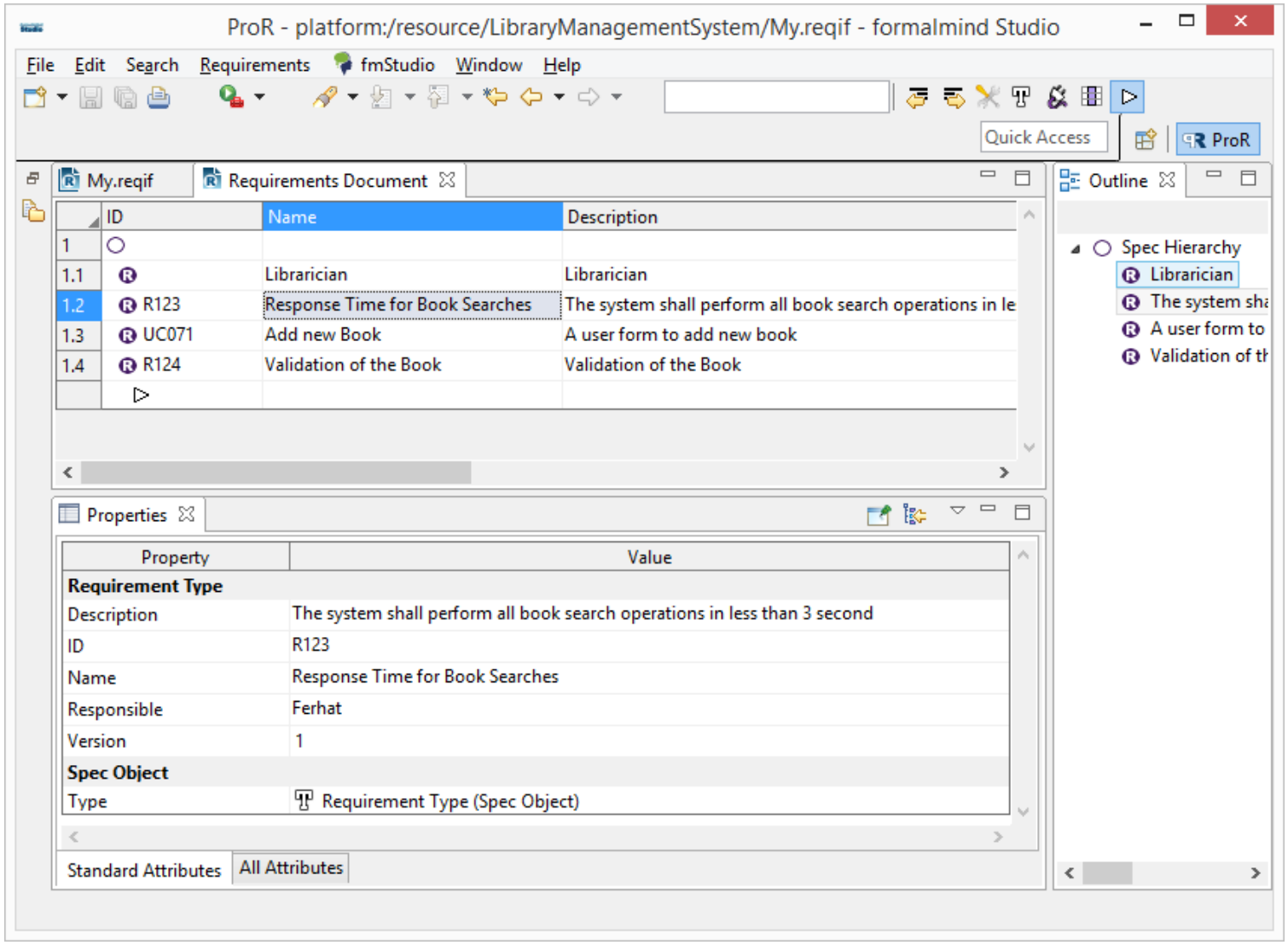


What is a model?

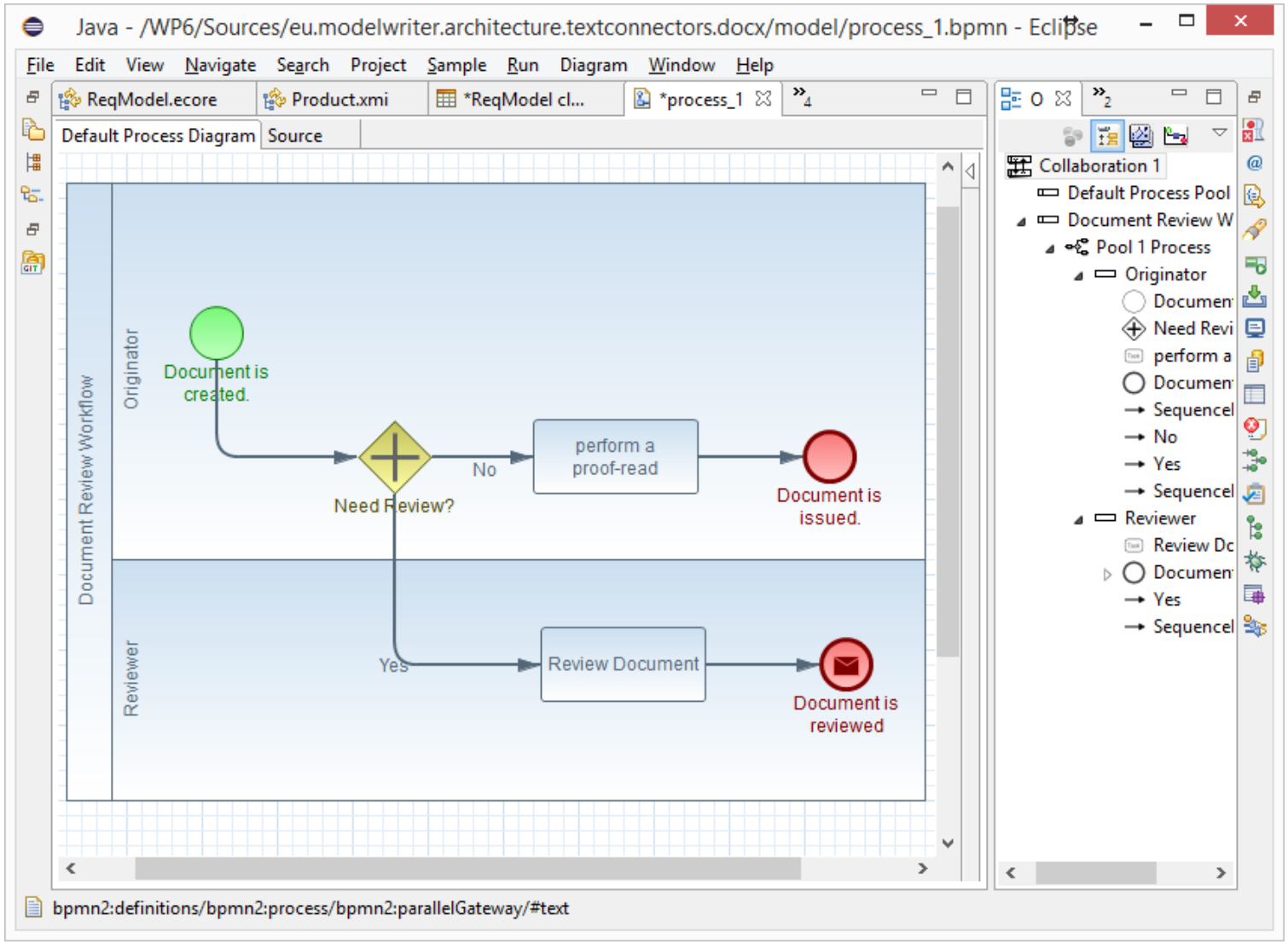


Everything is a model! (ReqIF Standard)

Requirements Interchange Format

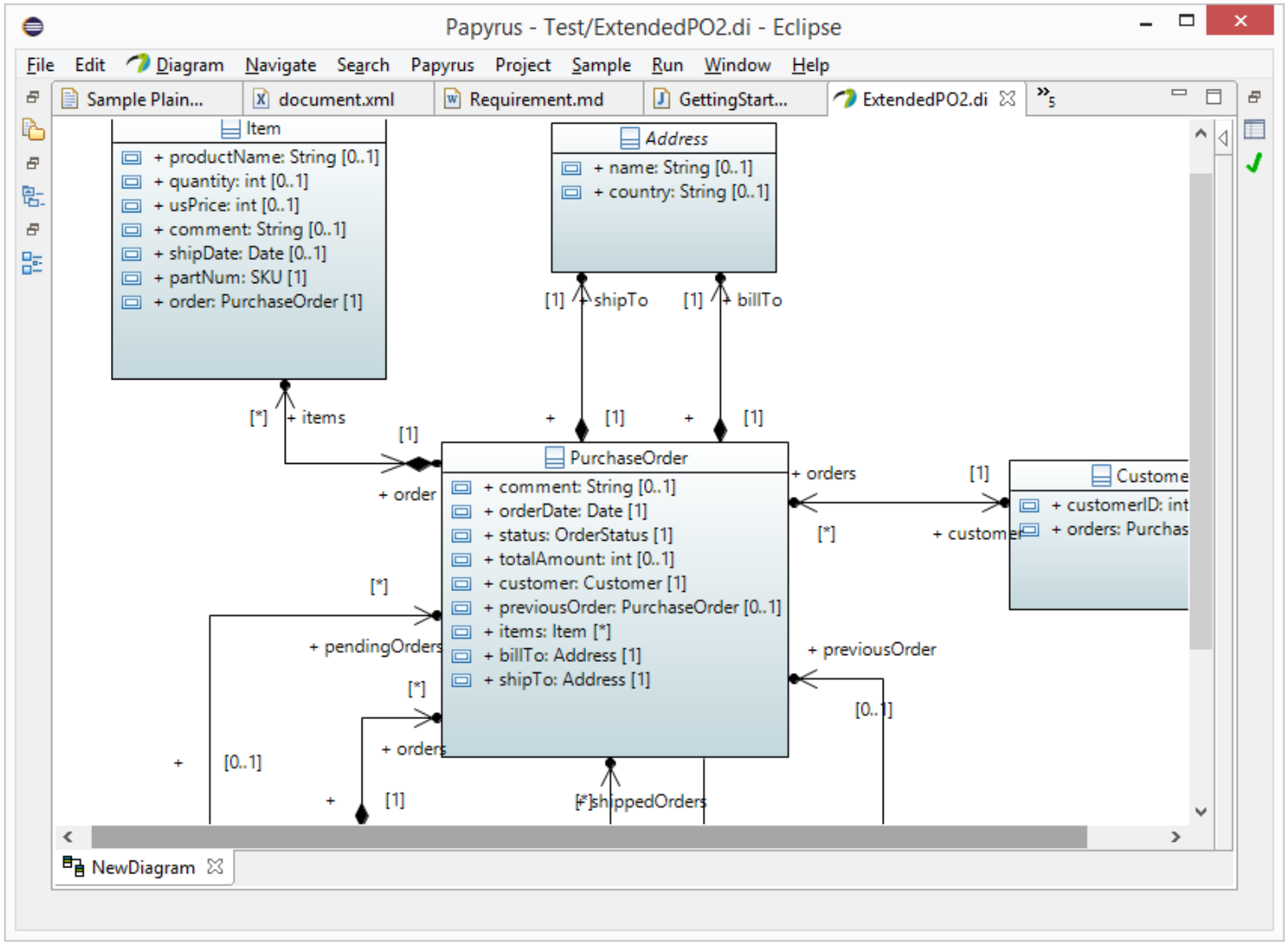


Everything is a model! (BPMN Standard)
Business Process Model & Notation

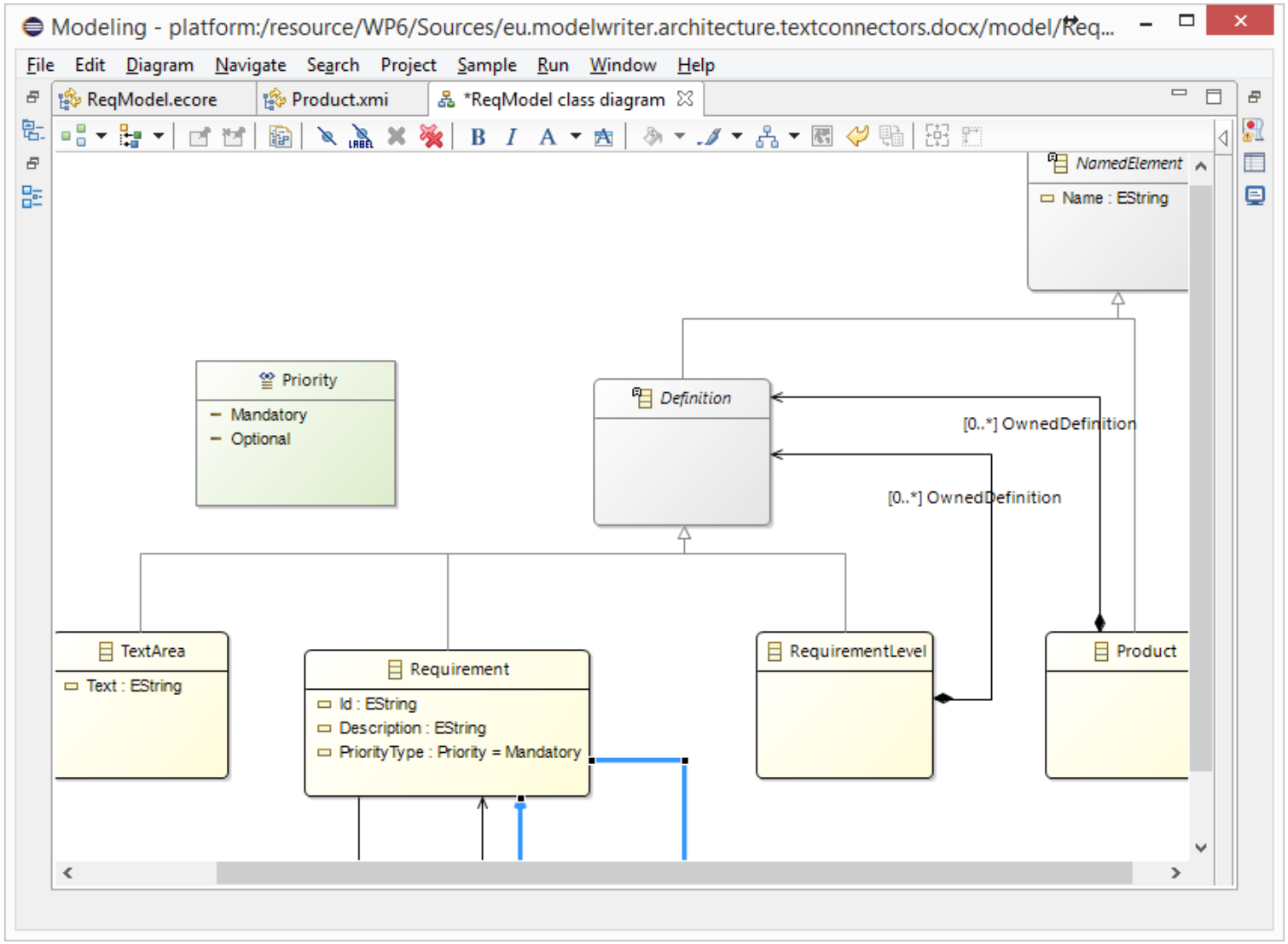


Everything is a model! (UML Standard)

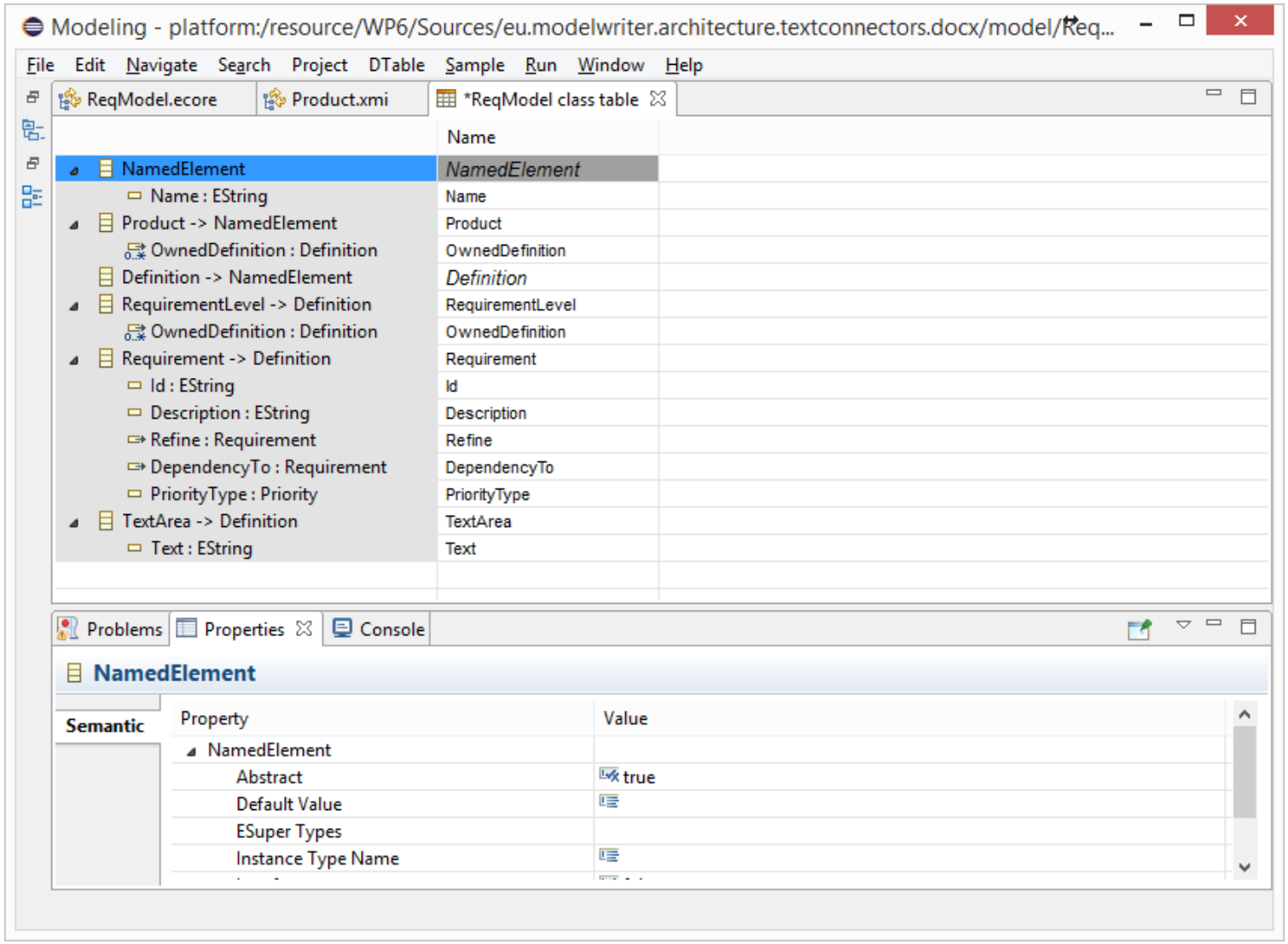
UML Modeling Languages



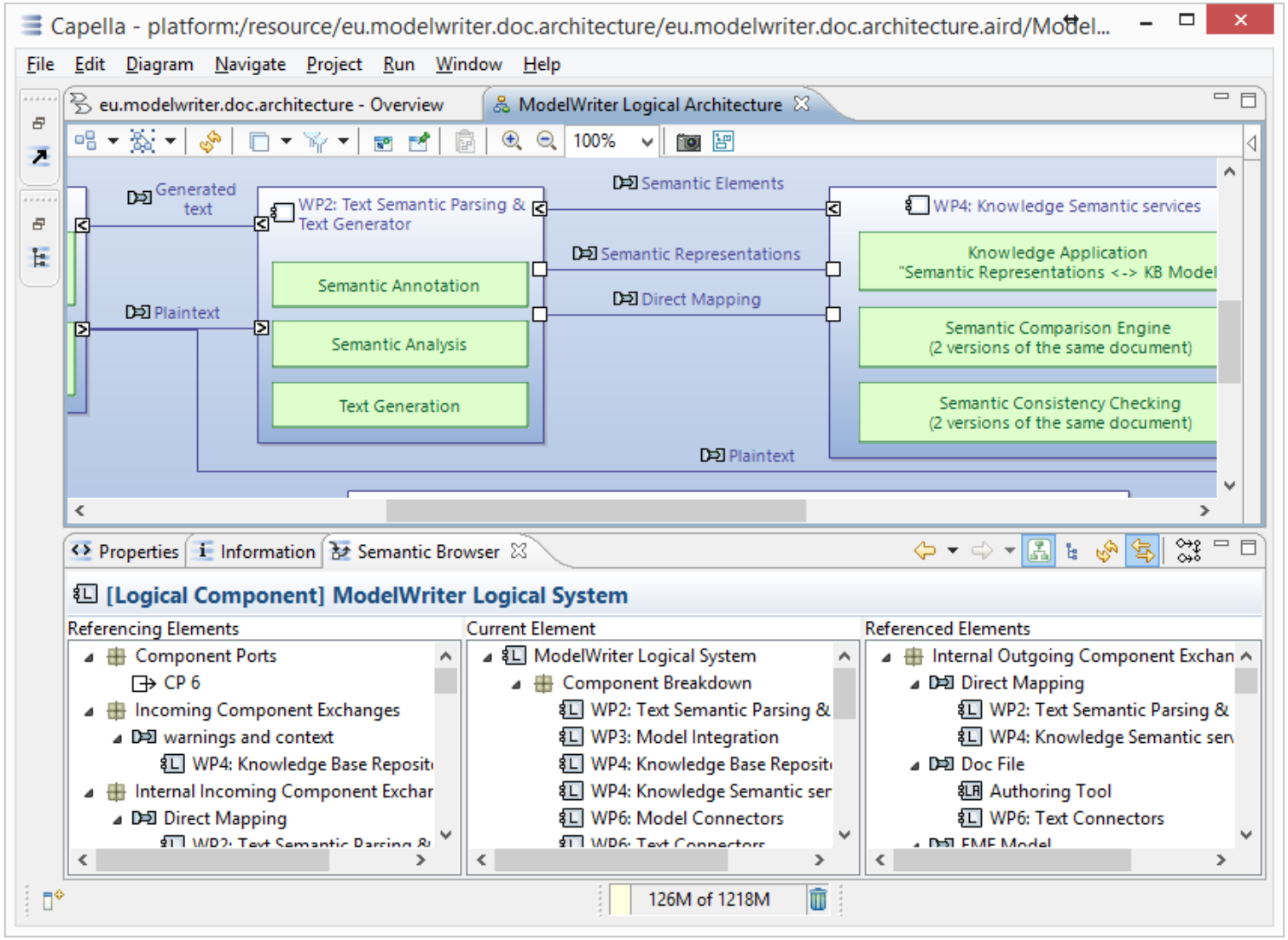
Everything is a model!
Eclipse Modeling Framework (EMF)



Everything is a model!
Tree-based or Tabular Representations

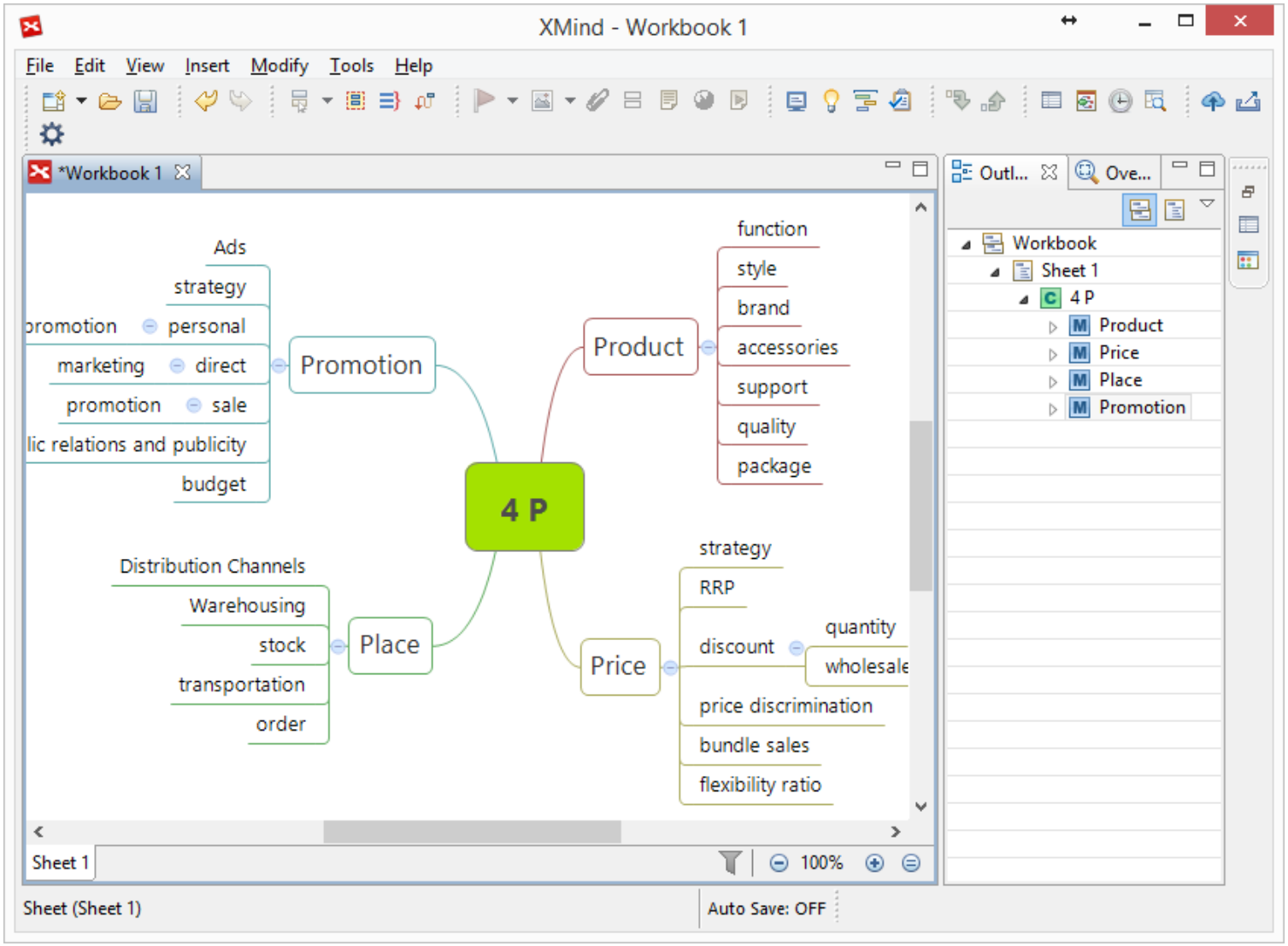


Everything is a model!
Software/System Architecture Design

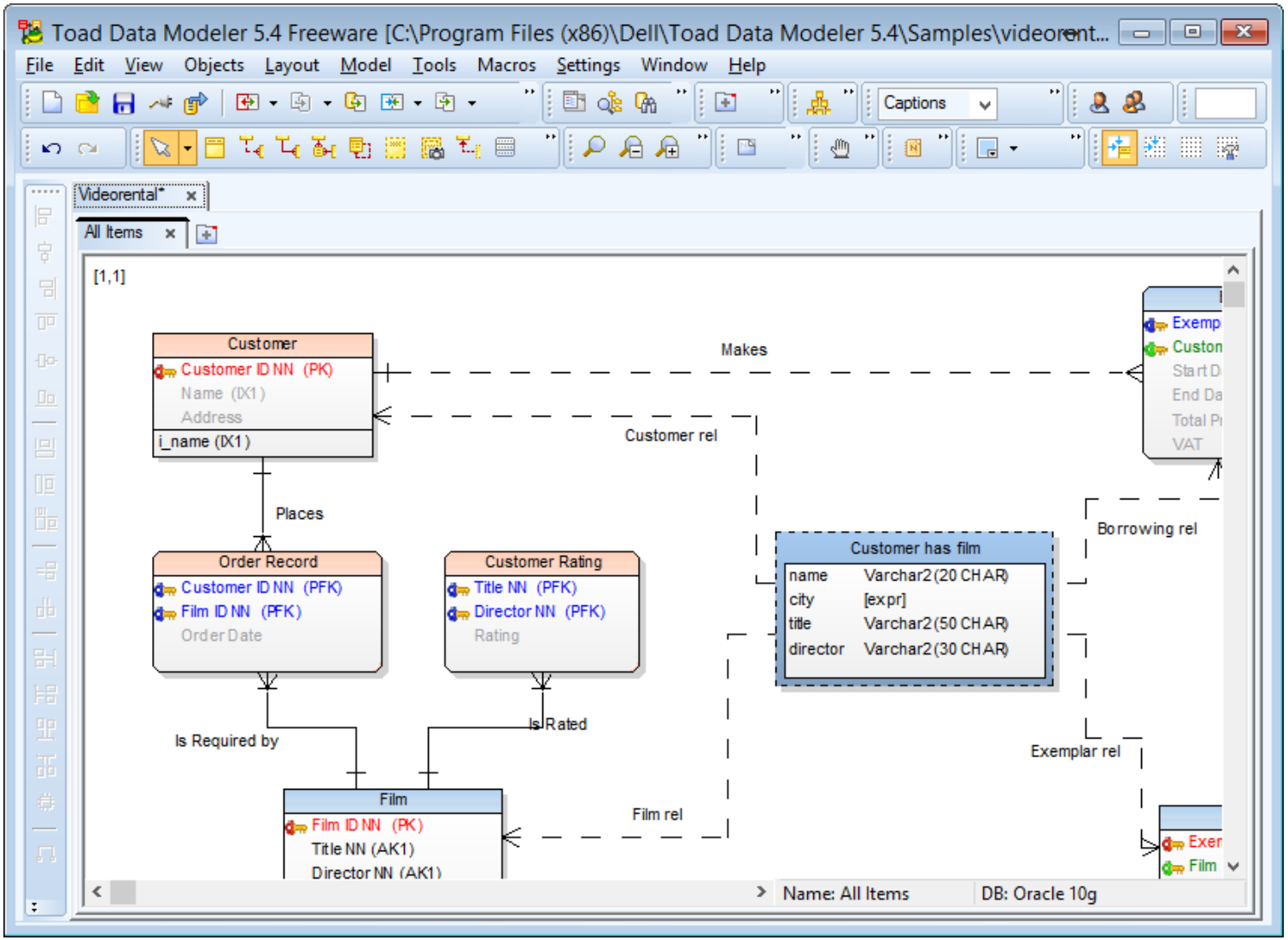


Everything is a model!

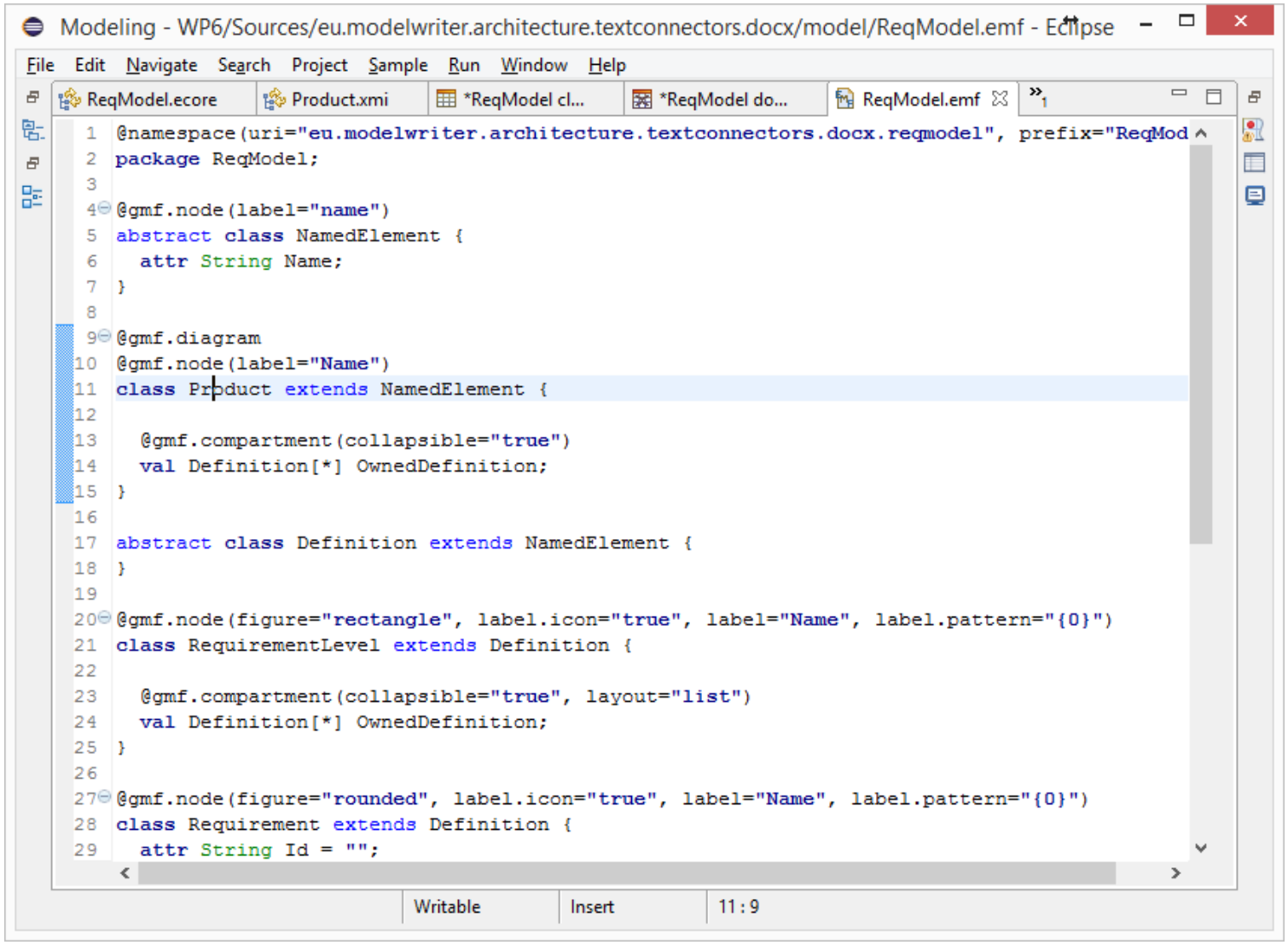
Topic Maps, Mind Maps, Vocabularies …



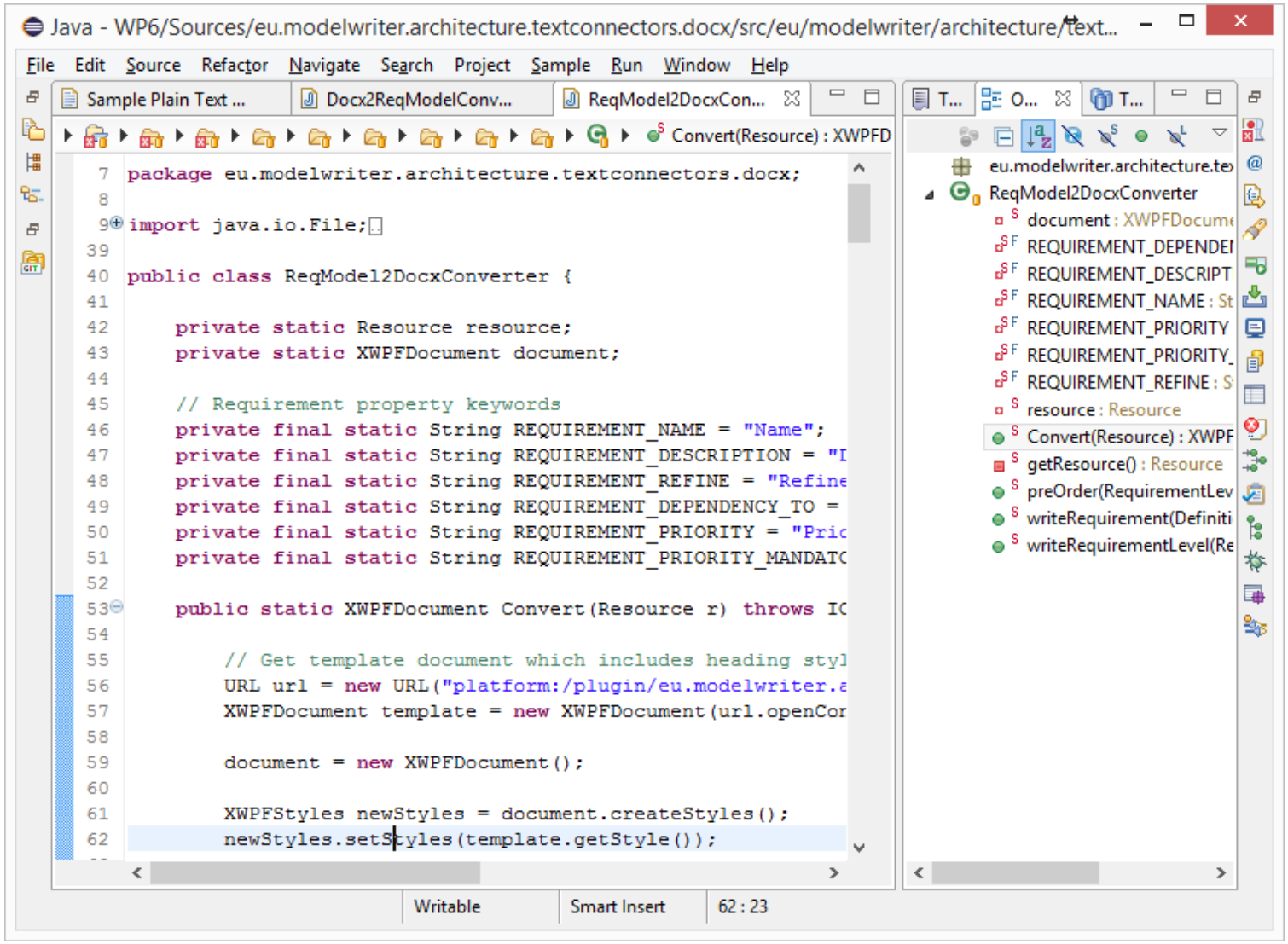
Everything is a model!
Databases (ER, IDEF1.x)



Everything is a model! (Textual Lang.)
Domain Specific Languages



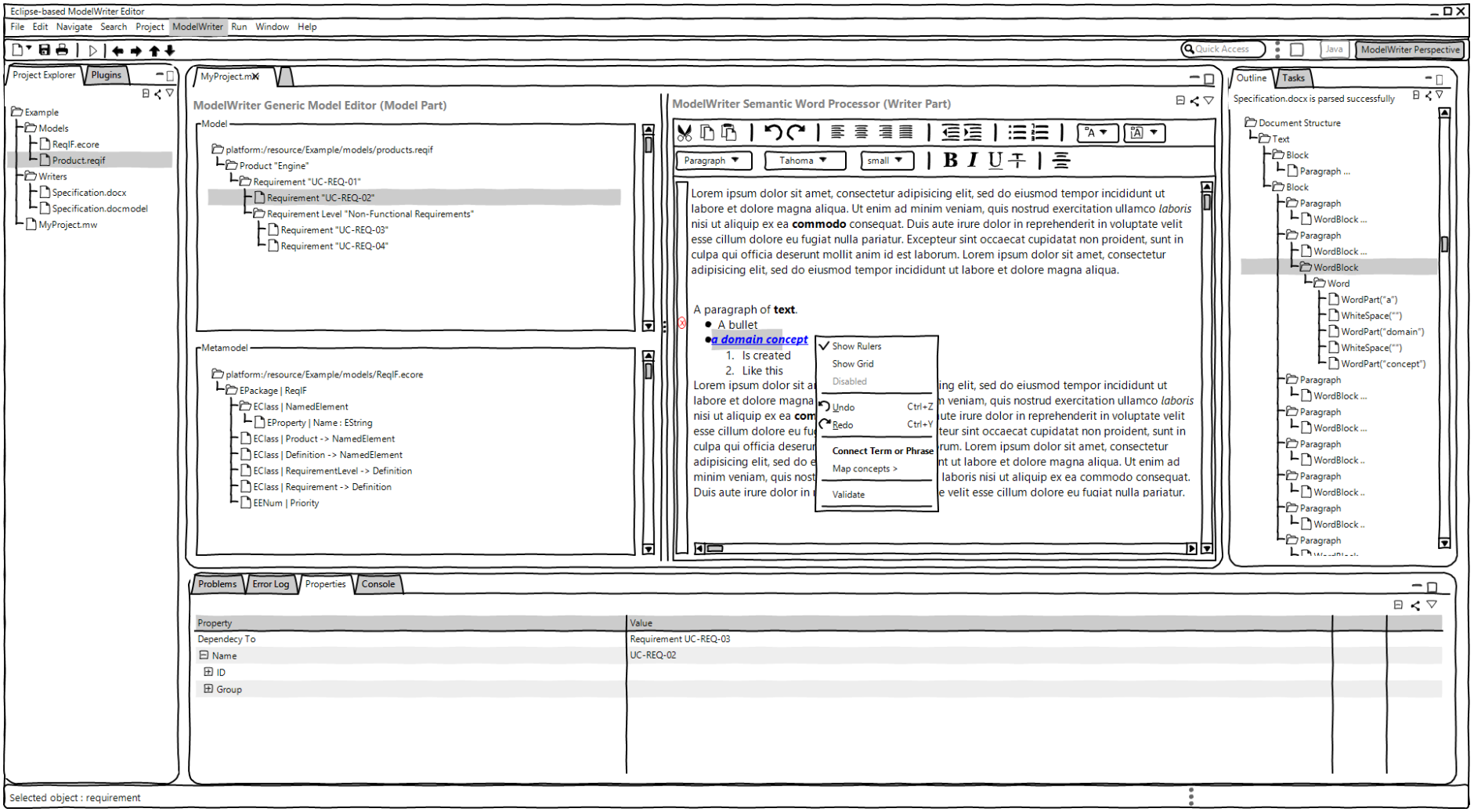
Everything is a model! (Java, C++, etc.)
Even Programing Languages (ASTs)



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

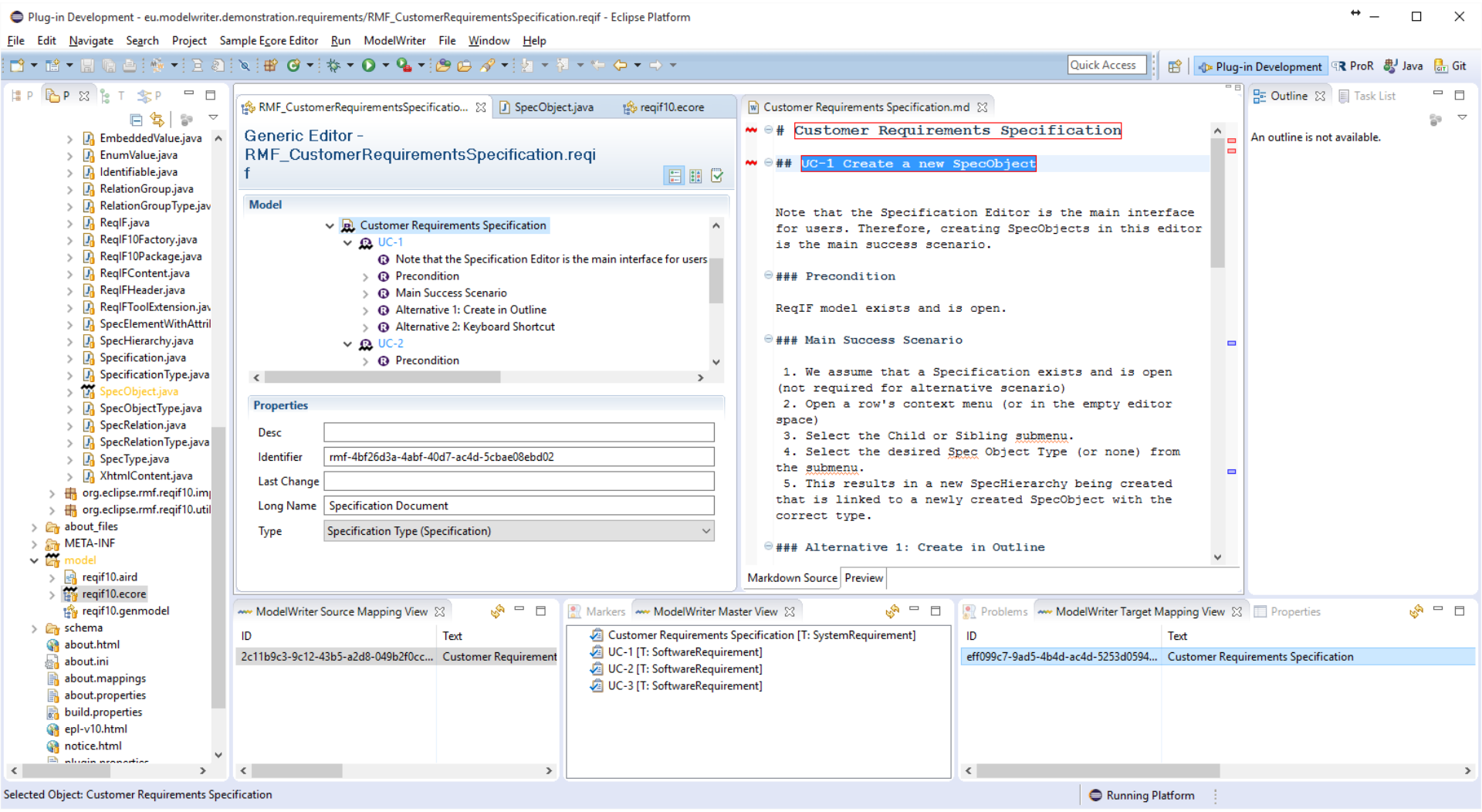
ITEA 3

ModelWriter – The Solution



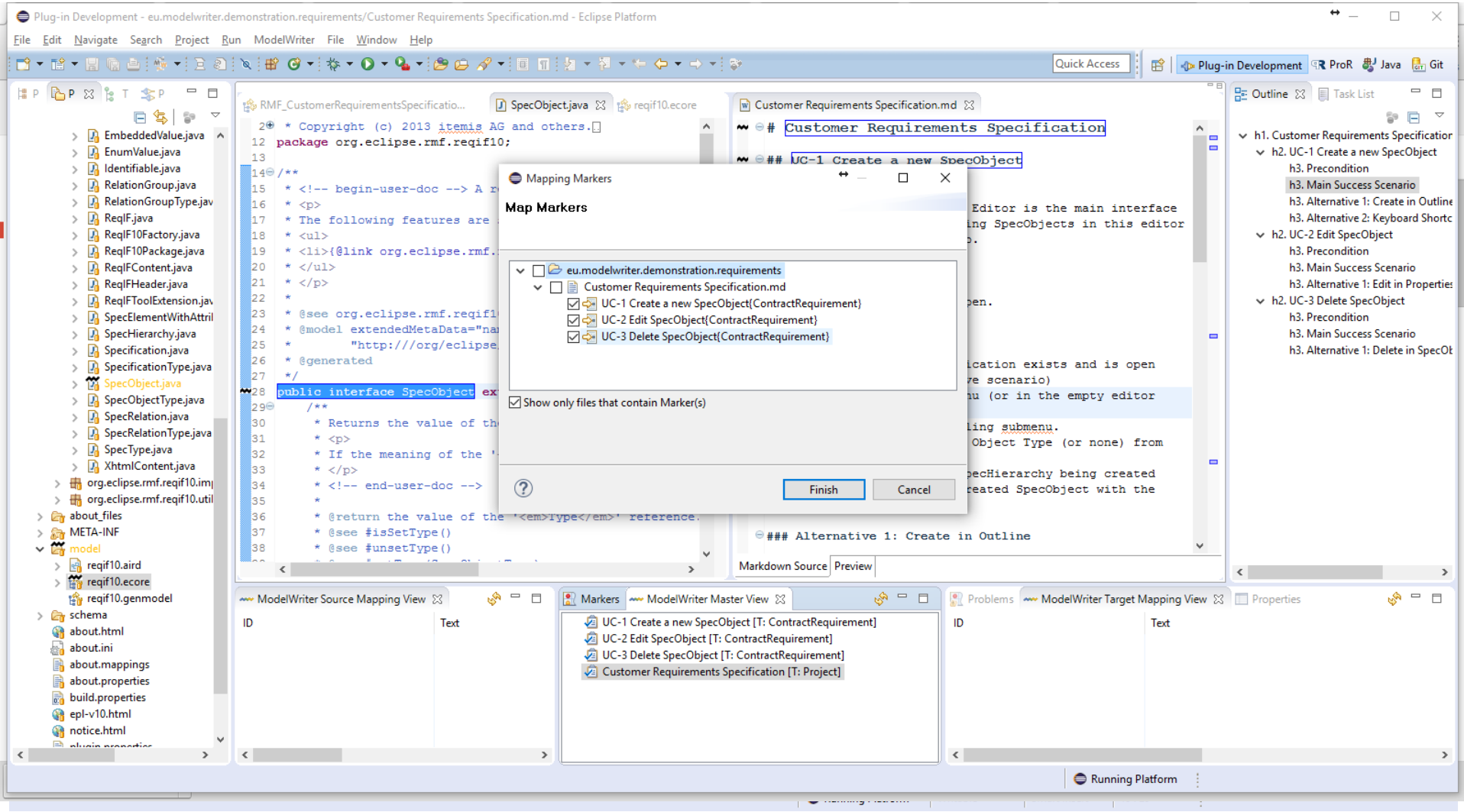
Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Capture



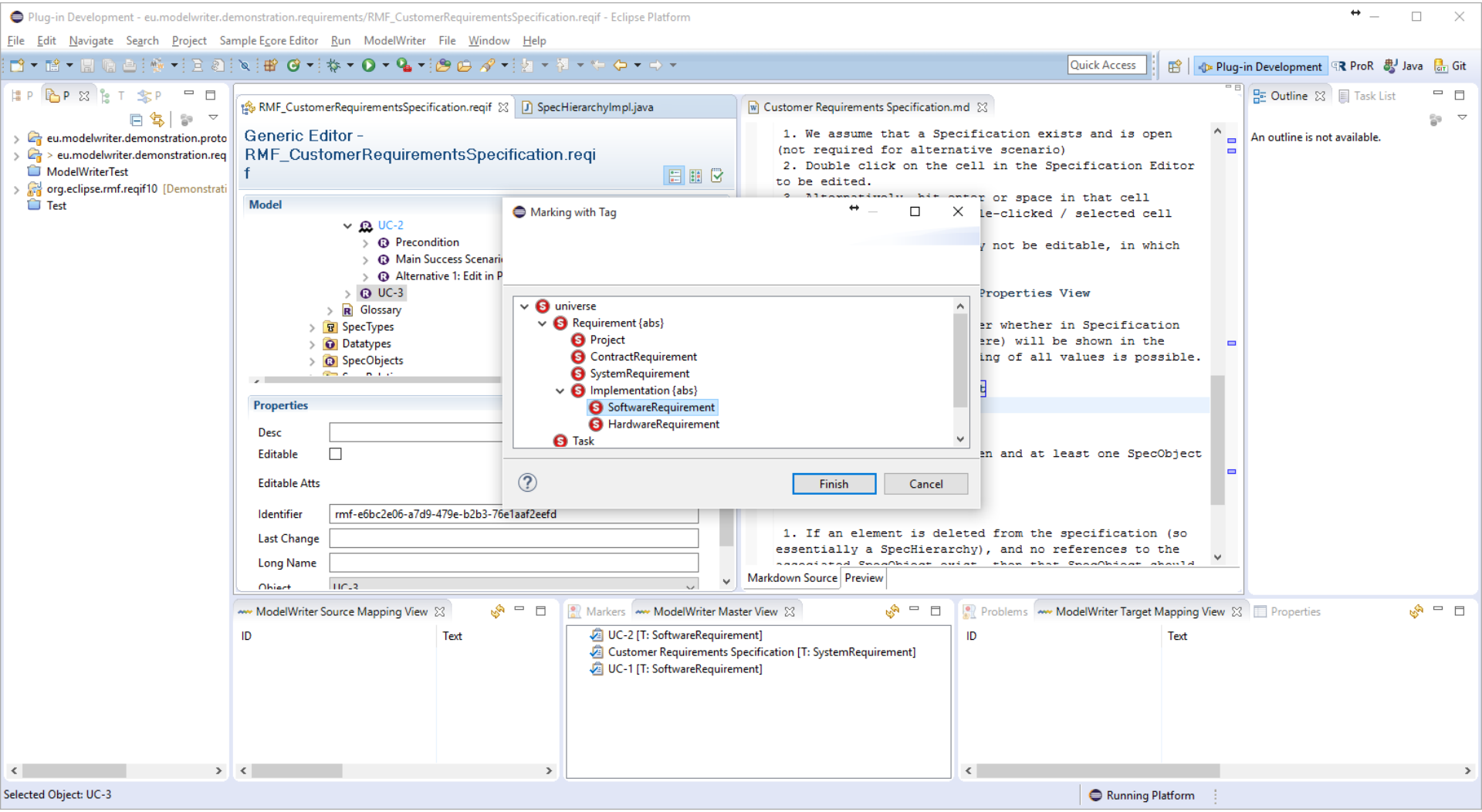
Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Capture



Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Capture

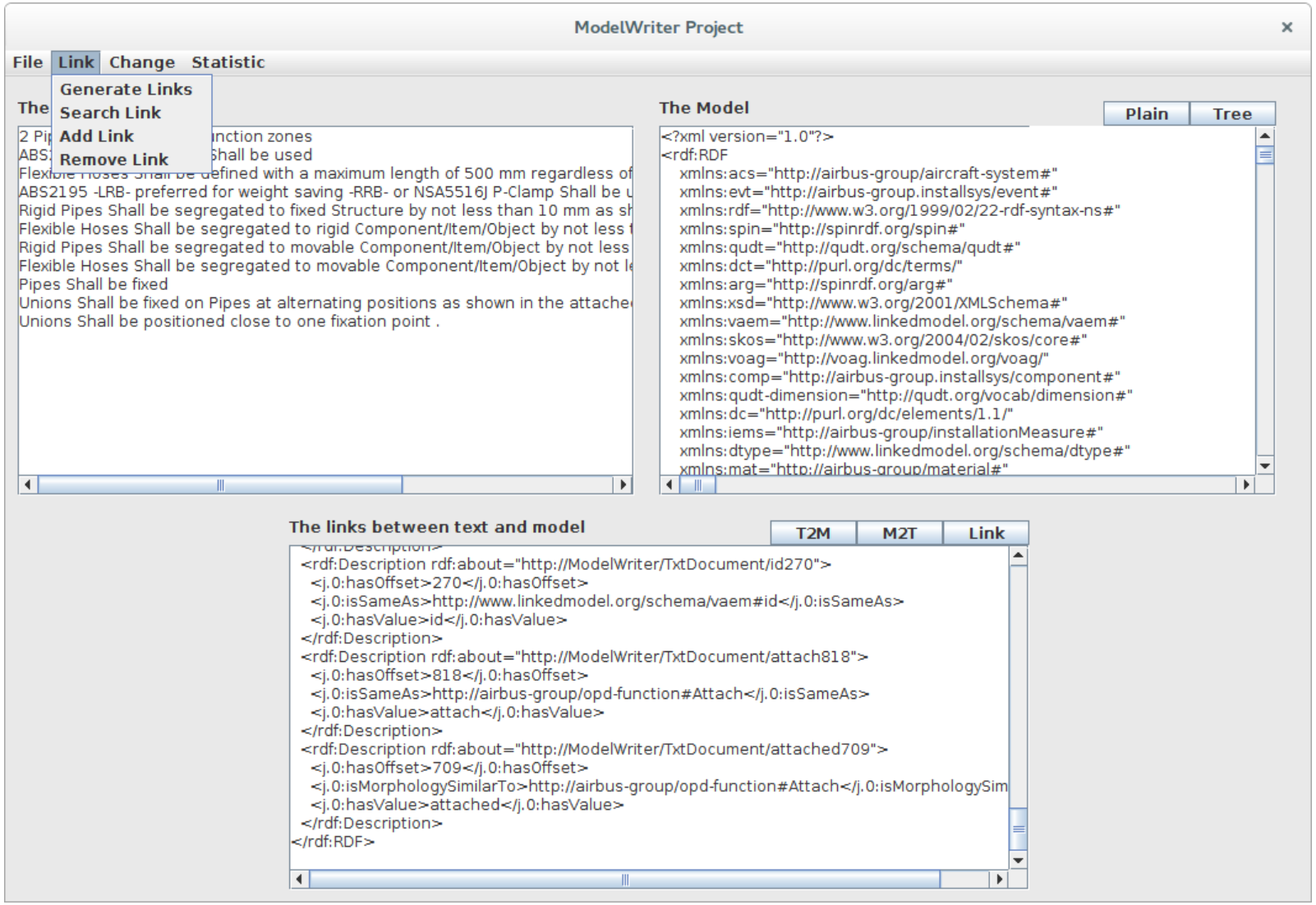


Text & Model-Synchronized Document Engineering Platform

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

ITEA 3

Solution – Knowledge Extraction



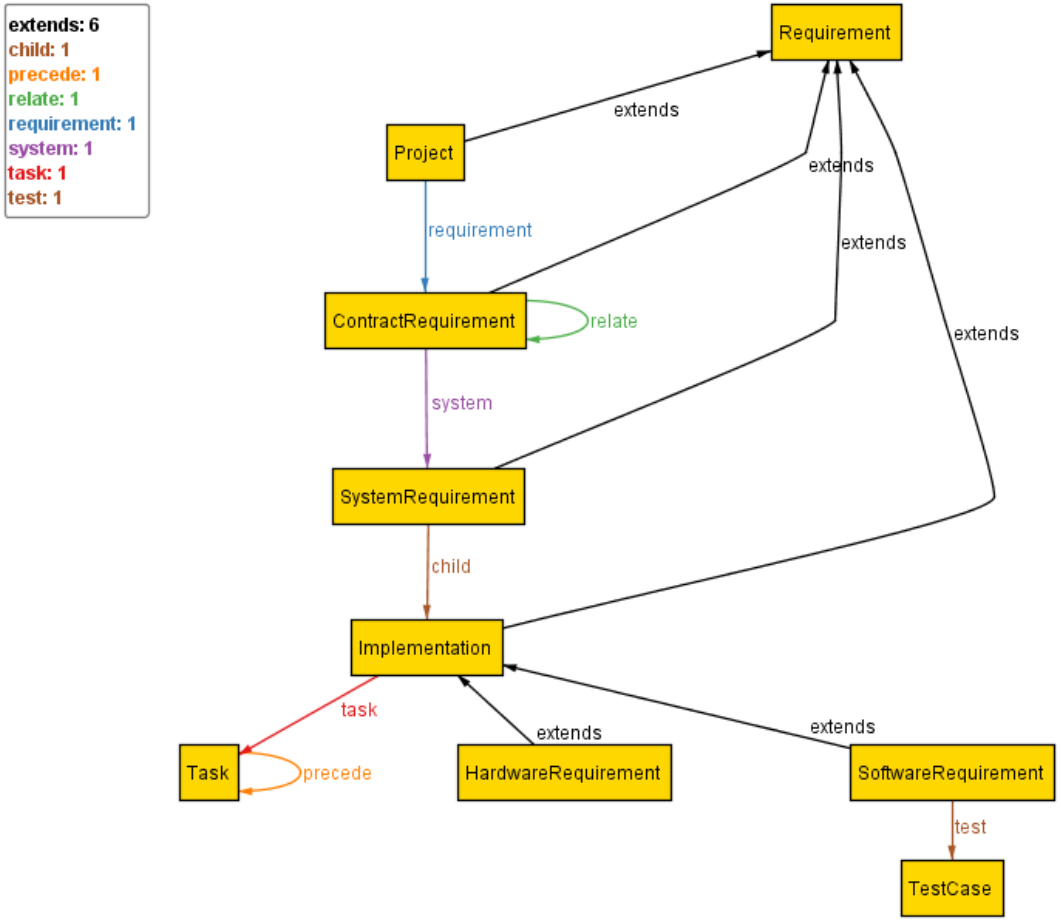
Text & Model-Synchronized Document Engineering Platform


Synchronization is maintained!

What about configuration/formalization of the platform?

ITEA 3

Configuration: Havelsan example



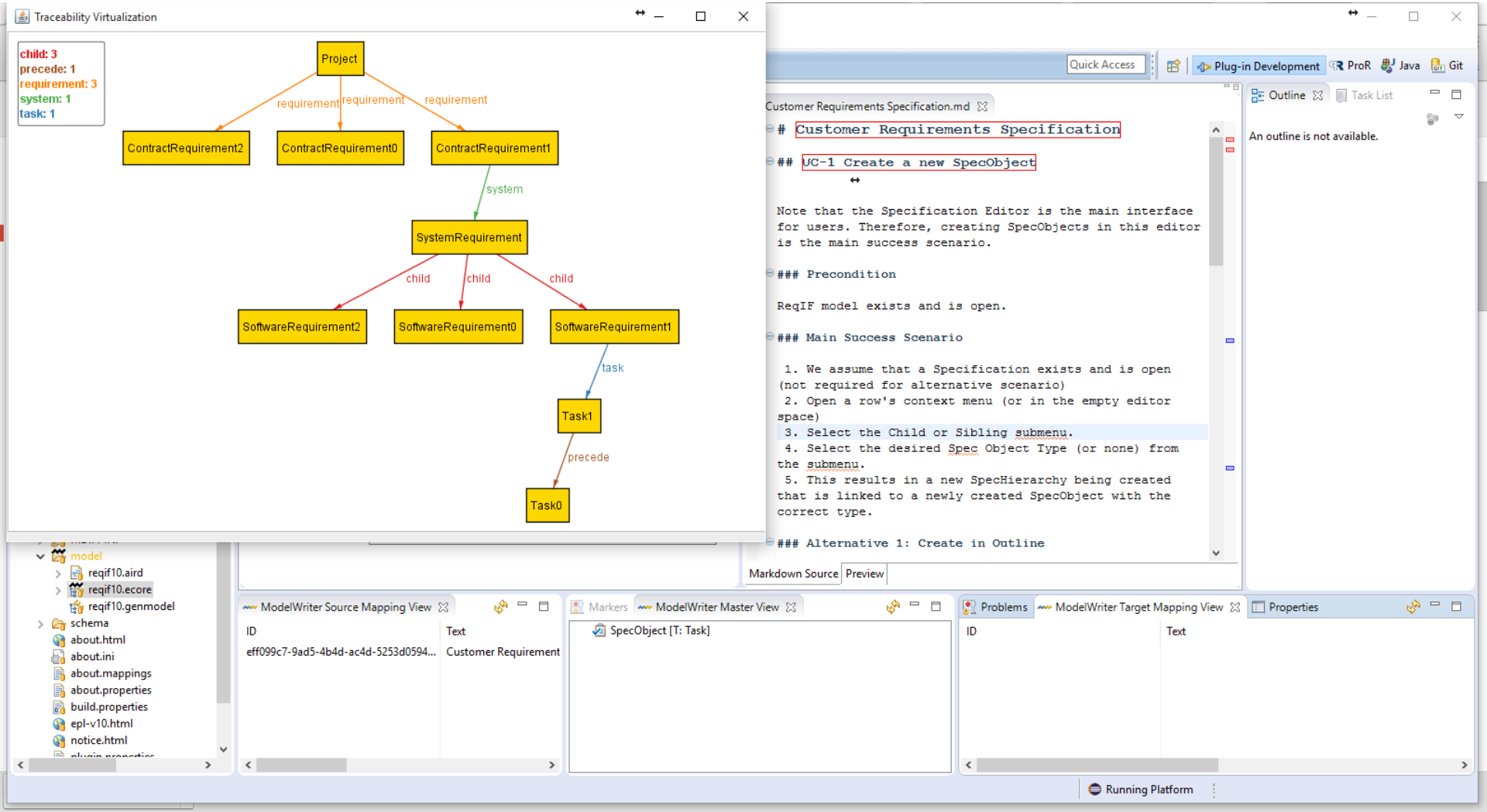

```
module Haveksan/Requirement

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

Is it possible to vizualize the trace links?

ITEA 3

Traceability: Havelsan example



A Formal Specification Model to configure the ModelWriter

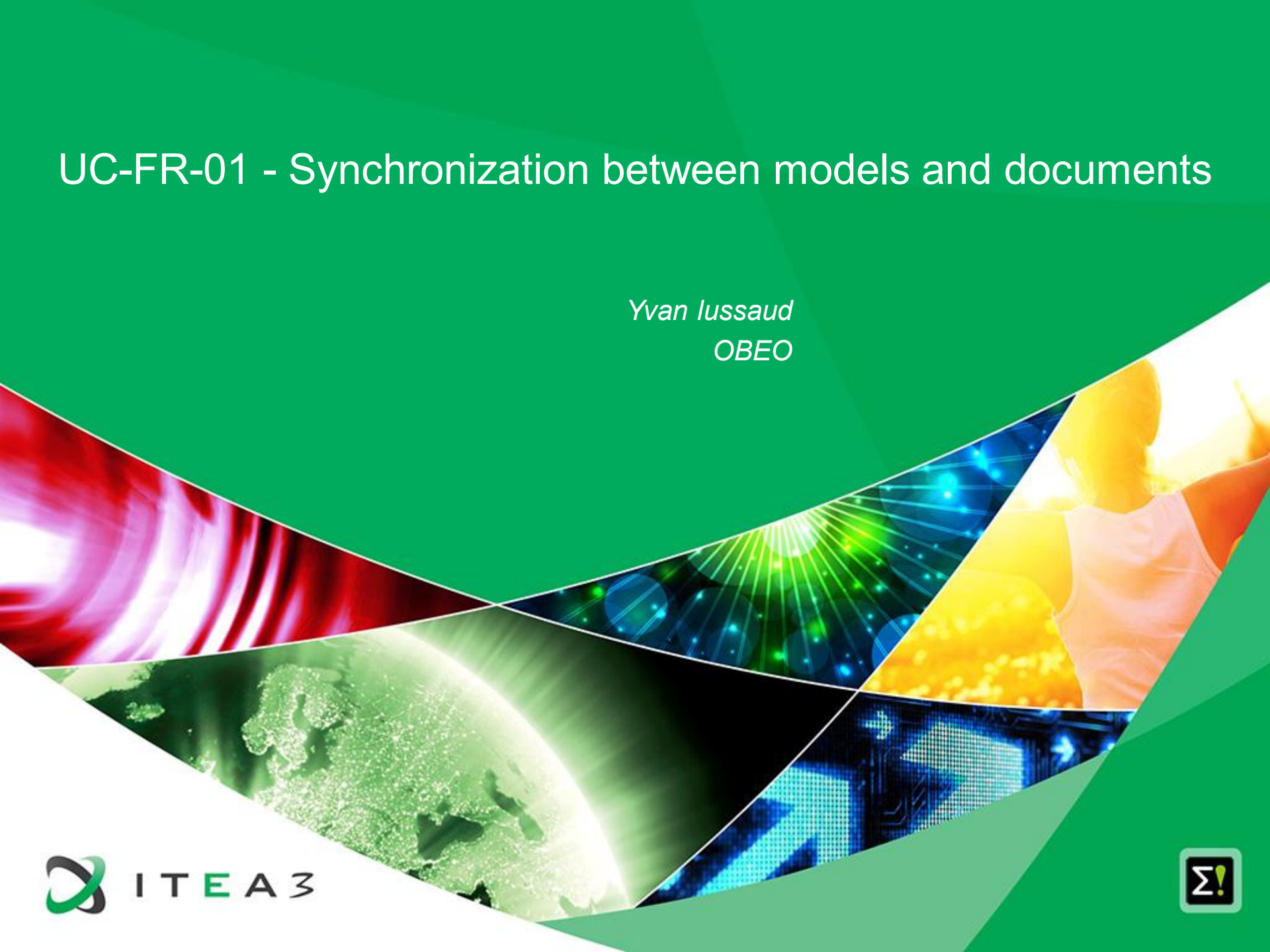
Thank you for your attention
We value your opinion and questions.

3 Progress on the Industrial Use Cases

Prof. Geylani Kardaş, Moderator
KoçSistem

UC-FR-01 - Synchronization between models and documents

Yvan Iussaud
OBEO



UC-FR-01
Development of Sirius

Artifacts
- Graphical Mapping DSL
- Source code
- Documentation (specifier, user)

Life cycle
- Release train Eclipse
- Dependencies

Team
- Core team
- External contributors

Methodology
- Code review
- Plan manual synchronization of artifacts

UC-FR-01
Problematic

Synchronization gap

- Development
- Artifacts knowledge is volatile

Validation

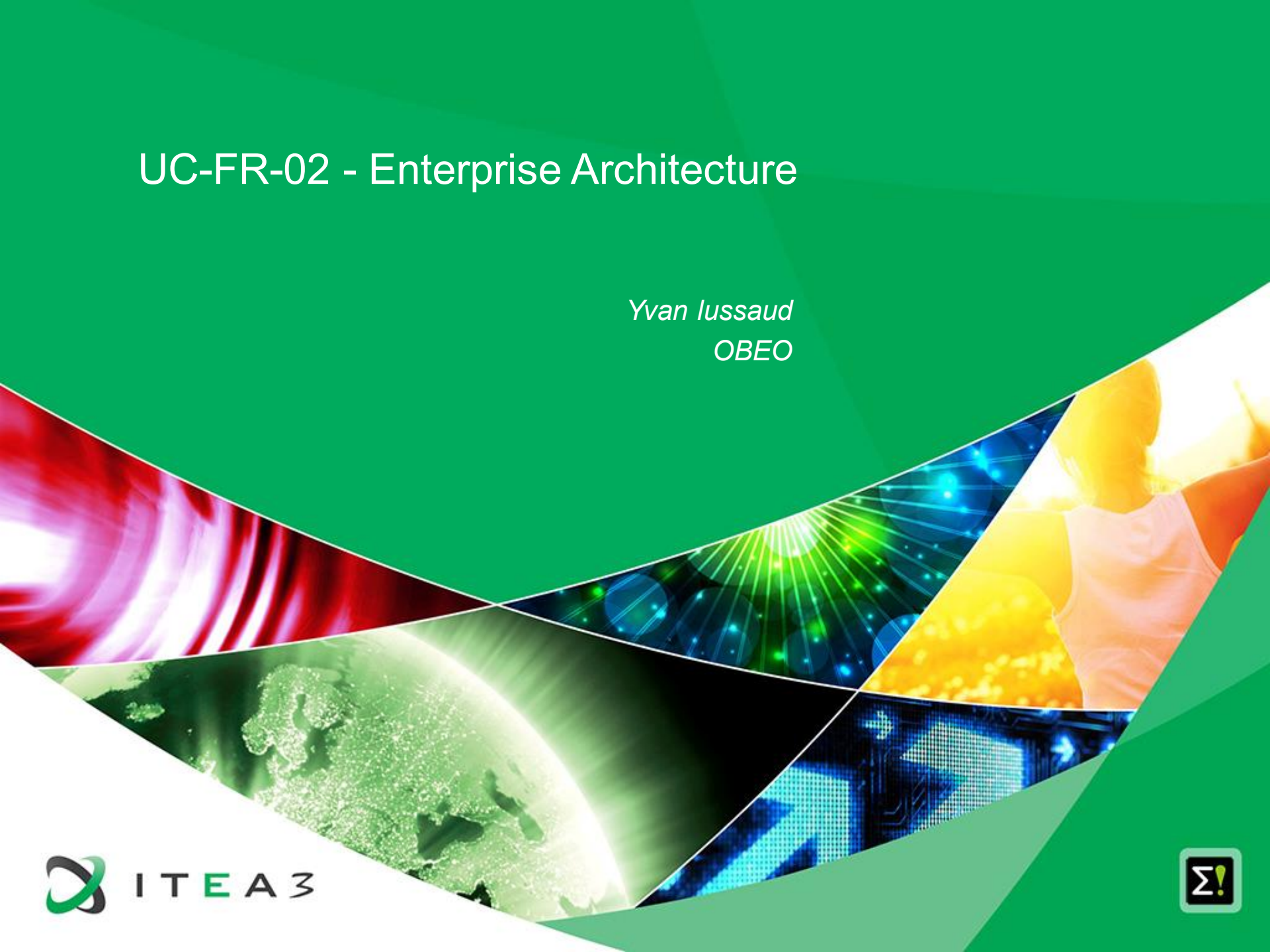
- Release
- Quality

UC-FR-01
Concern

- Mapping tooling
- Live validation feedback
- Fixing assistance
- Batch validation
- Contribute to Intent
- Internal validation of ModelWriter approach

UC-FR-02 - Enterprise Architecture

Yvan Iussaud
OBEO



UC-FR-02
Enterprise Architecture

Define As-Is state
- Import documents
- Consolidate the company model

Define To-Be state
- Identify possible scenario
- Evaluate possible scenario
- Modify the company model

Define trajectories
- Gap analysis
- Impact analysis
- Define milestones

UC-FR-02
Problematic

Documentation
- Keep track of sourced documents
- Keep representation and portal up-to-date

Synchronization gap
- Organizational changes
- Artifacts knowledge is volatile

Validation
- Produce documentation of organizational changes
- Quality

UC-FR-02
Concern

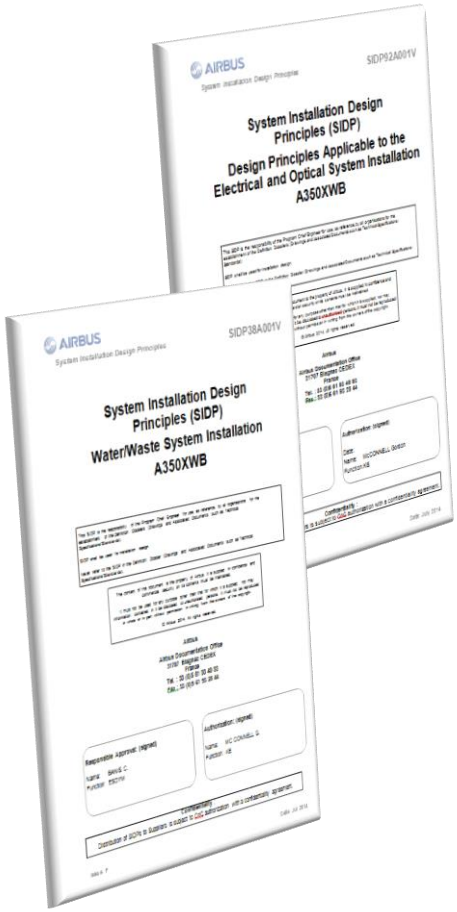
- Mapping tooling
- Live validation feedback
- Fixing assistance
- Batch validation
- Contribute to Intent
- Industrial use of ModelWriter tooling

UC-FR-03 - Synchronization of Regulation Documentation with a Design Rule Repository

Anne MONCEAUX
Airbus Group Innovation
Louis ROUCH, Ayhan MOHOVIC
Airbus SAS

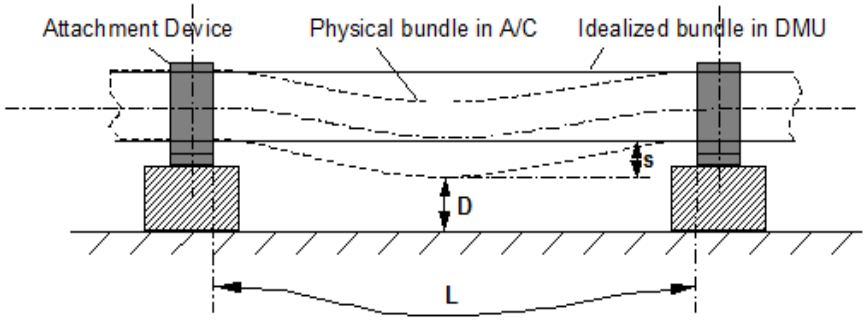
ITEA 3

SIDP: “System Installation Design Principles”



SIDP92A001V-A-784

For installation of optical and electrical harnesses additional clearance for sagging (s) shall be provided as detailed below:



s... Sagging of bundle (real behavior of physical bundle in A/C due to gravity, ageing, etc.)
D... Required Distance
L... Actual length of a bundle segment between two Attachment Points (as designed in DMU)

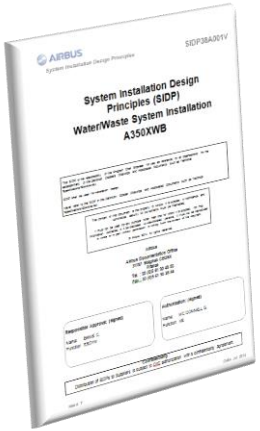
Figure 6: Sagging of bundles between attachment points

Note: Unless the bundle has a straight routing, L is bigger than the pitch between the Attachment Points.

66

Context and problem

- SIDP documents explain how to install the aircraft systems and attach them to the structure. They capitalize the best practices & proven technical solutions.
- SIDP are defined per ATA chapter (~functional domain) to be applied for each given A/C project: for example “A350 Electrical installation”



- SIPD are open to Extended Enterprise: installation tasks are performed by risk sharing partners.
- SIDP are living documents: during the aircraft development any new DP allowing to satisfy all targets/constraints can be added, assuming it is validated by Airbus dedicated committee.

Industrial high level needs

To improve SIDP creation, maintenance and consultation in order to:

- save costs by supporting DP consultation / retrieval in accordance to the needs
- avoid non-compliance of design caused by DP updating lead times
- keep traceability with upstream regulations and requirements and downstream design models

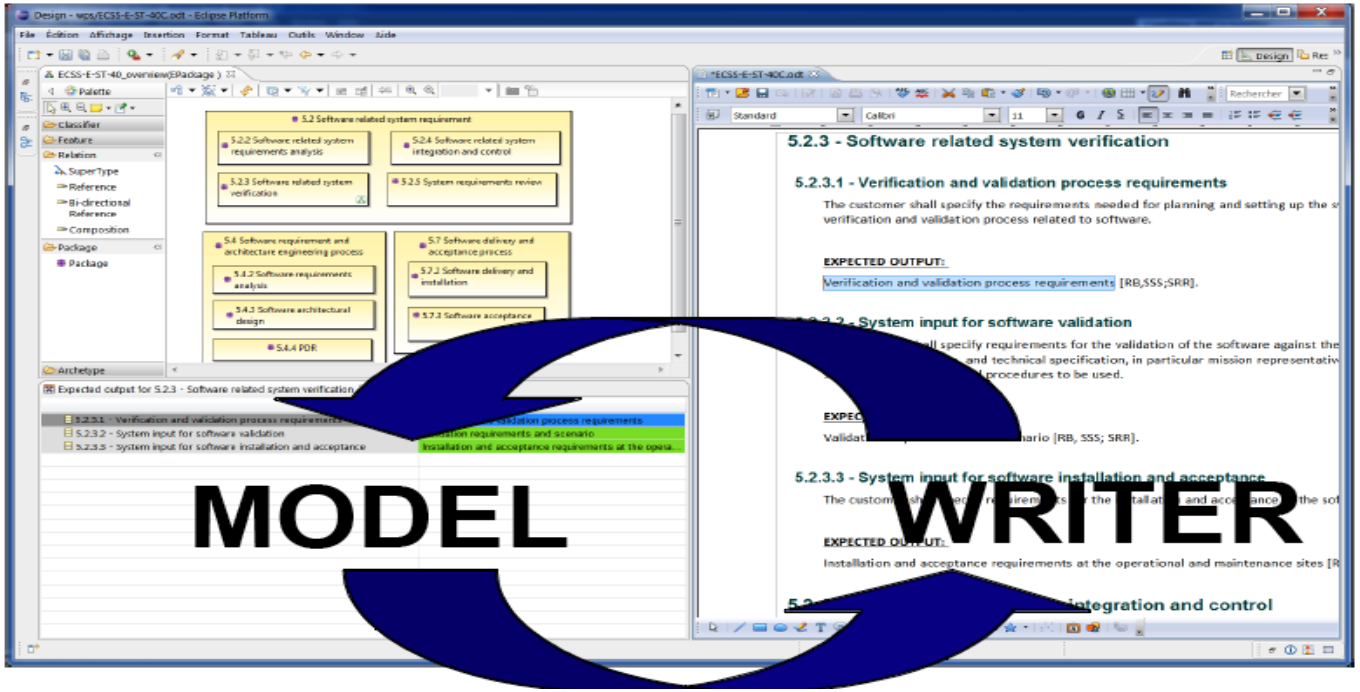
Synchronization of regulation documentation with a design rule repository

ITEA 3

UC-FR-03 Synchronization of regulation documentation with a design rule repository

Goal: synchronizing the SIDP database content with documentation

- Create links between text fragments & model elements (manual annotation?, semi-automatic?…)
- Manage consistent synchronization (manage changes)



UC-FR-03 Synchronization of regulation documentation with a design rule repository

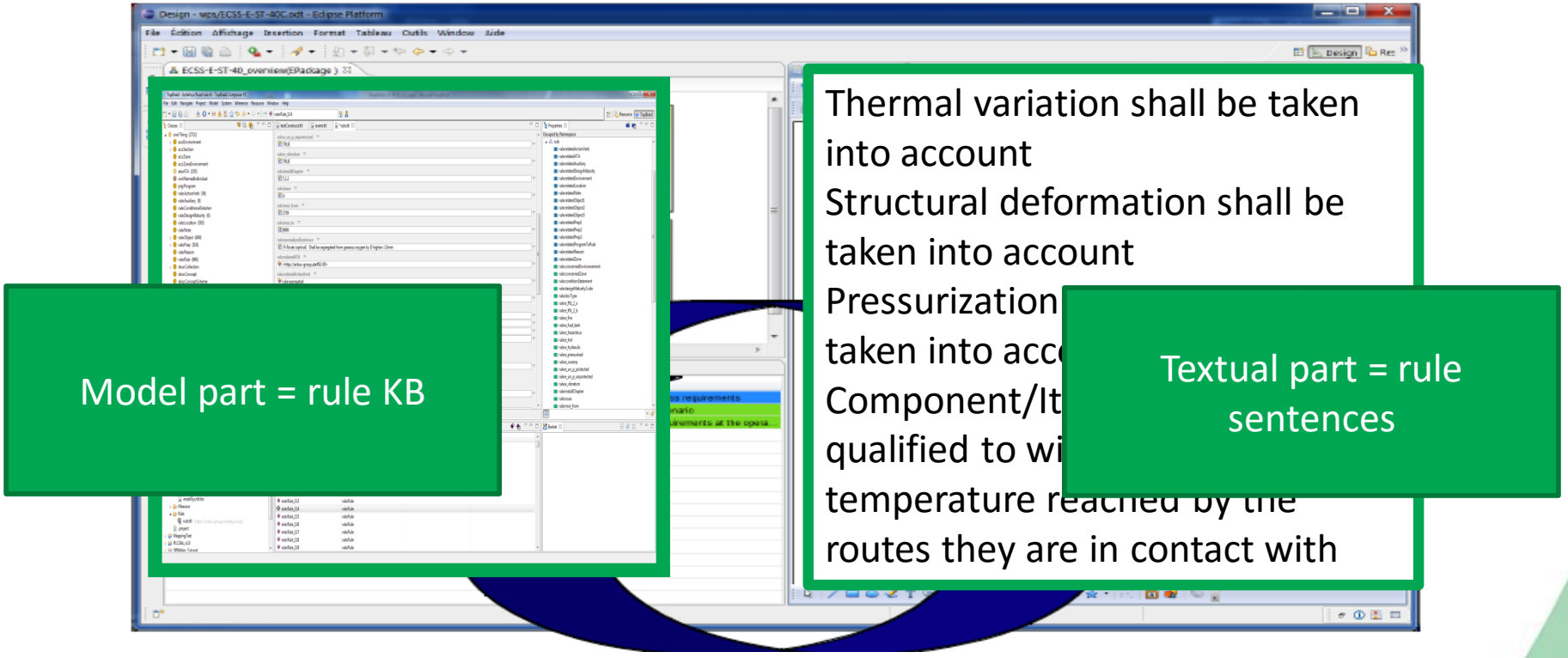
- Approach: limited case on Electrical Installation functional domain
- (Confidential Data - Non Disclosure Agreement finalized in June 2015).

- Text
  - 1 document: SIDP ATA 92
  - In our industrial context SIDP are edited using MSWord

- Models
  - An OWL model is built that reflects the DB schema: “Rule ontology” (30 classes, 35 object properties and 54 data properties)
  - Automatic population mechanism of the model from a csv BD export produces the KB (45781 triples)

UC-FR-03 Synchronization of regulation documentation with a design rule repository

Status 1st iteration: synchronizing the SIDP KB content and text part



UC-FR-04 - Production of a Context Specific Design Document

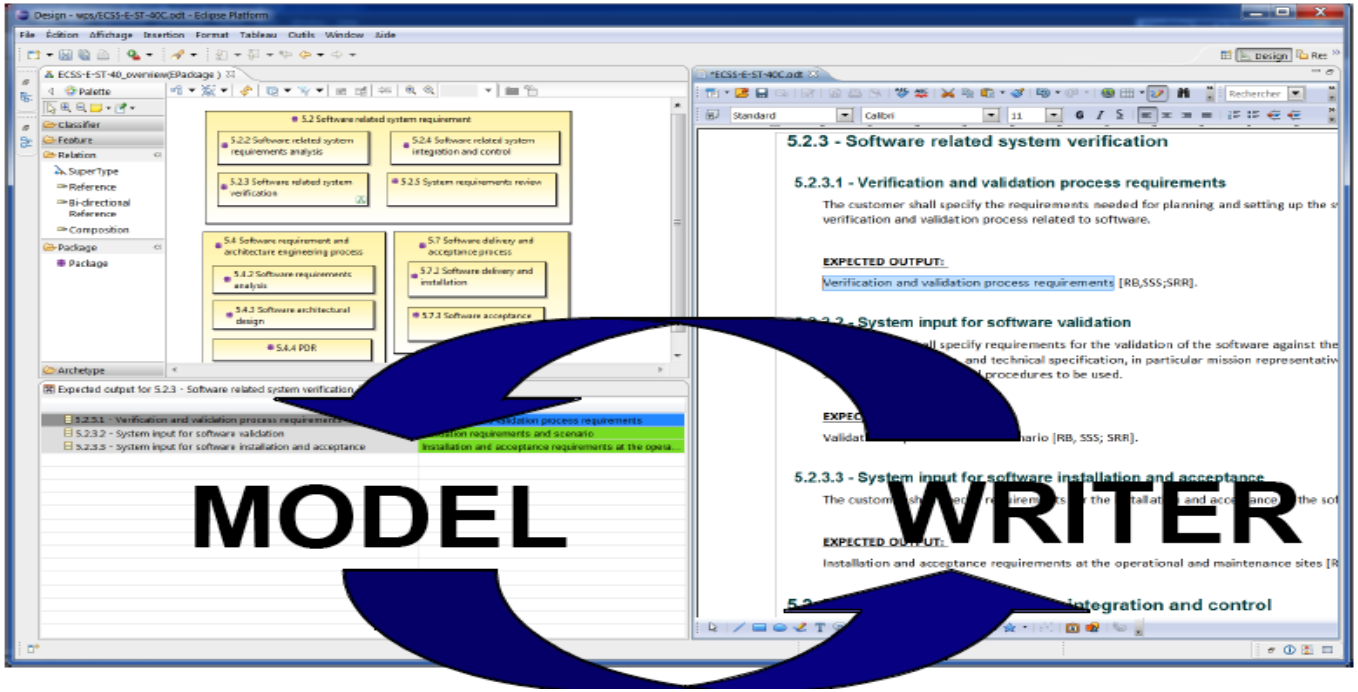
Anne MONCEAUX
Airbus Group Innovation
Louis ROUCH, Ayhan MOHOVIC
Airbus SAS

ITEA 3

UC-FR-04 Production of a context specific design document

Goal: producing document according to usage “needs”

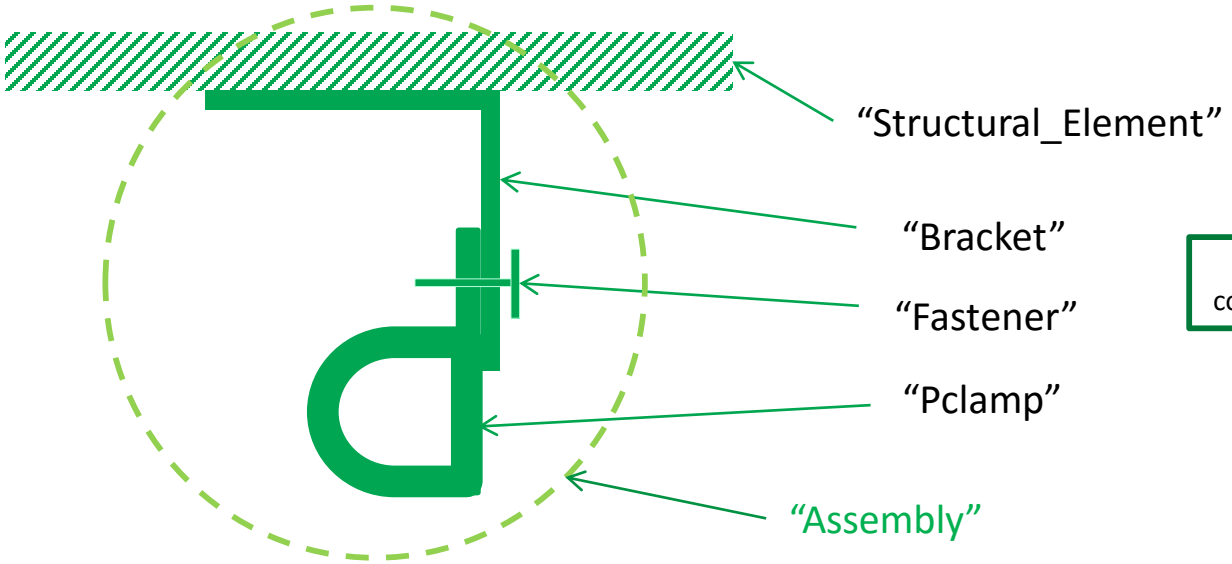
- Produce “filtered” document with subset of Design Principle textual elements according to usage “needs”

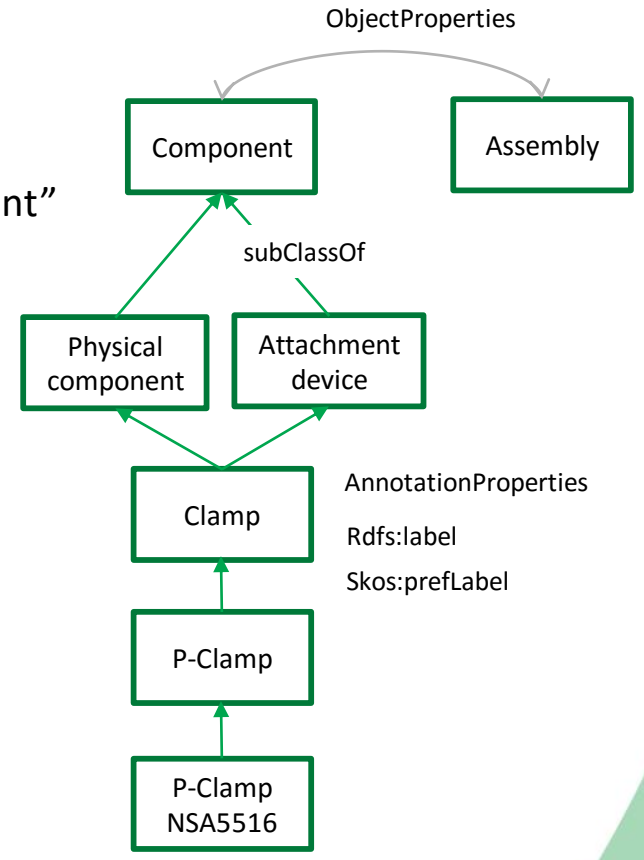


UC-FR-04 Production of a context specific design document

- Approach:
  - use case Electrical Installation functional domain
  - Confidential Data - Non Disclosure Agreement finalized in June 2015
  - Use model elements to retrieve relevant Design Principles
- Text
  - 1 document: SIDP ATA 92
  - In our industrial context SIDP are edited using MSWord
- Models
  - The previous Rule KB (populated Rule ontology)
  - Component ontology (476 classes, 21 ObjectProperties and 35 DataProperties)

Component classes taxonomy

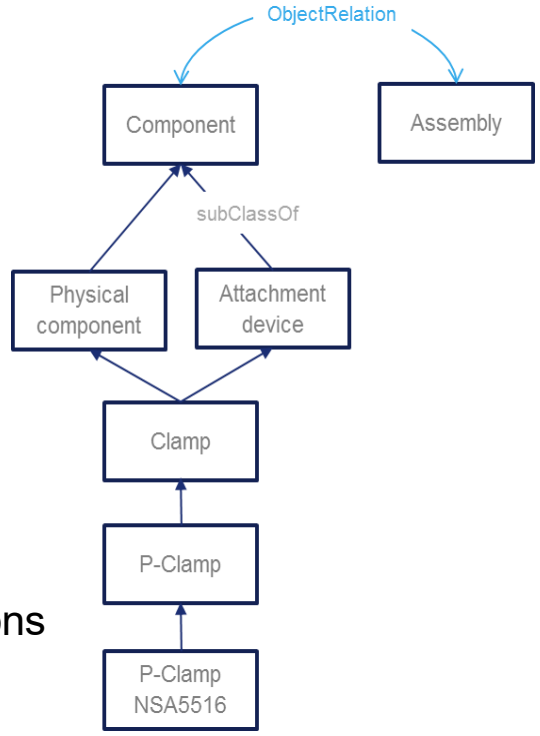




Component classes taxonomy

"P-clamp NSA5516 can be fixed on X with Y"

"Physical component" "Standard reference"

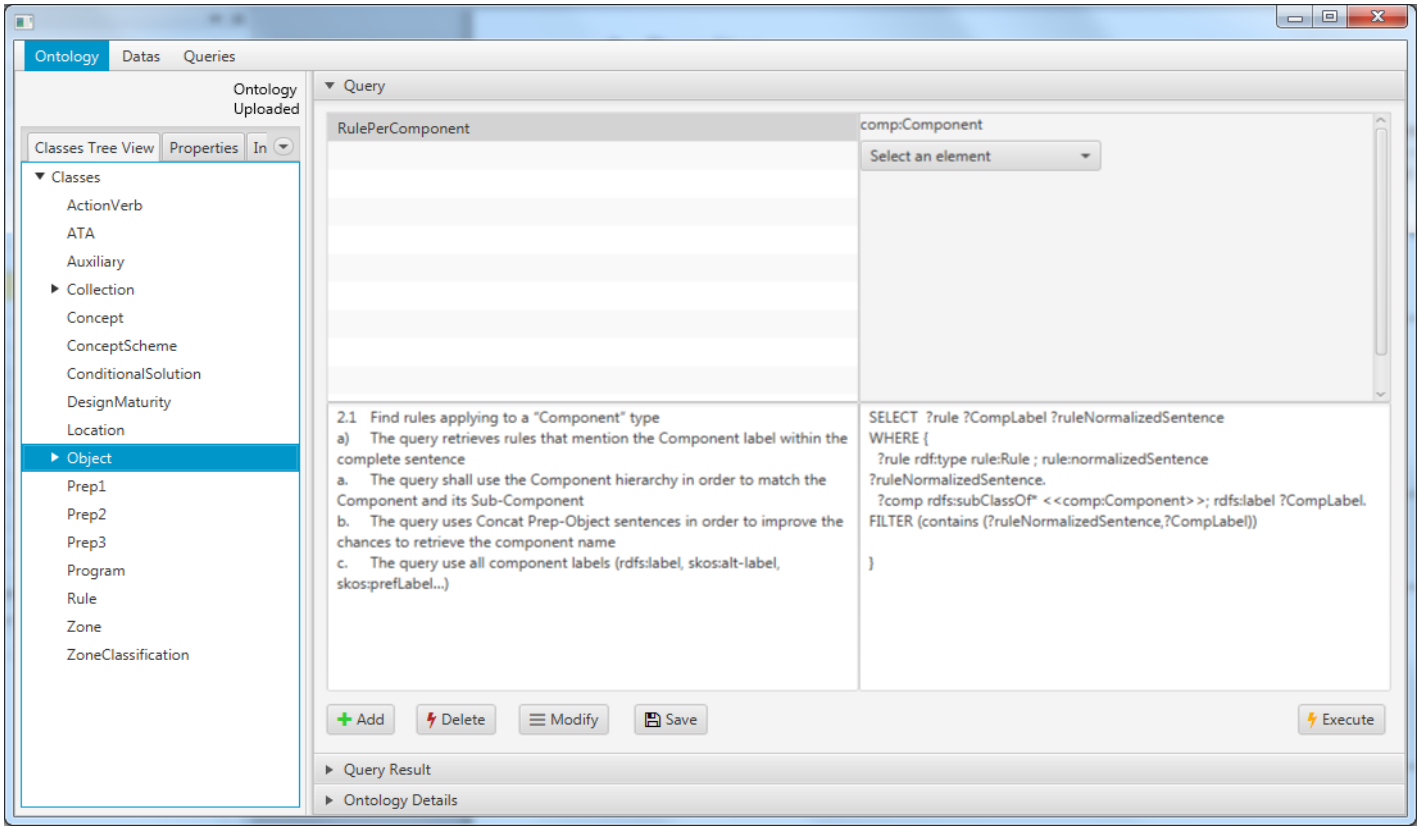


- NLP Parsing uses this taxonomy. Labels + assumptions such as a physical component may be referred to using its name or its reference or both concatenated
- Inference rule: a rule applying to a component type (Attachment device) applies to its subtypes (P-clamp)
- Document display : when searching rules applying to a component type (P-Camp) → retrieve and display rules applying to super-types

SPARQL queries

Status : model based queries specification

- Preliminary study with Loria (Text to RDF)
- 1 internship on RDFizer and Query management



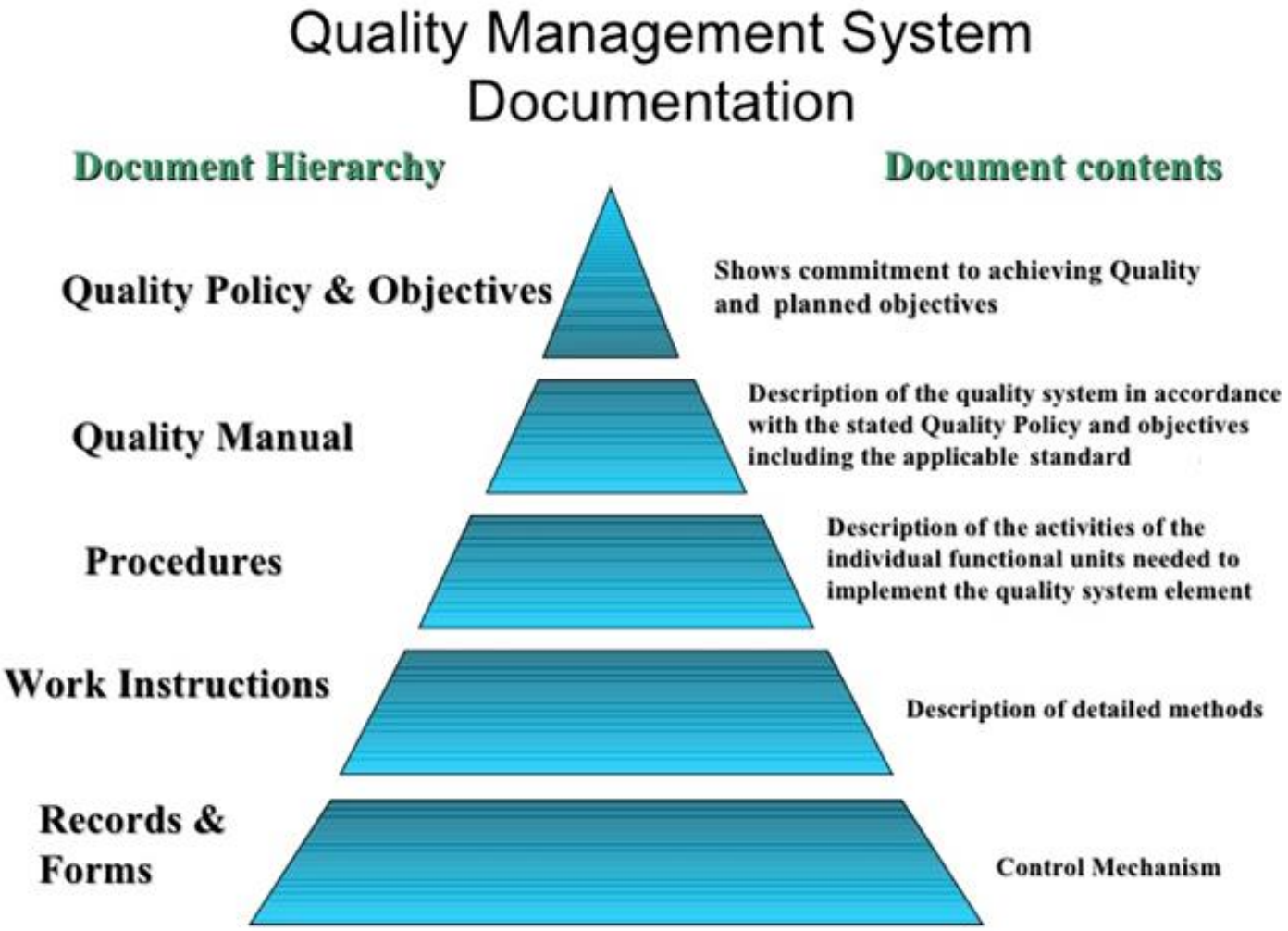
UC-TR-01 - Documents of Quality Assurance Department

Ersan GÜRDOĞAN
HISBIM

T1.4 – Industrial Use Cases for Turkish Consortium

UC-TR-01	Documents of Quality Assurance Department
Description	To create faster and more accurate the forms that are used in quality control progress and trigger related forms (re-work form triggering, revision needed, approved, rejected etc.)
Actors	Quality Managers, Quality Measurement Specialists, Quality Control Personnel, Quality Auditors, Production Crews
Assumptions	Quality control measurement units are standart Rejected and Approved products forms are standart Quality Certification standarts are always applied

TR-UC-01 Documents of Quality Assurance Department



TR-UC-01 Documents of Quality Assurance Department

What we expect ModelWriter about QDMS:

- Quality documents and forms are mostly too generic and similar. However some customers and users would like to customized documents. We would like to develop ModelWriter system could automatically generated QDMS related docs and forms with respect to Quality Test results.
- With sharing utility QDMS documents will be send to company managers, quality department authorised personnel, related customers' responsibles and suppliers2 responsibles if manufactured part comes from subcontractor.

What we have done until today about QDMS use case:

- Customer segmentation done, customer database and data relations are ready for ModelWriter
- Suppliers information are acqisied from database, ready to integrate ModelWriter system
- Models of Quality Documents and forms have been completed

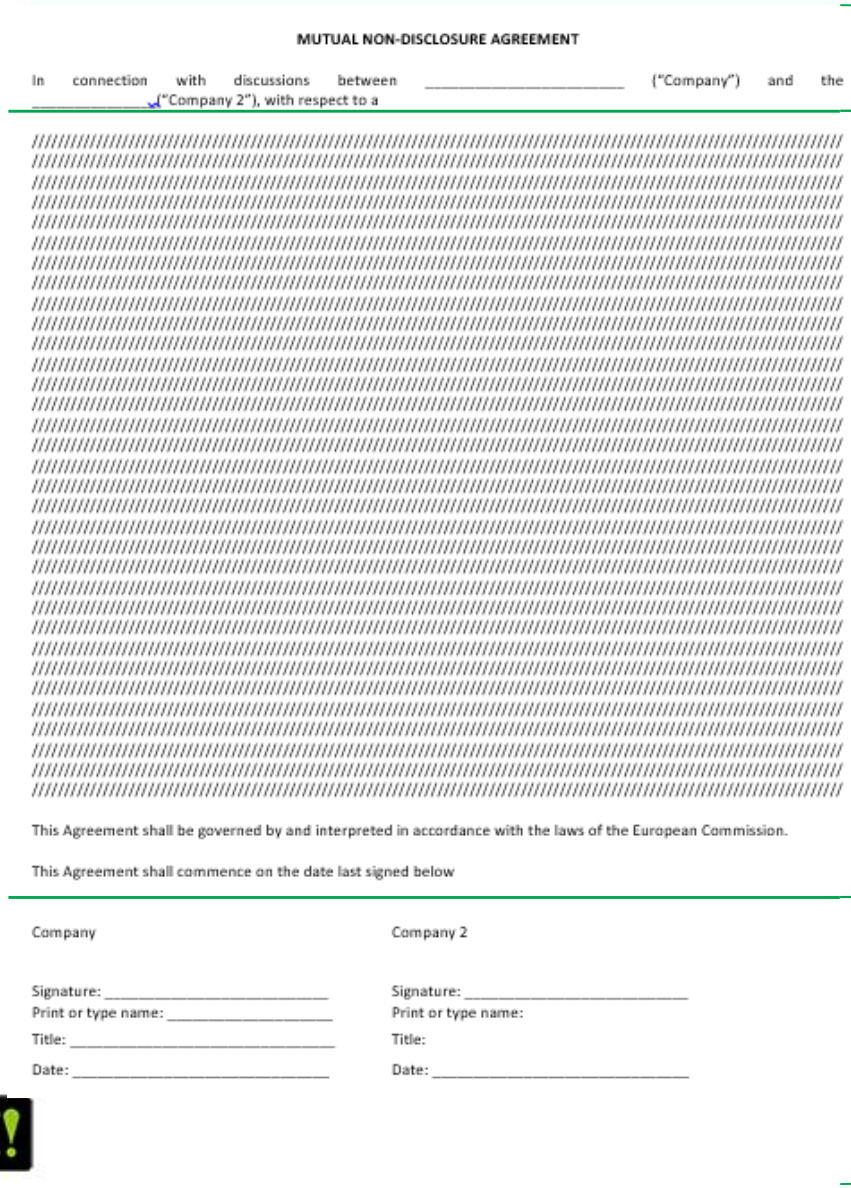
UC-TR-02 - Non-Disclosure Agreements

Ersan GÜRDOĞAN
HISBIM

T1.4 – Industrial Use Cases for Turkish Consortium

UC-TR-02	Non-Disclosure Agreements
Version	V1.0.0 dated 15-Nov-2014
Description	Non-Disclosure Agreement (NDA), also known as a confidentiality agreement (CA), confidential disclosure agreement (CDA), proprietary information agreement (PIA), or secrecy agreement (SA), is a legal contract between at least two parties that outlines confidential material, knowledge, or information that the parties wish to share with one another for certain purposes, but wish to restrict access to or by third parties. It is a contract through which the parties agree not to disclose information covered by the agreement. An NDA creates a confidential relationship between the parties to protect any type of confidential and proprietary information or trade secrets. As such, an NDA protects nonpublic business information. NDAs are commonly signed when two companies, individuals, or other entities (such as partnerships, societies, etc.) are considering doing business and need to understand the processes used in each other's business for the purpose of evaluating the potential business relationship
Actors	Responsible/Authorised personnel in both parties.

TR-UC-02 Non-Disclosure Agreements



MUTUAL NON-DISCLOSURE AGREEMENT

In connection with discussions between _____ ("Company") and the _____ ("Company 2"), with respect to a

This Agreement shall be governed by and interpreted in accordance with the laws of the European Commission.

This Agreement shall commence on the date last signed below

Company

Signature: _____
Print or type name: _____
Title: _____
Date: _____

Company 2

Signature: _____
Print or type name:
Title:
Date: _____

Company Information of Parties who sign NDA

Items of NDA with respect to contracted project(s) between parties, project related information such date of document, content of project etc.

Responsible names and their signatures from each party

TR-UC-02 Non-Disclosure Agreements

What we expect ModelWriter about NDAs:

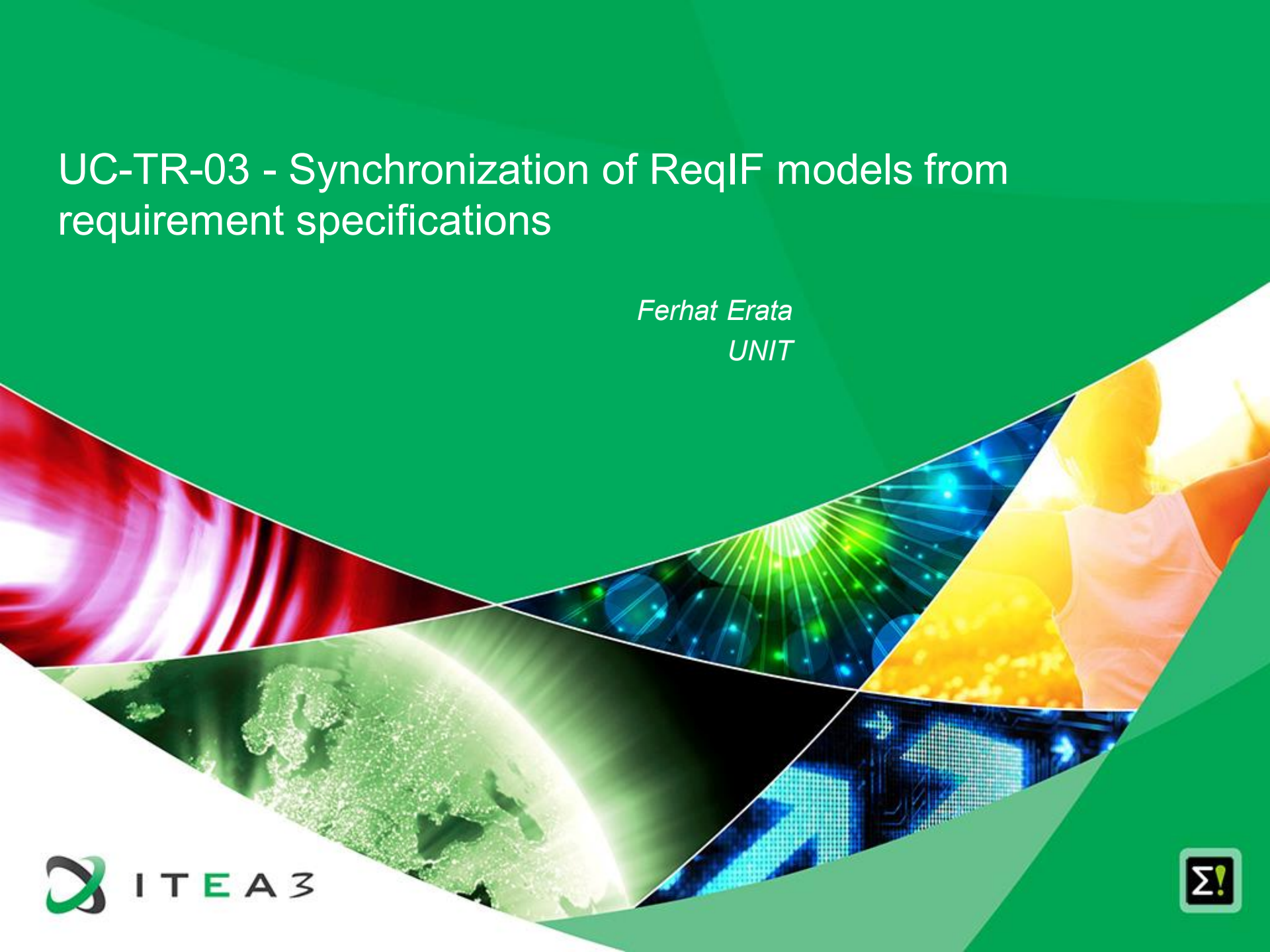
- Automatically captured parties information, main company and customer information
- Automatically filled names and signatures (if digital/electronical signatures)
- Automaticaly generate items according to given project ID
- With sharing utility ModelWriter will be able to send finalised and revised NDA to related parties and their contracted lawyers to review.

What we have done until today about NDA use case:

- Company information acquisition from database is done
- Customer segmentations completed ready to integrate and acquisite their data to ModelWriter
- Projects database relations have been built, ready to integrate ModelWriter engine

UC-TR-03 - Synchronization of ReqIF models from requirement specifications

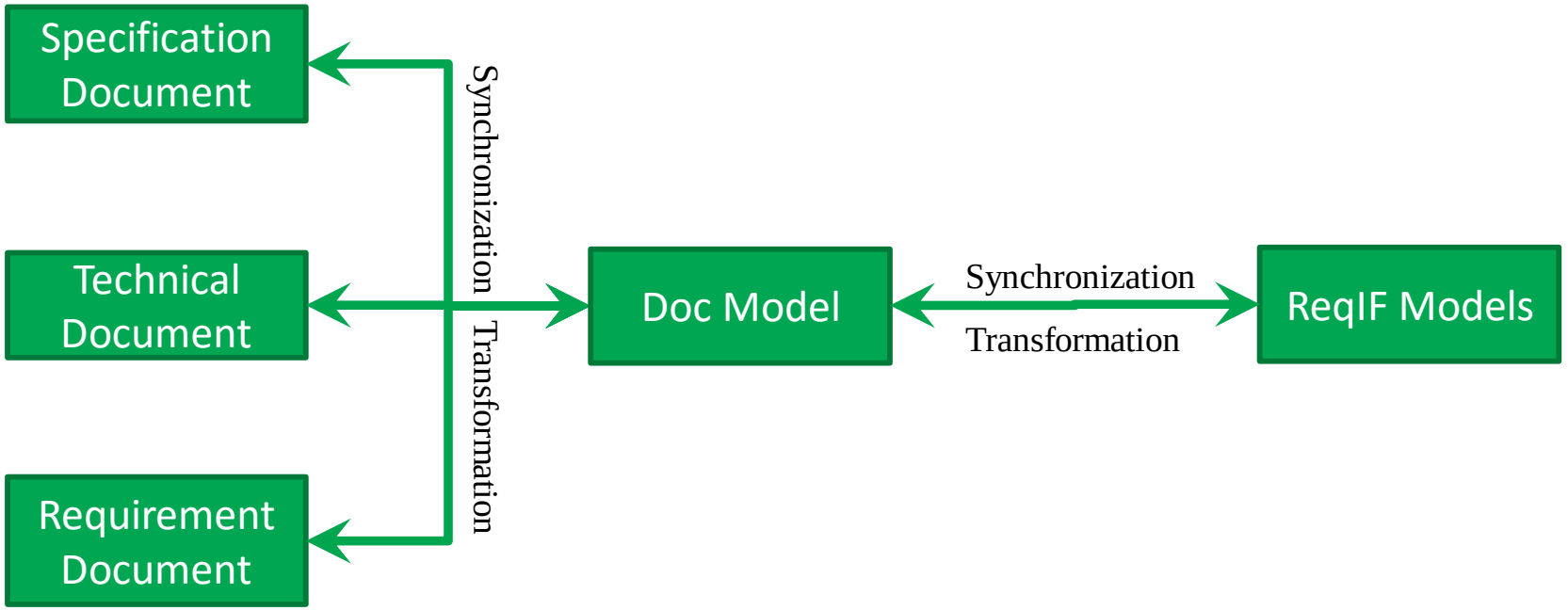
Ferhat Erata
UNIT



UC-TR-03 - Synchronization of ReqIF models from requirement specifications

- Technical documents are usually long and have complex structure
  - For example requirement or specification documents
- These documents keep changing in the time frame and need to be consistent with the other artifacts
  - For example, with a ReqIF model
- In this use case we aim to keep these documents and models synchronized
  - This will include bidirectional transformation of documents and models

UC-TR-03 - Synchronization of ReqIF models from requirement specifications



- At the current progress status of this UC:
  - The transformation is done in one way (left to right)
  - The synchronization is done only between DocModel and ReqIF models

UC-TR-03 - Synchronization of ReqIF models from requirement specifications

Sample Applications:

- Airbus SIDP templates ⇔ ReqIF
- Havelsan Requirement doc templates ⇔ ReqIF
- University Management System docs ⇔ ReqIF
- Eclipse RMF specifications ⇔ ReqIF

- The initial implemented version of this use case will be presented in the demonstration session.

UC-TR-03
Screenshots - Airbus SIDP templates

AIRBUS

Design Principles Applicable to the xx System Installation - Program

SIDPREF1

7 Design Principles

7.1 Design Principles Applicable to the Entire Design

General

SIDP92A001V-A-269

The effects of thermal variations, structural deformation, pressurization variation, etc., shall be taken into account.

SIDP92A001V-A-280

Each item in direct contact with ATA92 bundles shall be qualified to withstand conditions detailed in Table 3 below.

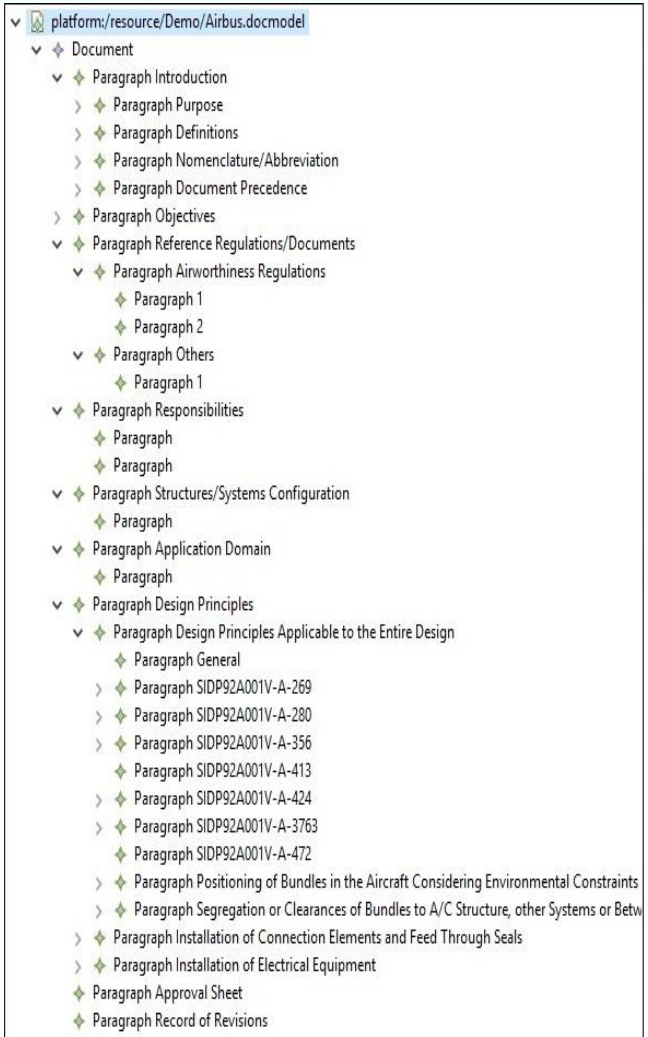
Type of Route	All areas except in high temperature zones		High temperature zones	
	Peak condition *	Continuous operating condition **	Peak condition *	Continuous operating condition **
G routes	170°C	170°C	260°C	260°C
P, X routes	150°C	95°C	260°C	200°C
S, R, T, U, V routes	85°C	85°C	130°C	130°C
Others types (M, S, Q,...)	135°C	95°C	260°C	200°C
* duration of 100 hours ** duration A/C life				

Table 3. Operating conditions for items in direct contact with ATA92 bundles

Attachment devices placed inside boxes, which contain power cables, shall withstand.

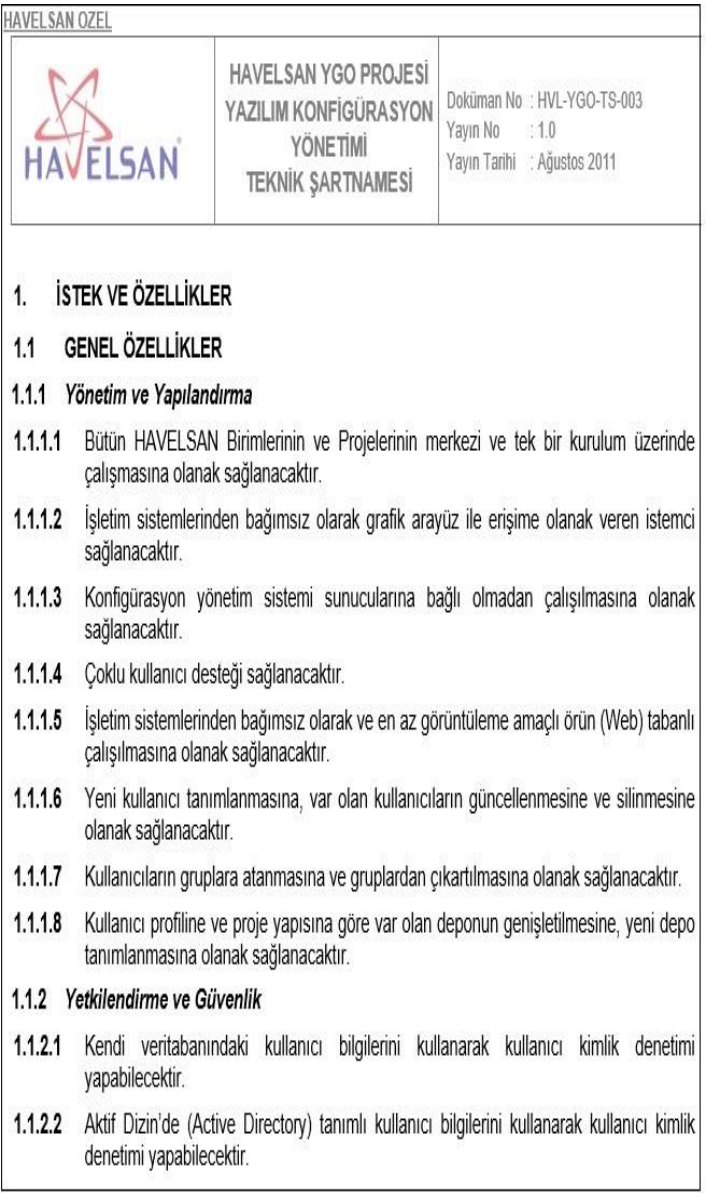
- a minimum of 150°C for peak condition and

- a minimum of 110°C for continuous operating condition.



ID	Description
1	Introduction
1.1	Purpose
1.2	Definitions
1.3	Nomenclature/Abbreviation
1.3.1	A/C Aircraft
1.3.2	ATA Air Transport Association
1.4	Document Precedence
2	Objectives
3	Reference Regulations/Documents
3.1	Airworthiness Regulations
3.1.1	JAR 25.607 Fasteners
3.1.2	REF Title
3.2	Others
3.2.1	A350XWB_SIDP V&V Policy PL0901917
4	Responsibilities
5	Structures/Systems Configuration
6	Application Domain
7	Design Principles
7.1	Design Principles Applicable to the Entire Design
7.1.1	General
7.1.3	SIDP92A001V-A-280
7.1.4	SIDP92A001V-A-356
7.1.4.1	Locking Of Bolted Fastenings
7.1.2	SIDP92A001V-A-269
7.1.5	SIDP92A001V-A-413
7.1.6	SIDP92A001V-A-424
7.1.6.1	Installation In Ceiling Area
7.1.7	SIDP92A001V-A-3763
7.1.7.1	Installation In Fuel Tanks
7.1.8	SIDP92A001V-A-472
7.1.9	Positioning of Bundles in the Aircraft Considering Environmental
7.1.9.1	General
7.1.9.2	SIDP92A001V-A-557
7.1.9.3	SIDP92A001V-A-579
7.1.10	Segregation or Clearances of Bundles to A/C Structure, other Syste
7.1.10.1	General Applications
7.1.10.1.1	Sagging (s)
7.1.10.1.2	SIDP92A001V-A-784

UC-TR-03
Screenshots - Havelsan Requirement docs



HAVELSAN OZEL

HAVELSAN YGO PROJESİ
YAZILIM KONFİGÜRASYON
YÖNETİMİ
TEKNİK ŞARTNAMESİ

Doküman No : HVL-YGO-TS-003
Yayın No : 1.0
Yayın Tarihi : Ağustos 2011

1. İSTEK VE ÖZELLİKLER

1.1 GENEL ÖZELLİKLER

1.1.1 *Yönetim ve Yapılandırma*

1.1.1.1 Bütün HAVELSAN Birimlerinin ve Projelerinin merkezi ve tek bir kurulum üzerinde çalışmasına olanak sağlanacaktır.

1.1.1.2 İşletim sistemlerinden bağımsız olarak grafik arayüz ile erişime olanak veren istemci sağlanacaktır.

1.1.1.3 Konfigürasyon yönetim sistemi sunucularına bağlı olmadan çalışılmasına olanak sağlanacaktır.

1.1.1.4 Çoklu kullanıcı desteği sağlanacaktır.

1.1.1.5 İşletim sistemlerinden bağımsız olarak ve en az görüntüleme amaçlı örün (Web) tabanlı çalışılmasına olanak sağlanacaktır.

1.1.1.6 Yeni kullanıcı tanımlanmasına, var olan kullanıcıların güncellenmesine ve silinmesine olanak sağlanacaktır.

1.1.1.7 Kullanıcıların gruplara atanmasına ve gruplardan çıkartılmasına olanak sağlanacaktır.

1.1.1.8 Kullanıcı profiline ve proje yapısına göre var olan deponun genişletilmesine, yeni depo tanımlanmasına olanak sağlanacaktır.

1.1.2 *Yetkilendirme ve Güvenlik*

1.1.2.1 Kendi veritabanındaki kullanıcı bilgilerini kullanarak kullanıcı kimlik denetimi yapabilecektir.

1.1.2.2 Aktif Dizin'de (Active Directory) tanımlı kullanıcı bilgilerini kullanarak kullanıcı kimlik denetimi yapabilecektir.

ID	Description
1	İSTEK VE ÖZELLİKLER
1.1	İSTEK VE ÖZELLİKLER
1.1	GENEL ÖZELLİKLER
1.1.1	Yönetim ve Yapılandırma
1.1.1.1	Bütün HAVELSAN Birimlerinin ve Projelerinin merkezi ve tek bir kurulum üzerinde çalış
1.1.1.2	İşletim sistemlerinden bağımsız olarak grafik arayüz ile erişime olanak veren istemci sa
1.1.1.3	Konfigürasyon yönetim sistemi sunucularına bağlı olmadan çalışılmasına olanak sağla
1.1.1.4	Çoklu kullanıcı desteği sağlanacaktır.
1.1.1.5	İşletim sistemlerinden bağımsız olarak ve en az görüntüleme amaçlı örün (Web) taban
1.1.1.6	Yeni kullanıcı tanımlanmasına, var olan kullanıcıların güncellenmesine ve silinmesine
1.1.1.7	Kullanıcıların gruplara atanmasına ve gruplardan çıkartılmasına olanak sağlanacaktır.
1.1.1.8	Kullanıcı profiline ve proje yapısına göre var olan deponun genişletilmesine, yeni depo
1.1.2	Yetkilendirme ve Güvenlik
1.1.2.1	Kendi veritabanındaki kullanıcı bilgilerini kullanarak kullanıcı kimlik denetimi yapabile
1.1.2.2	Aktif Dizin'de (Active Directory) tanımlı kullanıcı bilgilerini kullanarak kullanıcı kimlik d
1.1.2.3	Aktif Dizin'den elde edilmiş kullanıcı kimliğini, Tek Giriş (Single Sign On - SSO) ilkesine
1.1.2.4	Kimliği doğrulanmış kullanıcılar için yetkilendirme yapılacaktır.
1.1.2.5	Nesne erişimlerinin yetkilendirilmesi için, en az "Ekleme", "Silme", "Düzenleme", "Görü
1.1.2.6	Proje yönetim ve bakım işlevlerinin yetkilendirilmesi için "Yönetim" yetkisi sağlanacak
1.1.2.7	Proje yönetim ve bakım işlevleri ile nesneler üzerinde yapılabilen tüm iş ve işlemleri ya
1.1.2.8	Nesneler üzerinde yapılabilen tüm iş ve işlemleri yapma yetkisine sahip "Düzenleyici"
1.1.2.9	Nesneleri görüntüleme ve raporlama yetkisine sahip "Görüntüleyici" yetki profili sağla
1.1.2.10	Yetki profillerinin kullanıcılara/gruplara atanmasına olanak sağlanacaktır.
1.1.2.11	Kullanıcıların/grupların dosya/dizinler üzerinde seçilebilecek yetkiler ile yetkilendirilme
1.1.3	İşlevsellik
1.1.3.1	Konfigürasyon elemanlarının çalışma kopyasının yaratılmasına olanak sağlanacaktır.
1.1.3.2	Çalışma kopyasında değişiklik yapılan konfigürasyon elemanlarının depoya gönderilm
1.1.3.3	Konfigürasyon elemanı üzerinde yapılan değişikliğin depoya gönderilmesi sırasında yo
1.1.3.4	Komut satırından çalışma olanağı sağlanacaktır.
1.1.3.5	Depoya gönderme işleminin parçalanamaz şekilde (atomic commit) gerçekleşmesi sa
1.1.3.6	Yan dal (branch) açmaya olanak sağlanacaktır.

UC-TR-03
University Management System docs

1 Use Case UC1: Sign In

Primary Actor: Student, Lecturer.

Stakeholders and Interests:

- Student: Wants simple user interface, fast response, no system errors.
- Lecturer: ?

Preconditions:

- Student is registered.

Success Guarantee (Postcondition):

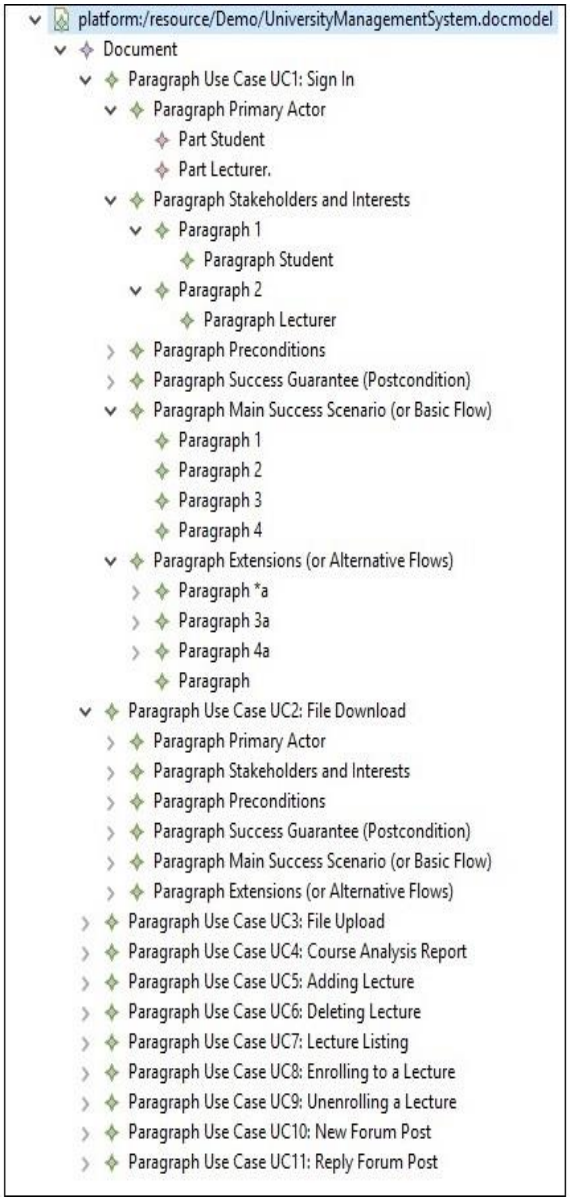
- Student is logged in.

Main Success Scenario (or Basic Flow):

1. Student visits system home page.
2. System shows home page with login form and sign up button.
3. Student enters his/her username and password then click login button.
4. System shows Student's home page.

Extensions (or Alternative Flows):

*a. At any time, system fails, to support recovery, ensure all transaction sensitive state
be recovered from any step of the scenario.
    1. Student restarts System and requests recovery of prior state.
    2. System reconstructs prior state.
        2a. System detects anomalies preventing recovery.
            1. System signals error to the Student, records the error, and
state.
3a. Student enters invalid username or password.
    1. System shows errors and request to Student to retry.
    2. Student enters his/her username and password.
4a. System detects failure to communicate with server.
    1. System signals error and rejects the request.



ID	Description
1	Use Case UC1: Sign In
1.1	Primary Actor
1.1.1	Student
1.1.2	Lecturer.
1.2	Stakeholders and Interests
1.2.1	Student
1.2.1.1	Wants simple user interface, fast response, no system errors.
1.2.1	Lecturer
1.2.1.1	?
1.3	Preconditions
1.3.1	Student is registered.
1.4	Success Guarantee (Postcondition)
1.4.1	Student is logged in.
1.5	Main Success Scenario (or Basic Flow)
1.5.1	Student visits system home page
1.5.2	System shows home page with login form and sign up button
1.5.3	Student enters his/her username and password then click login b
1.5.4	System shows Student's home page
1.6	Extensions (or Alternative Flows)
1.6.1	At any time, system fails, to support recovery, ensure all transacti state and events can be recovered from any step of the scenario
1.6.1.1	Student restarts System and requests recovery of prior state
1.6.1.2	System reconstructs prior state
1.6.1.2.1	System detects anomalies preventing recovery
1.6.1.2.1.1	System signals error to the Student, records the error, and enters
1.6.2	Student enters invalid username or password
1.6.2.1	System shows errors and request to Student to retry
1.6.2.2	Student enters his/her username and password
1.6.3	System detects failure to communicate with server
1.6.3.1	System signals error and rejects the request
2	Use Case UC2: File Download
2.1	Primary Actor
2.1.1	Student
2.2	Stakeholders and Interests
2.2.1	Student

UC-TR-03
Screenshot - Eclipse RMF specifications

1 Use Case UC1: Create a new SpecObject

Preconditions:

- ReqIF model exists and is open.

Main Success Scenario (or Basic Flow):

1. We assume that a Specification exists and is open (not required for alternative scenario)
2. Double click on the cell in the Specification Editor to be edited.
3. Select the Child or Sibling submenu.
4. Select the desired Spec Object Type (or none) from the submenu.
5. Note that some cells may not be editable, in which case nothing will happen.

Alternative 1 Create in Outline:

*a. The same workflow works for elements that are shown underneath "Specifications" in th
2a. It is also possible to create children of the "SpecObjects" folder in the outline, but in this
SpecHirarchy will be created.

Alternative 2 Keyboard Shortcut:

*a. The keyboard shortcut Ctrl-Enter will create a SpecHierarchy sibling to the currently sele
element and immediately go into edit mode in the currently selected column.

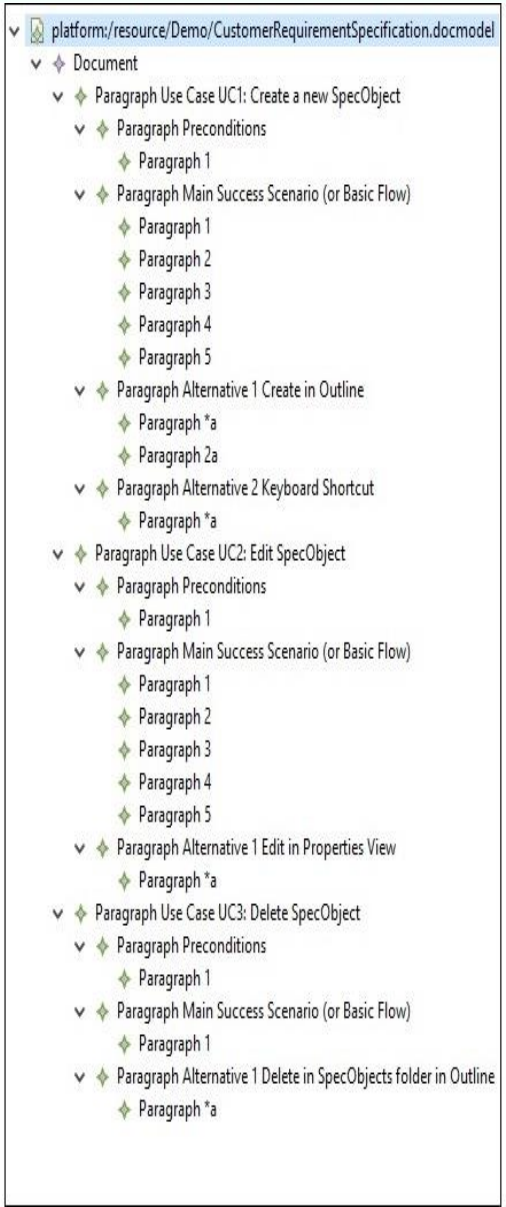
2 Use Case UC2: Edit SpecObject

Preconditions:

- A ReqIF model exists, is open and at least one SpecObject exisits.

Main Success Scenario (or Basic Flow):

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)



ID	Description
1	Use Case UC1: Create a new SpecObject
1.1	Preconditions
1.1.1	ReqIF model exists and is open.
1.2	Main Success Scenario (or Basic Flow)
1.2.1	We assume that a Specification exists and is open (not required for alternative scenario)
1.2.2	Double click on the cell in the Specification Editor to be edited
1.2.3	Select the Child or Sibling submenu
1.2.4	Select the desired Spec Object Type (or none) from the submenu
1.2.5	Note that some cells may not be editable, in which case nothing will happen
1.3	Alternative 1 Create in Outline
1.3.1	The same workflow works for elements that are shown underneath "Specifications" in the outilne
1.3.2	It is also possible to create children of the "SpecObjects" folder in the outline, but in this case, no SpecH
1.4	Alternative 2 Keyboard Shortcut
1.4.1	The keyboard shortcut Ctrl-Enter will create a SpecHierarchy sibling to the currently selected element ar
2	Use Case UC2: Edit SpecObject
2.1	Preconditions
2.1.1	A ReqIF model exists, is open and at least one SpecObject exisits.
2.2	Main Success Scenario (or Basic Flow)
2.2.1	We assume that a Specification exists and is open (not required for alternative scenario)
2.2.2	Open a row's context menu (or in the empty editor space)
2.2.3	Alternatively, hit enter or space in that cell
2.2.4	In both cases, the double-clicked / selected cell will switch to edit mode
2.2.5	This results in a new SpecHierarchy being created that is linked to a newly created SpecObject with the
2.3	Alternative 1 Edit in Properties View
2.3.1	A selected element (no matter whether in Specification Editor or Outline or elsewhere) will be shown in
3	Use Case UC3: Delete SpecObject
3.1	Preconditions
3.1.1	A ReqIF model exists, is open and at least one SpecObject exists.
3.2	Main Success Scenario (or Basic Flow)
3.2.1	If an element is deleted from the specification (so essentially a SpecHierarchy), and no references to the
3.3	Alternative 1 Delete in SpecObjects folder in Outline
3.3.1	If the SpecObject is deleted from the SpecObjects folder in the ouline, it will be removed, no matter wha

Next Steps

- Next steps:

  - ReqIF model transformation to documents (bi-directional transformation)
  - Synchronization between a document and Doc-Model (fully synchronization)

Model Writer

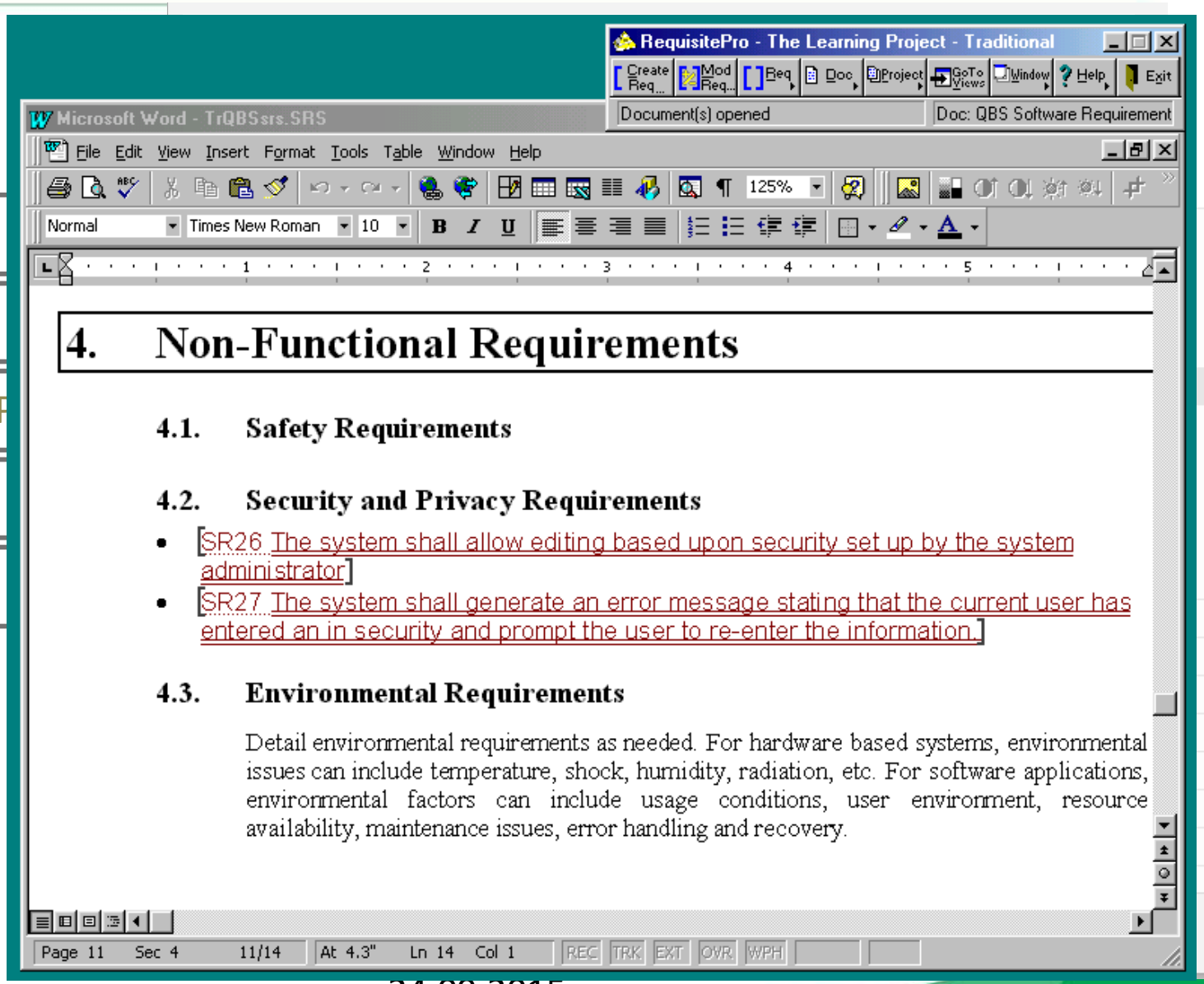
Work Package 1 - Industrial Use Cases and Requirements

Integration with Application Lifecycle Management tools

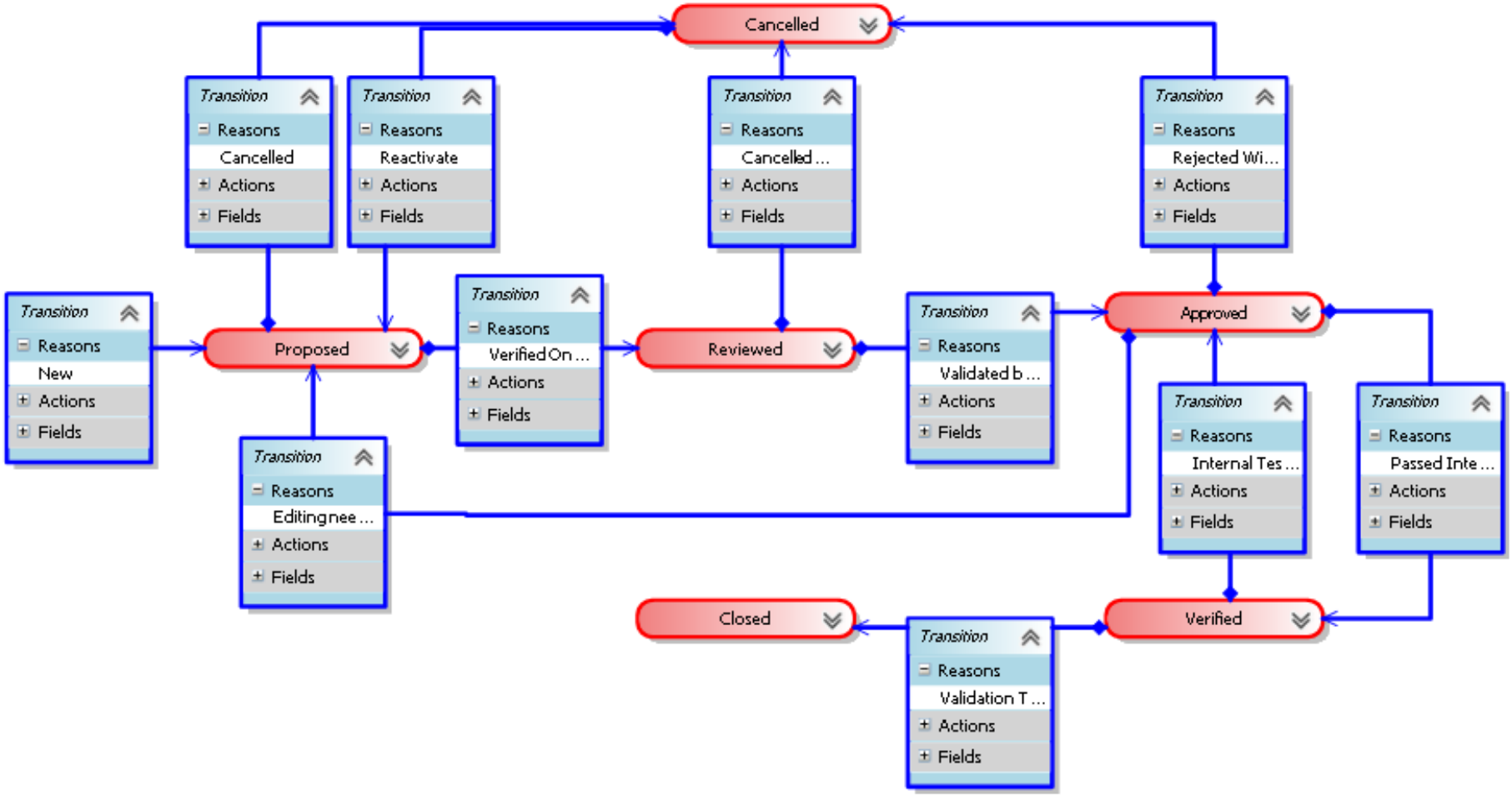
Eray TÜZÜN (HAVELSAN)
Yagup MACİT (HAVELSAN)

Requirement Work Item

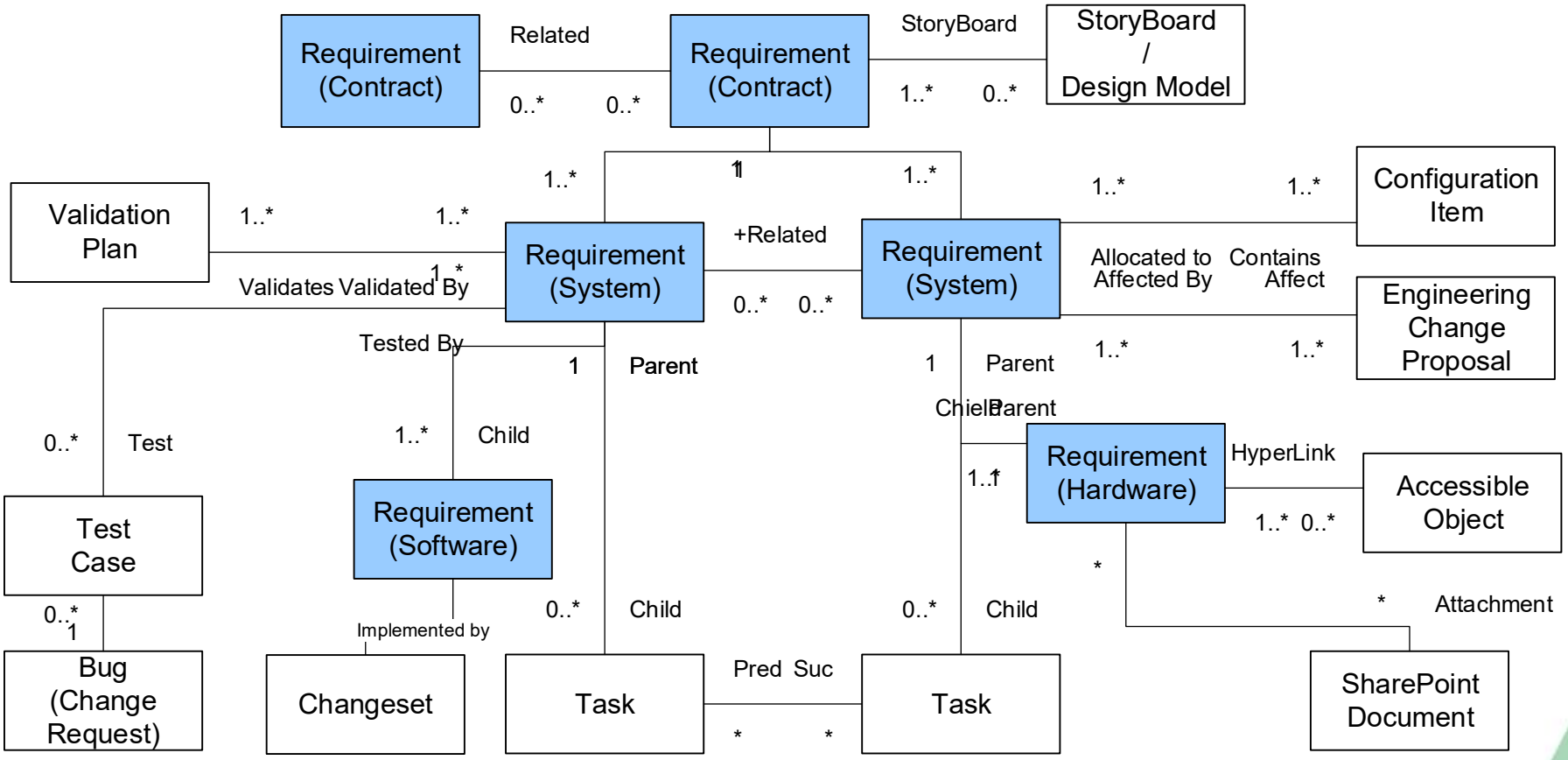
- Customized Form
- Attributes
- WorkItem Number F
- History
- Discussion



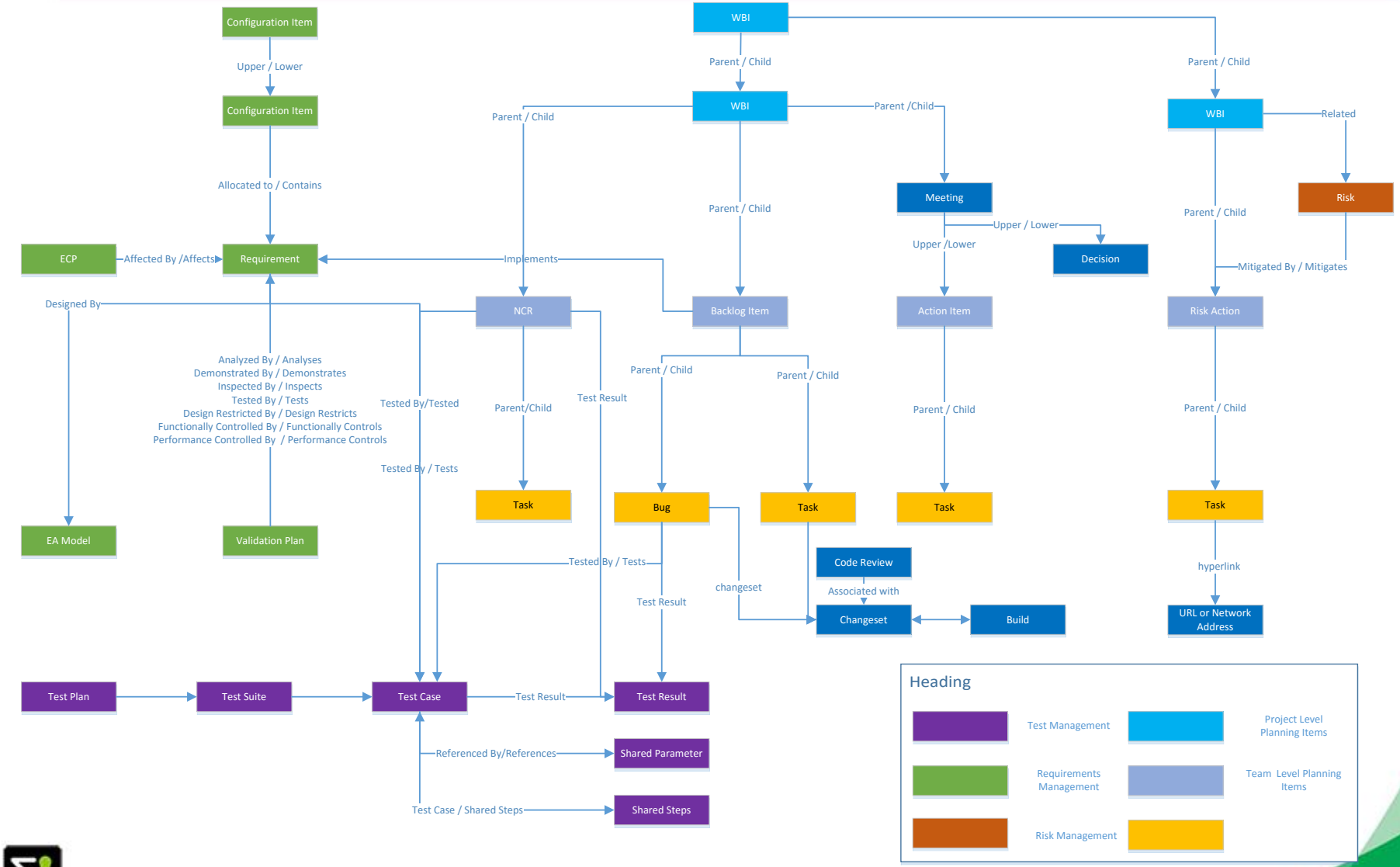
Requirement LifeCycle



Requirements Traceability



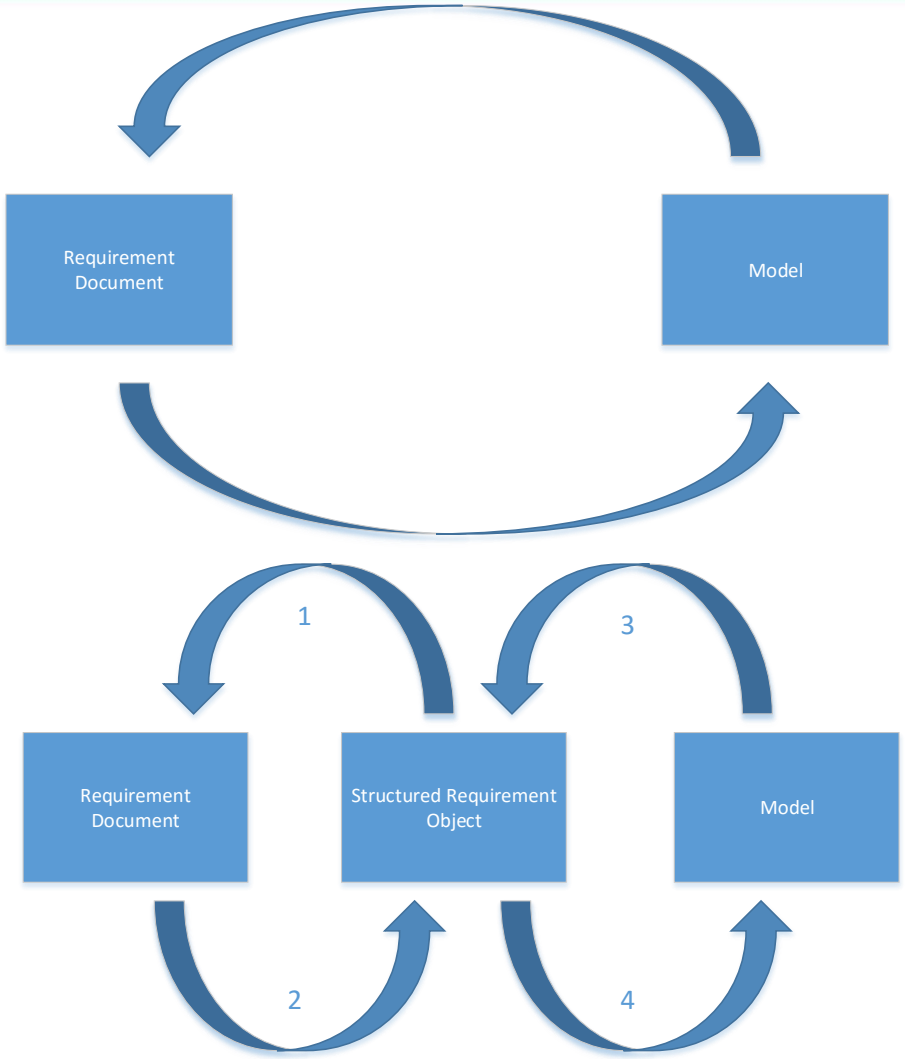
Requirements in ALM



Requirements in ALM

- Traceability with other artifacts is key
  - Requirements to other requirements
    - Customer/System/Software/Hardware..
    - Dependency relation between requirements
  - Requirements to tasks (Project management)
  - Requirements to Test Cases
  - Requirements to Design elements
  - Requirements to generated documents
  - Requirements to source code
  - Requirements to Build
  - Requirements to bugs
  - Requirements to risks
  - …

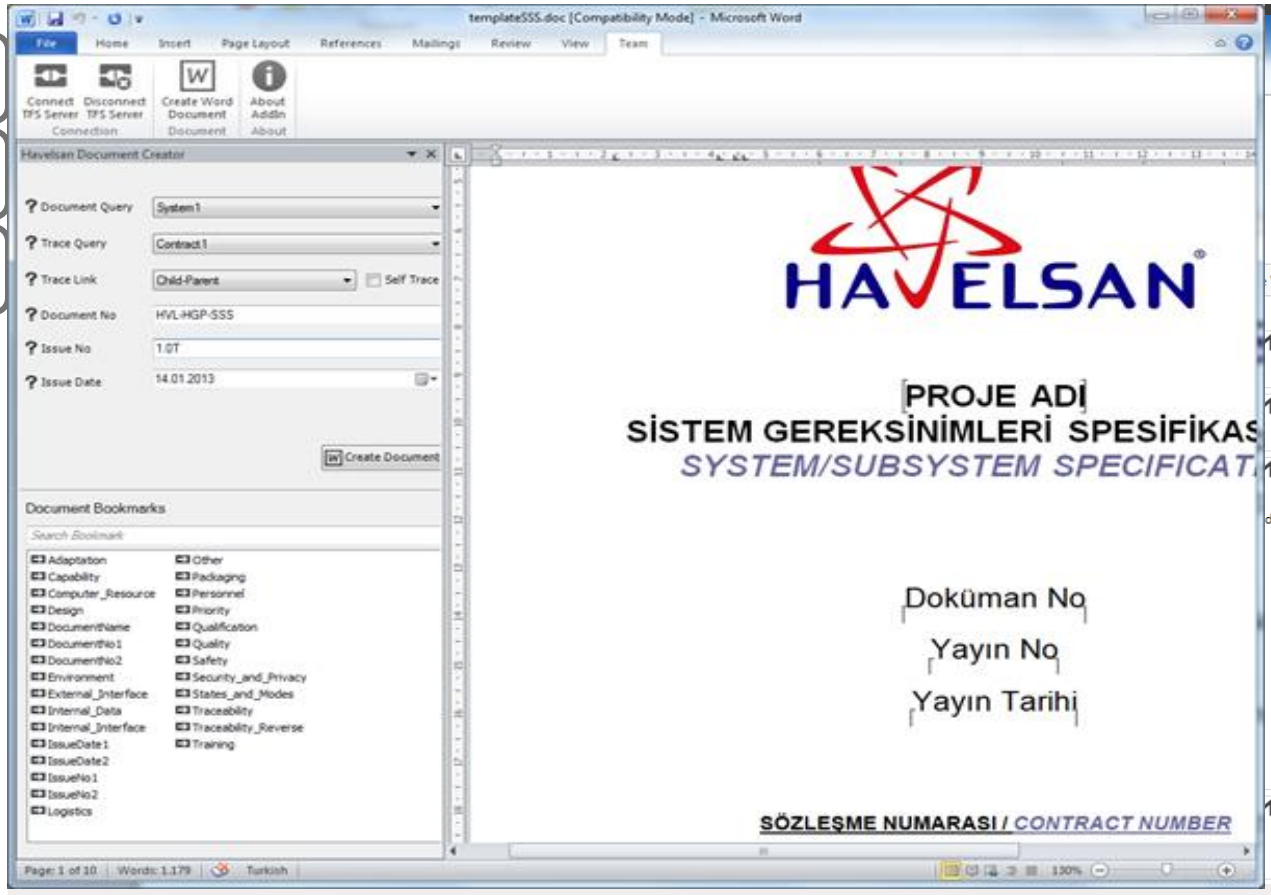
HAVELSAN Use case for ModelWriter



Currently we support scenario #1 and #4, and interested in Scenario #3, #2

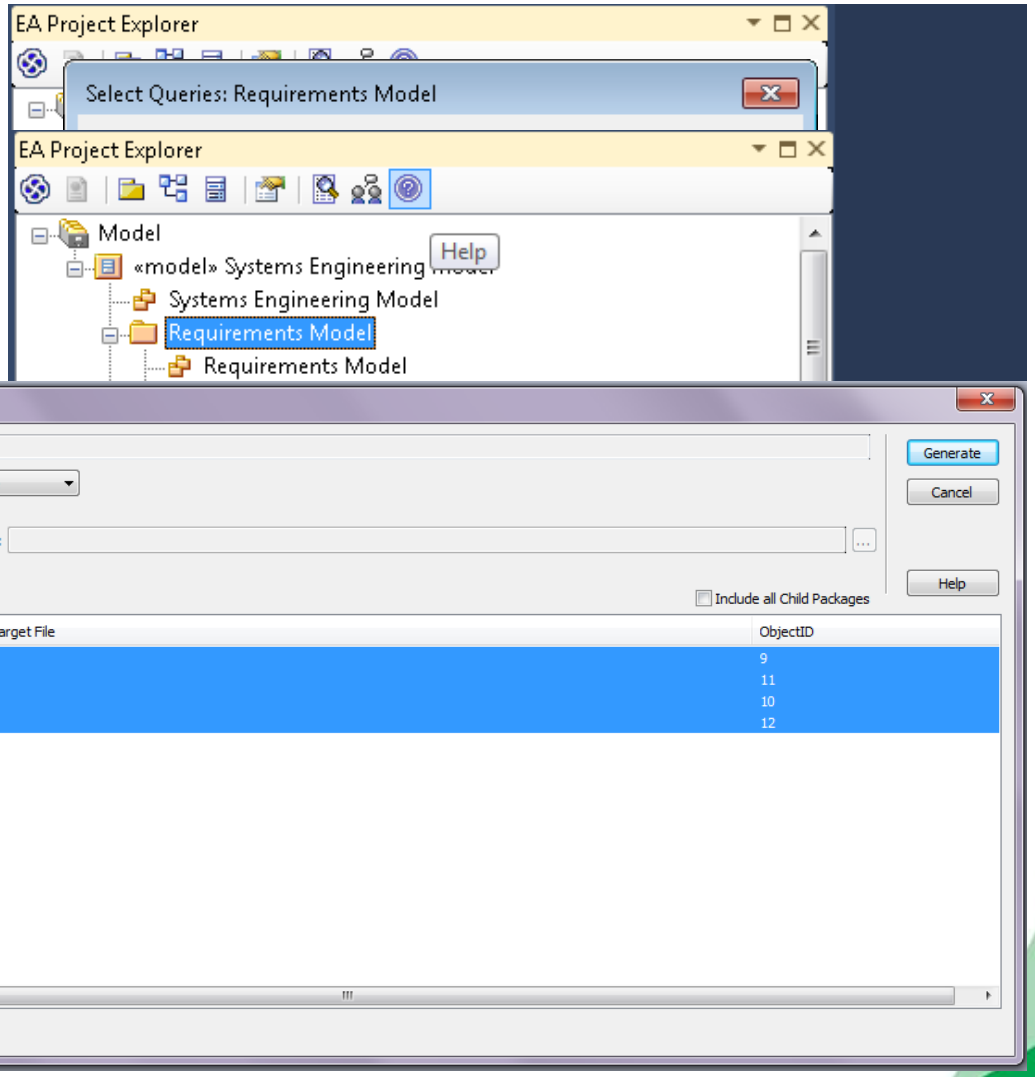
Havelsan Ext - Document Generation

- Preview
- Word Extension
- Support for Templates



Requirements - Design Model Traceability

- Requirements would be imported to the Design tool for traceability.
- Code generation from design



UC-TR-05 - Synchronous Business Process Design with Use Cases

Geylani Kardaş

KoçSistem

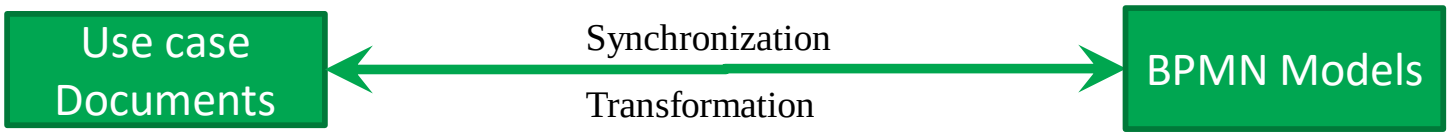
UC-TR-05 - Synchronous Business Process Design with Use Cases

- Use cases are one of the main approaches to represent the requirements.
  - A use case is a list of actions or event steps, typically defining the interactions between a role (a.k.a actor in the UML).

- BPMN provides a graphical notation for specifying the processes in a diagram based on a flowcharting technique (similar to activity diagrams in UML).
  - The aim is to support business process management, for both technical and business users, by providing a notation which enables to represent complex process semantics.

- However, the transformation and synchronization of use cases and BPMN models are challenges addressed in this UC.

UC-TR-05 - Synchronous Business Process Design with Use Cases



- Applications:
  - University Management System Use cases docs ⇔ BPMN
  - Eclipse RMF use case specifications ⇔ BPMN

- At the current state of progress this UC:
  - The transformations are done in one way (left to right)

UC-TR-05
University Management System Use cases

3 Use Case UC3: File Upload

Primary Actor: Student

Stakeholders and Interests:

- Student: Wants simple user interface, fast response, no system errors.

Preconditions: Student is identified and authenticated.

Success Guarantee (Postcondition): File is uploaded.

Main Success Scenario (or Basic Flow):

1. Student visits file upload page.
2. System opens file browser dialog.
3. Student chooses the file that she/he is wanted to upload.
4. System starts the upload process.

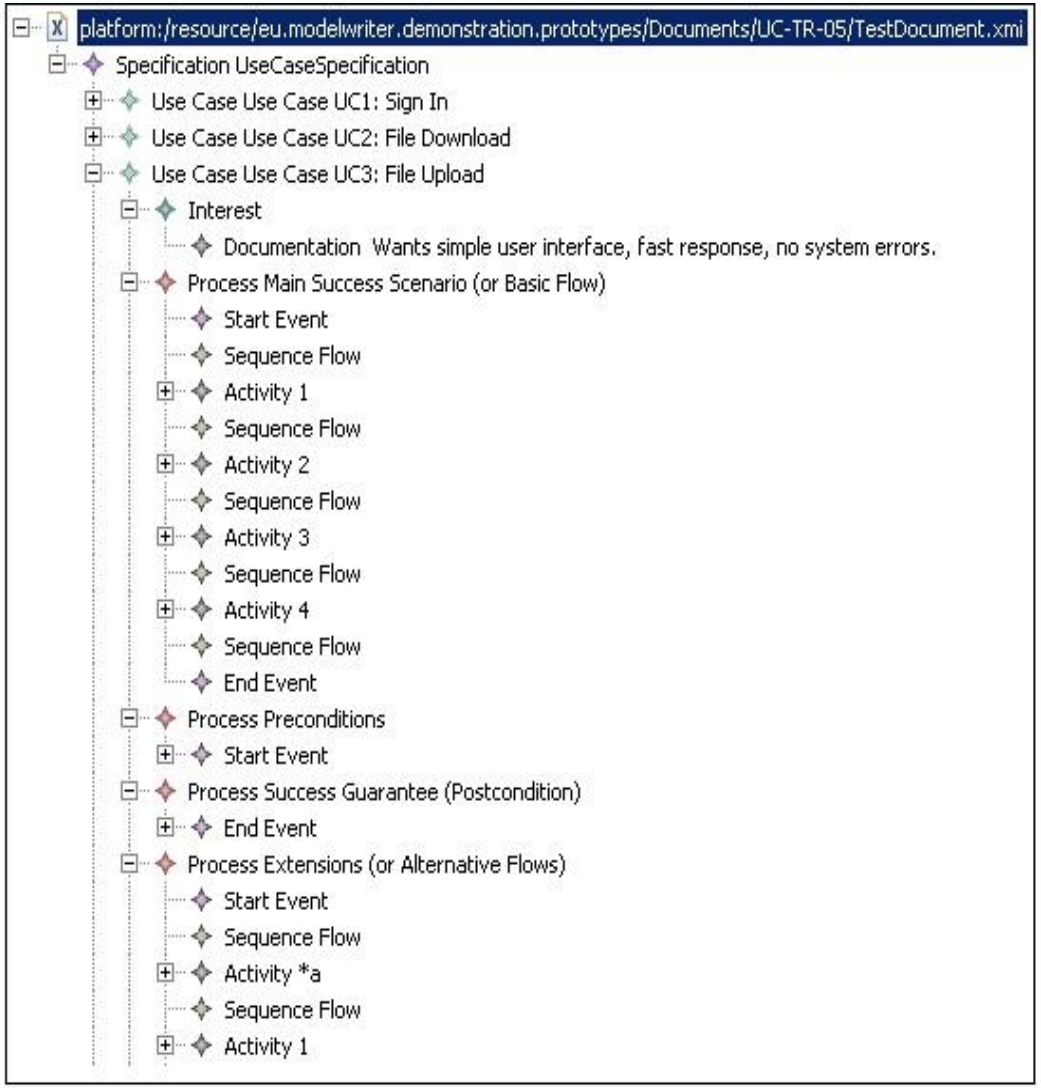
Extensions (or Alternative Flows):

*a. At any time, System fails: To support recovery, ensure all transaction sensitive state and events can be recovered from any step of the scenario.
    1. Student restarts System and requests recovery of prior state.
    2. System reconstructs prior state.

2a. System detects anomalies preventing recovery:
    1. System signals error to the Student, records the error, and enters a clean state.

3a. Invalid file:
    1. System shows the error and returns the file upload page.

4a. System detects failure to communicate with server:
    1. System signals error and rejects the request.



UC-TR-05
Eclipse RMF use case specifications

1 Use Case UC1: Create a new SpecObject

Preconditions:

- ReqIF model exists and is open.

Main Success Scenario (or Basic Flow):

1. We assume that a Specification exists and is open (not required for alternative scenario)
2. Double click on the cell in the Specification Editor to be edited.
3. Select the Child or Sibling submenu.
4. Select the desired Spec Object Type (or none) from the submenu.
5. Note that some cells may not be editable, in which case nothing will happen.

Alternative 1 Create in Outline:

*a. The same workflow works for elements that are shown underneath "Specifications" in the outilne.
2a. It is also possible to create children of the "SpecObjects" folder in the outline, but in this case, no SpecHirarchy will be created.

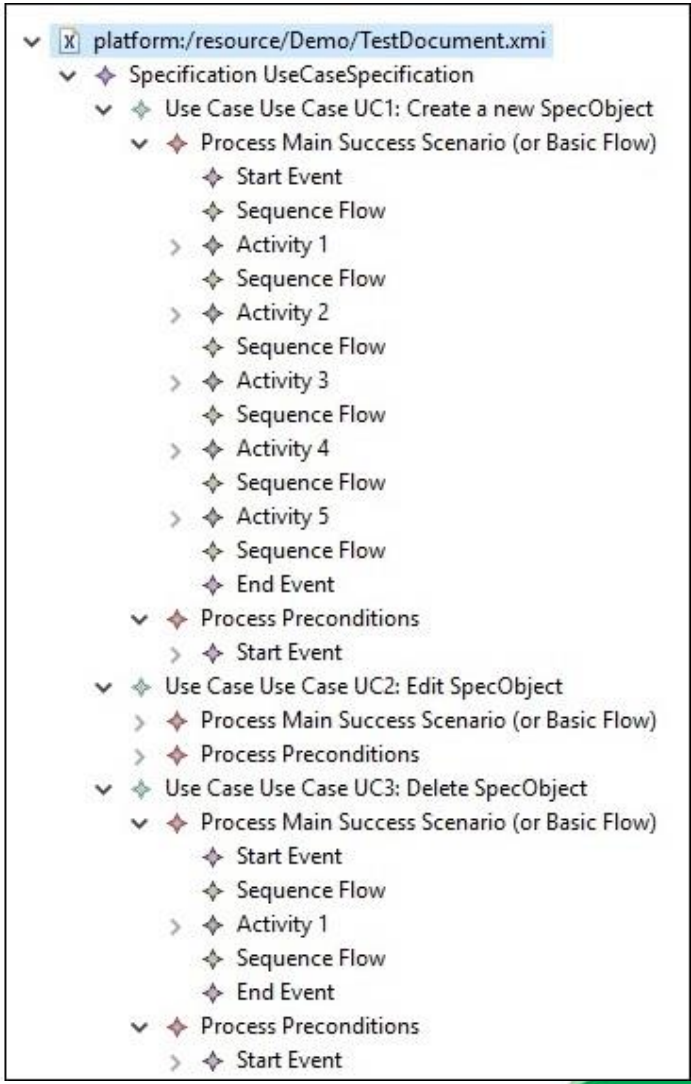
Alternative 2 Keyboard Shortcut:

*a. The keyboard shortcut Ctrl-Enter will create a SpecHierarchy sibling to the currently selected element and immediately go into edit mode in the currently selected column.

2 Use Case UC2: Edit SpecObject

Preconditions:

- A ReqIF model exists, is open and at least one SpecObject exisits.



UC-TR-05 - Synchronous Business Process Design with Use Cases

- Demonstration:
  - The initial implemented version of this use case will be presented in the demonstration session.

- Next steps:
  - BPMN model transformation to Use case documents (bi-directional transformation)
  - Synchronization between documents and BPMN Models
  - Using technique documents of Ford-Otosan;

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

4 Progress on Workpackages

Prof. Geylani Kardaş (Moderator)
KoçSistem

WP1 - Industrial use case and requirements

Anne MONCEAUX
AIRBUS GROUP

WP1

To describe and define the industrial, real life Use Cases, their associated requirements and evaluation method.

Tasks:

- T1.1 Evaluation Methods & Tools
- T1.2 Industrial Use Cases for Belgium Consortium
- T1.3 Industrial Use Cases for French Consortium
- T1.4 Industrial Use Cases for Turkish Consortium
- T1.5 Consolidated User Requirements and Review
- T1.6 Consolidated Software Requirements and Review
- T1.7 Annual Product Review
- T1.8 Technical Risk Assessment

T.1.1 Evaluation Methods & Tools

- UNIT, KOCSISTEM, AIRBUS, OBEO, HISBIM, MANTIS
  - To define evaluation methods, including the identification of metrics to quantify performance with and without ModelWriter

- Status
  - Survey of available evaluation method and tools
    - D1.1.1 Evaluation Methods & Tools

- Next:
  - Specification of use Cases KPI ; common KPI and selection of evaluation method and tools

T1.3 Industrial Use Cases for French Consortium

- OBEO, AIRBUS
- Status
  - Use case description
    - Discussed at 1st International ModelWriter Workshop in Izmir, Turkey
    - D1.3.1-Industrial Use cases for French consortium
  - Data collection
    - Part of the corpus data is provided by the partners.
    - Detailed description in *D2.1.2 Documentation of the corpora*
  - Public/private status
    - AIRBUS-OBEO-LORIA Non Disclosure Agreement finalized in June 2015
    - as some of the partners have not also decided for the privacy or publicity of the data, all of the corpus data is kept in a private repository in the GitHub
- *Remain to be done*
  - *Make public corpora available for all UC*

T1.4 Industrial Use Cases for Turkish Consortium

- MANTIS + UNIT + KOCSISTEM + HISBIM
- Status
  - Use cases description
    - D1.4.1 Industrial Use Cases for Turkish Consortium
  - Data collection
    - Part of the corpus data is provided by the partners.
    - However, some of the other partners have not decided on their corpus cases.
  - Public / private status
    - as some of the partners have not also decided for the privacy or publicity of the data, all of the corpus data is kept in a private repository in the GitHub

T1.5 Consolidated User Requirements and Review

- AIRBUS, OBEO, MANTIS, UNIT, KOCSISTEM, ALL
  - To share a common vision of User needs and expectations

- Status
  - UC driven User requirements capture
  - Consolidation through collective review (2nd International ModelWriter Workshop in Brussels, Belgium)
  - User requirements are stored & managed in GitHub
    - D1.5.1 Minutes of the User Requirements Review meeting
    - D1.5.2 User Requirements Document (URD) was automatically generate from Github
  - Technical Risks based on the defined requirements are identified in D1.8.1 Technical Risk Assessment v1.0

T1.6 Consolidated Software Requirements and Review

- AIRBUS, LORIA, UNIT, MANTIS, OBEO, KOCSISTEM, ALL
  - The [minimum] objective of the first year (Y1) is to integrate the key pieces of software together (modelling tools with a word processor within an IDE) to prove that we can have a unified prototype ModelWriter platform.

- Status
  - Technical partners refined software requirements based on URD
  - Consolidation through collective review (2nd International ModelWriter Workshop in Brussels, Belgium)
  - Software requirements are stored & managed in GitHub
    - D1.6.1 Minutes of the Software Requirements Review meeting
    - D1.6.2 Software Requirements Document (SRD) was automatically generate from Github
  - See also D1.8.1 Technical Risk Assessment v1.0

T1.8 Technical Risk Assessment

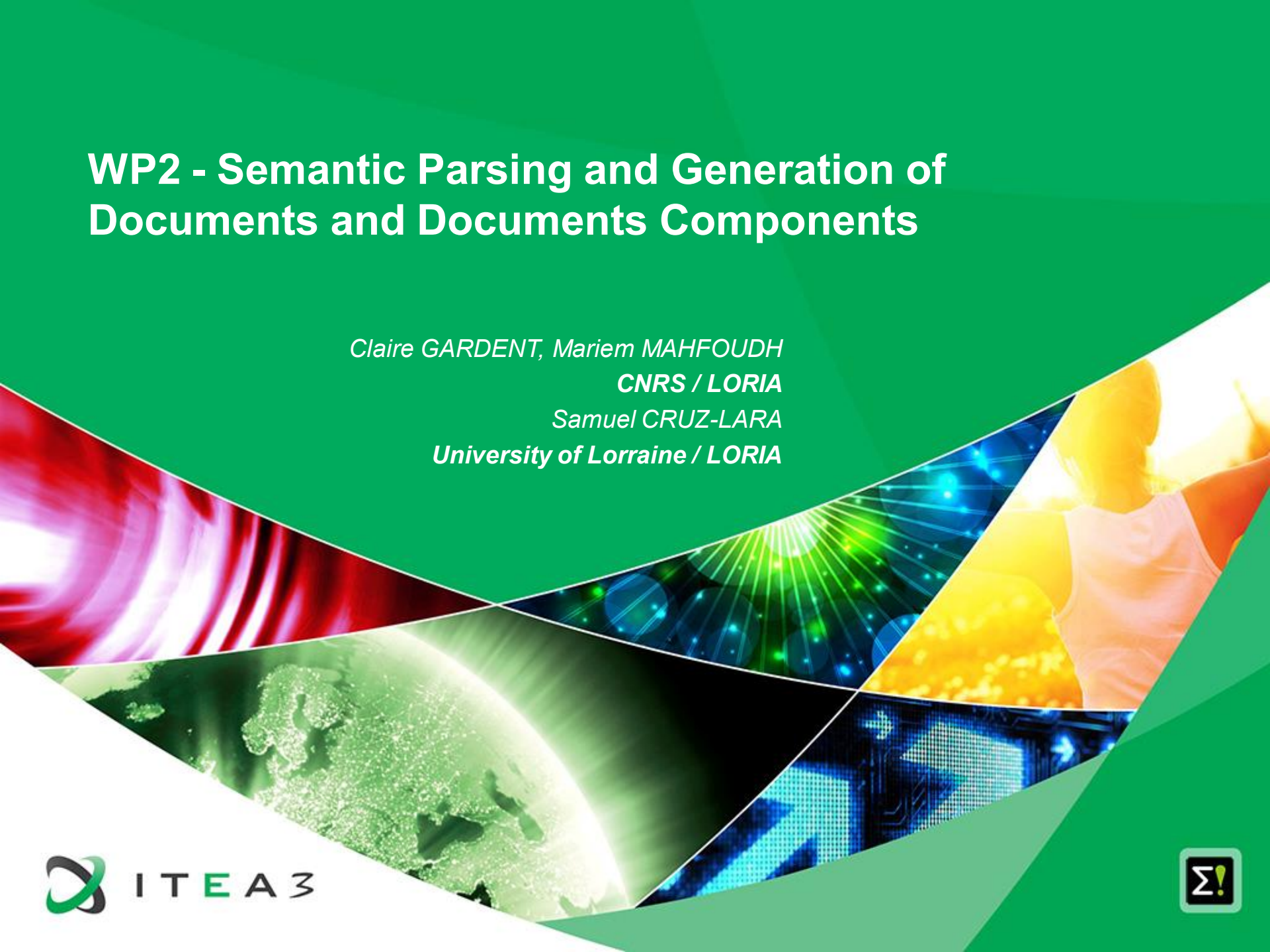
- OBEO, UNIT, KOCSISTEM + ALL
  - To identify, monitor and mitigate risks on the achievement of the project

- Status
  - 1st evaluation using Actuarial Approach of Technical Risk Assessment (TRA) of risks linked to requirements (URD, SDR) and technologies.
    - D1.8.1 Technical Risk Assessment v1.0

- *Next*
  - *The document may be up-dated throughout the project with special review at the same time as for the software requirements and the architectural design review, depending on the further details and requirements we get from the industrial use case providers.*

WP2 - Semantic Parsing and Generation of Documents and Documents Components

Claire GARDENT, Mariem MAHFOUDH
CNRS / LORIA
Samuel CRUZ-LARA
University of Lorraine / LORIA



ITEA 3

WP2

Goal: Provide tools and methods for:

- Annotating text fragments with model elements
- Converting texts to models and models to text

Tasks:

- T2.1 Data Collection
- T2.2 Semantic Parsing
- T2.3 Natural Language Generation
- T2.4 Definition of a common target semantic language
- T2.5 Development of a Semantic Parser and of a Natural Language Generator

T2.1 Data Collection

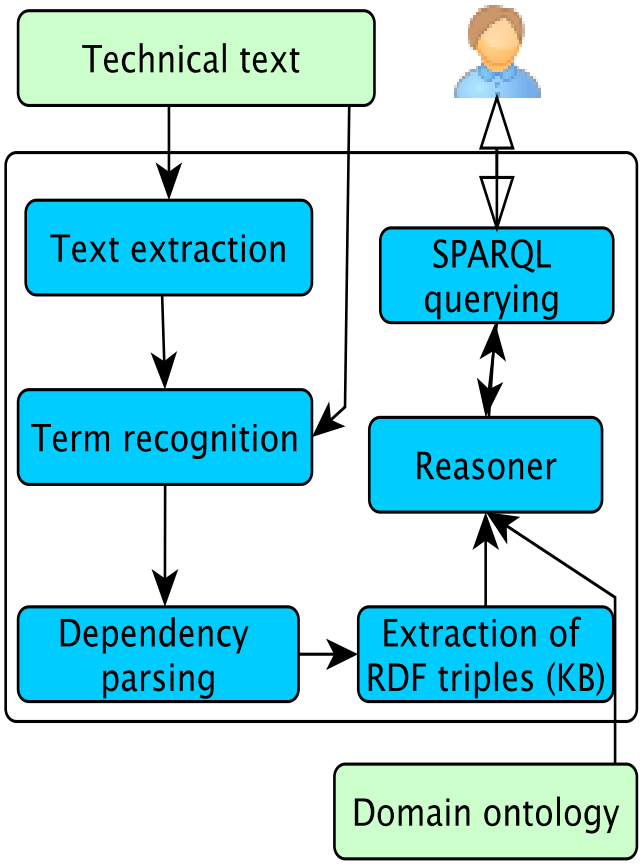
- AIRBUS (Confidential Data)
  - Text
    - System Installation Design Principles (SIDP) Documents
    - 986 semi-structured SIDP rules
  - Models
    - The Rule ontology represents the SIDP rules concepts. An OWL ontology composed of 30 classes, 35 object properties and 54 data properties.
    - The Component ontology represents the concepts and the vocabulary used in system installation rules. It is an OWL-DL ontology and it is composed in its current version of 476 classes, 21 ObjectProperties and 35 DataProperties.

Non Disclosure Agreement finalised in June 2015.

T2.1 Data Collection

- OBEO
  - Text
    - “TxStyle" Files: a set of files in natural language (i.e., English) related to the documentation of the application being modelled by Sirius
  - Models
    - Java Concepts: a list of Java identifiers (i.e., classes, interfaces, methods, etc.) related to Sirius
    - Ecore Concepts: a list of concepts related to Ecore (the Eclipse Modeling Framework meta model) and to Sirius

T2.2 Semantic Parsing



- Developed a prototype illustrating the automatic construction of an RDFS KB from text (CNRS/LORIA)
- « *Parsing Text into RDF* » B. Batouche, C. Gardent and A. Monceaux. SEPLN 2015, Alicante, Spain.
- Full scale Implementation applied to AIRBUS SIDP rules (Airbus)

T2.3 Natural Language Generation

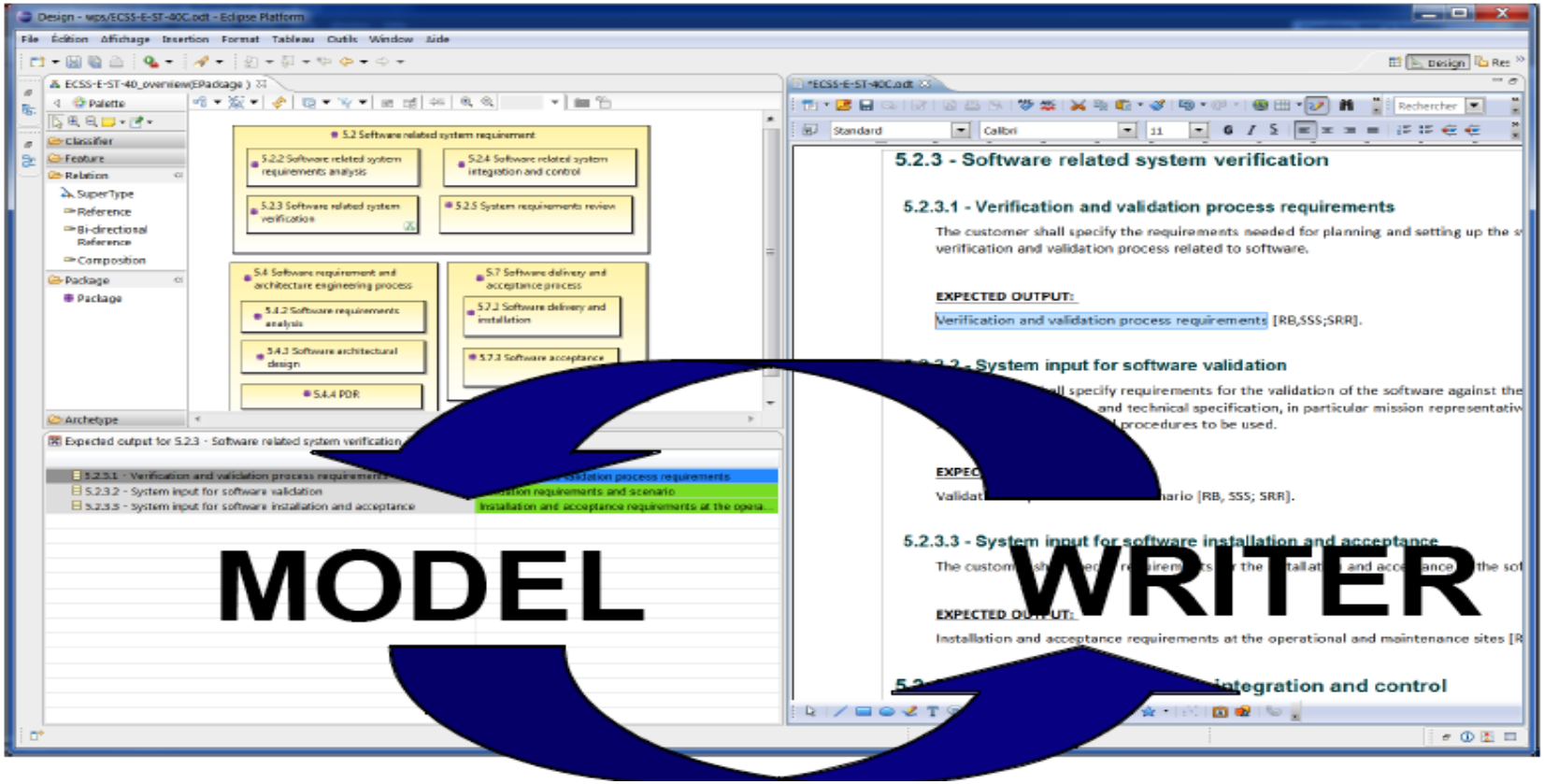
- D2.2.1 Report: Overview and Comparison of Existing Generators
- Keynote at SEPLN 2015, Alicante, Spain
- 2 Internships on generation from RDF data (ongoing)
  - Lexicalisation: automatic acquisition of a lexicon mapping RDF properties to natural language expressions
  - Document planning: automatic detection of typical document structures using DBPedia and Wikipedia

Semantic Annotation: Creating and Maintaining Synchronization Links, Checking their Consistency

ITEA 3

ModelWriter

Text & Model-Synchronized Document Engineering Platform



Creating synchronization links (relating text and model)
Checking the consistency of created synchronization links
Maintaining synchronization links (create, search, delete, modify)

Creating Synchronization Links

- Automatically
  - Exact matching: identified using String matching
    - Ex: Attach (text element) IsSameAs http://airbus-group/opd-function#Attach (ontology concept)
  - Morphological matching: identified using lemmatization and Stanford CoreNLP tools
    - Ex: Attached IsMorphologicallySimilarTo http://airbus-group/opd-function#Attach
  - Semantic matching: identified based on ontology and SKOS labels
    - Ex: Fixation isSynonymTo http://airbus-group.installsys/component#AttachmentPoint
- Manually
  - UserLink: Created by the user

Checking the Consistency of Synchronization Links

- Consistency check based on ontology's axioms and properties

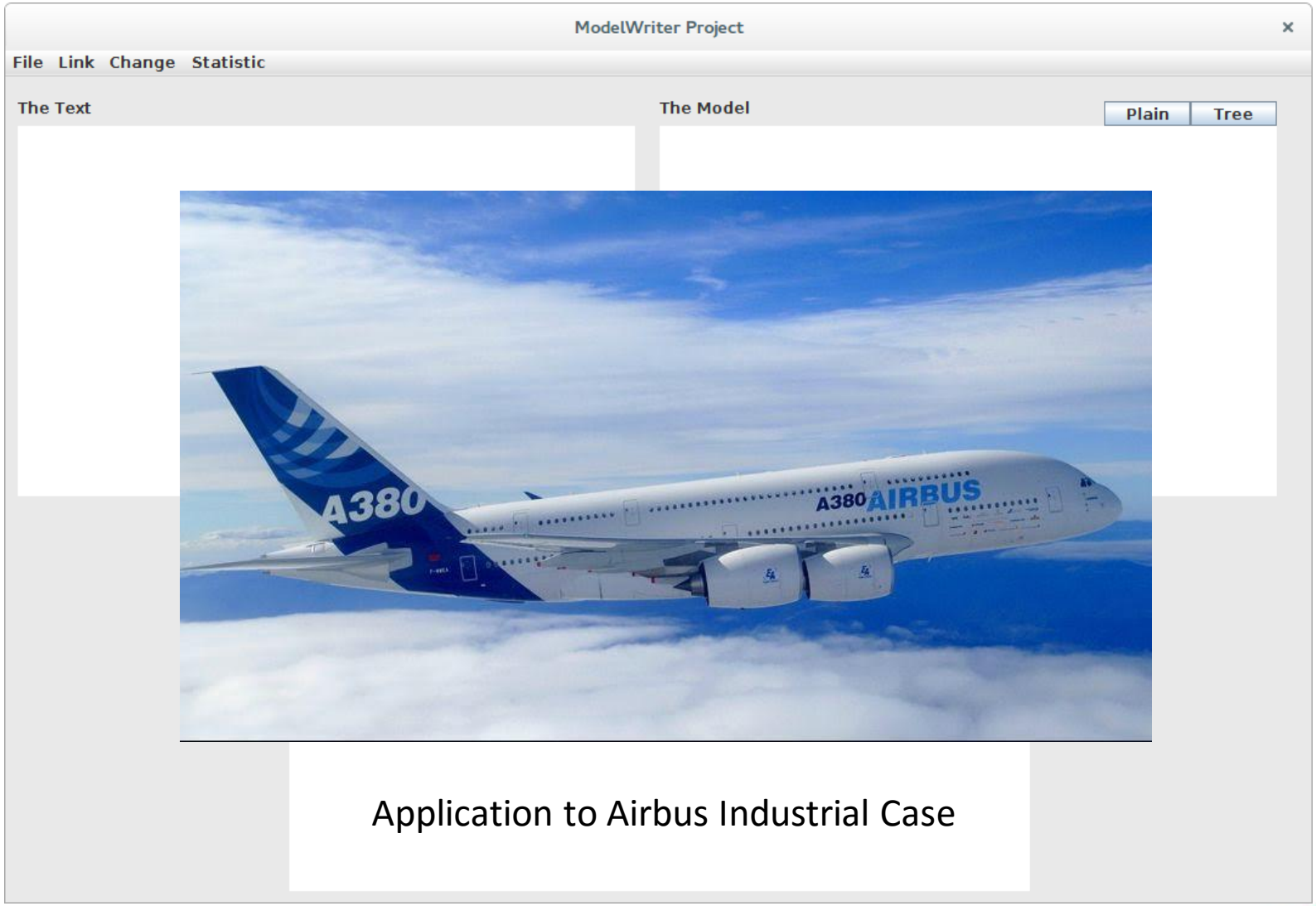
  - Rule:
    - If a text element and an ontology concept are semantically disjoint, then they cannot be synchronized
      - Ex: rigid Component cannot be synchronized with http://airbus-group.installsys/component#FlexibleComponent

Maintaining Synchronization Links

- Link maintenance operations:
  - Add New Link (user given)
  - Search Link (create or retrieve)
  - Remove Link

- Synchronisation between text and links
  - RenameTextElement
  - Add TextElement
  - RemoveTextElement

Semantic Annotation, Links Synchronization and Consistency Check



Application to Airbus Industrial Case

Semantic Annotation, Links Synchronization and Consistency Check

- Semantic parsing and consistency check:
  - The prototype can be accessed on the GitHub Model Writer repository:
    - https://github.com/ModelWriter/WP2/tree/master/Tool

Semantic Annotation

ITEA 3

Semantic Annotation

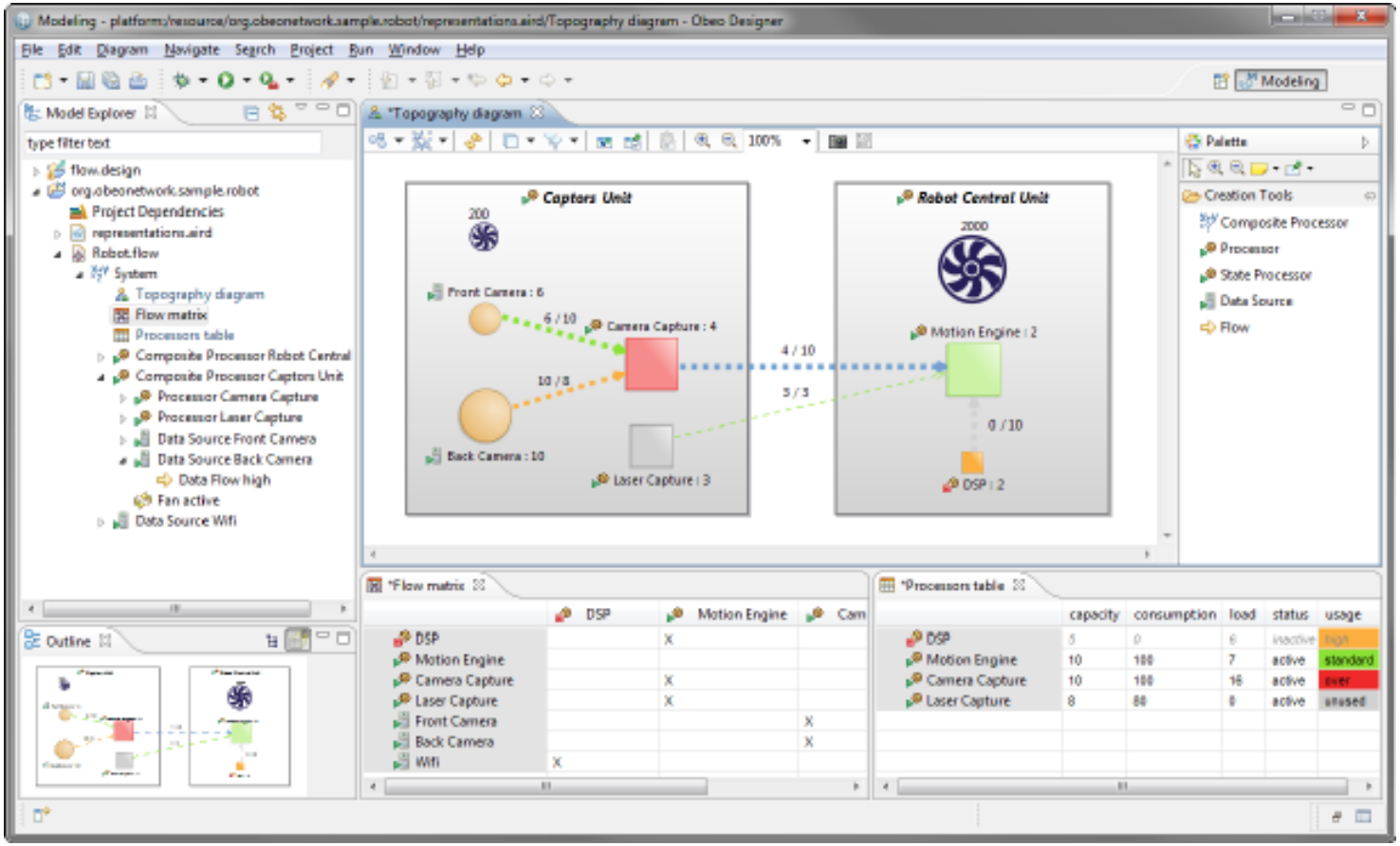
- OBEO Corpora
  - EMF (Eclipse Modeling Framework)
    - The EMF project is a modeling framework and code generation facility for building tools and other applications based on a structured data model
    - From a model specification described in XMI, EMF provides tools and runtime support to produce a set of Java classes for the model, along with a set of adapter classes that enable viewing and command-based editing of the model, and a basic editor
  - SIRIUS
    - Is an Eclipse project based on EMF

Semantic Annotation

- SIRIUS
  - A modeling workbench created with Sirius is composed of a set of Eclipse editors (diagrams, tables and trees) that allow the users to create, edit and visualize EMF models
  - The editors are defined by a model that defines the complete structure of the modeling workbench, its behavior and all the edition and navigation tools
  - For supporting specific need for customization, Sirius is extensible in many ways, notably by providing new kinds of representations, new query languages and by being able to call Java code to interact with Eclipse or any other system

Semantic Annotation

- SIRIUS

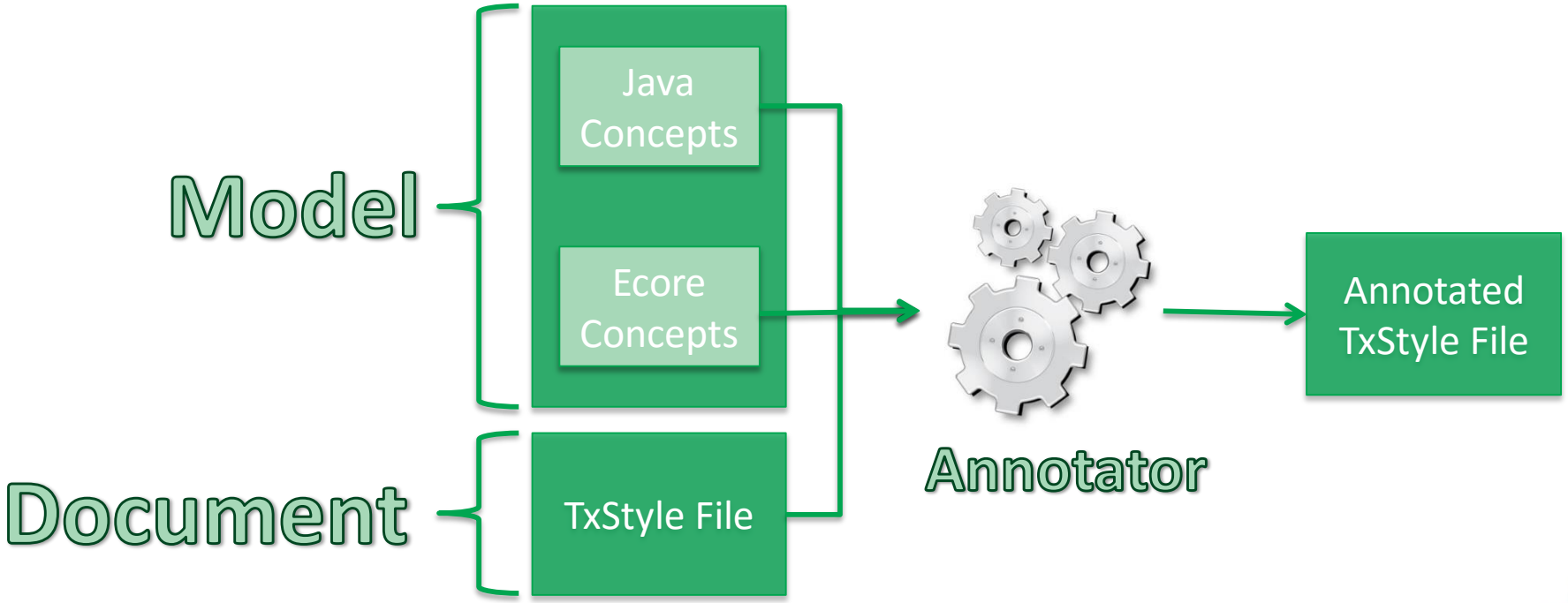


Semantic Annotation

- OBEO Corpora
  - ***Java Concepts***: a list of Java identifiers (i.e., classes, interfaces, methods, etc.) related to Sirius
  - ***Ecore Concepts***: a list of concepts related to Ecore (the EMF meta model) and to Sirius
  - ***“TxStyle" Files***: a set of files in natural language (i.e., English) related to the documentation of the application being modeled by Sirius

Semantic Annotation

- A semantic annotator
  - We have developed is a basic prototype allowing to annotate the “TxStyle” files by establishing links to Java Concepts and to Ecore Concepts



Semantic Annotation

- A semantic annotator
  - The prototype can be accessed on the GitHub Model Writer repository:
    - https://github.com/ModelWriter/WP6/tree/master/EcoreConcepts-JavaConcepts-Annotator

WP3 - Model to/from Knowledge Base Synchronization Mechanism

Moharram Challenger, R&D Director
UNIT Information Technologies R&D

WP3

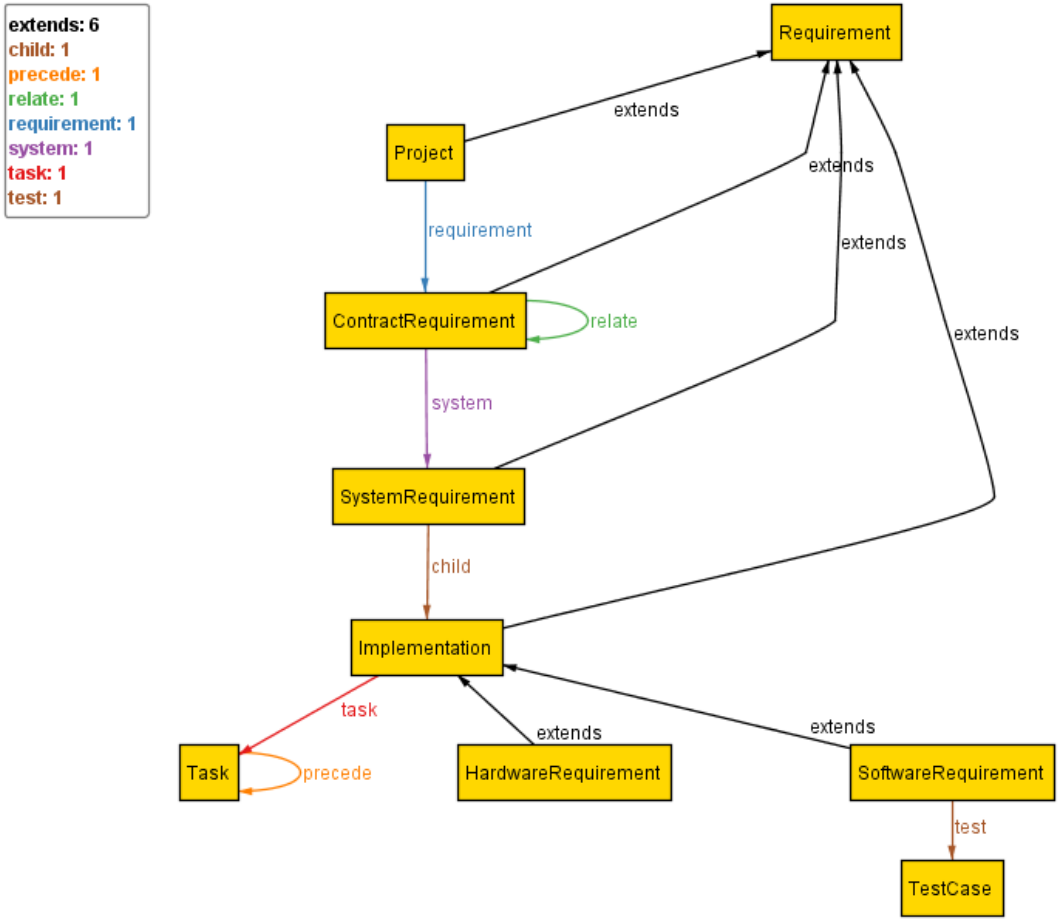
- Objective: provide the **synchronization mechanism** to keep the "**user-visible models**" consistent with the "**KB-stored models**"

- This will consist of the following main **plug-in** components:
  - **Transformation Manager**: provides the infrastructure to register and launch transformations.
  - **Configuration Manager**: for personalizing the behaviour of the framework to meet the needs of a specific standard / organization / project / individual.
  - **Traceability Manager**: keeps links between elements of user-visible models and elements of the KB.
  - **Synchronization Manager**: triggering transformations when synchronization is needed.

Tasks

- T3.1 - Review of M2M transformation approaches
- T3.2 - Specification and design of the M2M Transformation Framework
- T3.3 – Development of the **Transformation Manager** component
- T3.4 – Development of the **Configuration Manager** component
- T3.5 – Development of the **Traceability Manager** component
- T3.6 – Development of the **Synchronization Manager** component
- T3.7 – Design of the model-to-model transformations
- T3.8 – Implementation of the model-to-model transformations
- T3.9 – Validation of the M2M Transformation Framework

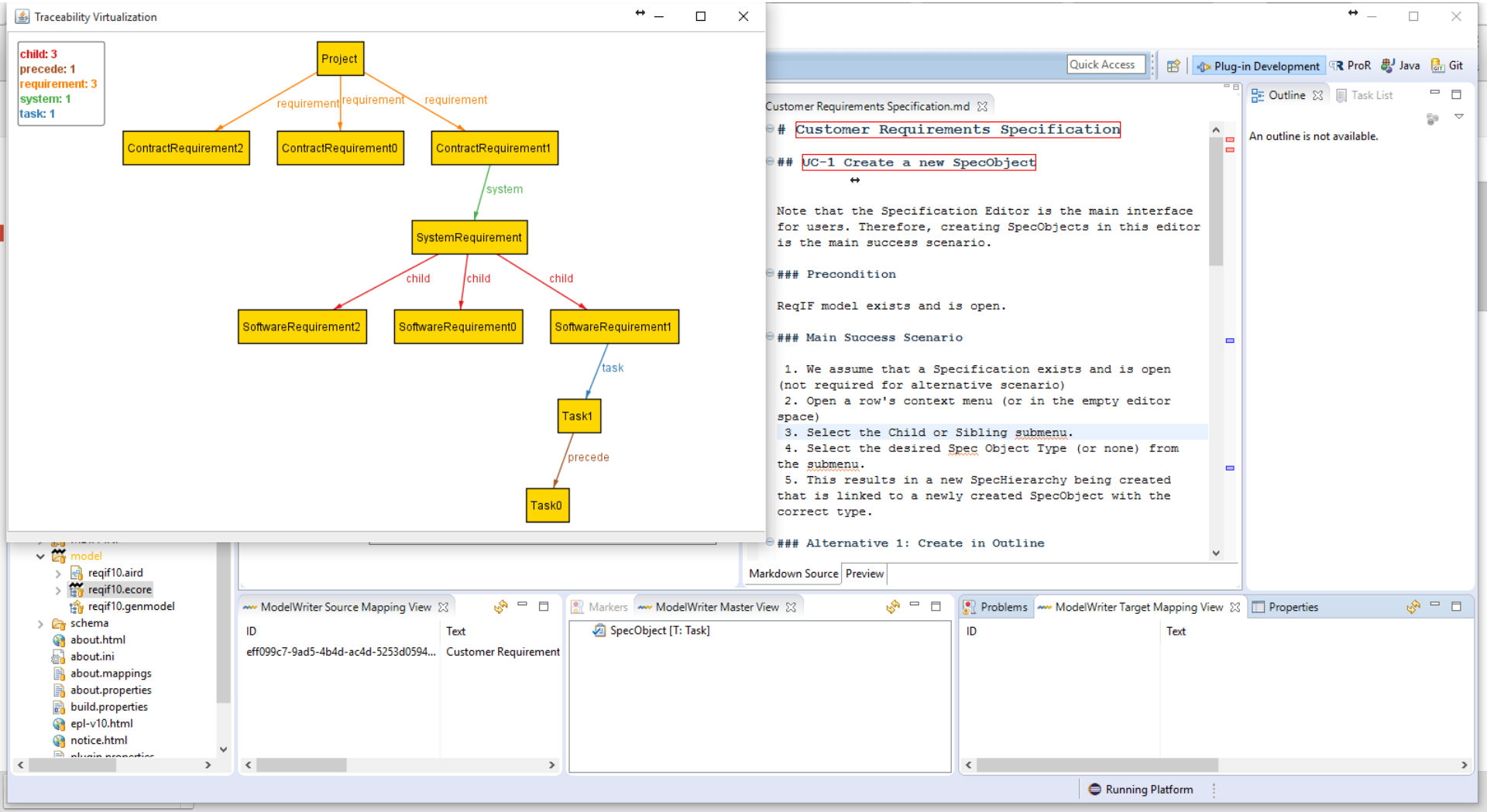
Configuration: Havelsan example




```
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

Traceability: Havelsan example



A Formal Specification Model to configure the ModelWriter

T3.1 - Review of M2M transformation approaches

- UNIT, KOCSISTEM
  - A systematic review of model-to-model transformation approaches, and a selection of the most convenient and widely used in the industry for inclusion into the ModelWriter tool

- Status
  - Survey of available approaches and tools are available at:
    - D3.1.1Review of model-to-model transformation approaches and technologies

- Next:
  - The document may be updated based on the new approaches and tools in SotA

T3.2 - Specification and design of the M2M Transformation Framework

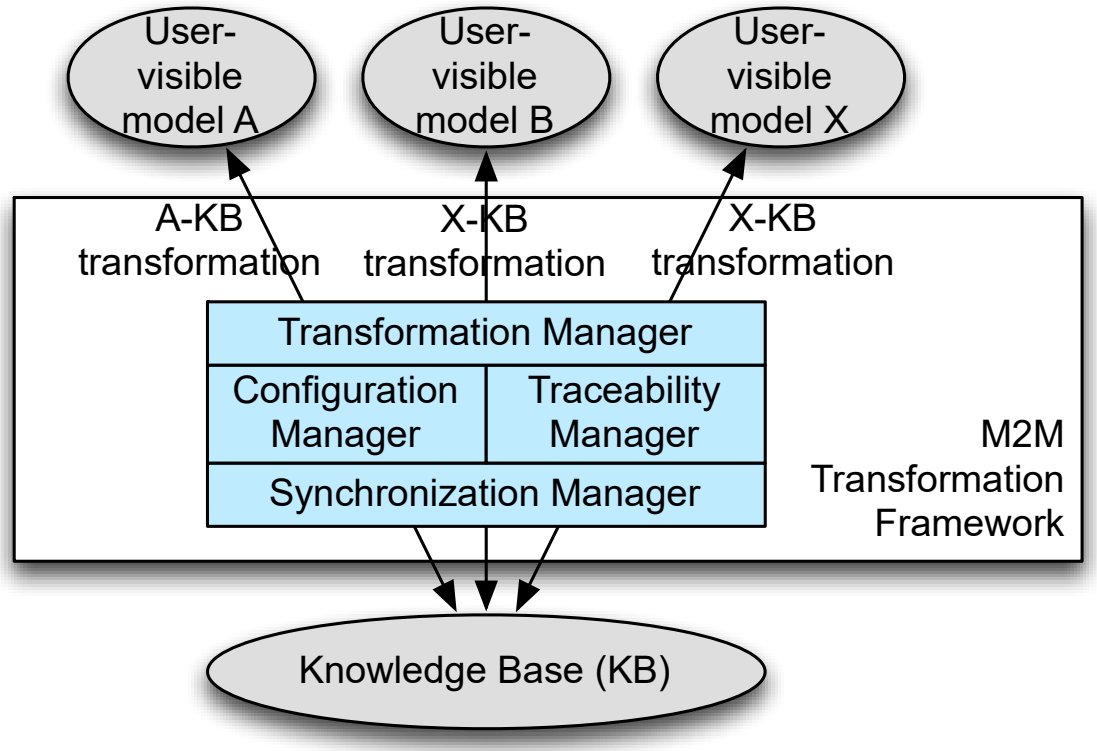
- UNIT + KOCSISTEM
  - Objective: Designing the M2M Transformation Framework whose main goal is to make the ModelWriter tool able to launch M2M transformations

- Status:
  - D3.2.1 - M2M Transformation Framework architectural design document (incl. Transformation, Configuration, Traceability, and Synchronization architectural design)

- Next:
  - The architecture may be updated based on the new needs during the project progress.

T3.2 - Specification and design of the M2M Transformation Framework

- Overview of the components of the M2M Transformation Framework:



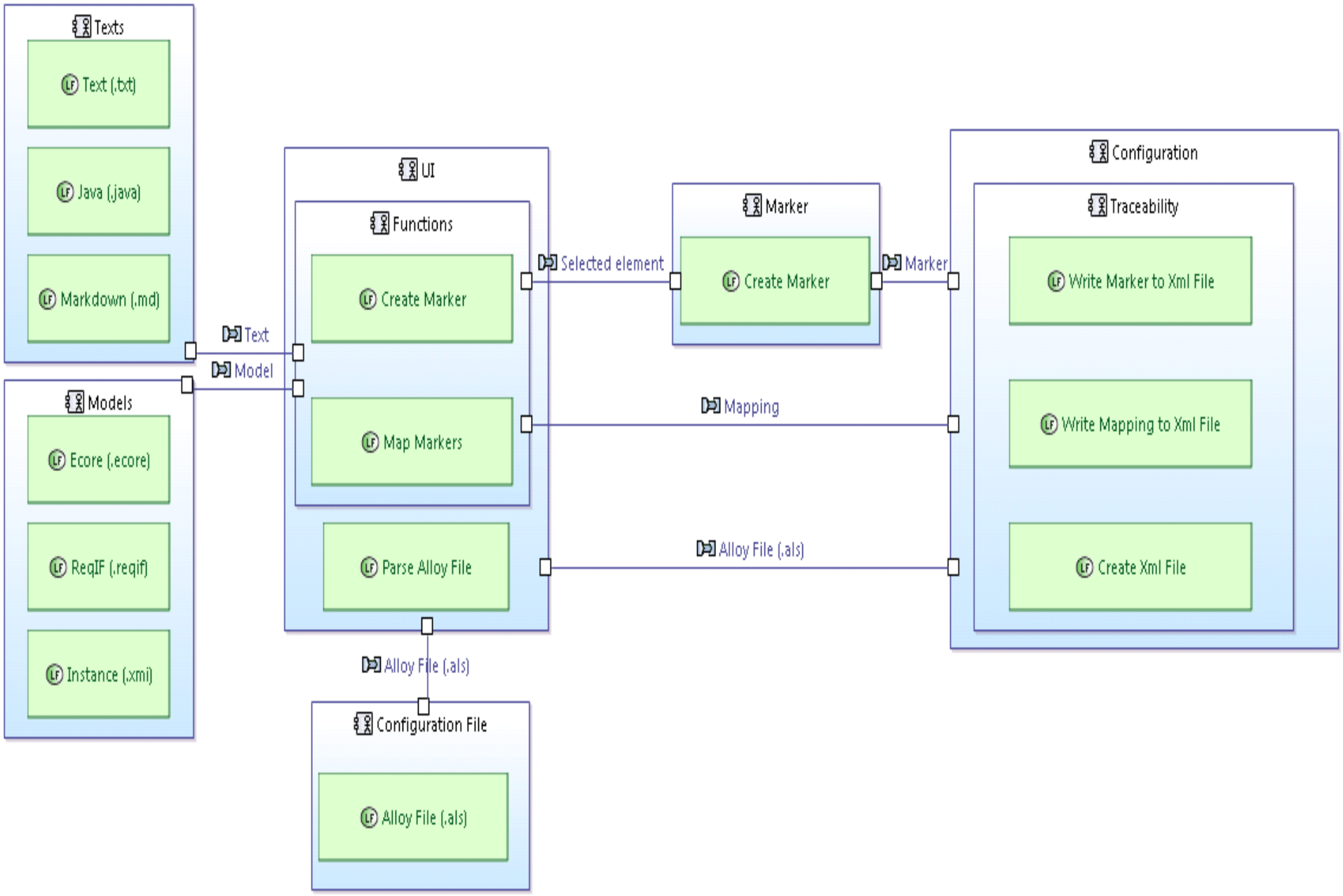
T3.3, T3.4, T3.5, T3.6 (UNIT + KOCSISTEM)

- Development of the:
  - T3.3 Transformation Manager component
  - T3.4 Configuration Manager component
  - T3.5 Traceability Manager component
  - T3.6 Synchronization Manager component

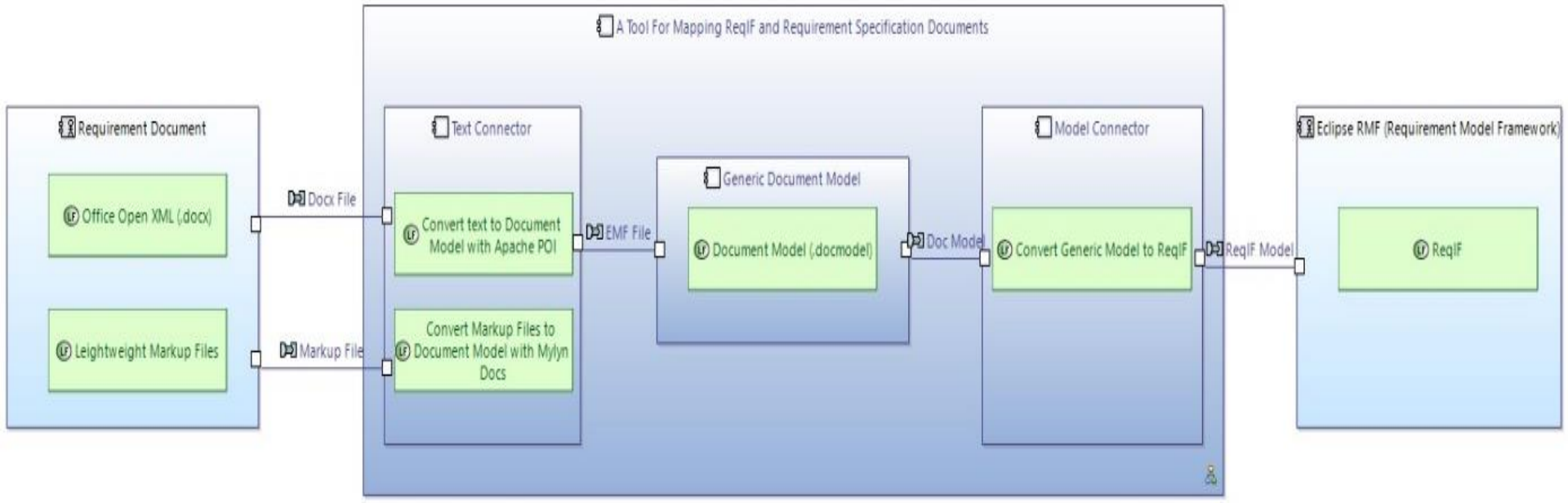
- Status:
  - These tasks has software deliverables which are developed and are available at GitHub

- Next:
  - These components will be updated.

T3.3, T3.4, T3.5, T3.6 (UNIT + KOCSISTEM)



T3.3, T3.4, T3.5, T3.6 (UNIT + KOCSISTEM)



T3.3, T3.4, T3.5, T3.6 (UNIT + KOCSISTEM)

- The fully functional demonstration of the main components of WP3 (T3.3, T3.4, T3.5, T3.6) will be presented at demonstration session.

WP4 – Knowledge base Design and Implementation

Prof. Dr. Erhan Mengusoglu
MANTIS

WP4 Knowledge Base Design and Implementation

- Design and implement the ModelWriter's federated Knowledge Base itself, hosting multiple formalisms.
- Design and implement its bi-directional text-model synchronization mechanism.
- Design and implement its API.
- Design and implement a set of specialised modules (plug-ins) that exploit the Knowledge Base in ways that make the tasks of Technical Authors much more productive, e.g. consistency checks.
- Design and implement the collaborative functions linking and hierarchically organizing multiple ModelWriter KBs used by different Technical Authors on different sites.

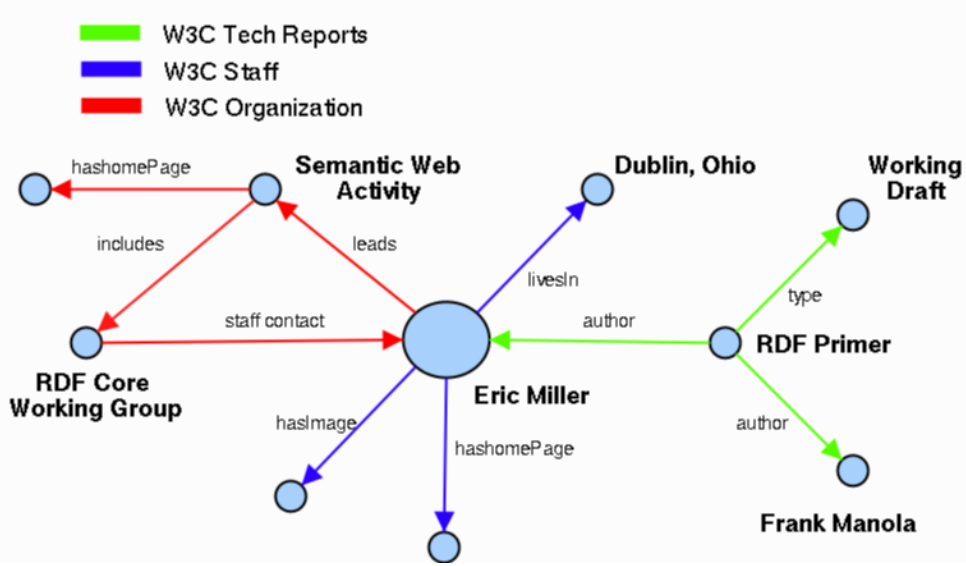
WP4 Knowledge Base Design and Implementation

- Plug-in #1 – This provides consistency and completeness checks within the same software lifecycle document, allowing automatic quality review of the content (meaning).
- Plug-in #2 – This provides consistency and completeness checks between related set of documents.
- Plug-in #3 – This provides semantic comparison between two versions of the same software lifecycle document (i.e. what conceptual changes have happened).

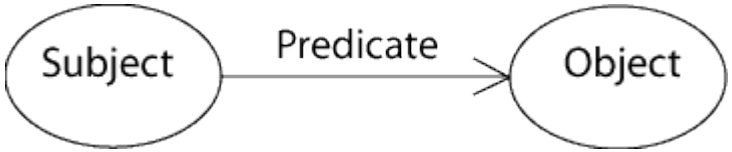
WP4 Knowledge Base Design and Implementation

- T4.1 - Design of the Knowledge Base (MANTIS + OBEO + KUL2 + UNIT + KOCSISTEM)
- T4.2 - API of the Knowledge Base (KOCSISTEM + KUL2 + + OBEO + UNIT + HISBIM)
- T4.3 - Implementation of the Knowledge Base (KUL2 + MANTIS + HISBIM)
- T4.4 – Plug-in #1: ModelWriter-assisted requirements review (KUL2 + MANTIS)
- T4.5 – Knowledge Base serialization and reuse plug-in (MANTIS)
- T4.6 – Plug-in #3: ModelWriter-assisted semantic comparison of 2 documents (OBEO + MANTIS + HISBIM)
- T4.7 – Plug-in #2: ModelWriter-assisted compliance review (MANTIS + UNIT + AIRBUS +SOGETI)
- T4.8 – Internal bi-directional synchronization mechanism (OBEO + UNIT)
- T4.9 – External synchronization mechanism for collaborating ModelWriters (HISBIM)

WP4 - T4.1 – Design of the Knowledge Base



Semantic Web



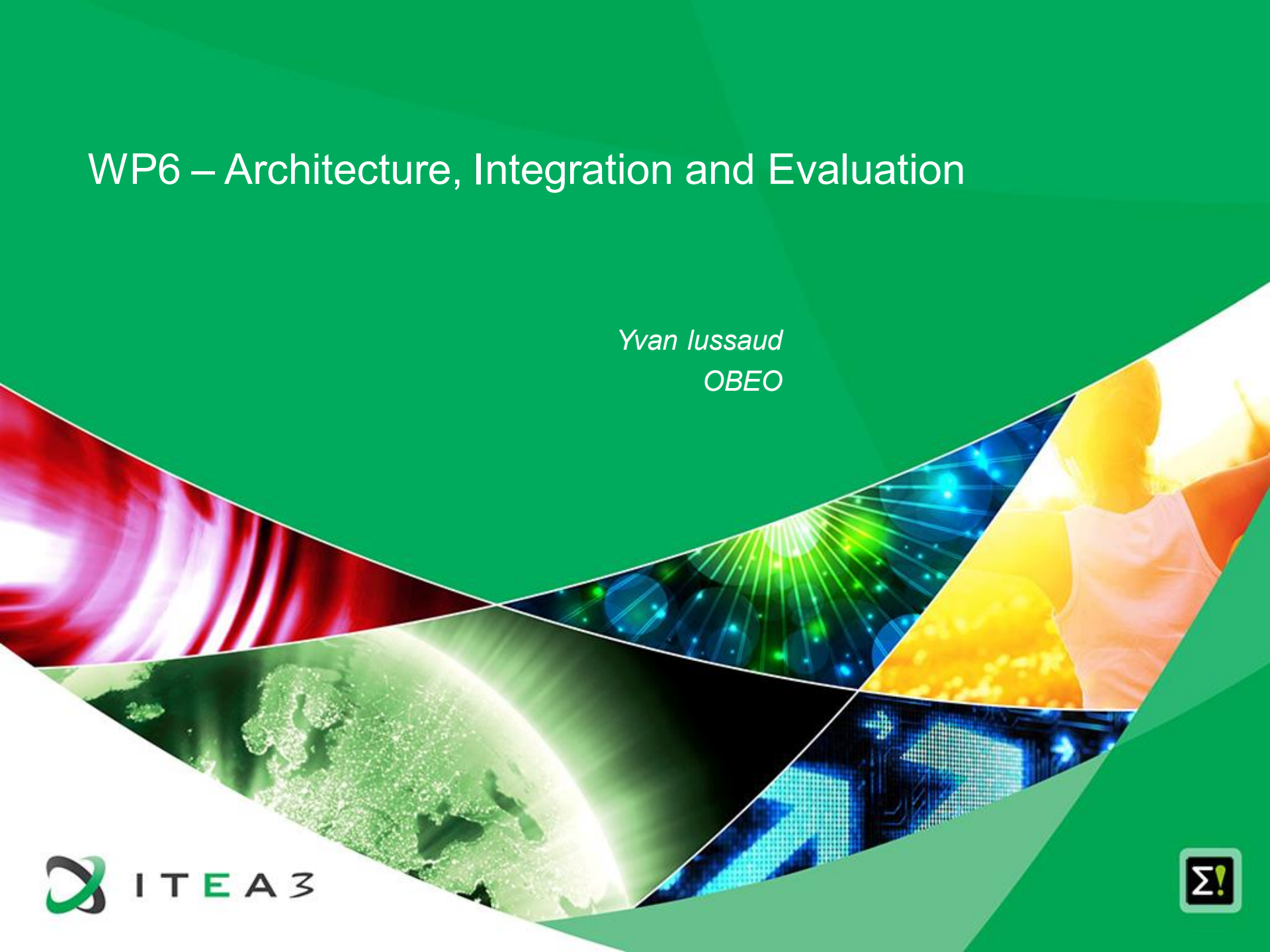
RDF Expression

WP5 – Project Management

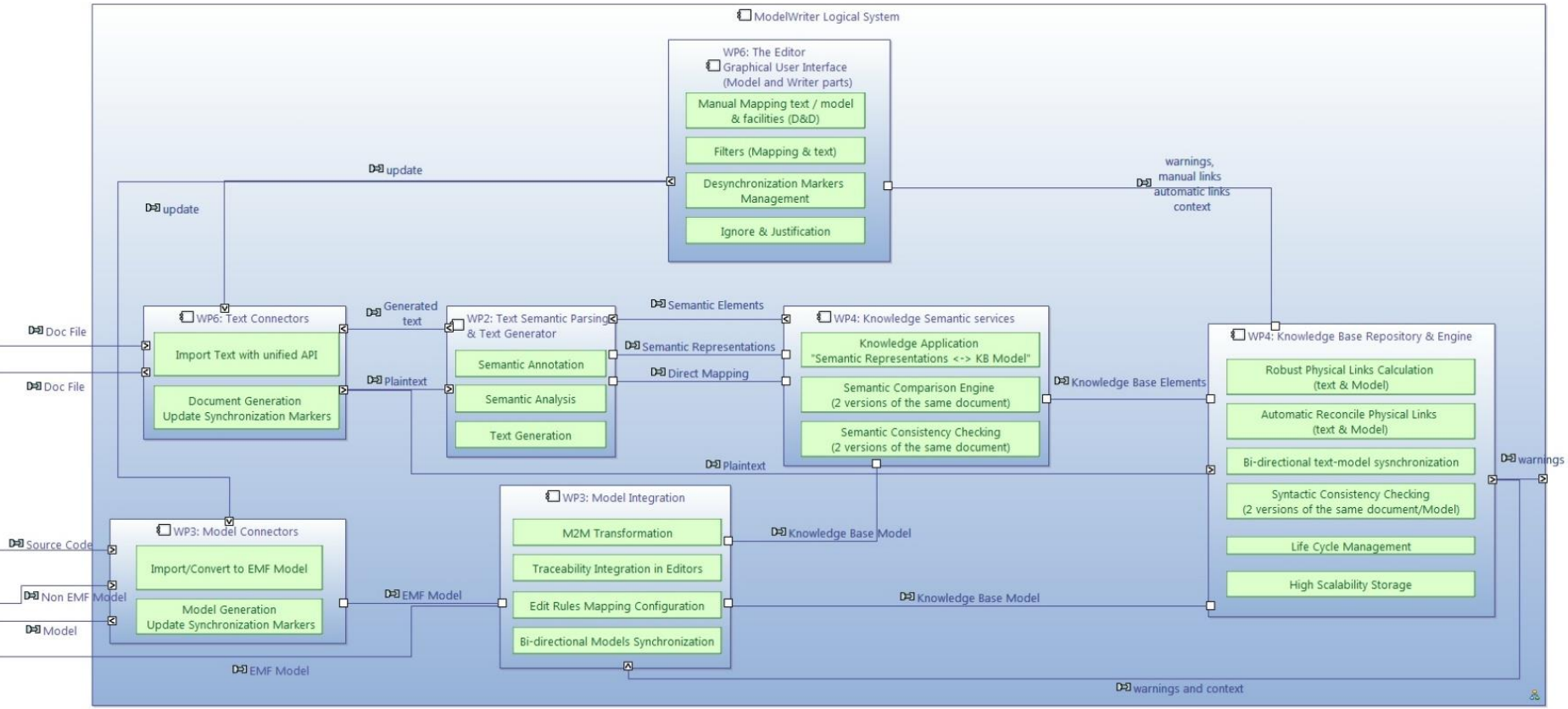
Moharram Challenger, R&D Director
UNIT Information Technologies Ltd.

WP6 – Architecture, Integration and Evaluation

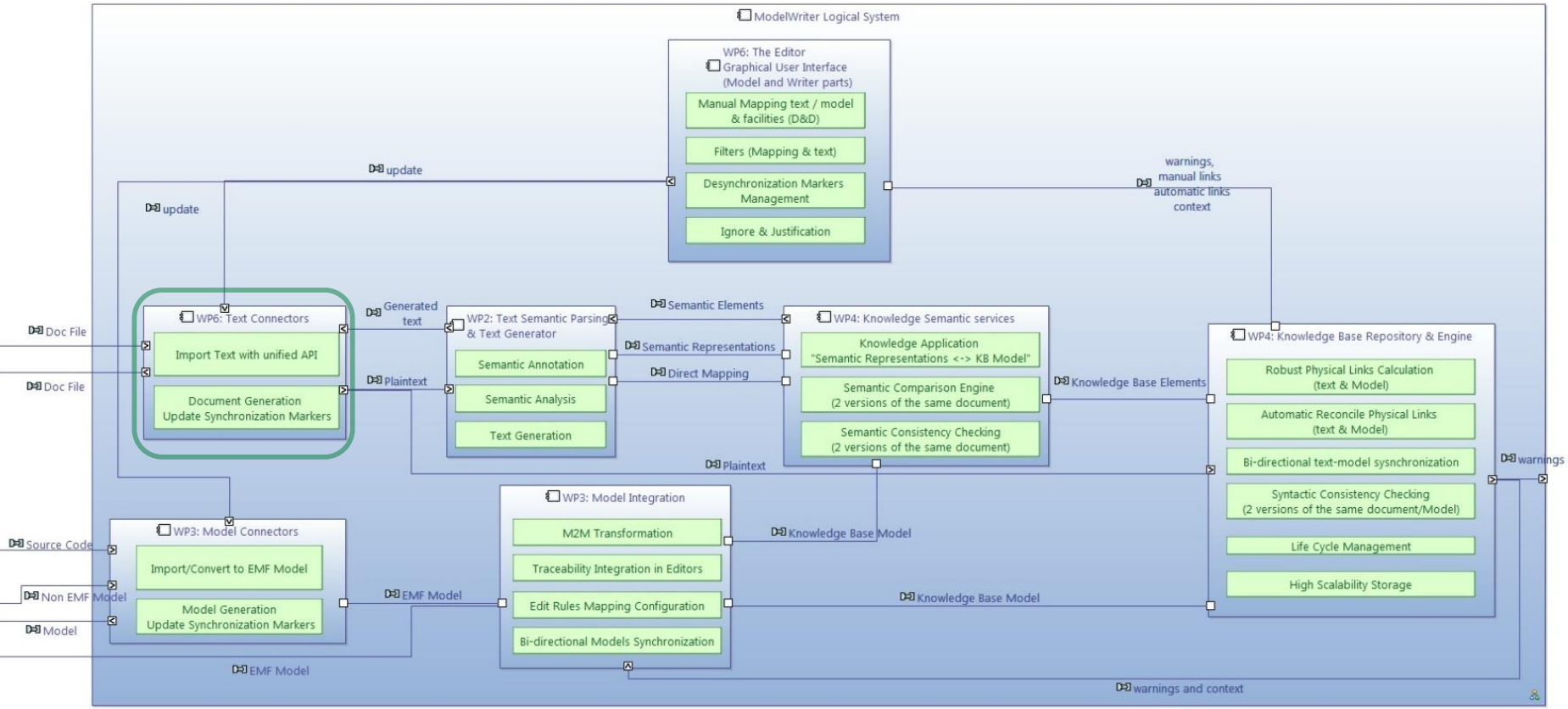
Yvan Iussaud
OBEO



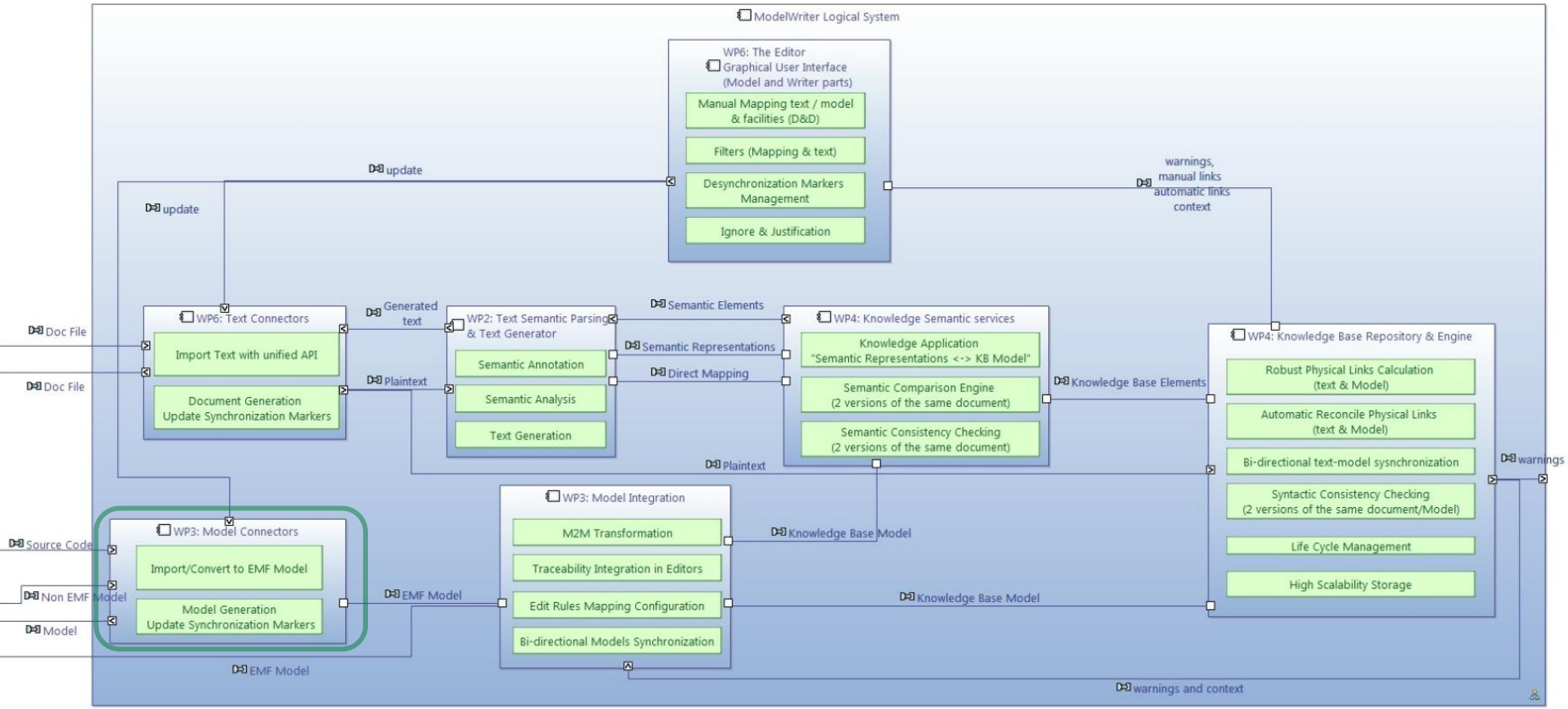
WP6
ModelWriter Functional Architecture



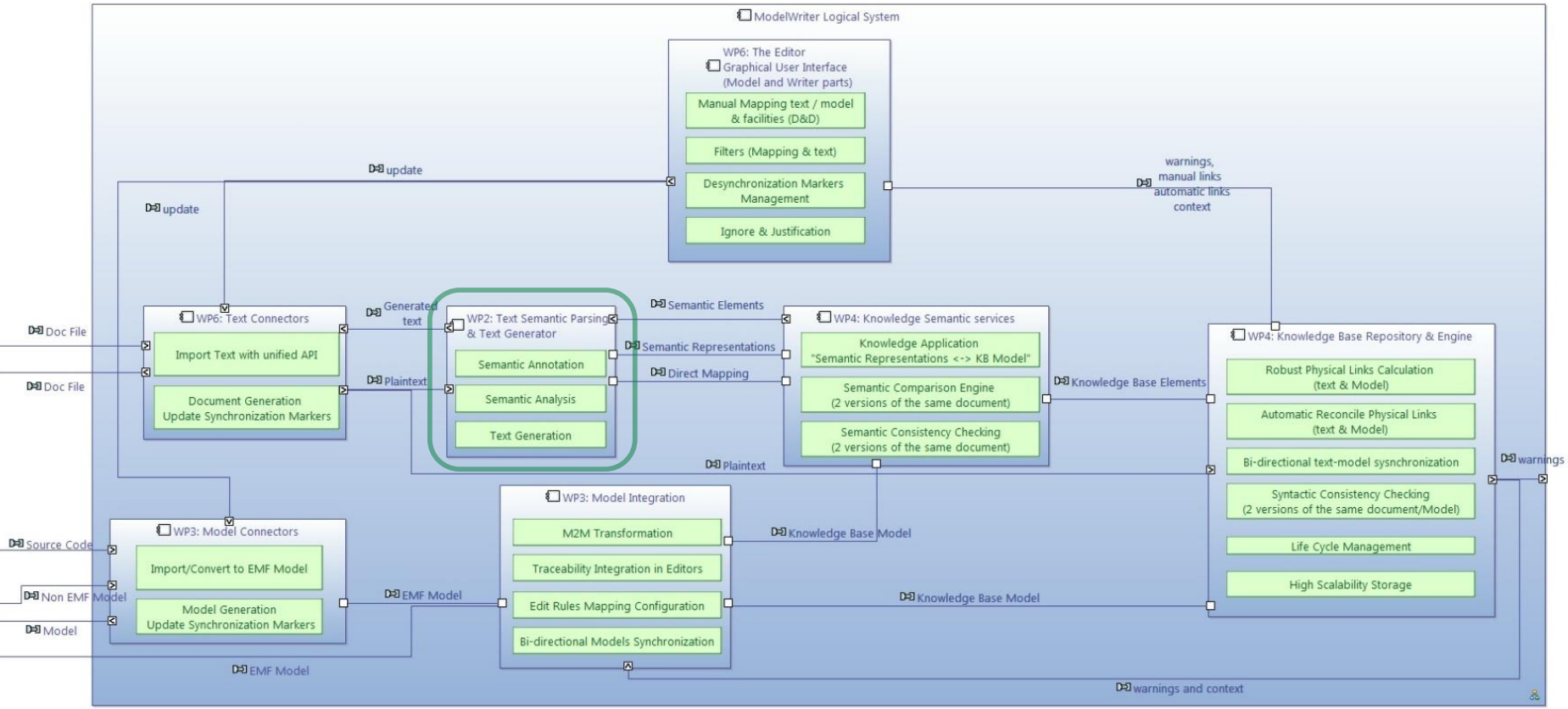
WP6
Text connectors



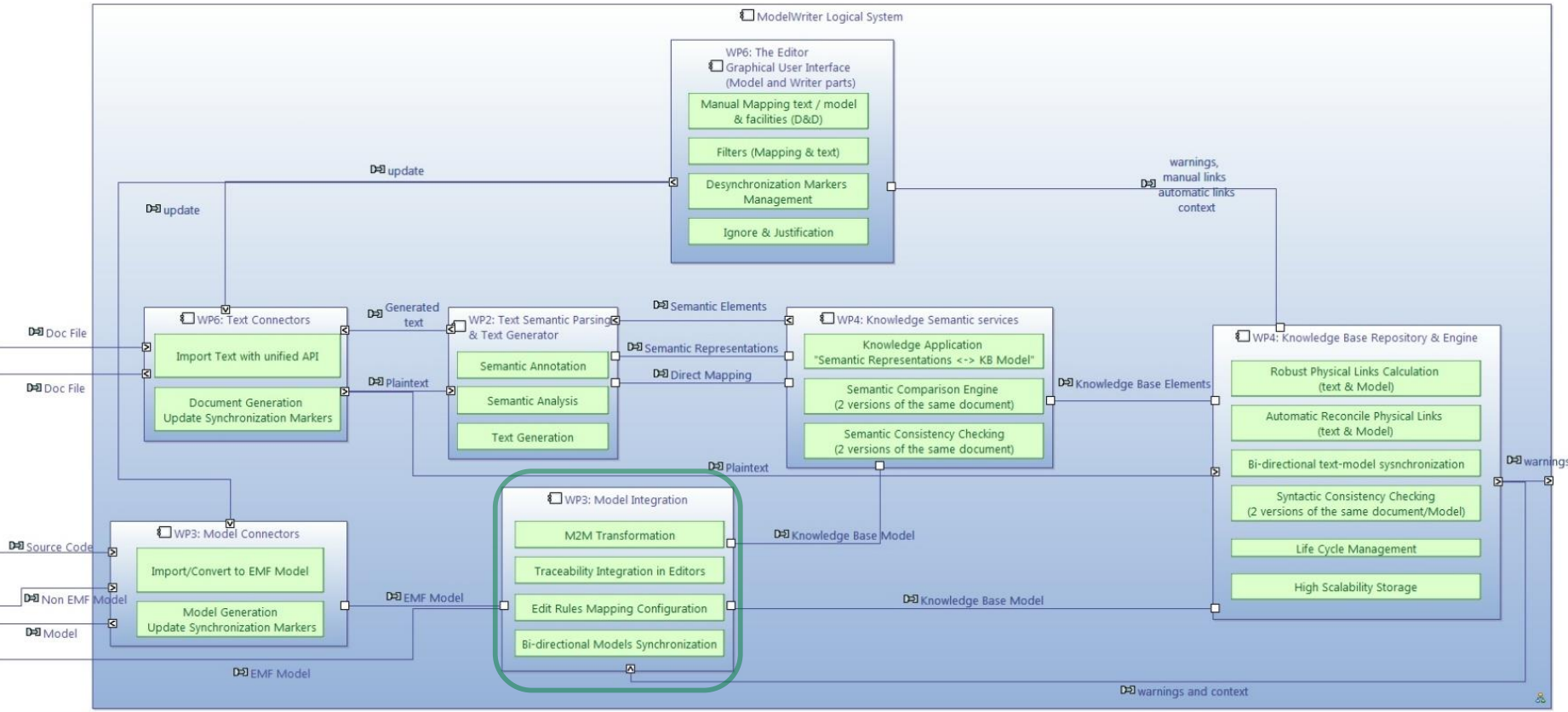
WP6
Model connectors



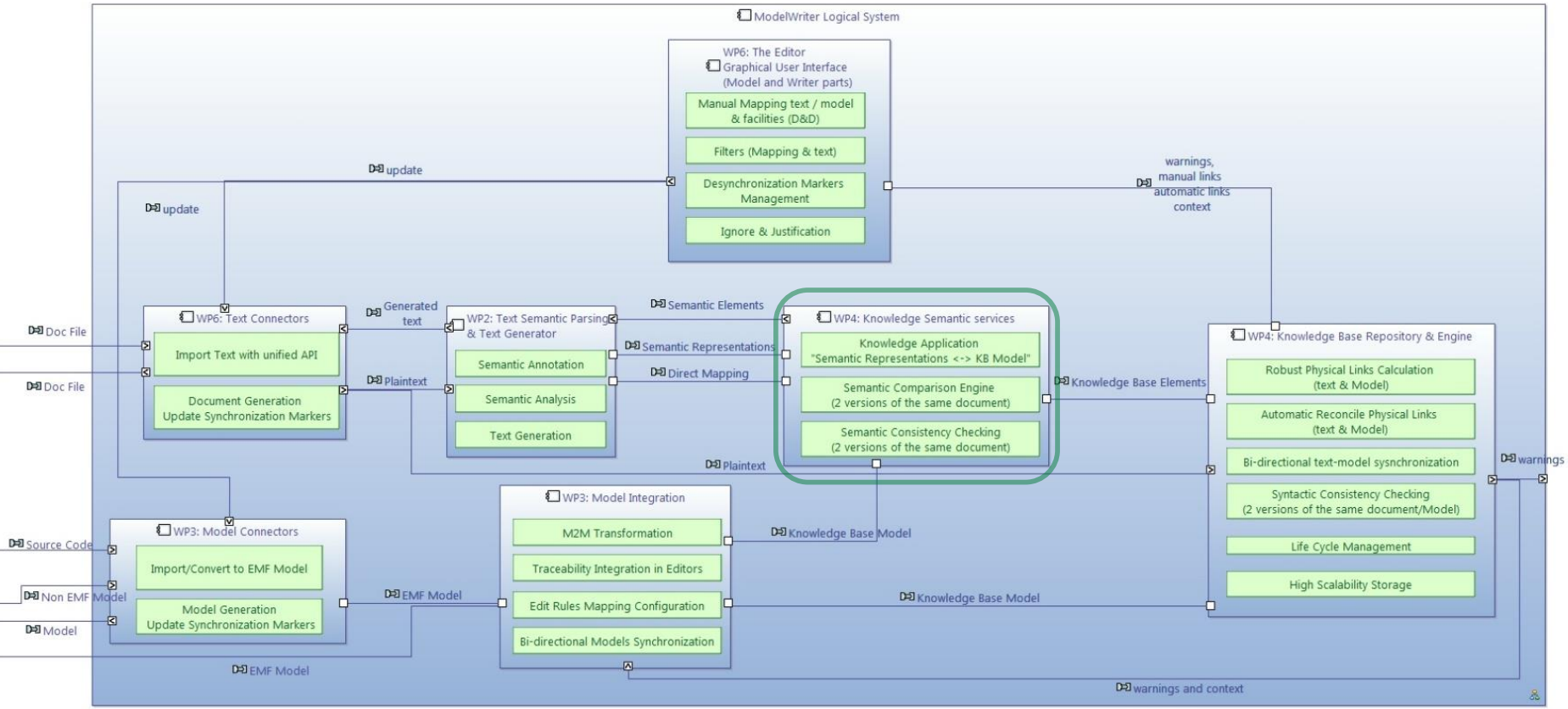
WP6
Text semantic parsing and text generator



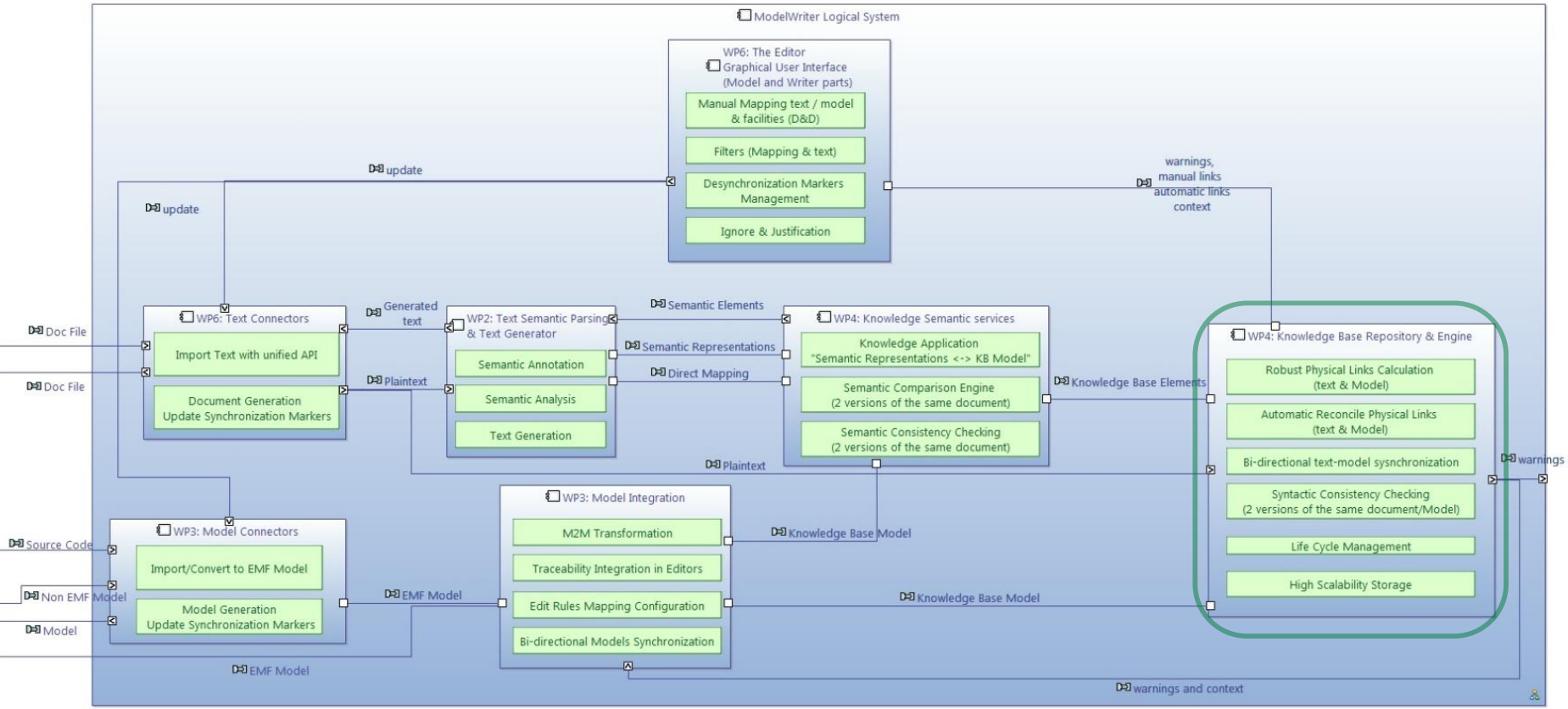
WP6
Model Integration



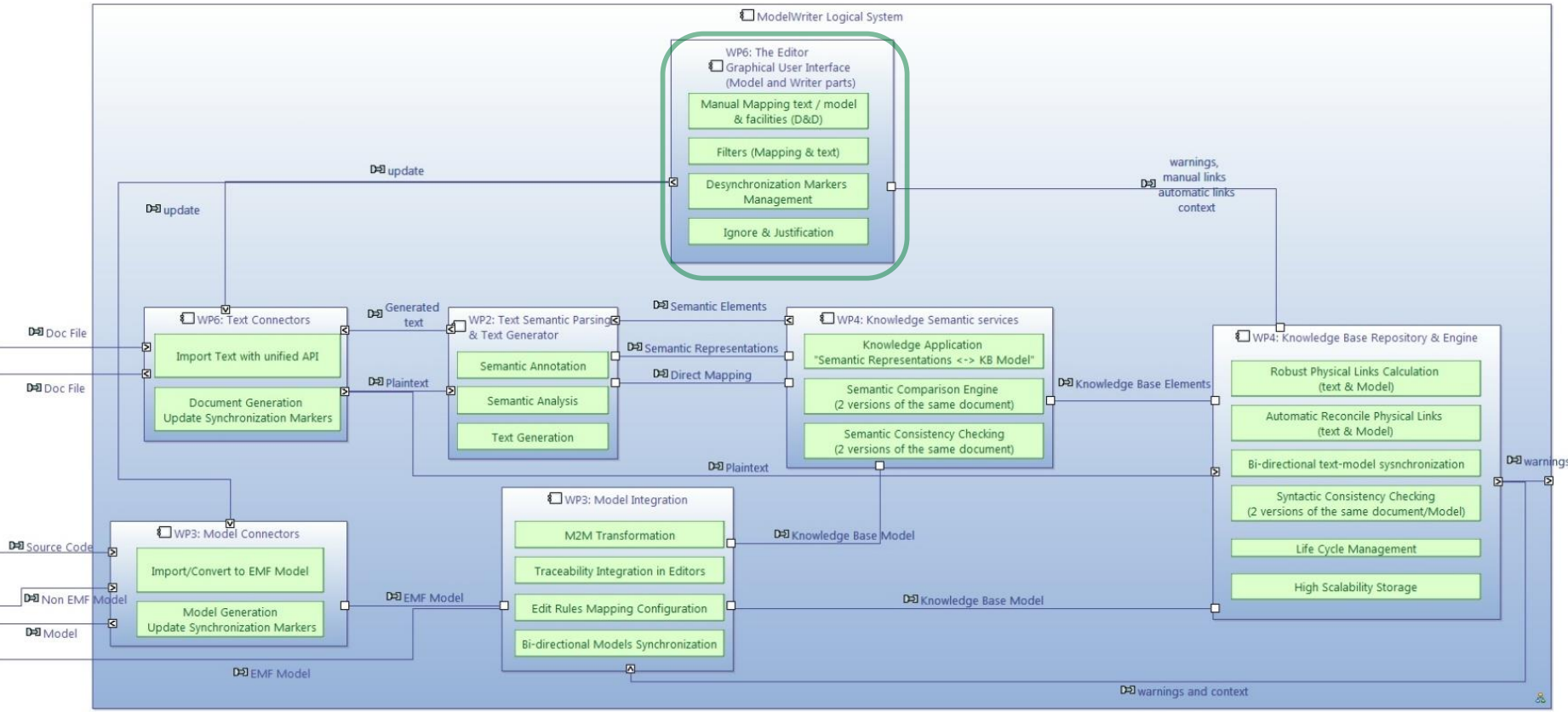
WP6
Knowledge Semantic services



WP6
Knowledge base repository and engine



WP6
Editors and GUI (Model and Writer parts)



WP6
Integration

Source code

- Github repository
- Checkstyle and code templates
- Target platforms

Continuous integration

- Jenkins
- Ease the release process

Next steps

- integrate existing components
- Provide an update site and an Eclipse product

WP6
Evaluation

Unit testing
- JUnit
- Code coverage (Eclemma)

Integration testing
- Functional testing via GUI (RCPTT)
- Jenkins will run all tests on a daily basis

Use cases
- Drives features and enhancements
- Milestone functional testing

WP7 – Standardization, Dissemination and Exploitation

Yvan Iussaud
OBEO

ITEA 3

WP7
Standardization

Specification and verification of ALM platform
- Open source software - traceability

Change impact analysis and visualization
- Open source software – relational calculus

System installation component ontology
- De facto standard – Airbus vocabulary

Semantic annotator
- Open source software – API for text annotation

Synchronization engine prototype
- Open source software – Eclipse Intent contribution

WP7
Dissemination

International ModelWriter workshops
- 5 workshops – 2 open workshops

Parsing text into RDF
- Publication poster – propose a RDF-based method for querying the content of a text

5th Turkish software Architecture conference
- Develop an open source community for model and text synchronization

Keynote on text generation at SEPLN 2015
- Spanish Natural Language Processing Conference – academics and industrials – a project session

Keynote speech at International Workshop on Advanced Topic in Software engineering
- Present Eclipse ecosystem and Modeling approach to software engineering

WP7
Exploitation

Collaboration between UNIT and HAVELSAN
- Traceability in ALM platform – applied to Microsoft Team Foundation Server – support of KoçSistem

Requirement documents and ReqIF standard synchronization
- Prototype – automatic synchronization between ReqIf models and requirement documentation

CSV to OWL transformation
- Generates a triple dataset to populate the SIDP (System Installation Design Principle) rule model

SIDP installation rule model
- Model of SIDP installation rules using RDFS and OWL languages

WP7
Exploitation

Enhancement in text connector for Airbus
- Syntactical parsing of SIDP rules based on templates

Collaboration/Participation of FORD-Otosan
- Long term support – semantic parsing and traceability for Product Life Cycle documents

Collaboration between Obeo and Airbus
- Discussions on topics related to the ModelWriter scope

Expertise on document extraction
- Improve expertise on information extraction for reverse engineering purpose

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

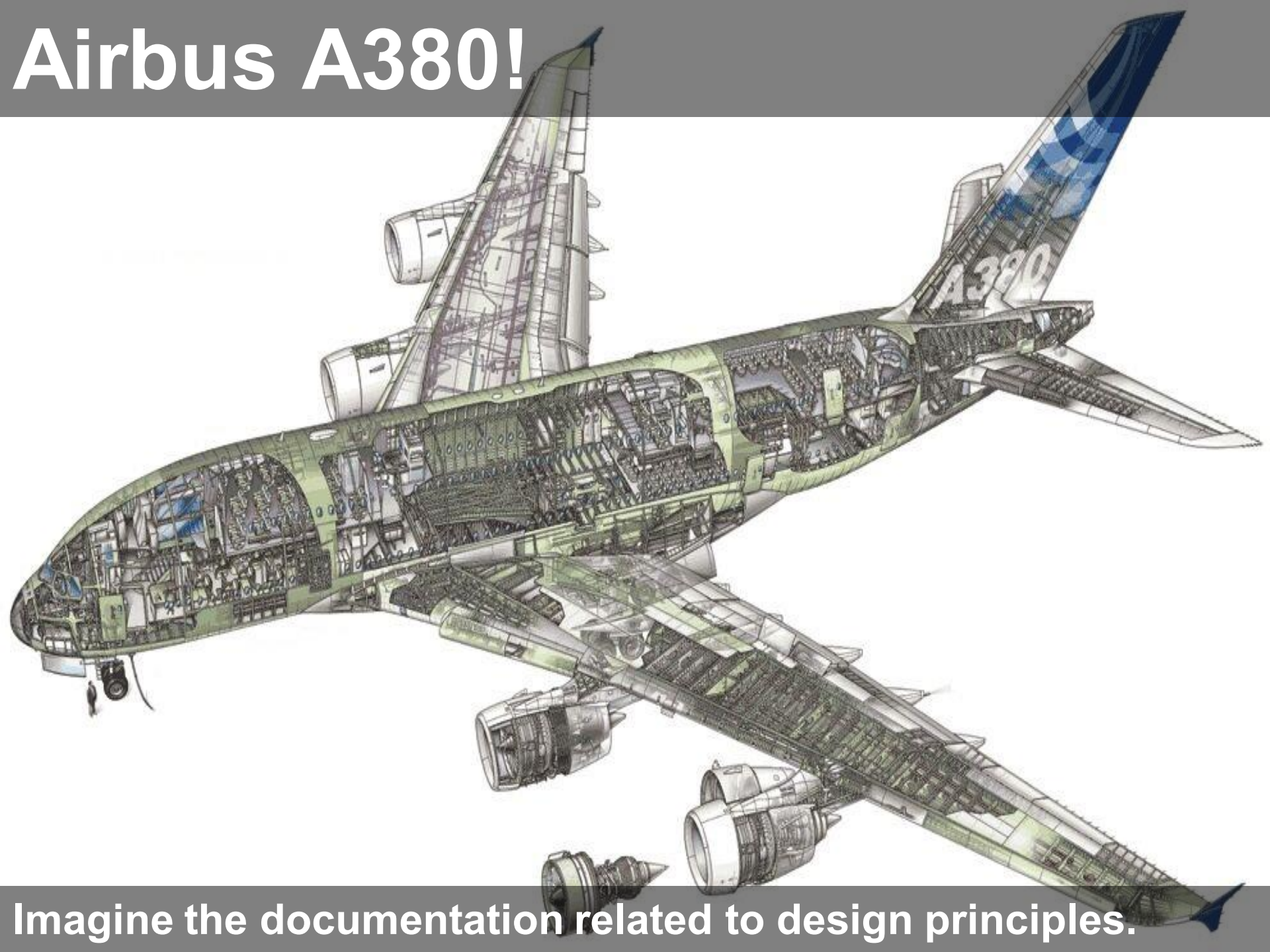
5 The Project Idea with Demonstrations

Ferhat Erata (UNIT, ModelWriter Project Leader)
Dr. Mariem Mahfoudh (CNRS/LORIA)

ITEA 3

What is the problem?

Airbus A380!

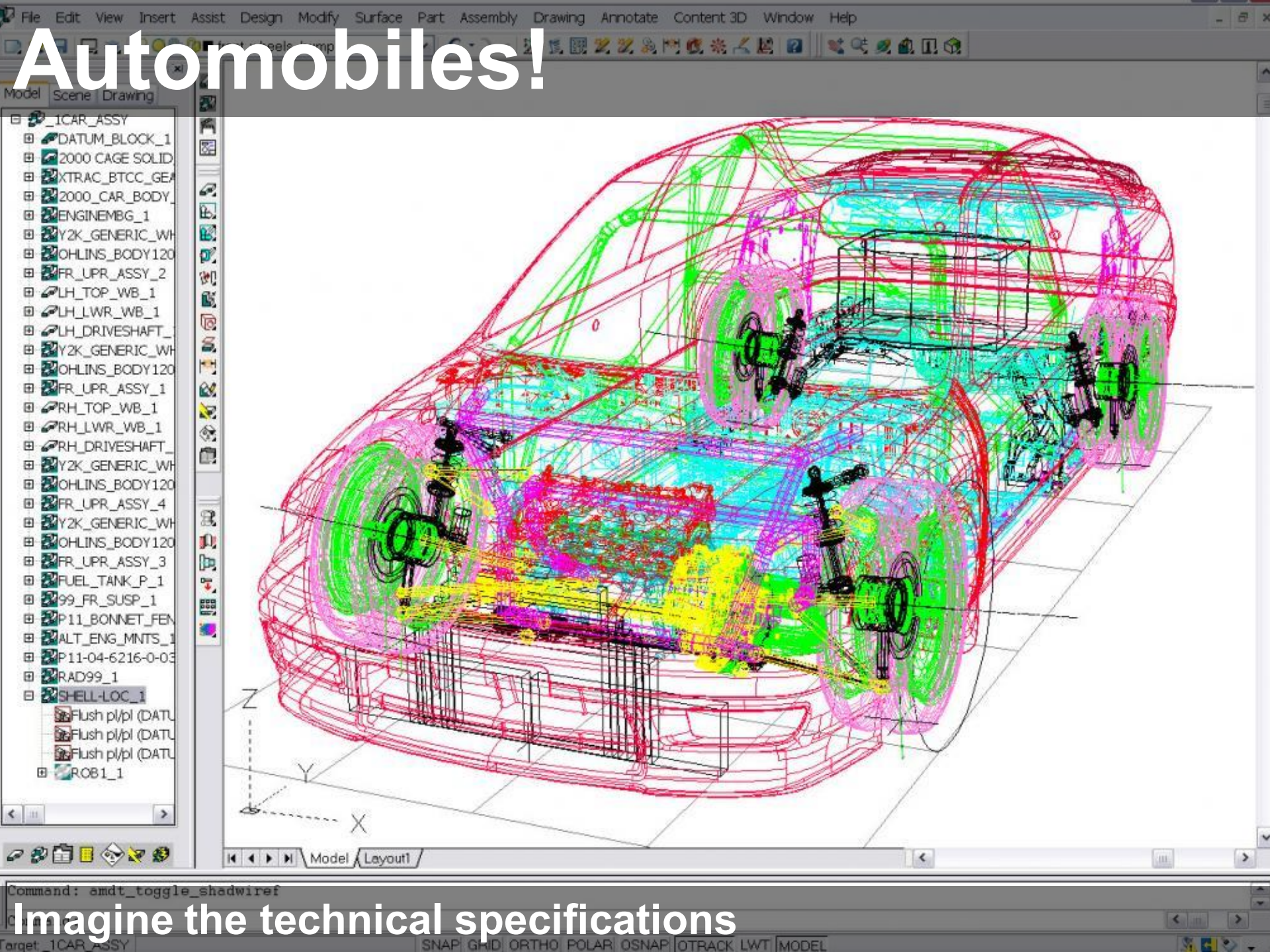


Imagine the documentation related to design principles.

Plants!

Imagine the documentation of a construction site.

Automobiles!

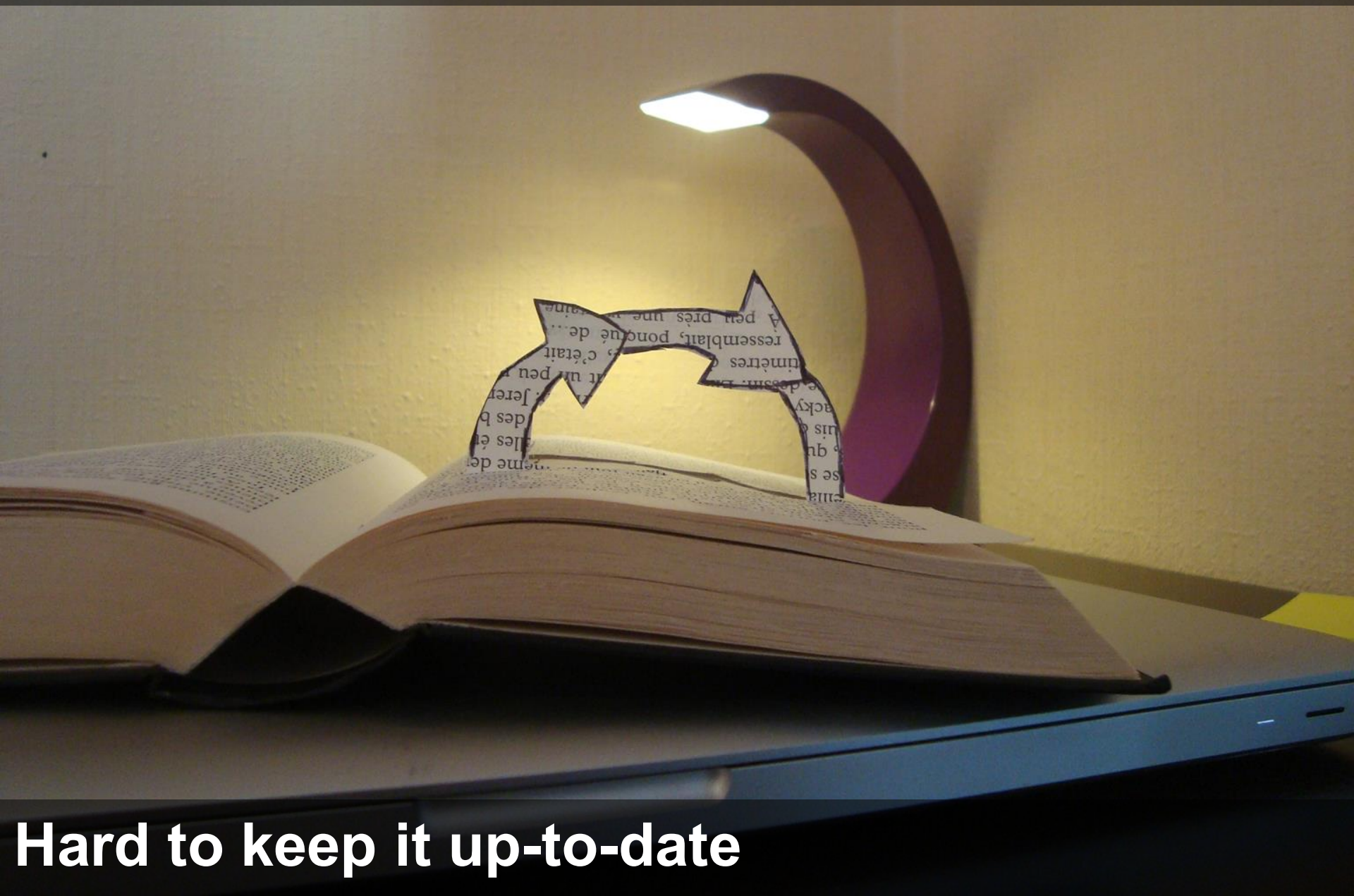


Imagine the technical specifications

Documentation!

Write once … and never look at after

Documentation!

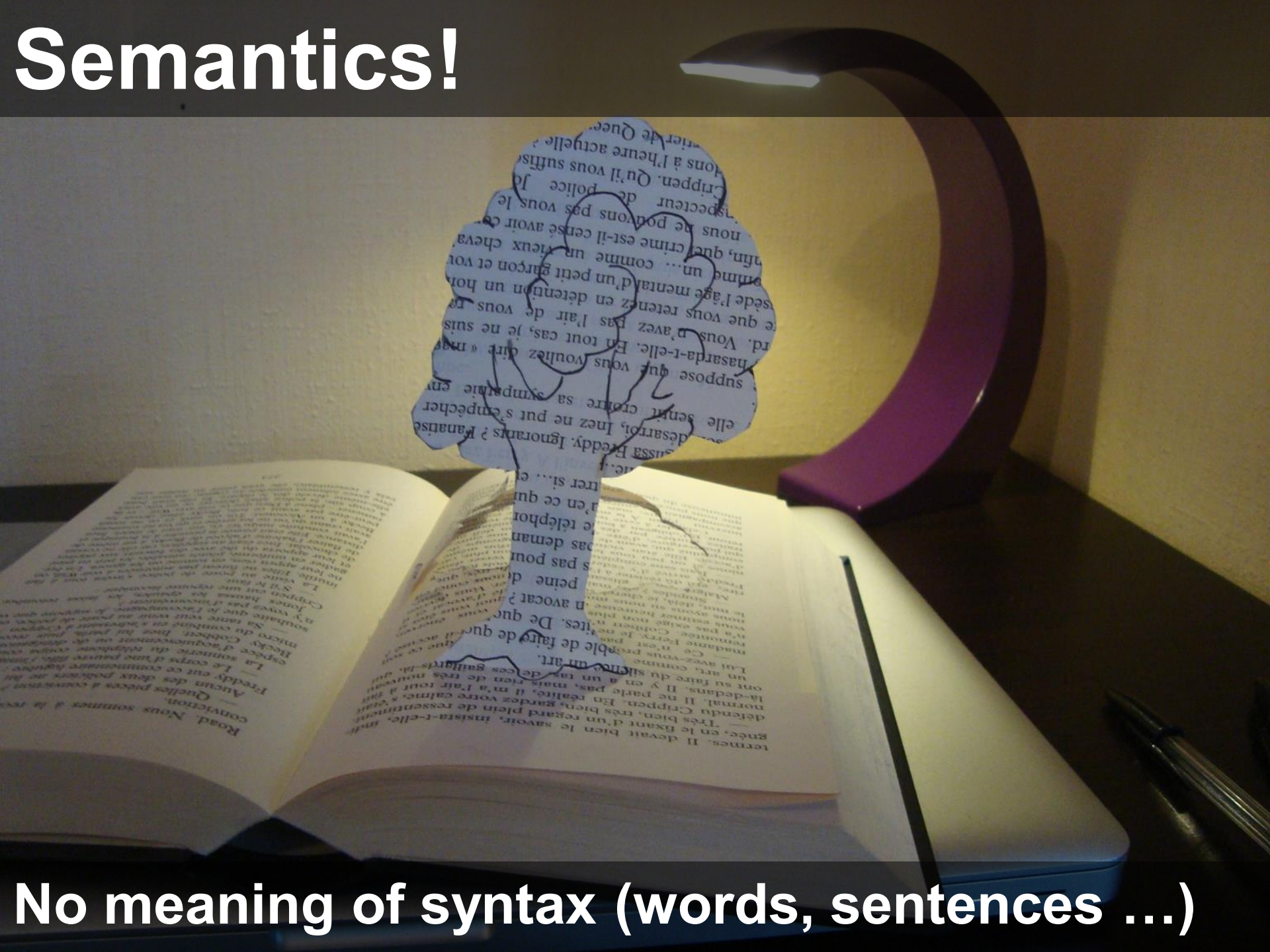


Hard to keep it up-to-date

Documentation!

Useful for users / Boring to write

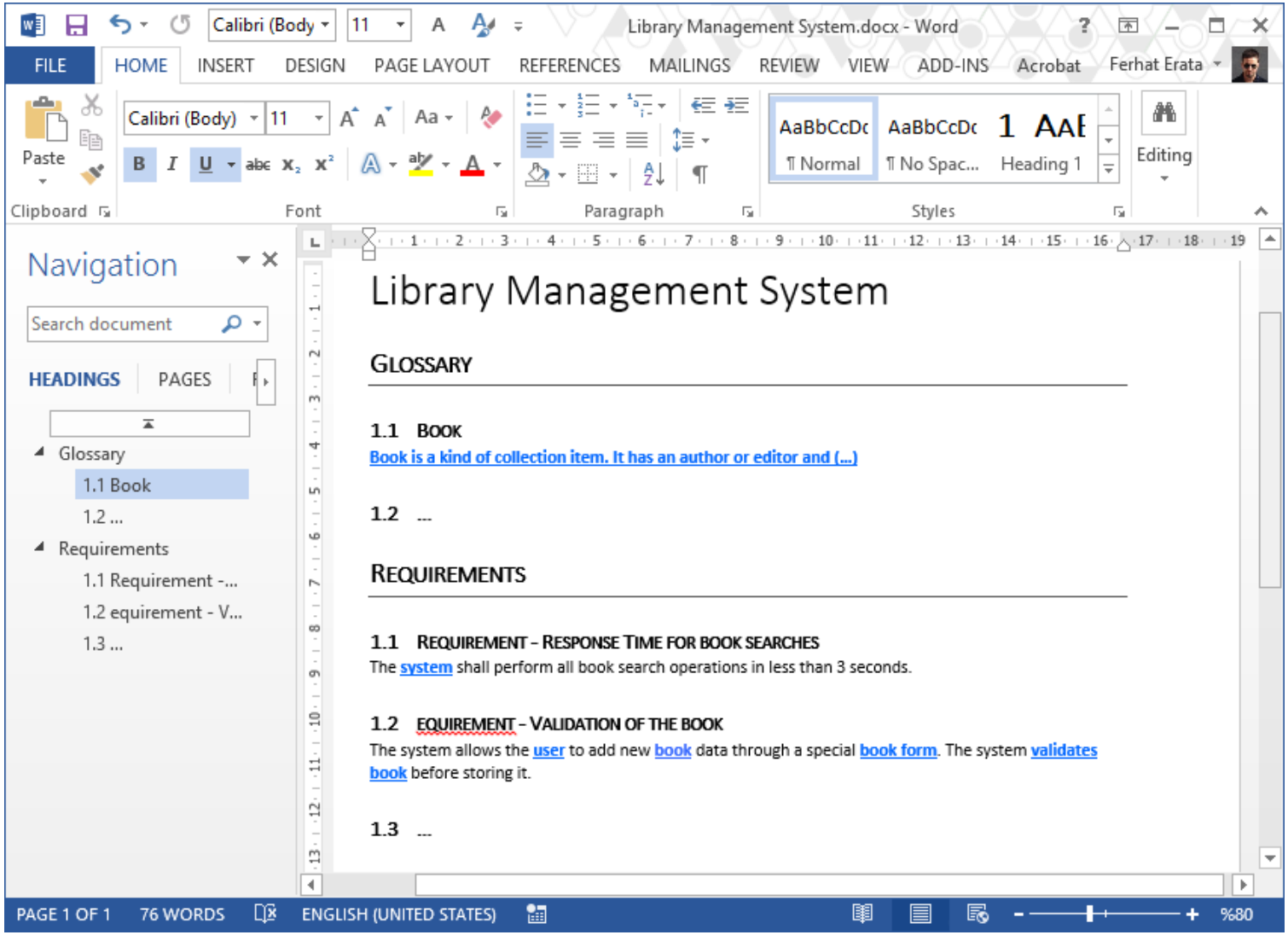
Semantics!



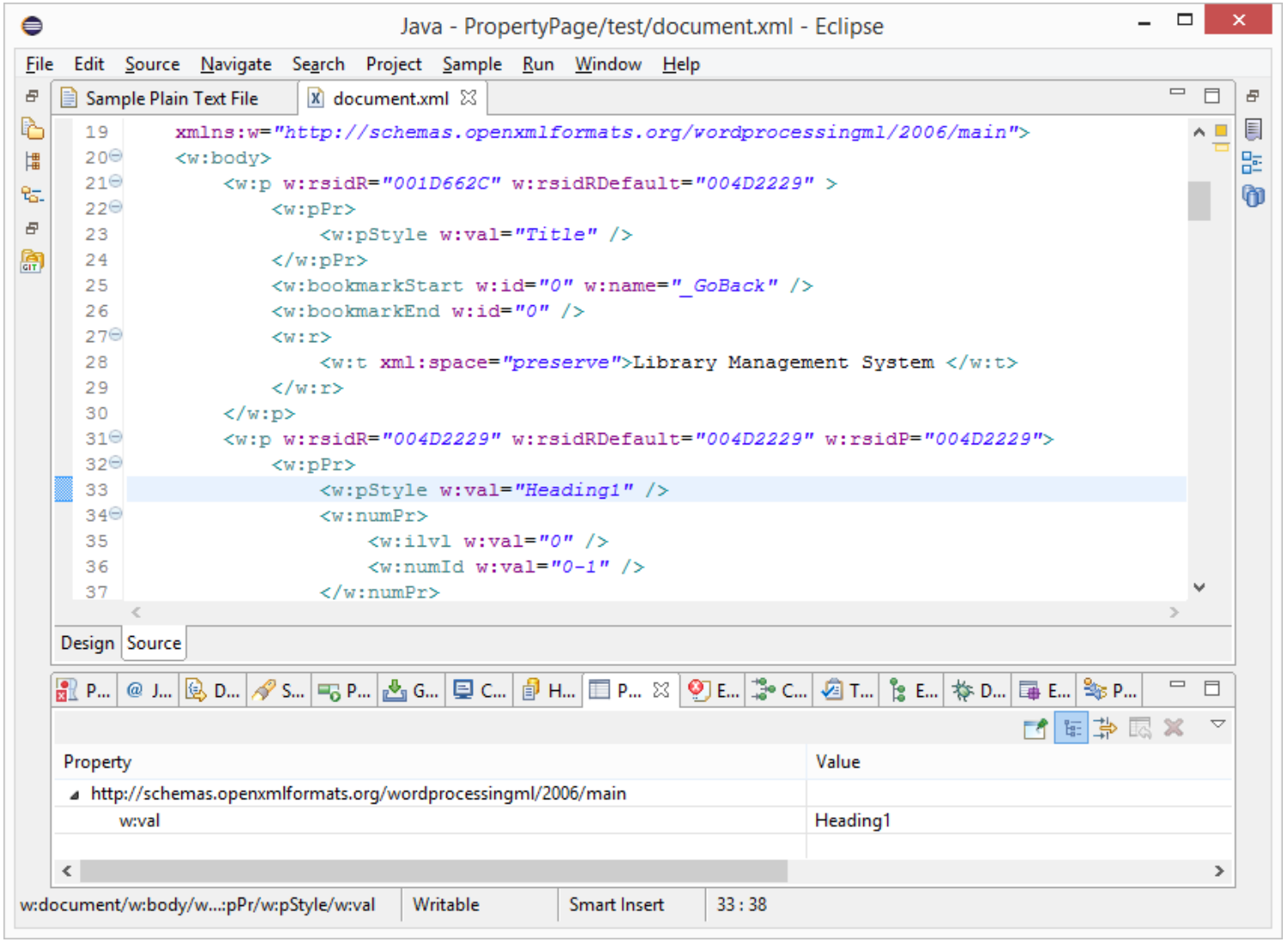
No meaning of syntax (words, sentences …)

What is a text?

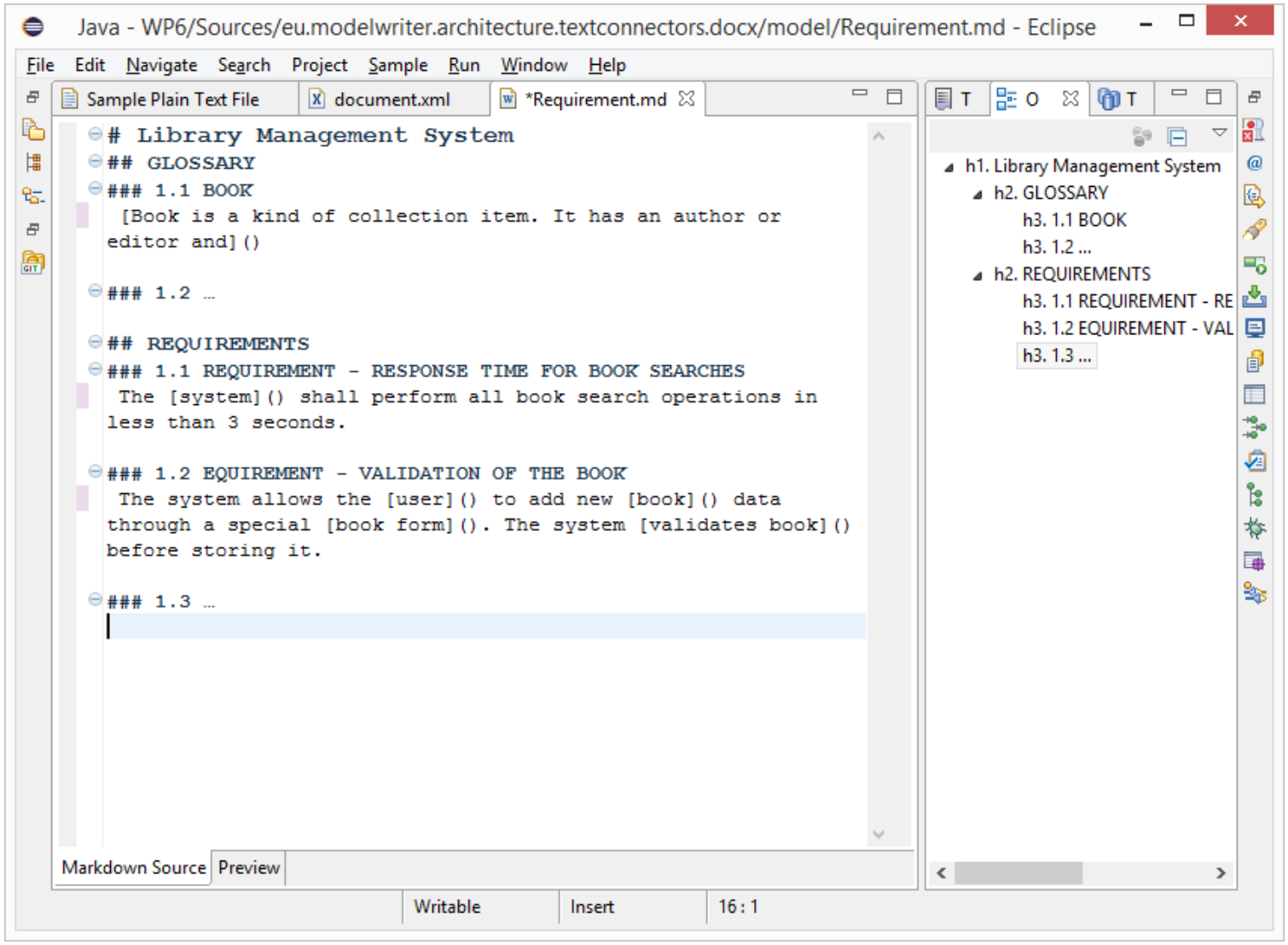
What is a text? (document file formats)
Office Open XML (.docx) (ISO/IEC 29500)



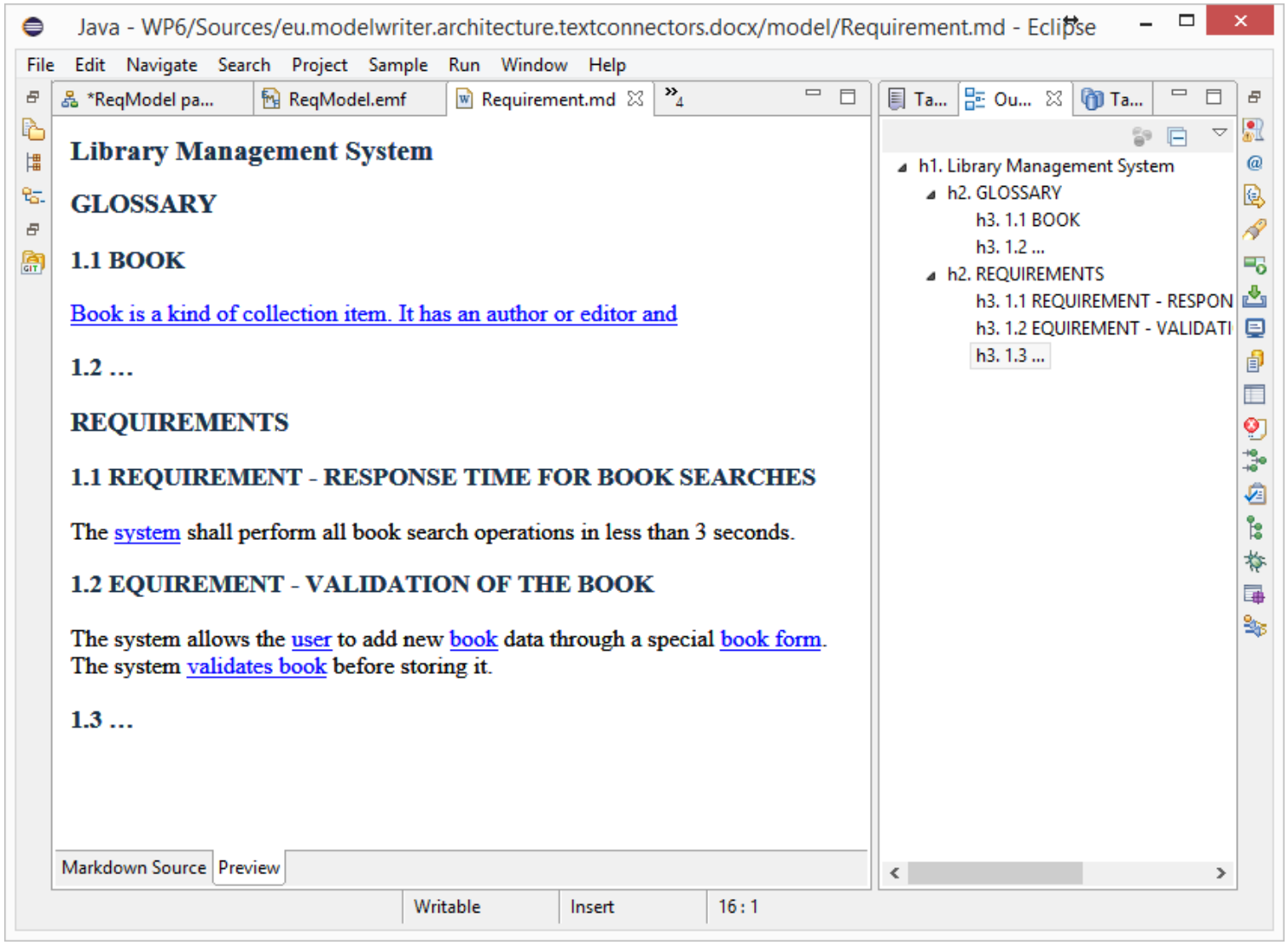
What is a text? (document file formats)
Office Open XML (.docx) (ISO/IEC 29500)



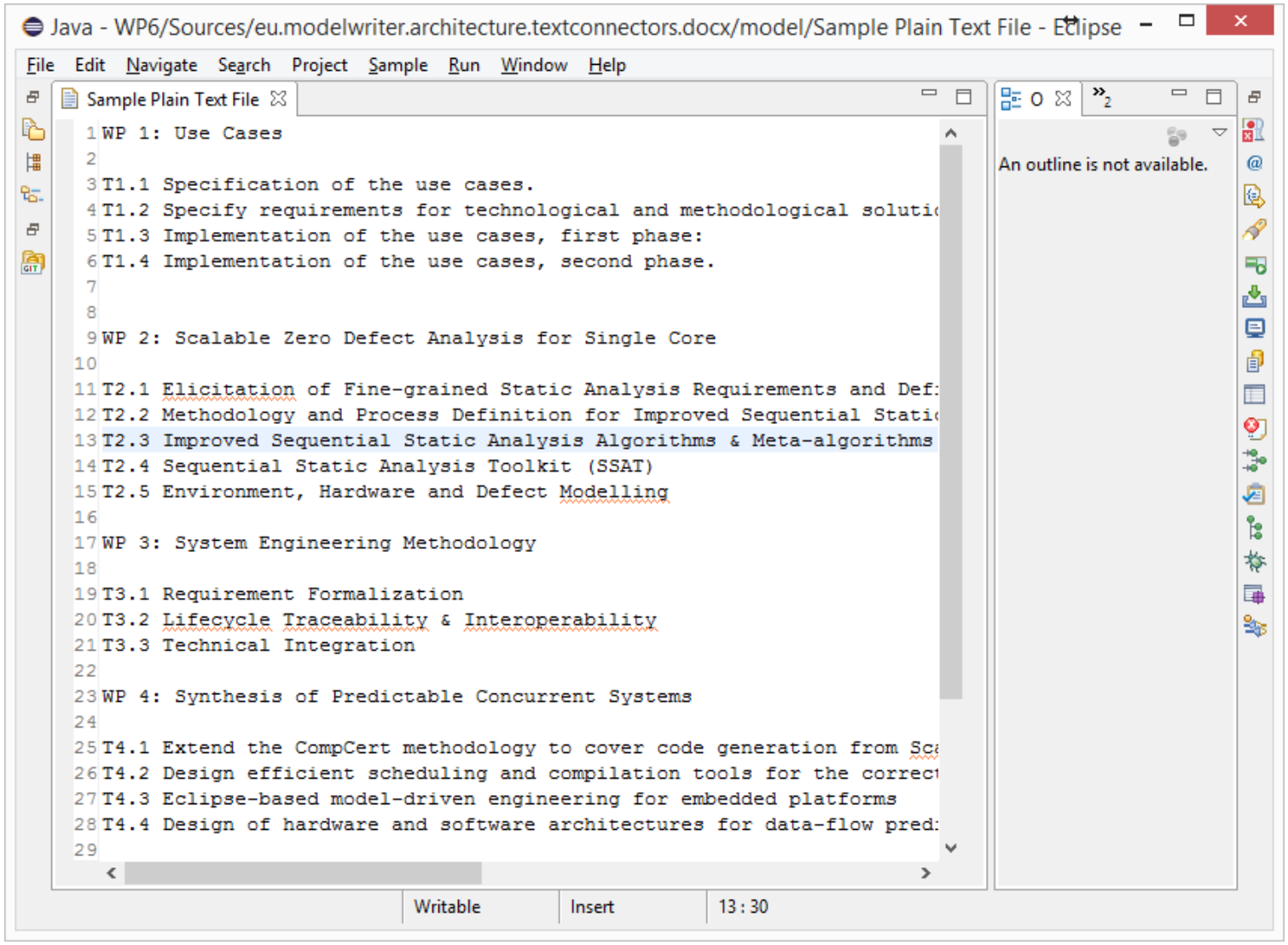
What is a text? (.md source file)
text/markdown (ICANN Standard)



What is a text? (HTML Preview)
text/markdown (ICANN Standard)

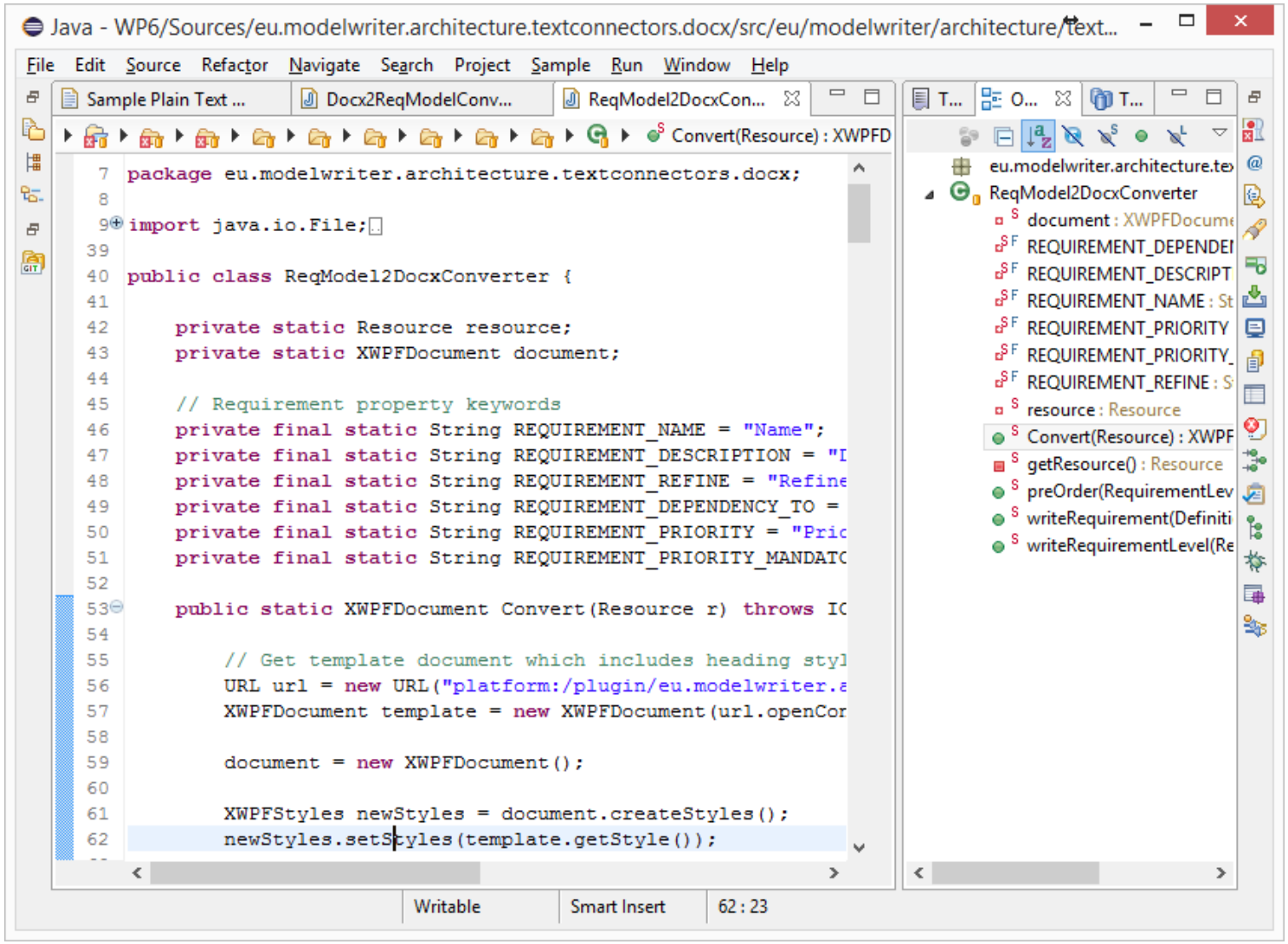


What is a text? (unformatted text)
text/plain (ICANN Standard)

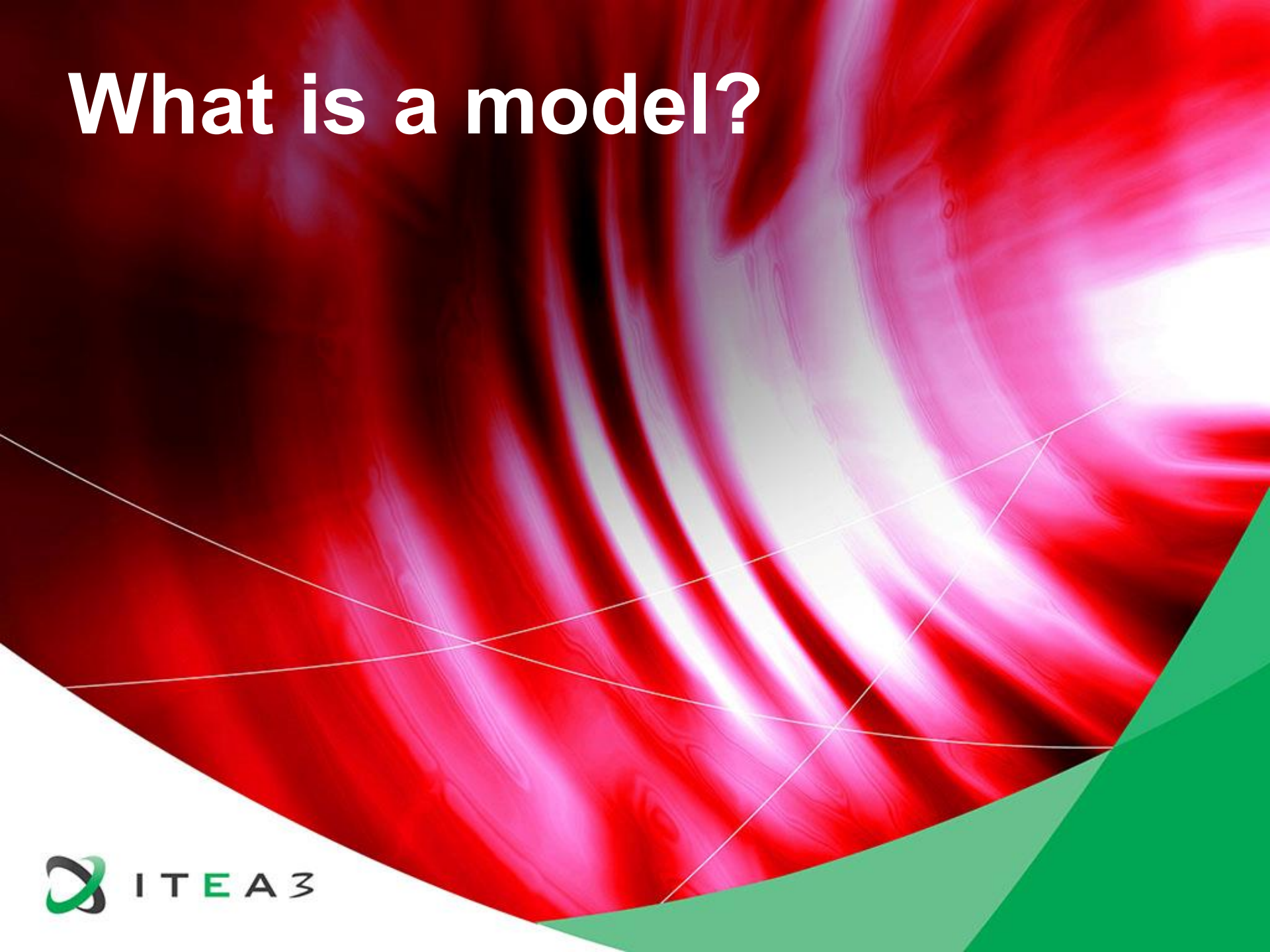


What is a text? (code files)

Java, C++ … Programing Languages

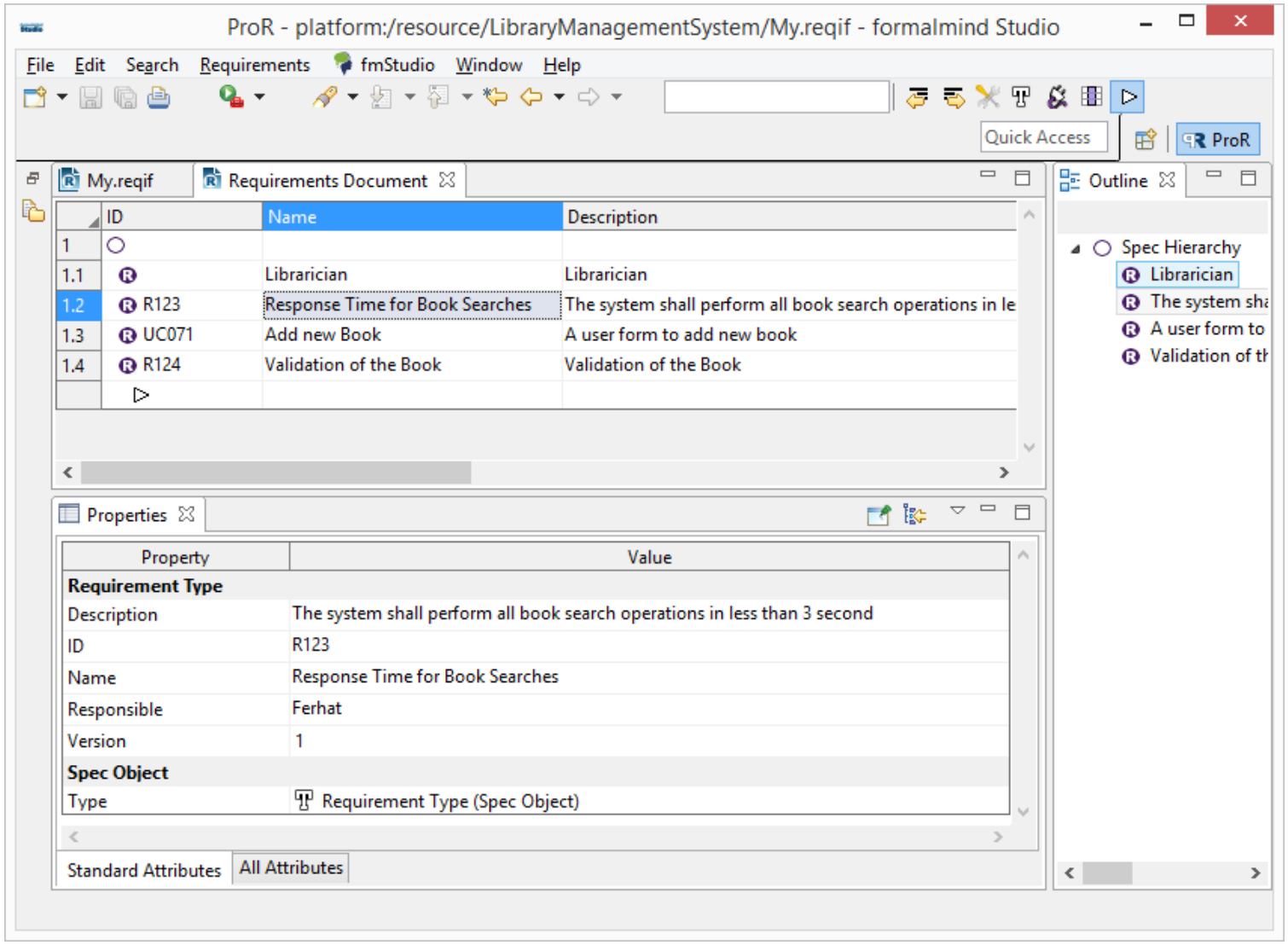


What is a model?

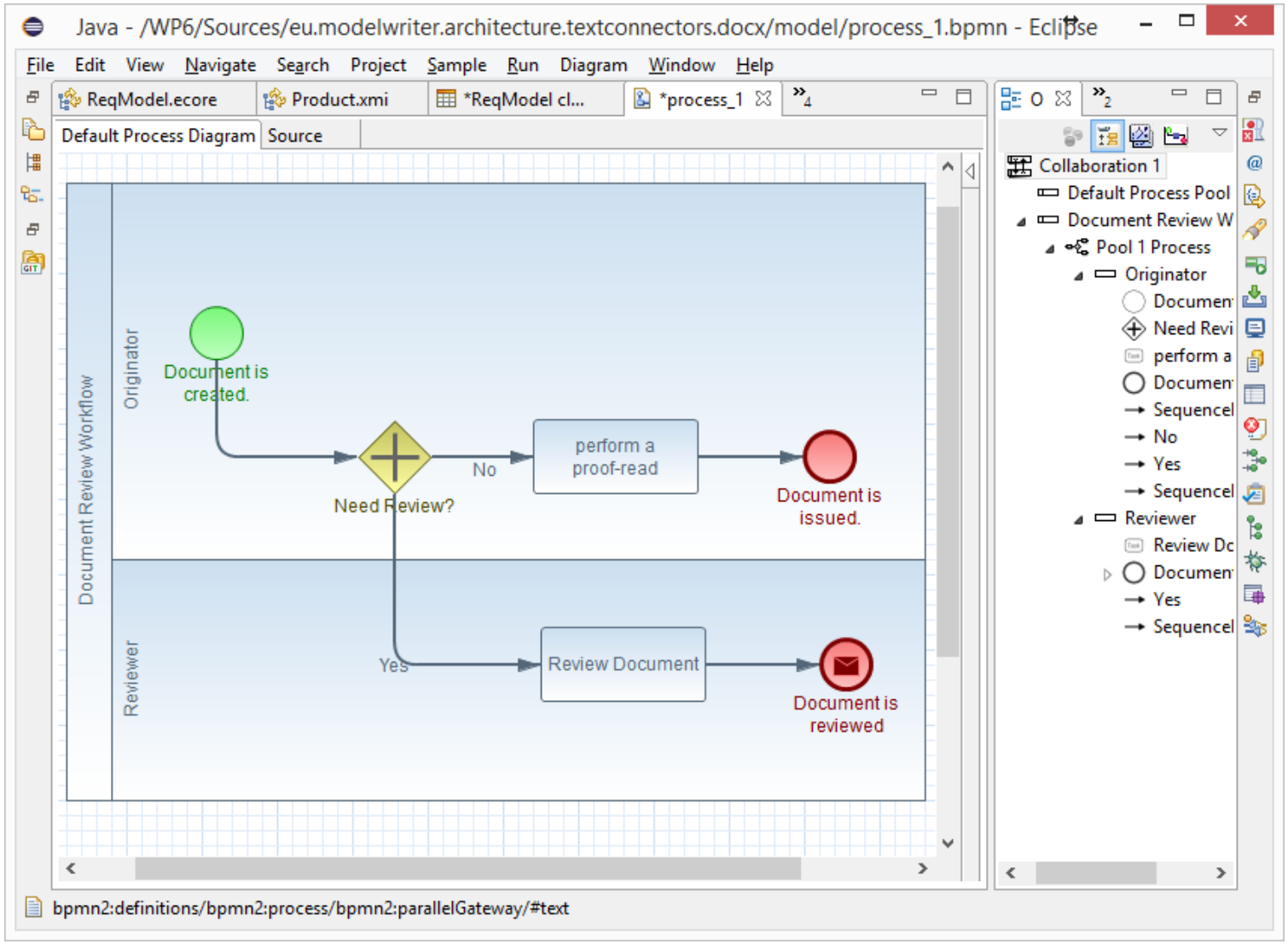


Everything is a model! (ReqIF Standard)

Requirements Interchange Format

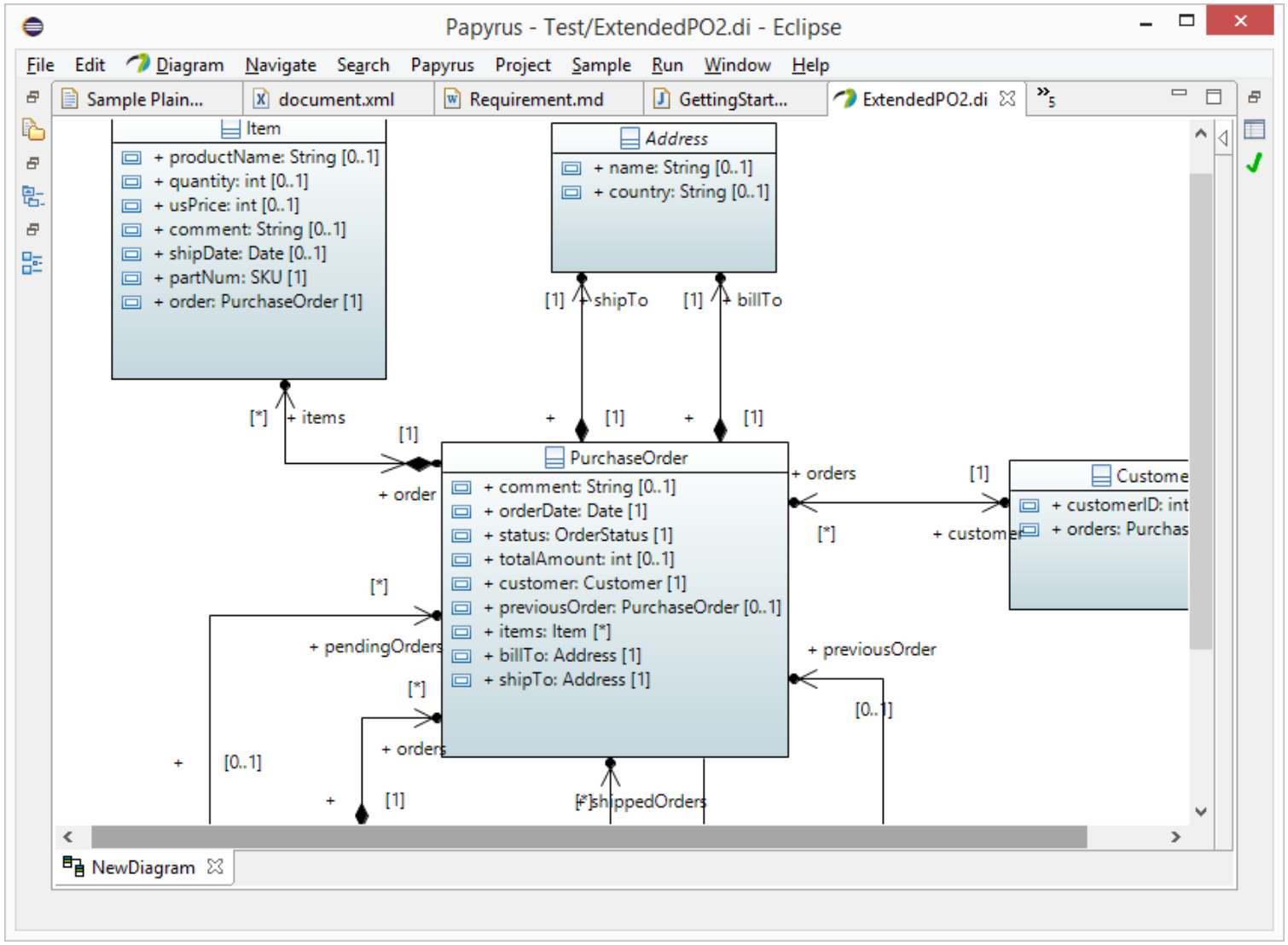


Everything is a model! (BPMN Standard)
Business Process Model & Notation

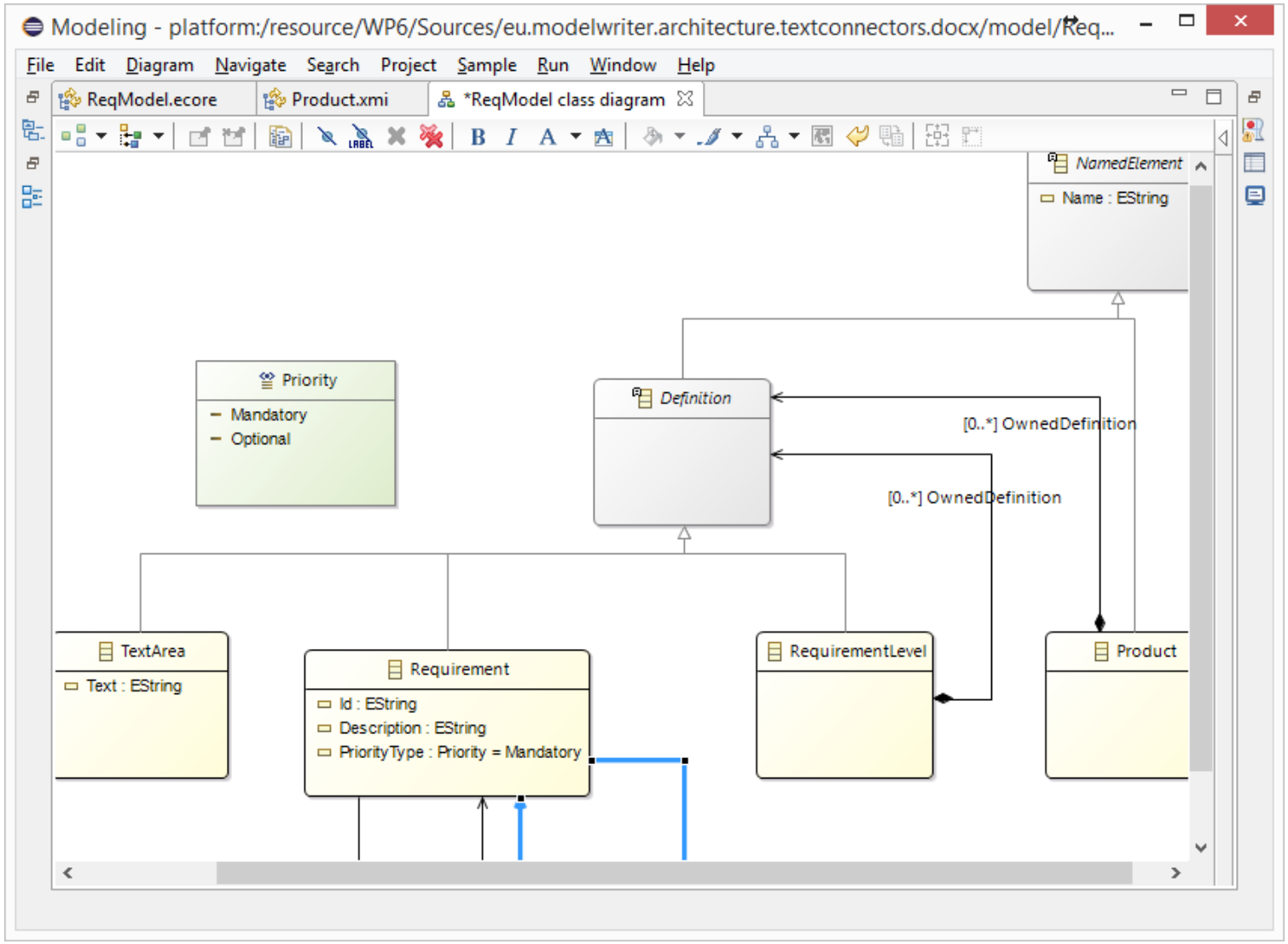


Everything is a model! (UML Standard)

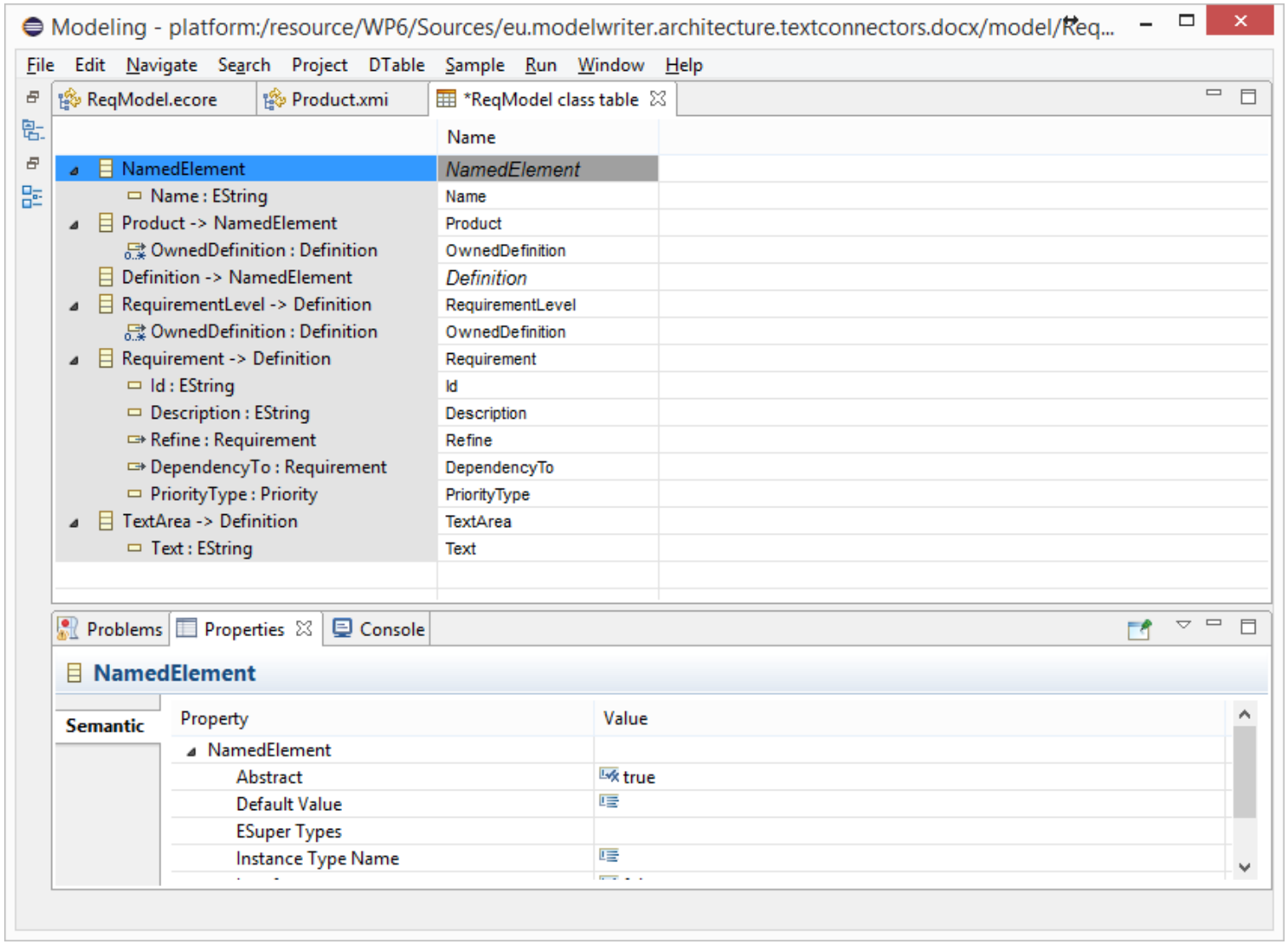
UML Modeling Languages



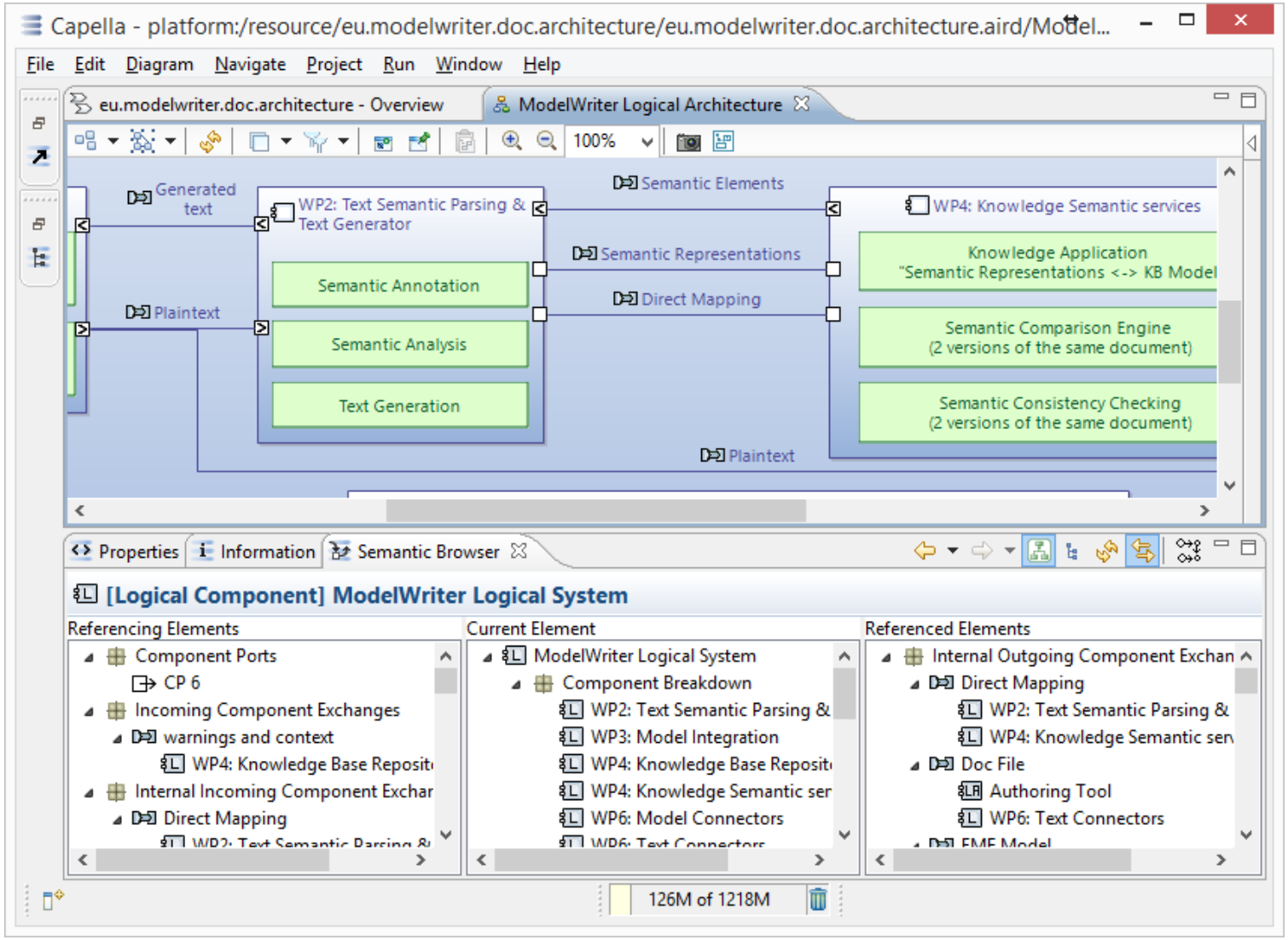
Everything is a model!
Eclipse Modeling Framework (EMF)



Everything is a model!
Tree-based or Tabular Representations

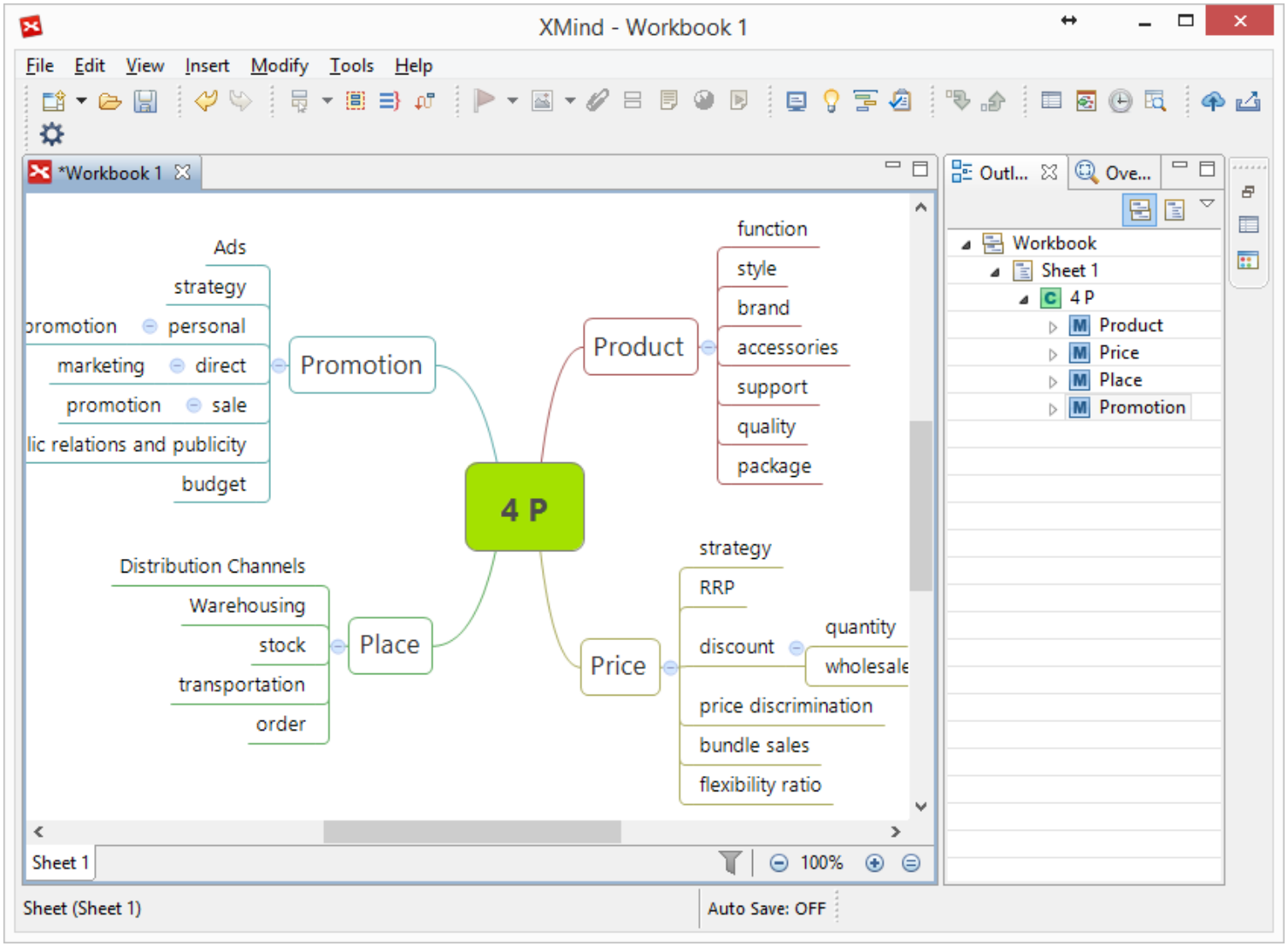


Everything is a model!
Software/System Architecture Design

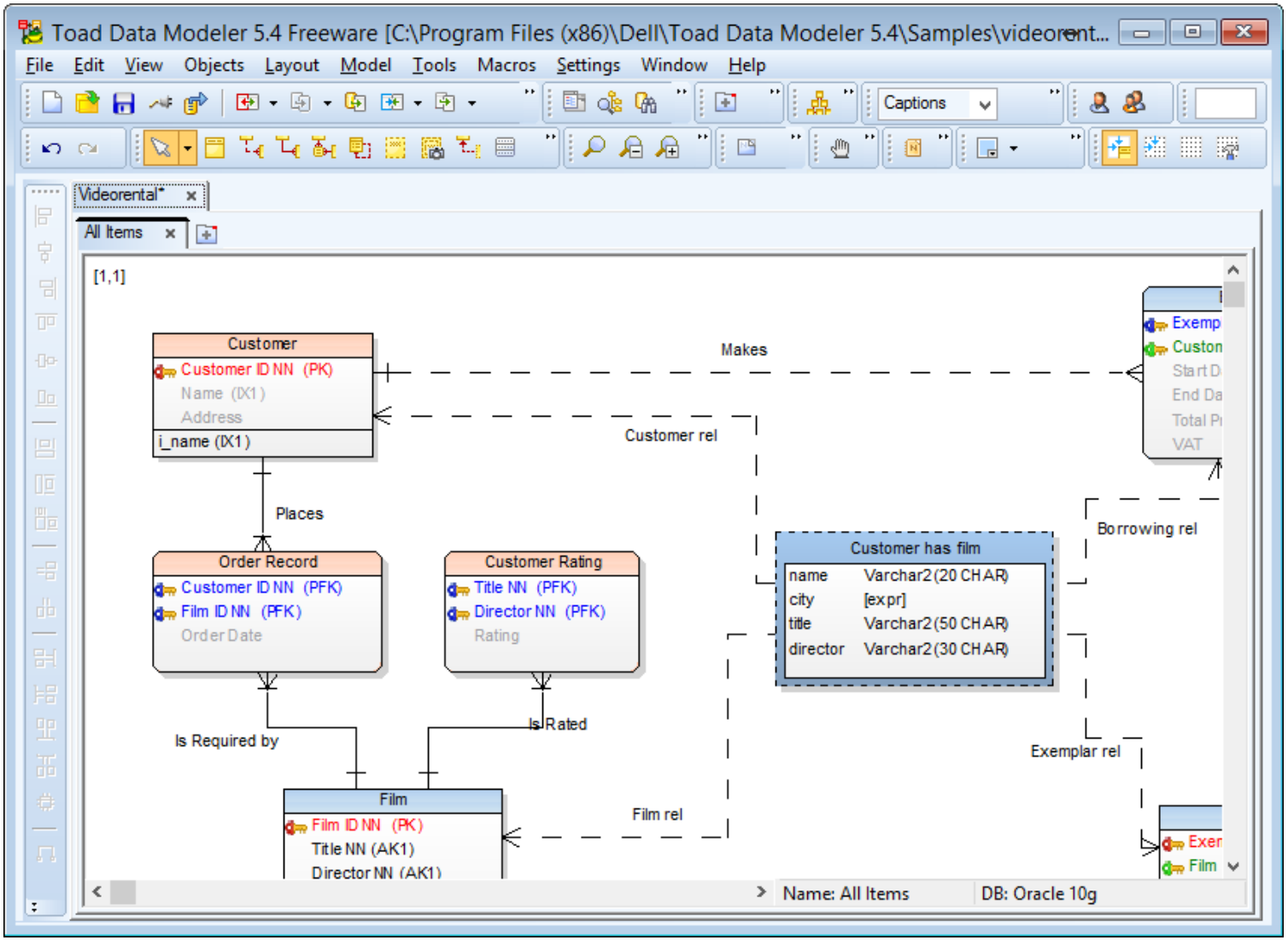


Everything is a model!

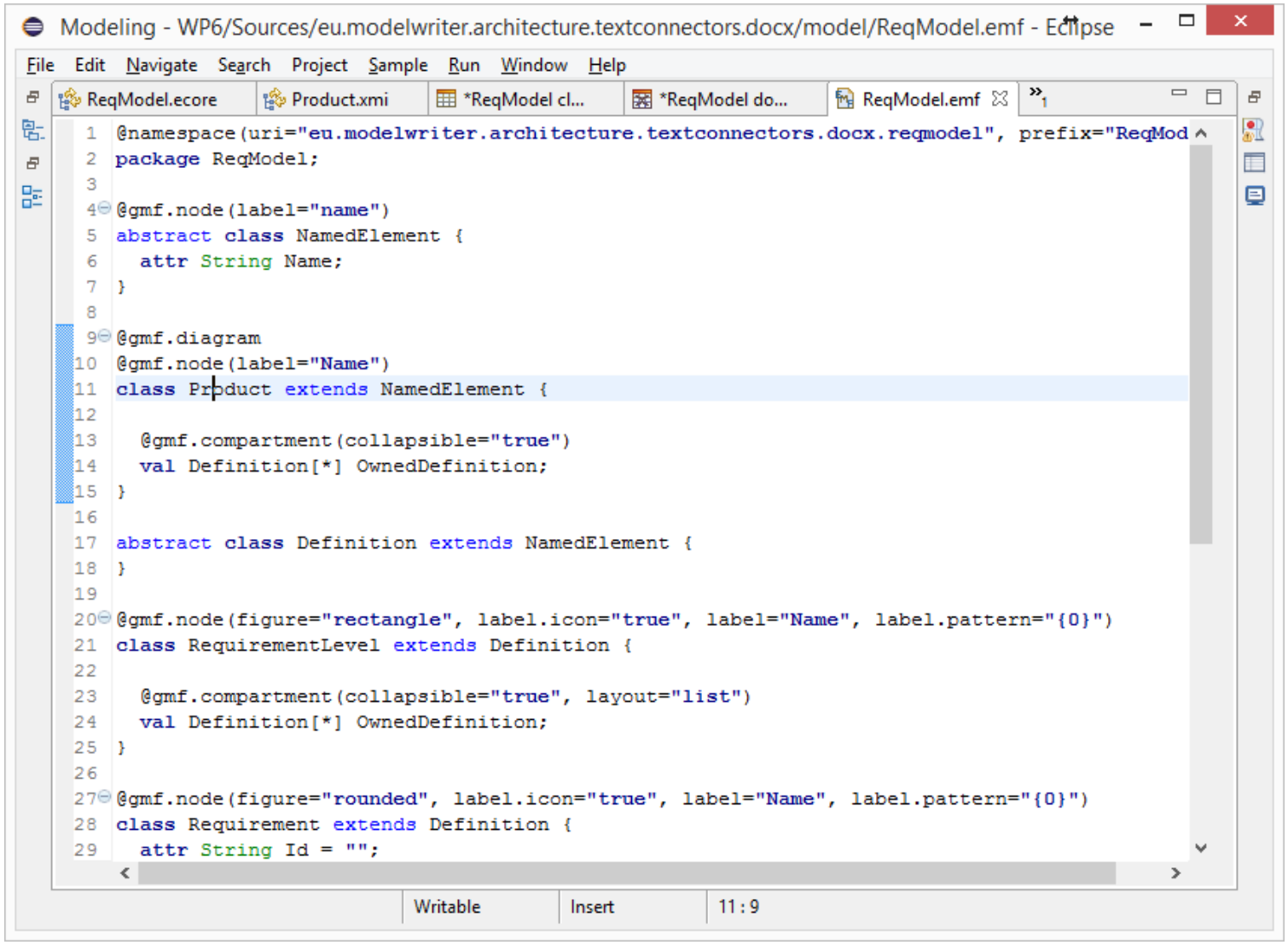
Topic Maps, Mind Maps, Vocabularies …



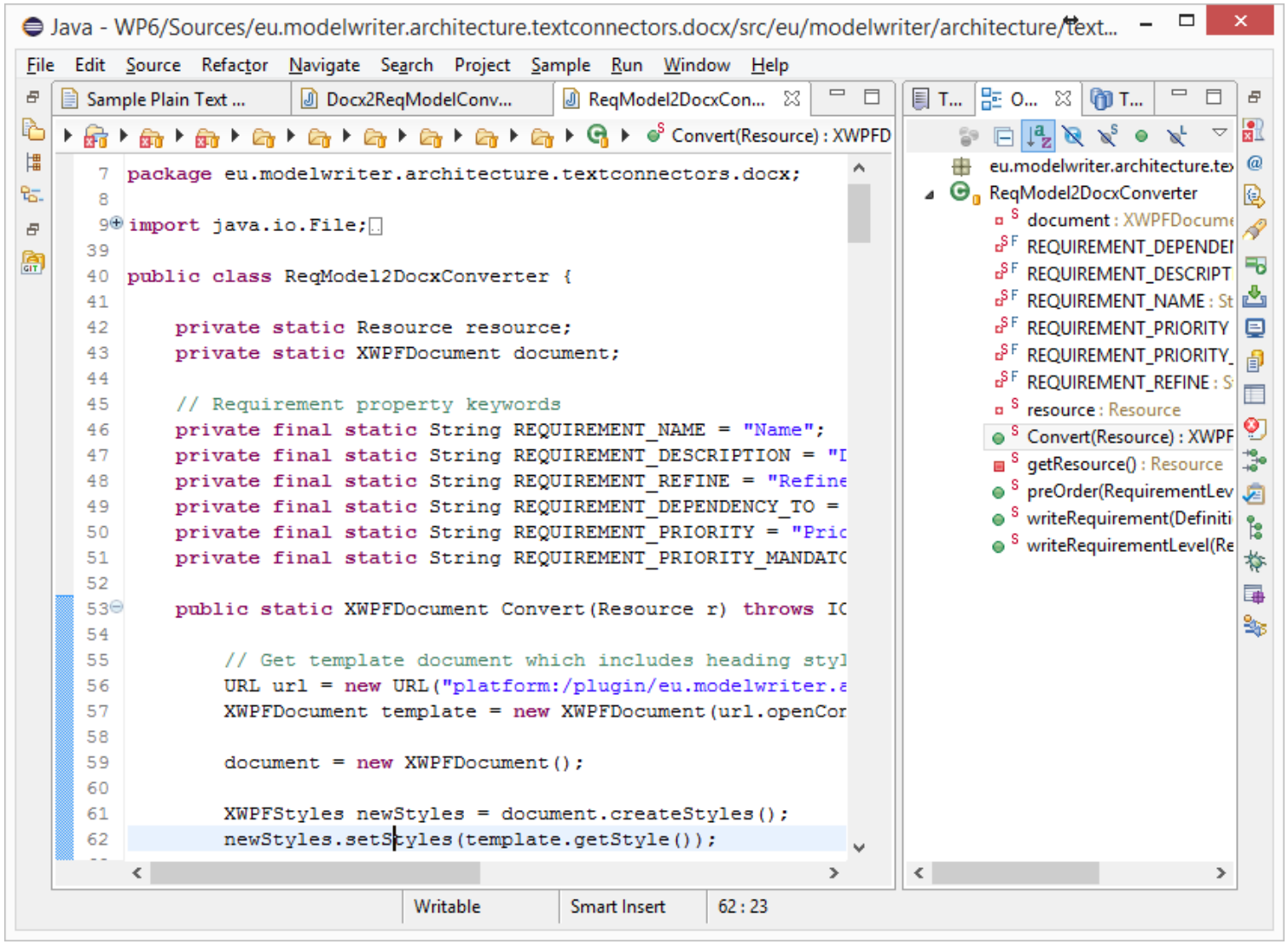
Everything is a model!
Databases (ER, IDEF1.x)



Everything is a model! (Textual Lang.)
Domain Specific Languages



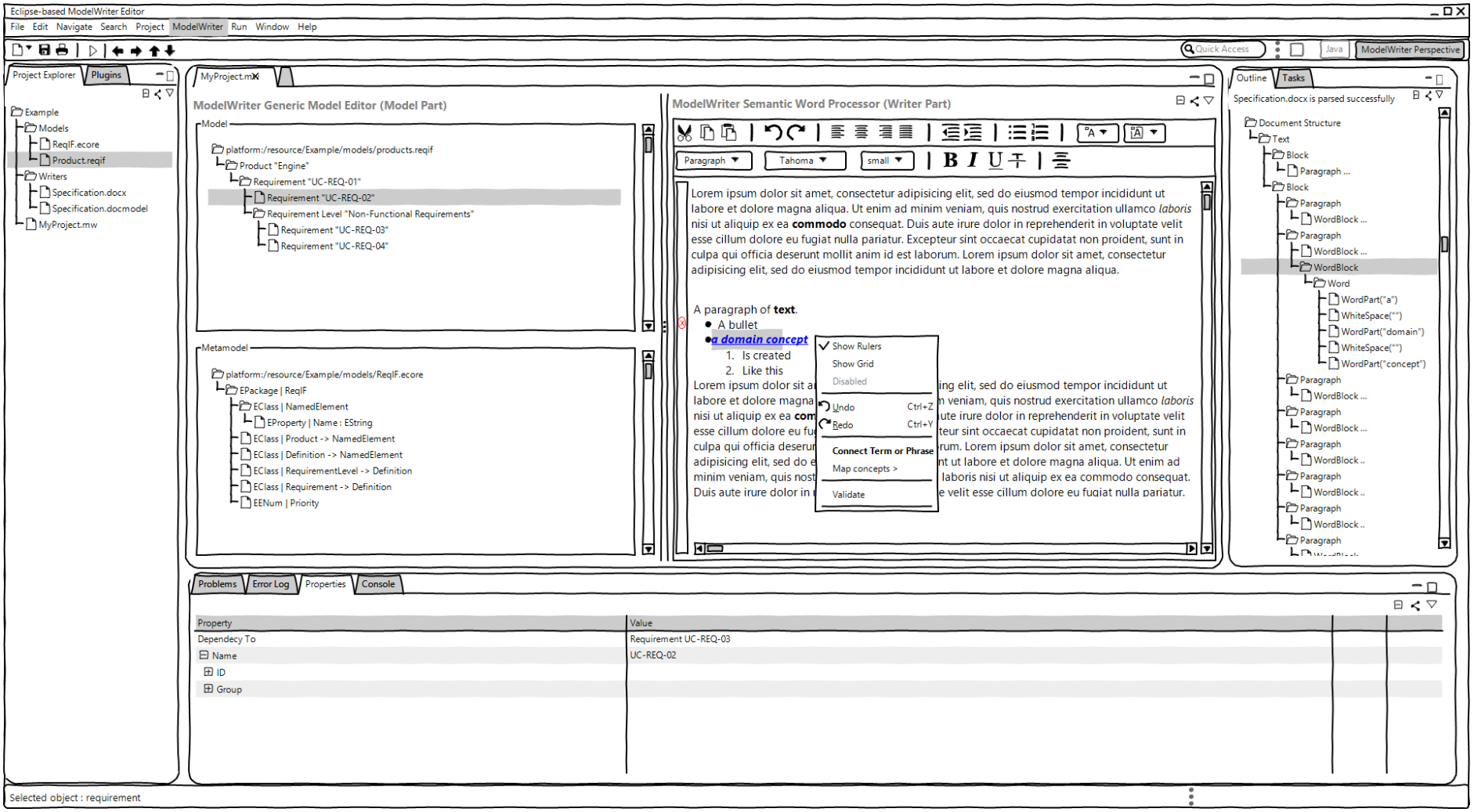
Everything is a model! (Java, C++, etc.)
Even Programing Languages (ASTs)



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

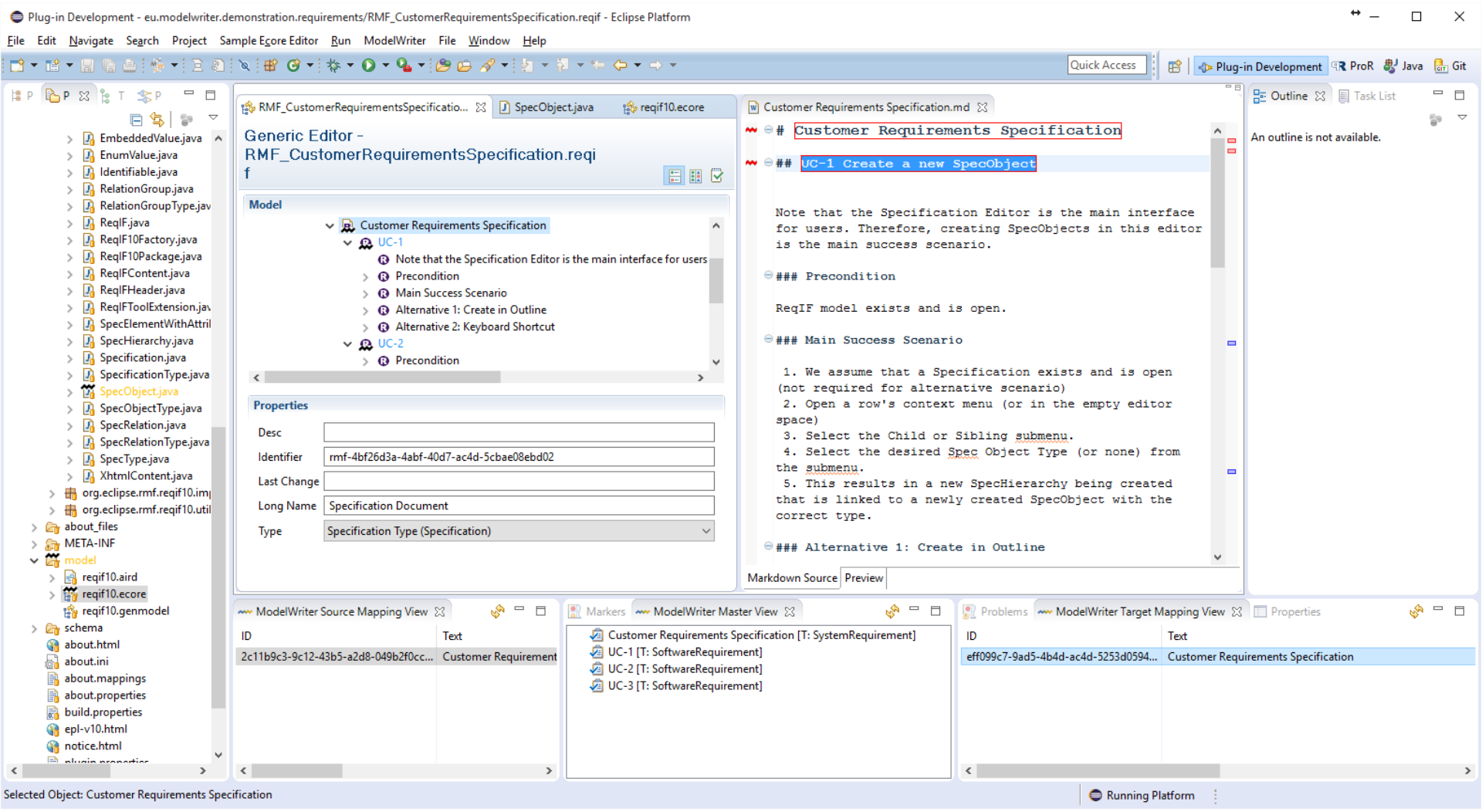
ITEA 3

ModelWriter – The Solution



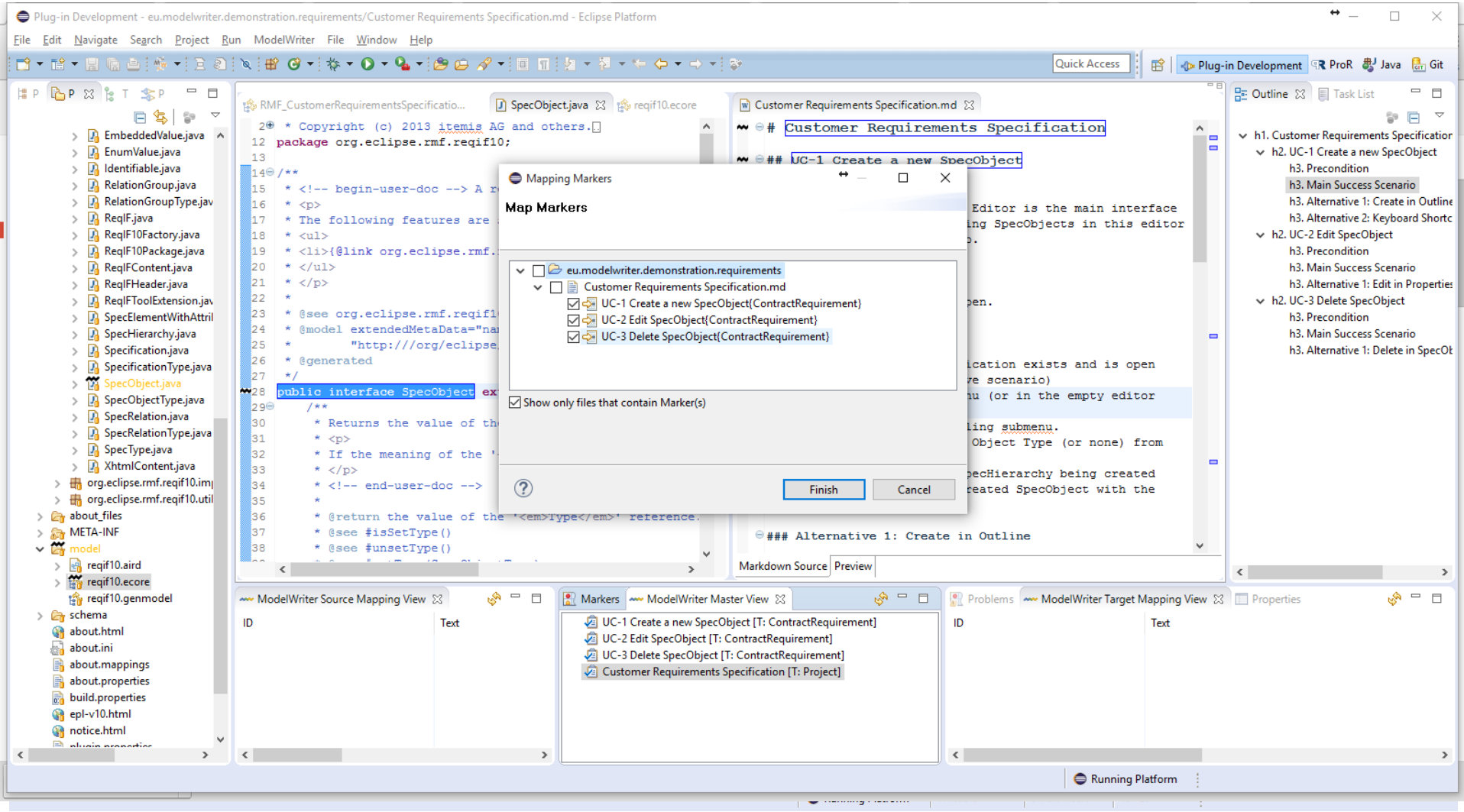
Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Capture



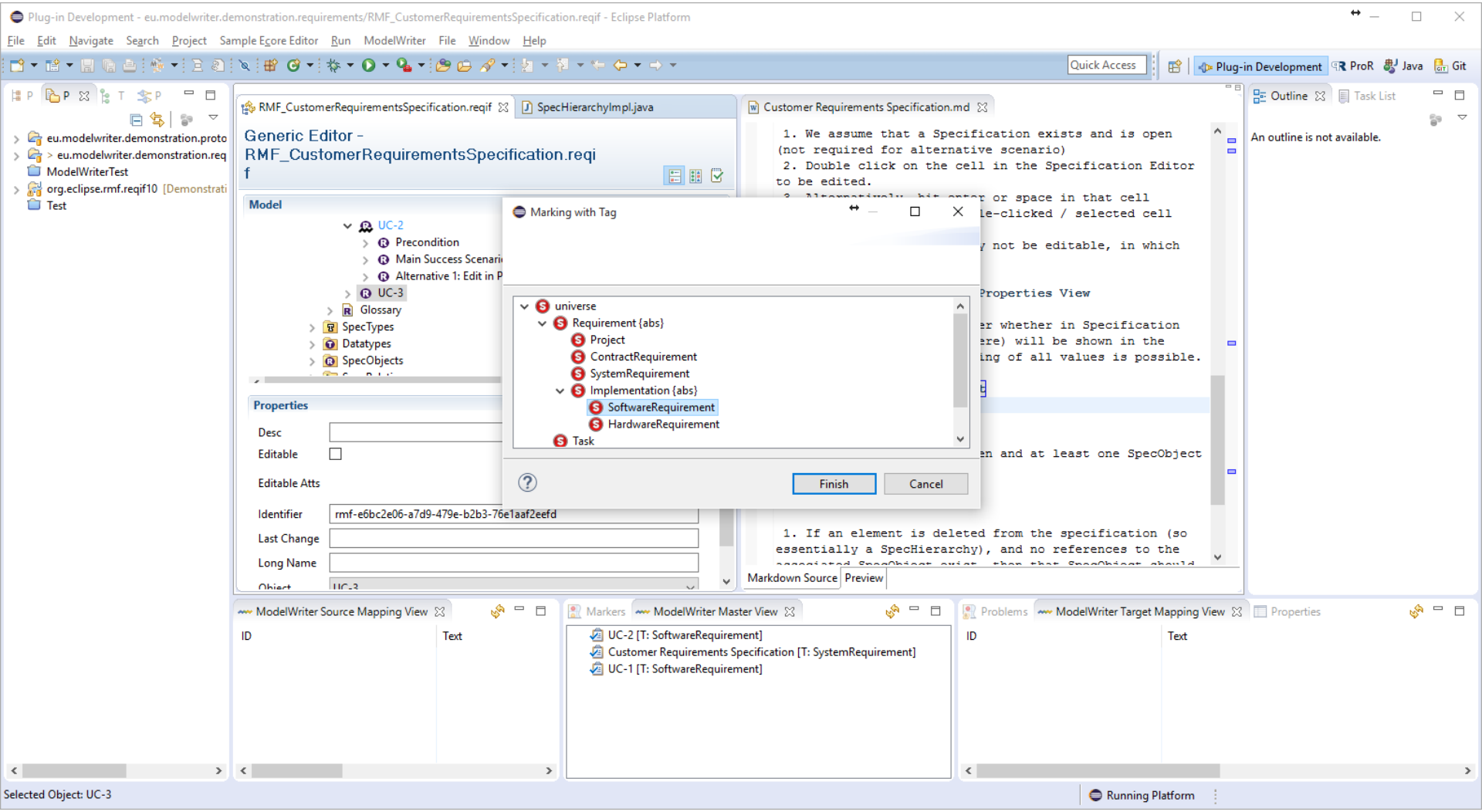
Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Capture



Text & Model-Synchronized Document Engineering Platform

Solution – Knowledge Capture

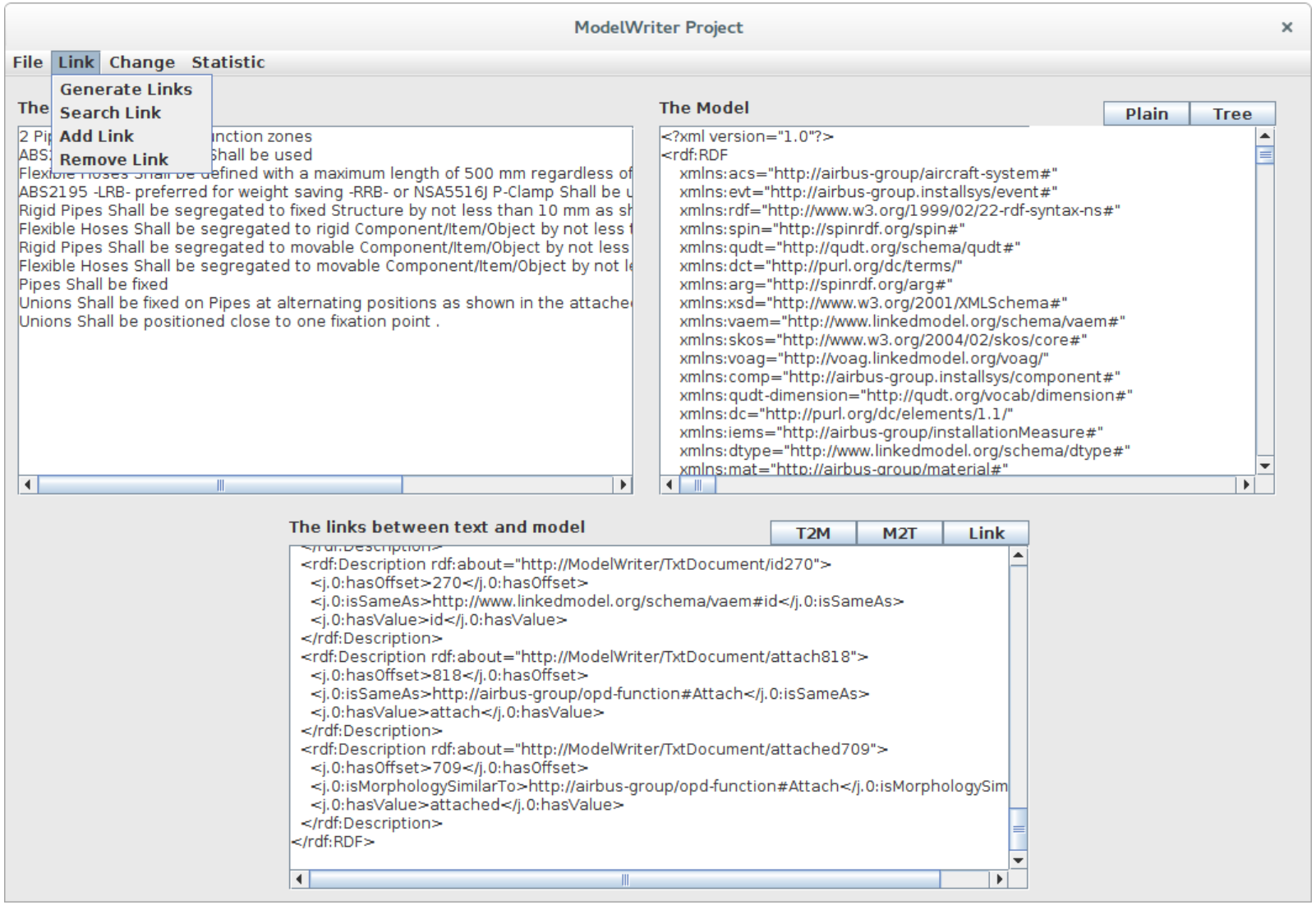


Text & Model-Synchronized Document Engineering Platform

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

ITEA 3

Solution – Knowledge Extraction



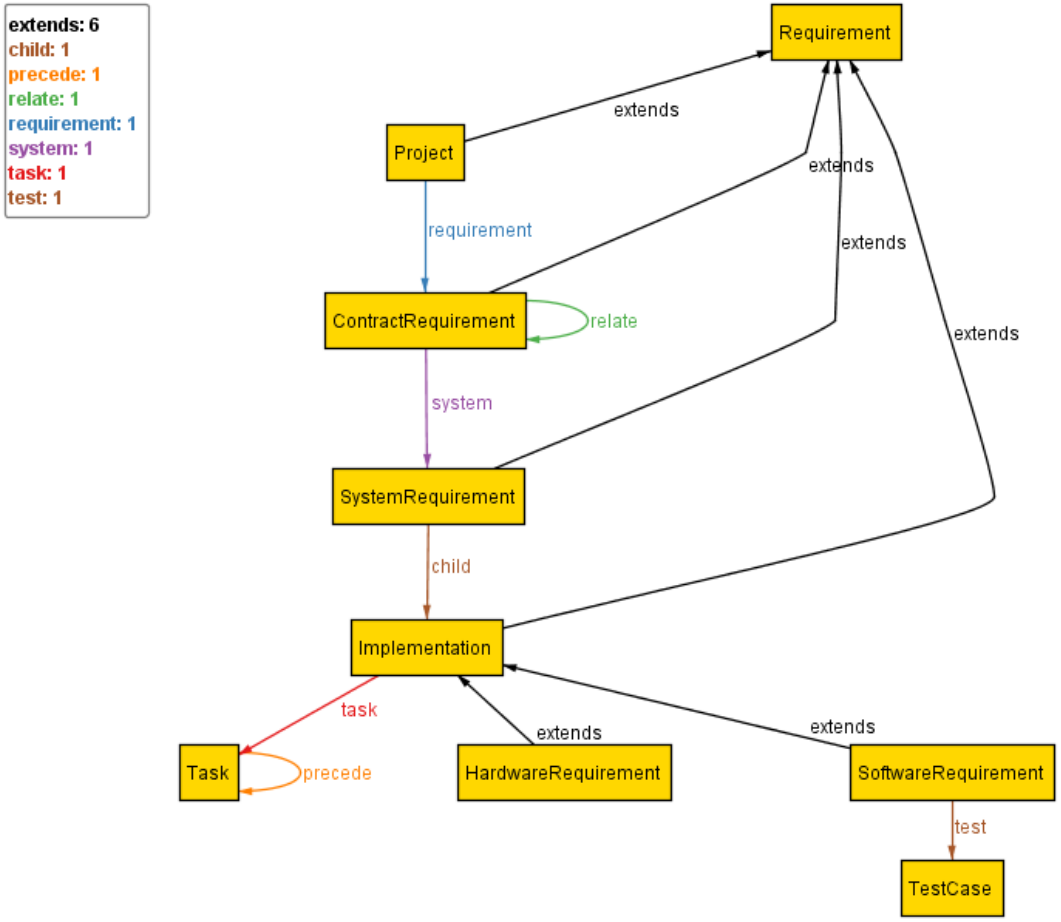
Text & Model-Synchronized Document Engineering Platform


Synchronization is maintained!

What about configuration/formalization of the platform?

ITEA 3

Configuration: Havelsan example



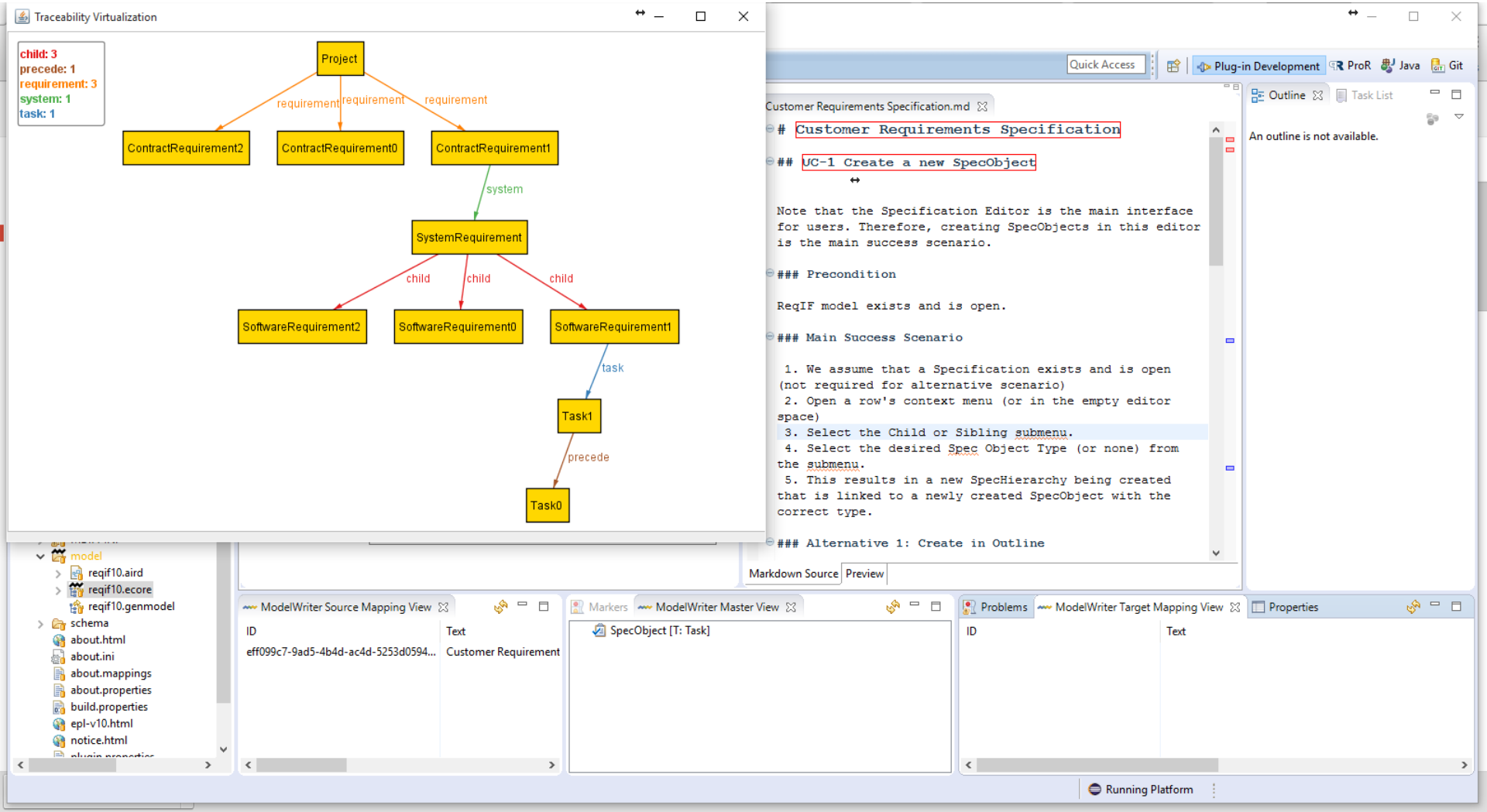

```
module Haveksan/Requirement

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

Is it possible to vizualize the trace links?

ITEA 3

Traceability: Havelsan example



A Formal Specification Model to configure the ModelWriter

6 Summary of the Current Status

Ferhat Erata
UNIT, ModelWriter Project Leader

Several Achievements

- Exploitation of ModelWriter in ITEA3-ASSUME
  - New system (Exploitation)
- Specification & Verification of ALM Platform
  - Open Source Software (Standardisation)
- Change Impact Analysis & Visualization
  - Open Source Software (Standardisation)
- System Installation Component Ontology
  - De facto standard (Standardisation)
- Semantic Annotator
  - Open Source Software (Standardisation)
- CSV to OWL transformation program
  - New product (Exploitation)
- SIDP Installation Rule Model
  - New product (Exploitation)

Summary of the first year

- We have a clear project structure and objectives.
- We positioned new industrial partners.
- We managed to restore the consortium with early changes in the leaderships
- We have still the same ambition.
- We have end users, clear needs; have enough tool & technology providers
- We have already significant Exploitation Related Achivements
- We have developed core ModelWriter
  - Knowledge Capture & Knowledge extraction
- All software components are platform independent and open source

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