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

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.

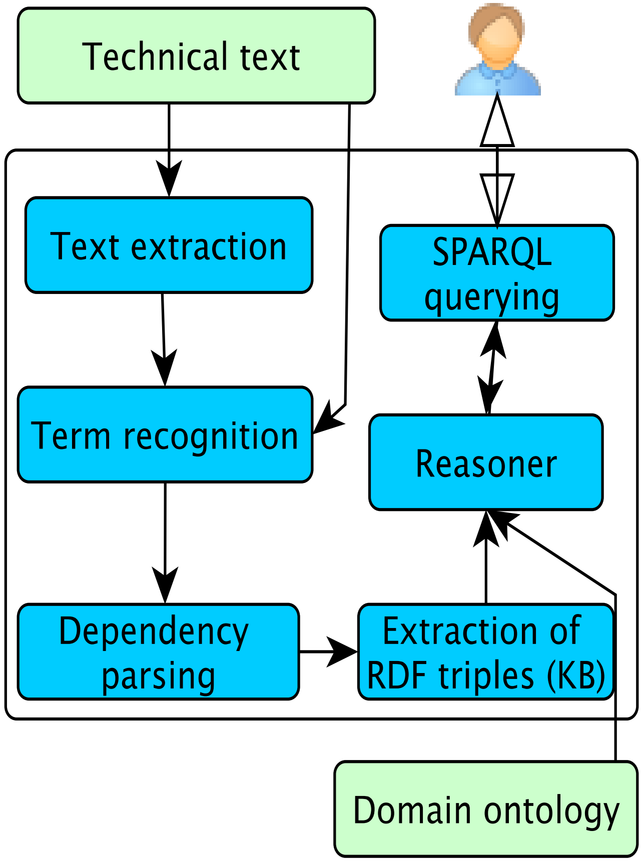
Non Disclosure Agreement finalised in June 2015.

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

Model Driven Company

T2.2 Semantic Parsing

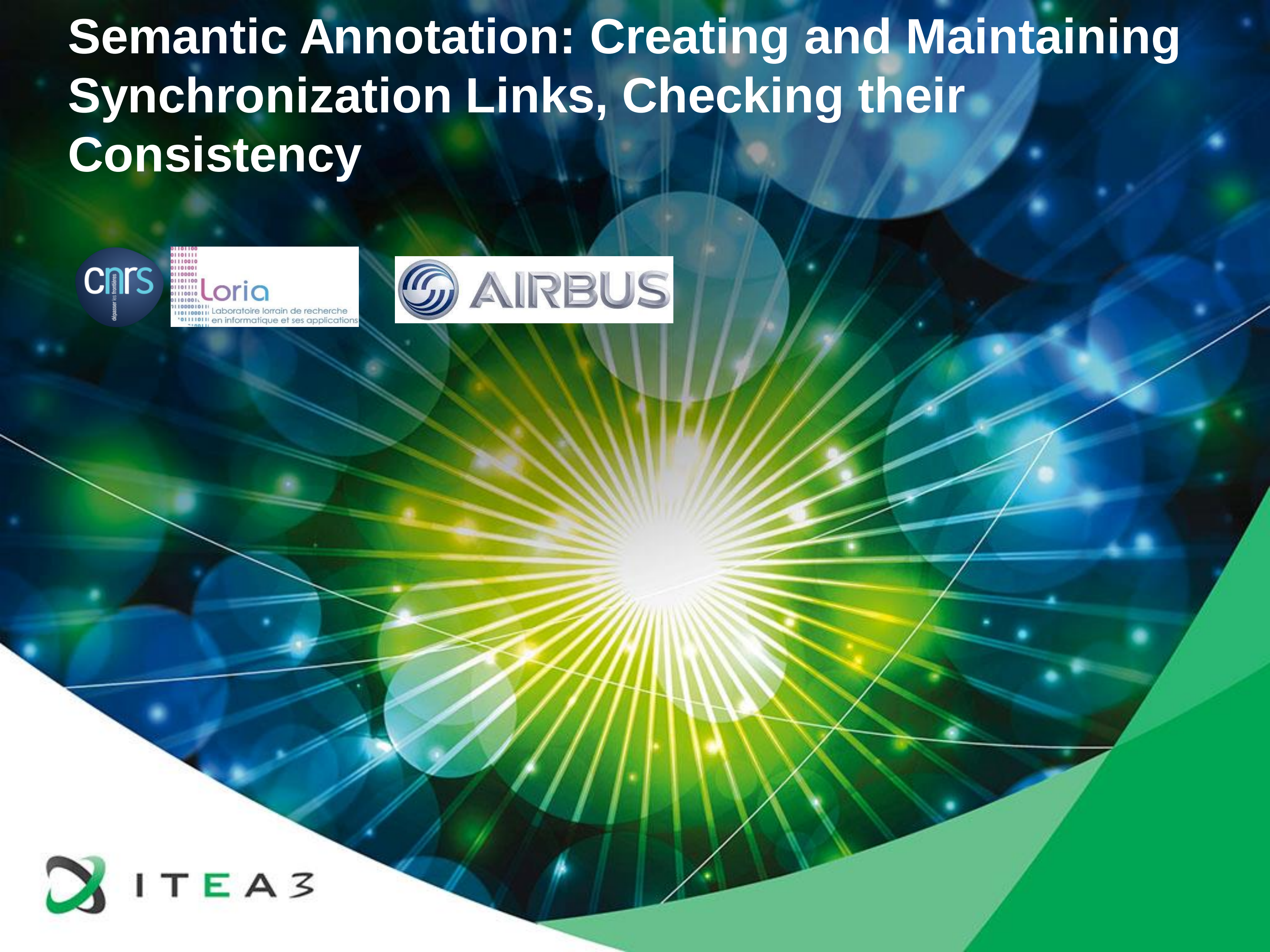


- Developed a prototype illustrating the automatic construction of an RDFS KB from text (CNRS/LORIA)
- « *Parsing Text into RDF* » B. Batouche, C. Gardent and A. Monceaux. SEPLN 2015, Alicante, Spain.
- Full scale Implementation applied to AIRBUS SIDP rules (Airbus)

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



cnrs

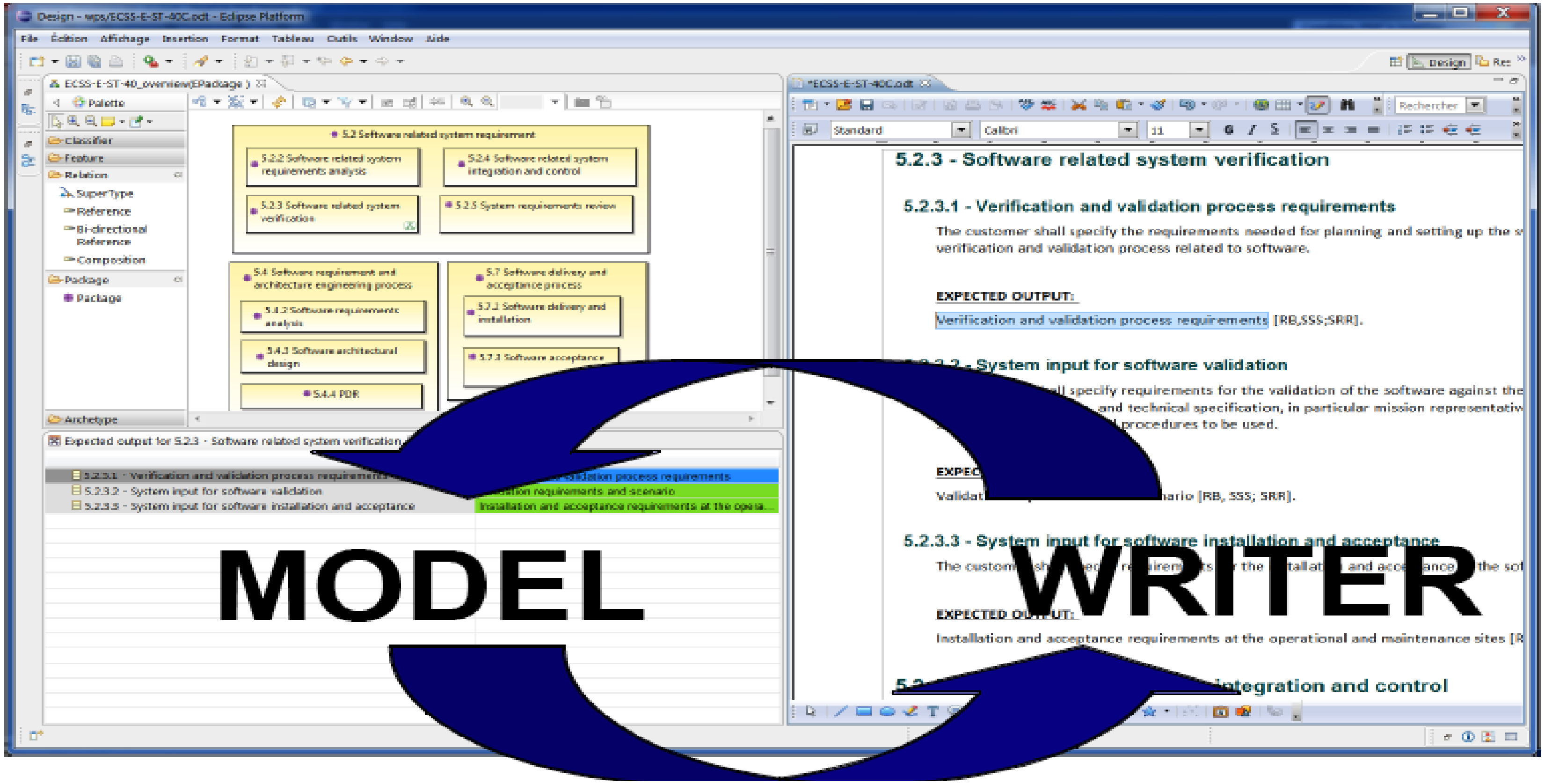
Loria
Laboratoire lorrain de recherche
en informatique et ses applications

AIRBUS

ITEA3

ModelWriter

Text & Model-Synchronized Document Engineering Platform


Design - wps/ECSS-E-ST-40C.odt - Eclipse Platform
File Édition Affichage Insertion Format Tableau Outils Window Aide
ECSS-E-ST-40_overview(EPackage)
Palette
Classifier
Feature
Relation
SuperType
Reference
Bi-directional Reference
Composition
Package
Package
Archetype
5.2 Software related system requirement
5.2.2 Software related system requirements analysis
5.2.4 Software related system integration and control
5.2.3 Software related system verification
5.2.5 System requirements review
5.4 Software requirement and architecture engineering process
5.4.2 Software requirements analysis
5.4.3 Software architectural design
5.4.4 PDR
5.7 Software delivery and acceptance process
5.7.2 Software delivery and installation
5.7.3 Software acceptance
*ECSS-E-ST-40C.odt
Design
Res
Rechercher
Standard
Calibri
11
MODEL
WRITER

Expected output for 5.2.3 - Software related system verification

5.2.3.1 - Verification and validation process requirements	…alidation process requirements
5.2.3.2 - System input for software validation	…ation requirements and scenario
5.2.3.3 - System input for software installation and acceptance	Installation and acceptance requirements at the opera…

5.2.3 - Software related system verification

5.2.3.1 - Verification and validation process requirements

The customer shall specify the requirements needed for planning and setting up the s verification and validation process related to software.

EXPECTED OUTPUT:

Verification and validation process requirements [RB,SSS;SRR].

…2.2 - System input for software validation

…ll specify requirements for the validation of the software against the …and technical specification, in particular mission representativ… …procedures to be used.

EXPEC…

Validat… …ario [RB, SSS; SRR].

5.2.3.3 - System input for software installation and acceptance

The custom… sh… …ed… re…irem…ts …r the …tallat… and acce…ance… the sof

EXPECTED OU…UT:

Installation and acceptance requirements at the operational and maintenance sites [R

5.2… …integration and control

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

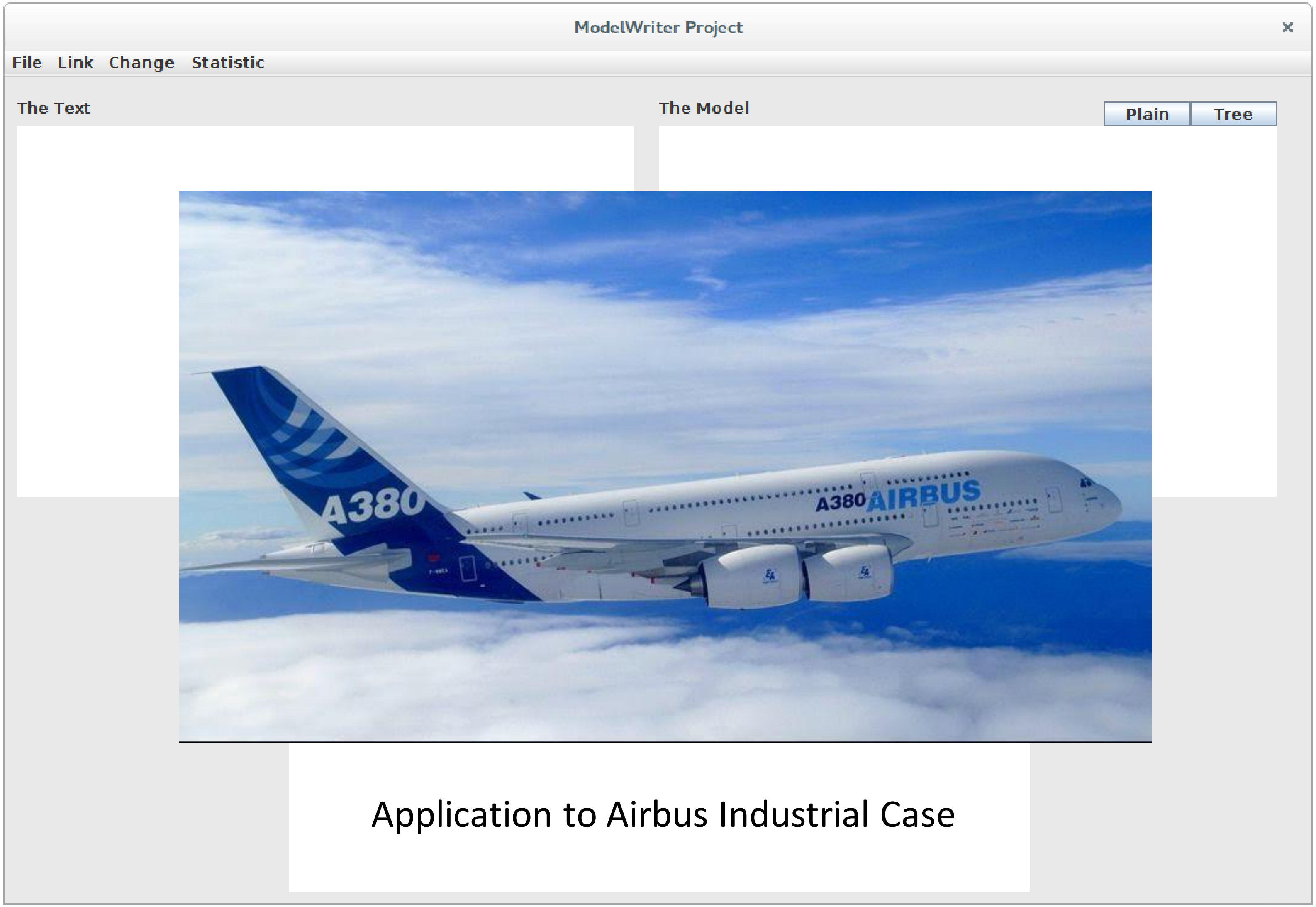
  - Rule:
    - If a text element and an ontology concept are semantically disjoint, then they cannot be synchronized
      - Ex: rigid Component cannot be synchronized with http://airbus-group.installsys/component#FlexibleComponent

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

cnrs dépasser les frontières

Loria Laboratoire lorrain de recherche en informatique et ses applications

Obeo Model Driven Company

ITEA3

Semantic Annotation

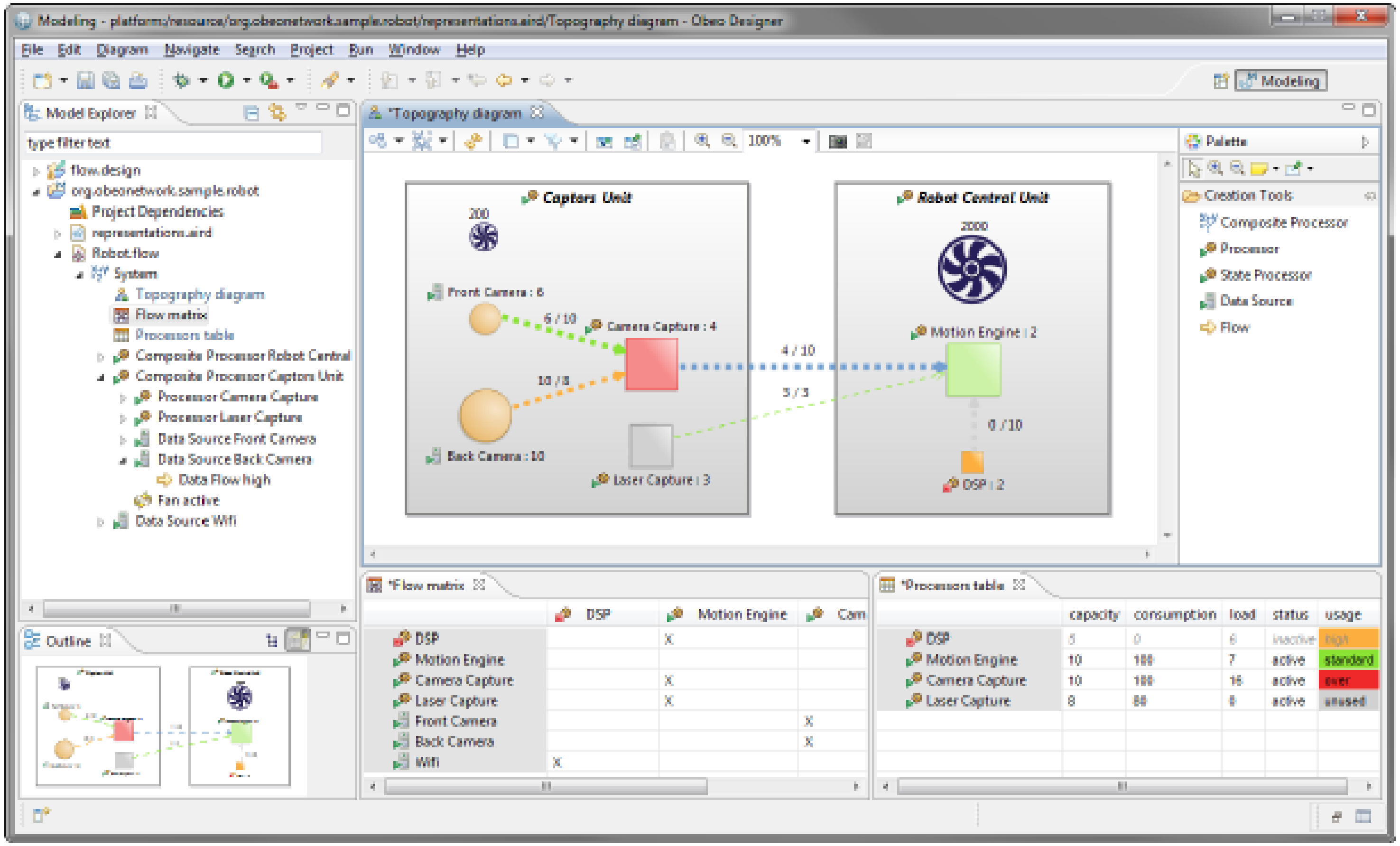
- OBEO Corpora
  - EMF (Eclipse Modeling Framework)
    - The EMF project is a modeling framework and code generation facility for building tools and other applications based on a structured data model
    - From a model specification described in XMI, EMF provides tools and runtime support to produce 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



*Flow matrix

	DSP	Motion Engine	Cam
DSP		X	
Motion Engine			
Camera Capture		X	
Laser Capture		X	
Front Camera			X
Back Camera			X
Wifi	X		

*Processors table

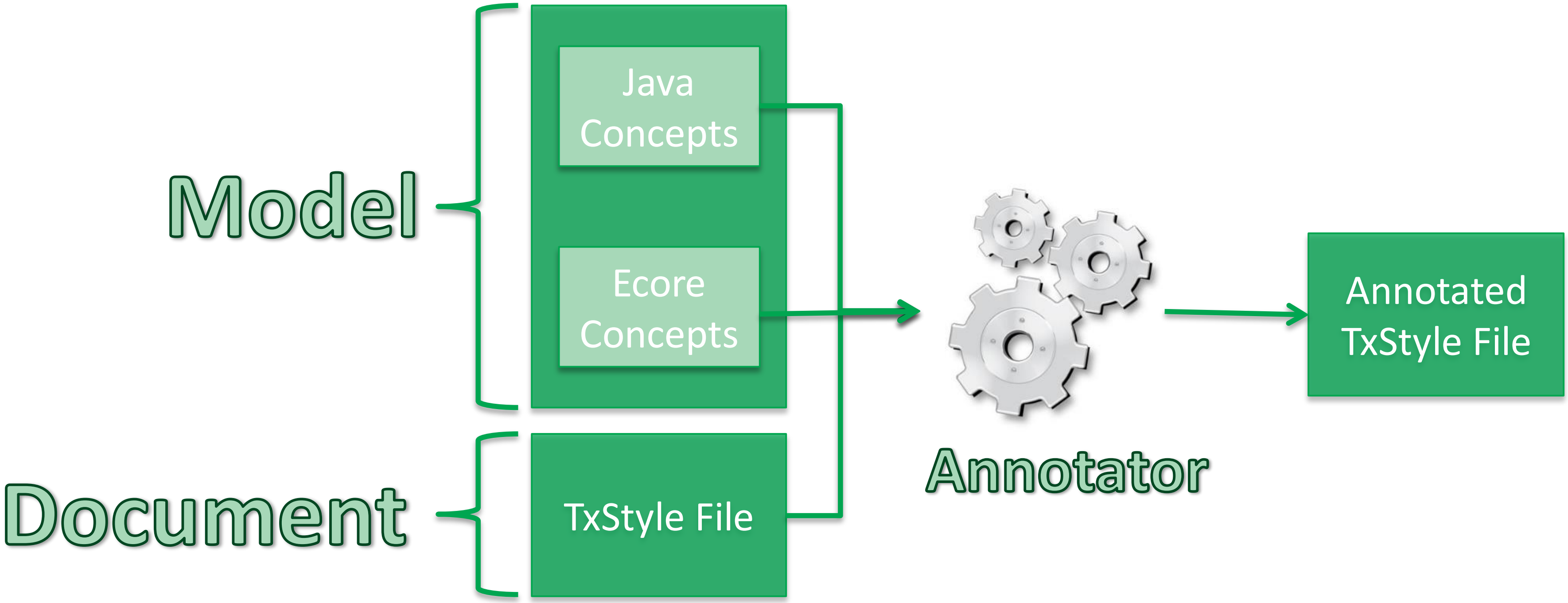
	capacity	consumption	load	status	usage
DSP	5	0	6	inactive	high
Motion Engine	10	100	7	active	standard
Camera Capture	10	100	16	active	over
Laser Capture	8	80	8	active	unused

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

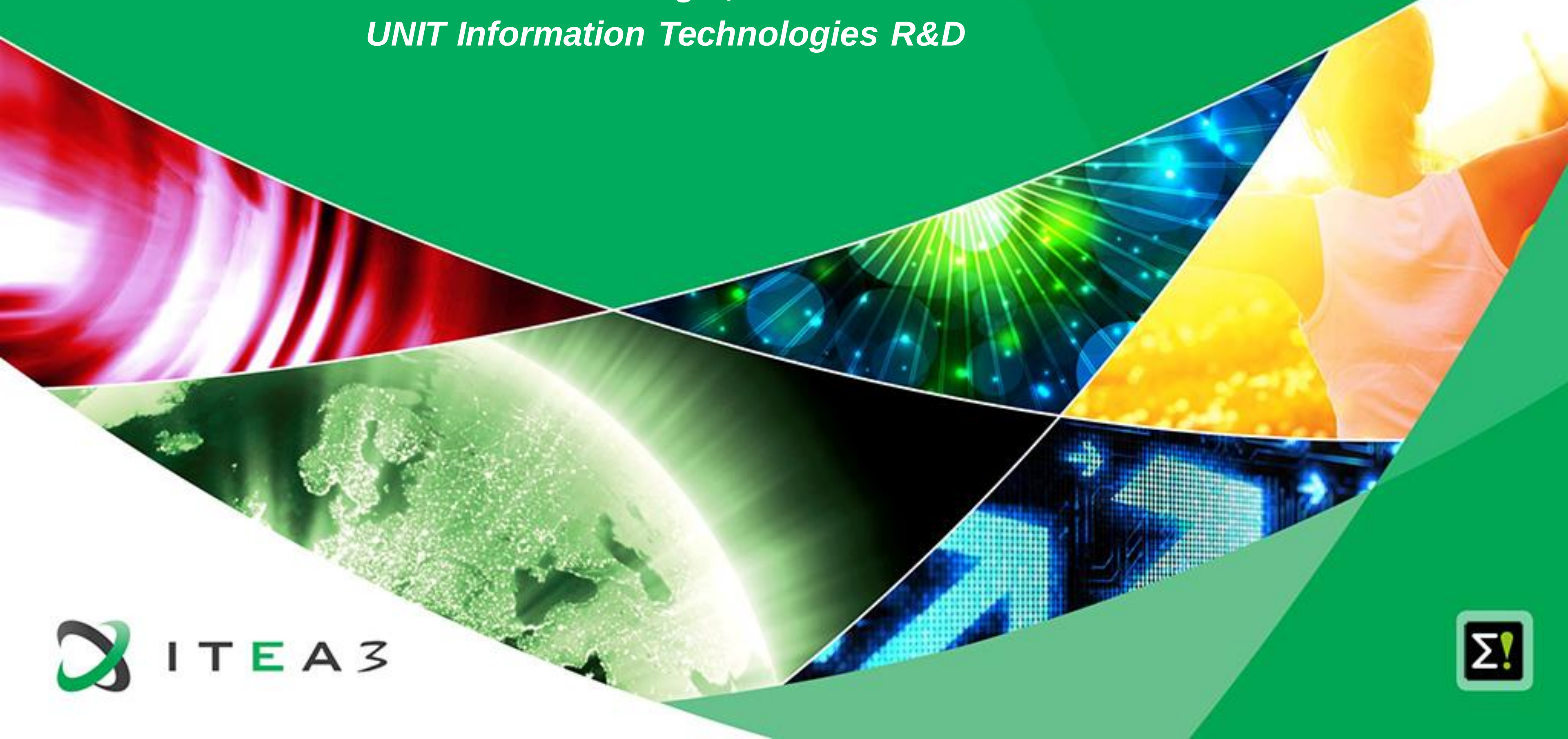


Semantic Annotation

- 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

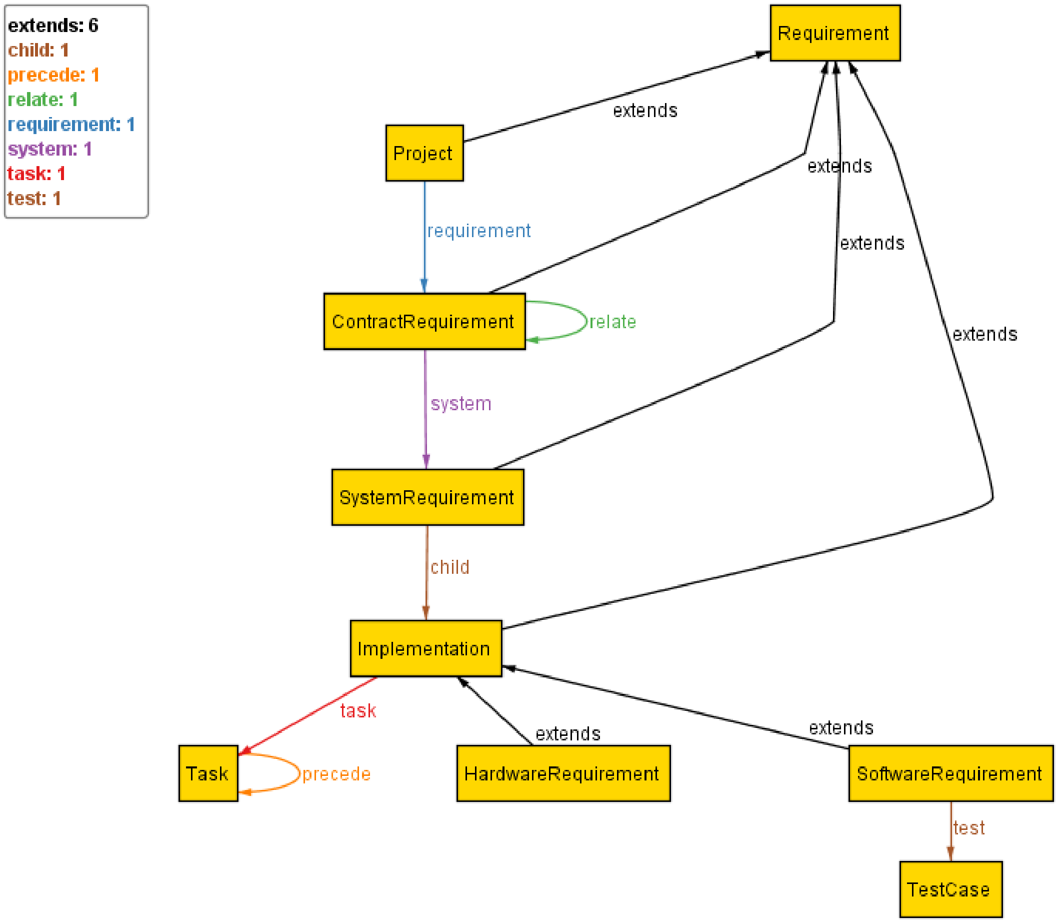
- Objective: provide the **synchronization mechanism** to keep the "**user-visible models**" consistent with the "**KB-stored models**"

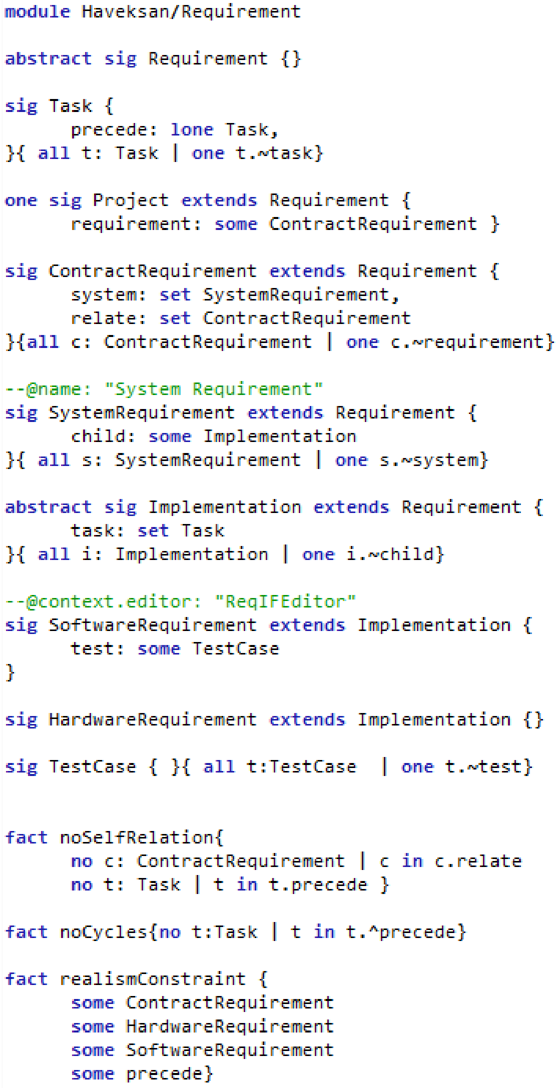
- This will consist of the following main **plug-in** components:
  - **Transformation Manager**: provides the infrastructure to register and launch transformations.
  - **Configuration Manager**: for personalizing the behaviour of the framework to meet the needs of a specific standard / organization / 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

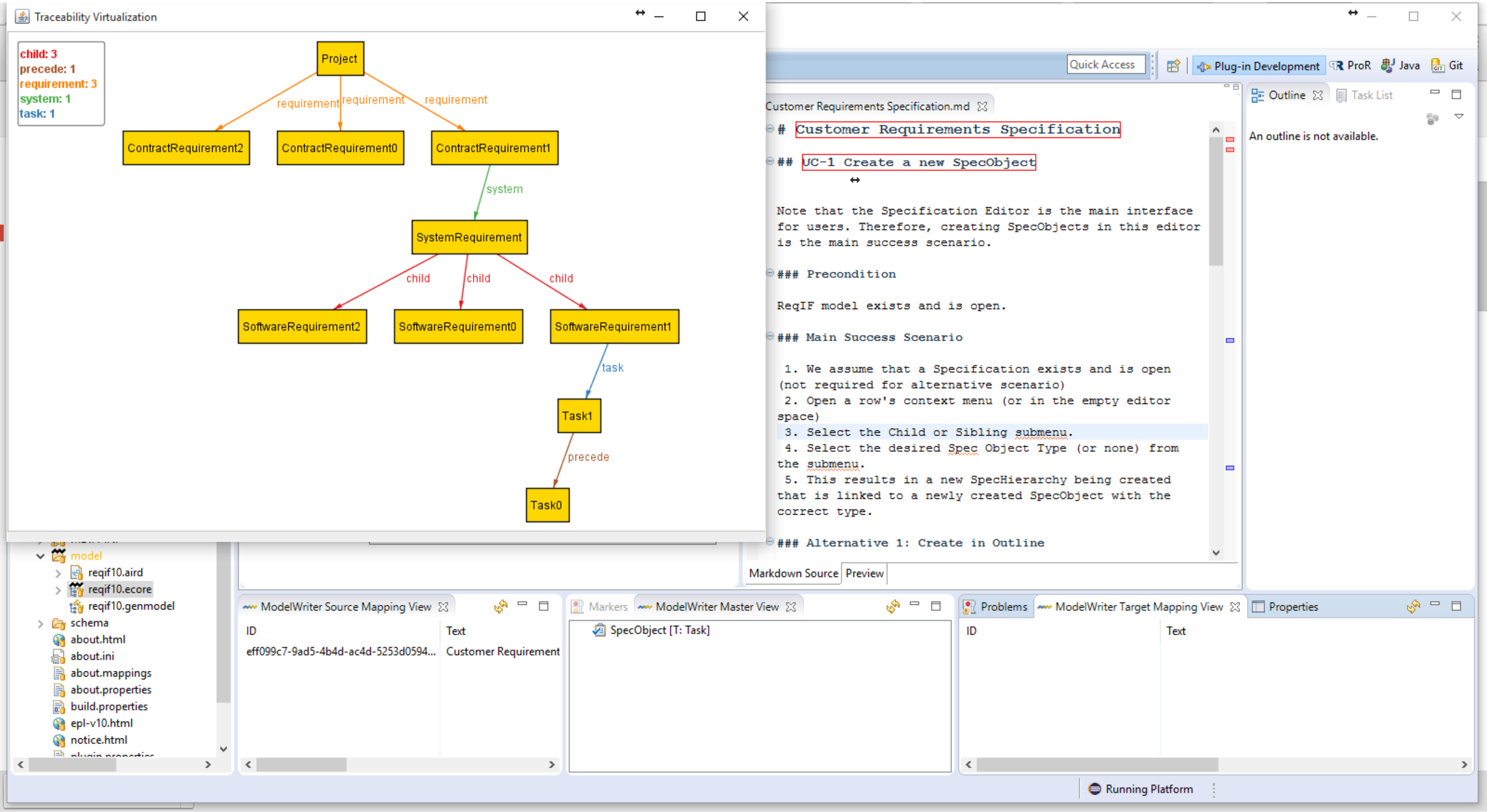
Configuration: Havelsan example





```
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}
```

Traceability: Havelsan example



```
# Customer Requirements Specification

## UC-1 Create a new SpecObject

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

### Precondition

ReqIF model exists and is open.

### Main Success Scenario

 1. We assume that a Specification exists and is open
(not required for alternative scenario)
 2. Open a row's context menu (or in the empty editor
space)
 3. Select the Child or Sibling submenu.
 4. Select the desired Spec Object Type (or none) from
the submenu.
 5. This results in a new SpecHierarchy being created
that is linked to a newly created SpecObject with the
correct type.

### Alternative 1: Create in Outline
```

An outline is not available.

ID	Text
eff099c7-9ad5-4b4d-ac4d-5253d0594...	Customer Requirement

SpecObject [T: Task]

ID	Text

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

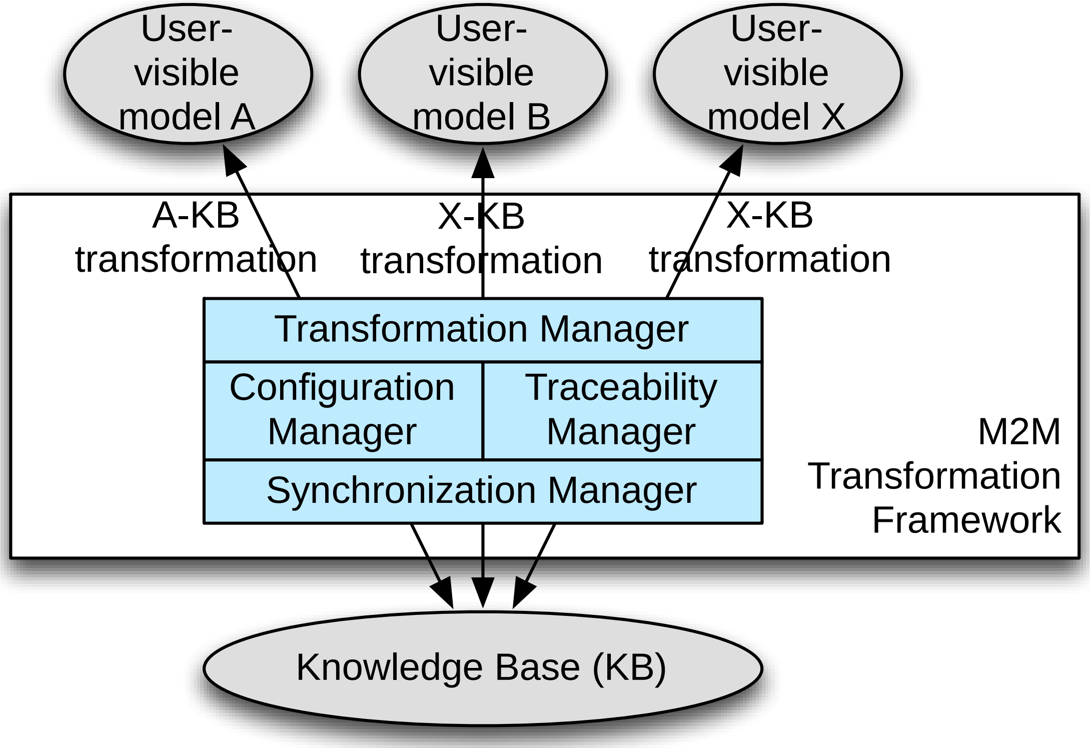
- UNIT + KOCSISTEM
  - Objective: Designing the M2M Transformation Framework whose main goal is to make the ModelWriter tool able to launch M2M transformations

- 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:



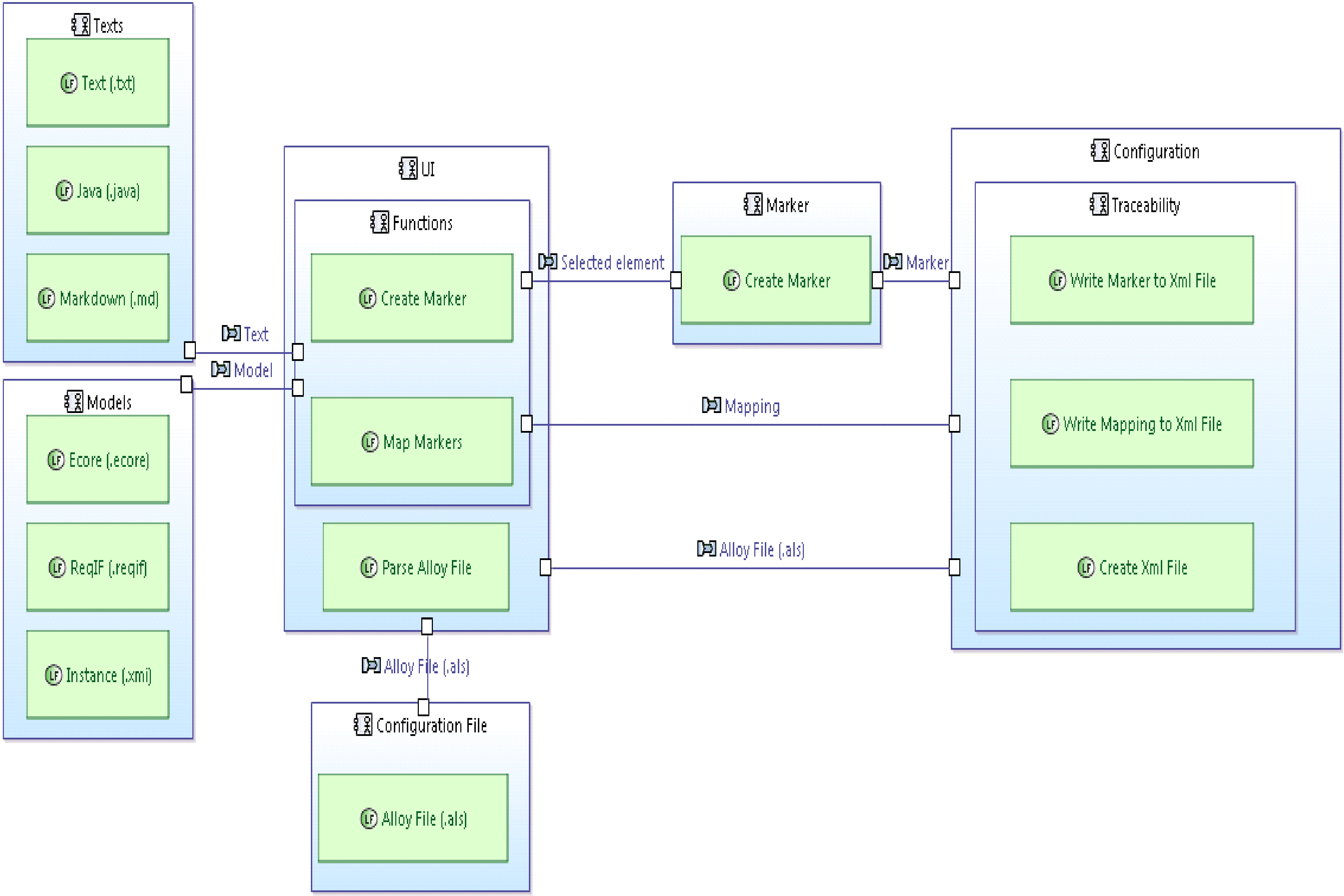
T3.3, T3.4, T3.5, T3.6 (UNIT + KOCSISTEM)

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

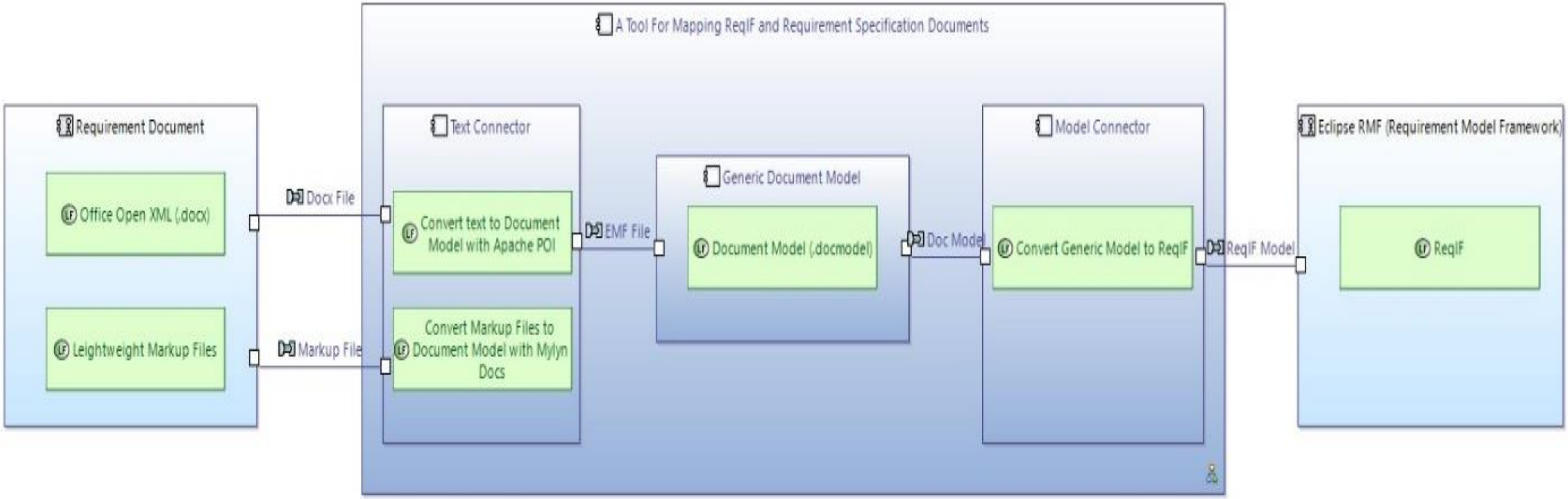
- Status:
  - These tasks has software deliverables which are developed and are available at GitHub

- 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)


A Tool For Mapping ReqIF and Requirement Specification Documents
Requirement Document
Office Open XML (.docx)
Leightweight Markup Files
Docx File
Markup File
Text Connector
Convert text to Document Model with Apache POI
Convert Markup Files to Document Model with Mylyn Docs
EMF File
Generic Document Model
Document Model (.docmodel)
Doc Model
Model Connector
Convert Generic Model to ReqIF
ReqIF Model
Eclipse RMF (Requirement Model Framework)
ReqIF

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.

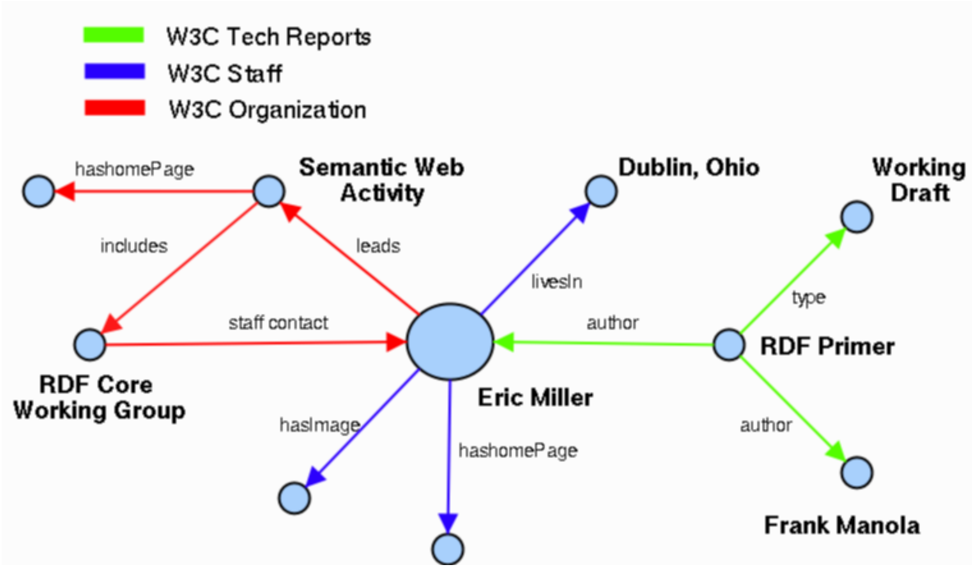
WP4 Knowledge Base Design and Implementation

- 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).

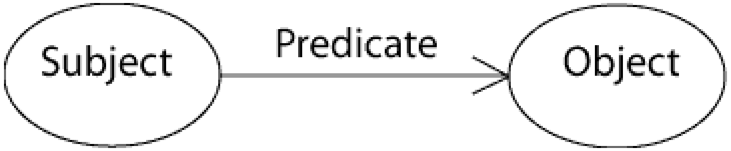
WP4 Knowledge Base Design and Implementation

- 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.

WP5

- Objectives:
  - To perform overall project governance, and to establish and maintain a communication and controlling infrastructure to run the project smoothly.

- Status:
  - Project Coordination Committee (PCC) and Technical Coordination Committee (TCC) are constituted.
  - Collaboration mechanism is stablished
  - Development management environment is created

- Next:
  - The 1st release will be issued

WP5

- T5.1 – Communication Management and Collaboration Infrastructure (UNIT + WP7 leader)
- T5.2 – Project Coordination and Reporting (UNIT + Country Coordinators + WP Leaders)
- T5.3 – Project Controls (UNIT + WP Leaders)
- T5.4 – Closing Project (UNIT + WP Leaders)

T5.1 Communication Management and Infrastructure

- Mailing lists: @modelwriter.eu
- Source Code Management: GitHub.com/modelwriter
- Project Co-authoring and document management: Google doc and GitHub
- Scrum Management: Waffle and GitHub
- Issue and Bug Tracking, etc: GitHub

T5.2 Project Coordination and Reporting

- Yearly ITEA review meetings
- National review meetings (e.g. in every 6 month in Turkey)
- International workshops: face to face meetings (in each 3 months)
- Monthly International Telco meetings
- Weekly collaboration meeting: e.g. UNIT-KS
- Action specific meetings: e.g. Airbus-UNIT action

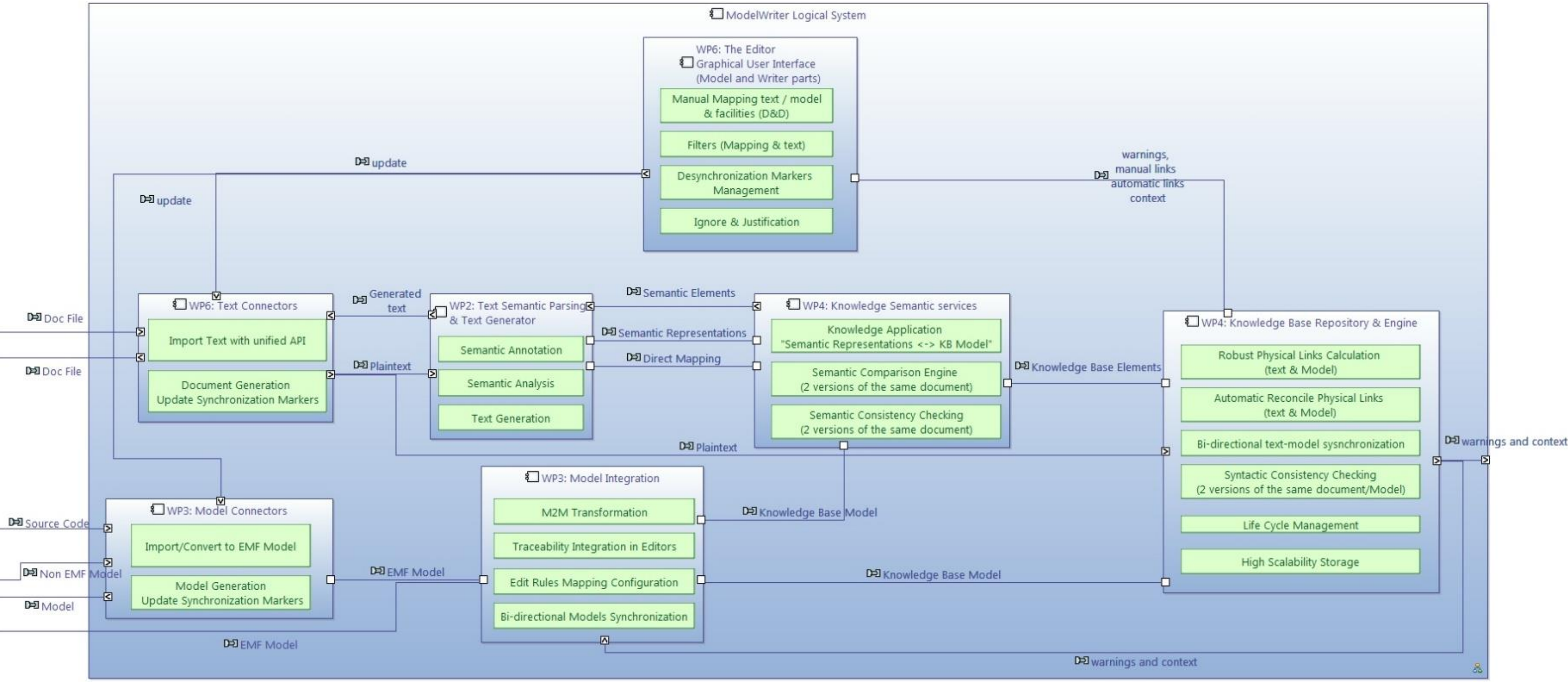
T5.3 – Project Controls

- Monitoring the deliverables
- Monitoring the requirements
- Decision mechanism for addressing issues, changes, risks
- Progress controls by means of
  - monitoring,
  - regular meetings and
  - reports such as
    - Kick-off and closing report
    - Project management plan
    - Yearly ITEA PPR
    - National progress reports

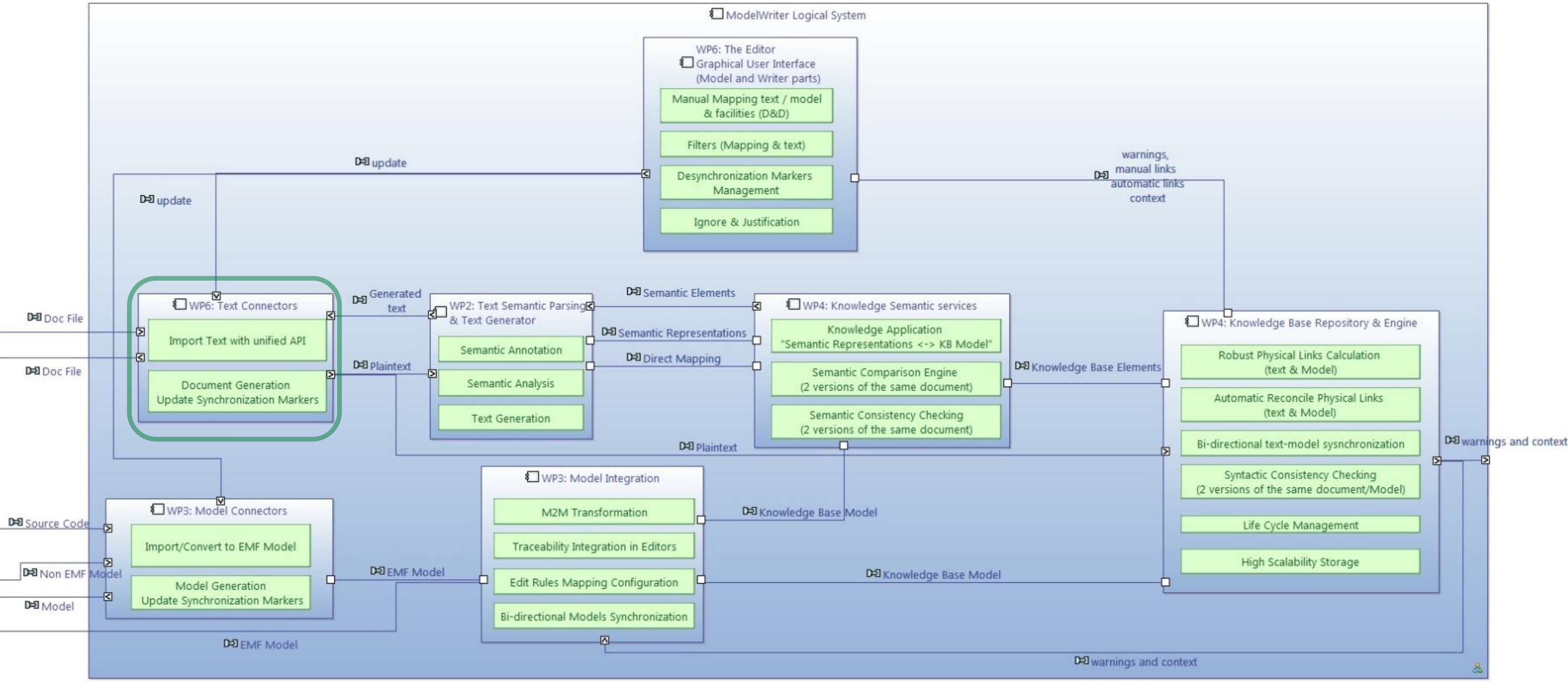
WP6 – Architecture, Integration and Evaluation

Yvan Iussaud
OBEO

WP6
ModelWriter Functional Architecture


ModelWriter Logical System
WP6: The Editor
Graphical User Interface
(Model and Writer parts)
Manual Mapping text / model
& facilities (D&D)
Filters (Mapping & text)
Desynchronization Markers
Management
Ignore & Justification
update
update
warnings,
manual links
automatic links
context
WP6: Text Connectors
Import Text with unified API
Document Generation
Update Synchronization Markers
Doc File
Doc File
Generated
text
Plaintext
WP2: Text Semantic Parsing
& Text Generator
Semantic Annotation
Semantic Analysis
Text Generation
Semantic Elements
Semantic Representations
Direct Mapping
WP4: Knowledge Semantic services
Knowledge Application
"Semantic Representations <-> KB Model"
Semantic Comparison Engine
(2 versions of the same document)
Semantic Consistency Checking
(2 versions of the same document)
Knowledge Base Elements
WP4: Knowledge Base Repository & Engine
Robust Physical Links Calculation
(text & Model)
Automatic Reconcile Physical Links
(text & Model)
Bi-directional text-model sysnchronization
Syntactic Consistency Checking
(2 versions of the same document/Model)
Life Cycle Management
High Scalability Storage
warnings and context
Plaintext
WP3: Model Integration
M2M Transformation
Traceability Integration in Editors
Edit Rules Mapping Configuration
Bi-directional Models Synchronization
Knowledge Base Model
Knowledge Base Model
WP3: Model Connectors
Import/Convert to EMF Model
Model Generation
Update Synchronization Markers
Source Code
Non EMF Model
Model
EMF Model
EMF Model
warnings and context

WP6
Text connectors



ModelWriter Logical System

WP6: The Editor
Graphical User Interface
(Model and Writer parts)

- Manual Mapping text / model & facilities (D&D)
- Filters (Mapping & text)
- Desynchronization Markers Management
- Ignore & Justification

WP6: Text Connectors

- Import Text with unified API
- Document Generation Update Synchronization Markers

WP2: Text Semantic Parsing & Text Generator

- Semantic Annotation
- Semantic Analysis
- Text Generation

WP4: Knowledge Semantic services

- Knowledge Application "Semantic Representations <-> KB Model"
- Semantic Comparison Engine (2 versions of the same document)
- Semantic Consistency Checking (2 versions of the same document)

WP4: Knowledge Base Repository & Engine

- Robust Physical Links Calculation (text & Model)
- Automatic Reconcile Physical Links (text & Model)
- Bi-directional text-model sysnchronization
- Syntactic Consistency Checking (2 versions of the same document/Model)
- Life Cycle Management
- High Scalability Storage

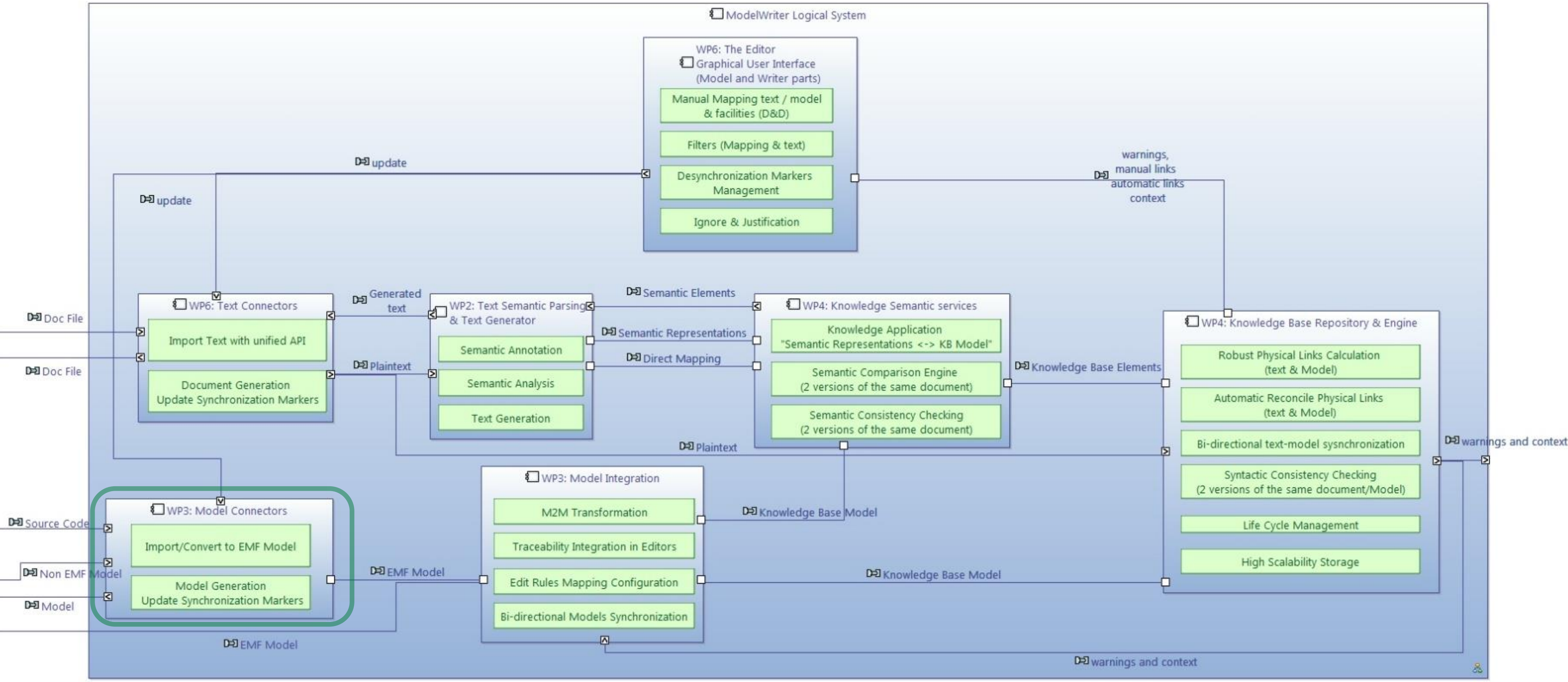
WP3: Model Connectors

- Import/Convert to EMF Model
- Model Generation Update Synchronization Markers

WP3: Model Integration

- M2M Transformation
- Traceability Integration in Editors
- Edit Rules Mapping Configuration
- Bi-directional Models Synchronization

WP6
Model connectors



ModelWriter Logical System

WP6: The Editor
Graphical User Interface
(Model and Writer parts)

Manual Mapping text / model & facilities (D&D)

Filters (Mapping & text)

Desynchronization Markers Management

Ignore & Justification

update

update

warnings, manual links automatic links context

WP6: Text Connectors

Import Text with unified API

Document Generation
Update Synchronization Markers

Doc File

Doc File

Generated text

Plaintext

WP2: Text Semantic Parsing & Text Generator

Semantic Annotation

Semantic Analysis

Text Generation

Semantic Elements

Semantic Representations

Direct Mapping

WP4: Knowledge Semantic services

Knowledge Application
"Semantic Representations <-> KB Model"

Semantic Comparison Engine
(2 versions of the same document)

Semantic Consistency Checking
(2 versions of the same document)

Knowledge Base Elements

WP4: Knowledge Base Repository & Engine

Robust Physical Links Calculation
(text & Model)

Automatic Reconcile Physical Links
(text & Model)

Bi-directional text-model sysnchronization

Syntactic Consistency Checking
(2 versions of the same document/Model)

Life Cycle Management

High Scalability Storage

warnings and context

Plaintext

WP3: Model Connectors

Import/Convert to EMF Model

Model Generation
Update Synchronization Markers

Source Code

Non EMF Model

Model

EMF Model

EMF Model

WP3: Model Integration

M2M Transformation

Traceability Integration in Editors

Edit Rules Mapping Configuration

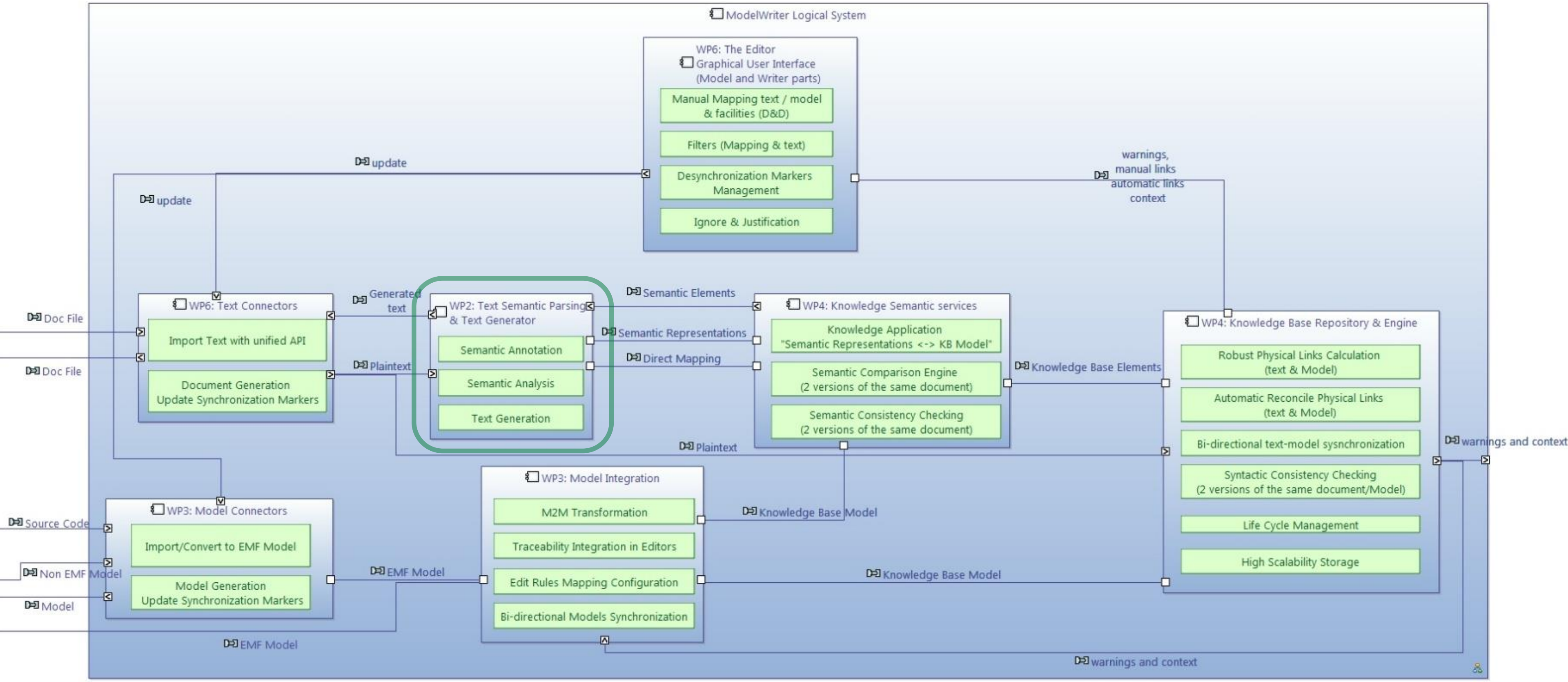
Bi-directional Models Synchronization

Knowledge Base Model

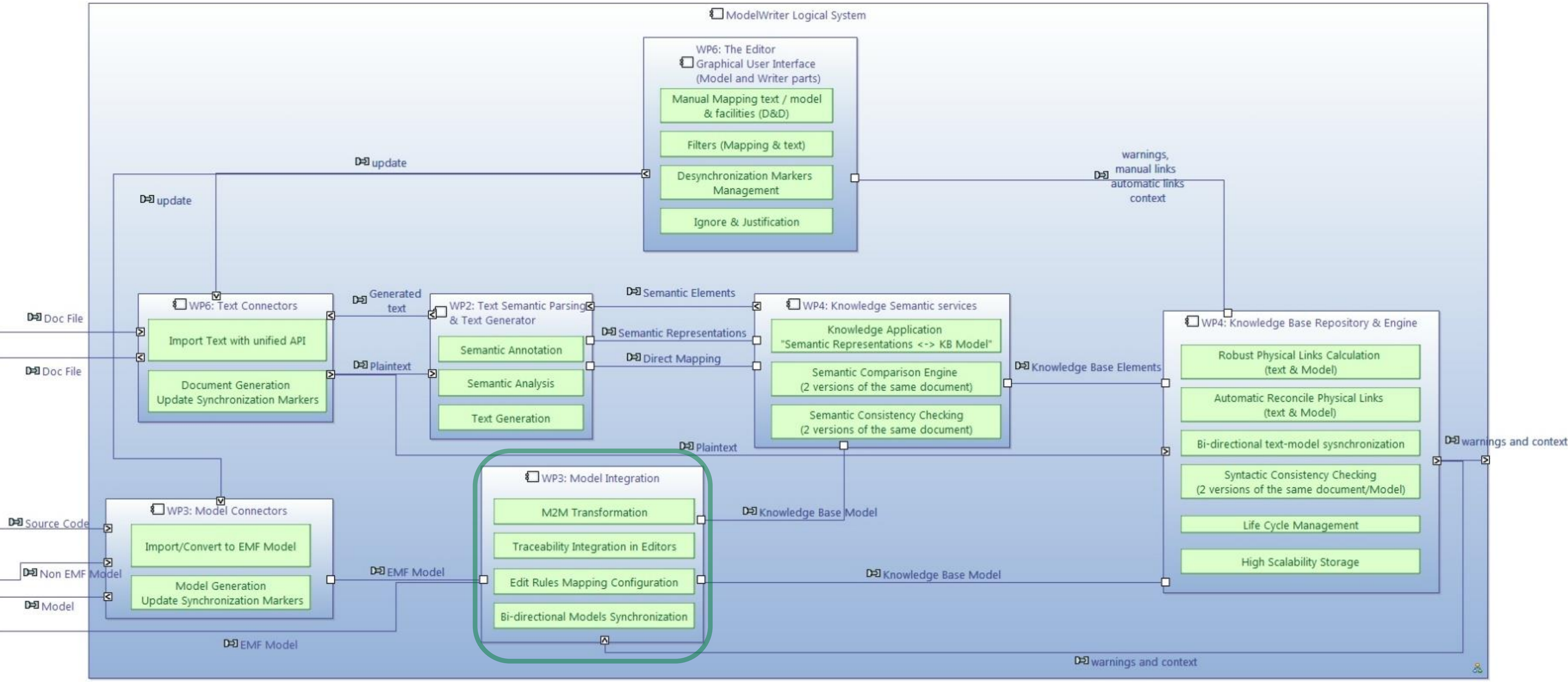
Knowledge Base Model

warnings and context

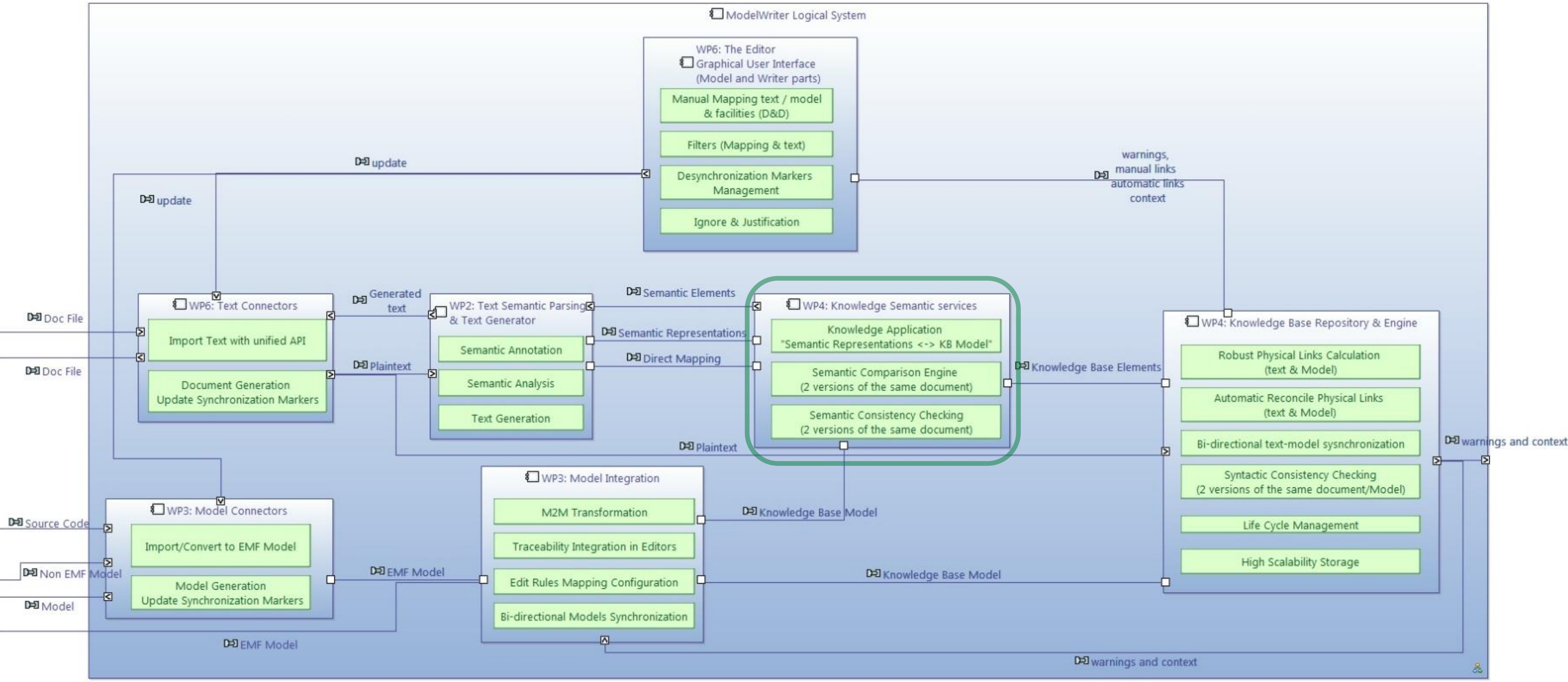
WP6
Text semantic parsing and text generator


ModelWriter Logical System
WP6: The Editor
Graphical User Interface
(Model and Writer parts)
Manual Mapping text / model
& facilities (D&D)
Filters (Mapping & text)
Desynchronization Markers
Management
Ignore & Justification
update
update
warnings,
manual links
automatic links
context
WP6: Text Connectors
Import Text with unified API
Document Generation
Update Synchronization Markers
Doc File
Doc File
Generated
text
Plaintext
WP2: Text Semantic Parsing
& Text Generator
Semantic Annotation
Semantic Analysis
Text Generation
Semantic Elements
Semantic Representations
Direct Mapping
WP4: Knowledge Semantic services
Knowledge Application
"Semantic Representations <-> KB Model"
Semantic Comparison Engine
(2 versions of the same document)
Semantic Consistency Checking
(2 versions of the same document)
Knowledge Base Elements
WP4: Knowledge Base Repository & Engine
Robust Physical Links Calculation
(text & Model)
Automatic Reconcile Physical Links
(text & Model)
Bi-directional text-model sysnchronization
Syntactic Consistency Checking
(2 versions of the same document/Model)
Life Cycle Management
High Scalability Storage
warnings and context
Plaintext
WP3: Model Integration
M2M Transformation
Traceability Integration in Editors
Edit Rules Mapping Configuration
Bi-directional Models Synchronization
Knowledge Base Model
Knowledge Base Model
WP3: Model Connectors
Import/Convert to EMF Model
Model Generation
Update Synchronization Markers
Source Code
Non EMF Model
Model
EMF Model
EMF Model
warnings and context

WP6
Model Integration


ModelWriter Logical System
WP6: The Editor
Graphical User Interface
(Model and Writer parts)
Manual Mapping text / model
& facilities (D&D)
Filters (Mapping & text)
Desynchronization Markers
Management
Ignore & Justification
update
update
warnings,
manual links
automatic links
context
Doc File
Doc File
WP6: Text Connectors
Import Text with unified API
Document Generation
Update Synchronization Markers
Generated
text
Plaintext
WP2: Text Semantic Parsing
& Text Generator
Semantic Annotation
Semantic Analysis
Text Generation
Semantic Elements
Semantic Representations
Direct Mapping
WP4: Knowledge Semantic services
Knowledge Application
"Semantic Representations <-> KB Model"
Semantic Comparison Engine
(2 versions of the same document)
Semantic Consistency Checking
(2 versions of the same document)
Knowledge Base Elements
WP4: Knowledge Base Repository & Engine
Robust Physical Links Calculation
(text & Model)
Automatic Reconcile Physical Links
(text & Model)
Bi-directional text-model sysnchronization
Syntactic Consistency Checking
(2 versions of the same document/Model)
Life Cycle Management
High Scalability Storage
warnings and context
Plaintext
WP3: Model Integration
M2M Transformation
Traceability Integration in Editors
Edit Rules Mapping Configuration
Bi-directional Models Synchronization
Knowledge Base Model
Knowledge Base Model
Source Code
Non EMF Model
Model
WP3: Model Connectors
Import/Convert to EMF Model
Model Generation
Update Synchronization Markers
EMF Model
EMF Model
warnings and context

WP6
Knowledge Semantic services



ModelWriter Logical System

WP6: The Editor
Graphical User Interface
(Model and Writer parts)
- Manual Mapping text / model & facilities (D&D)
- Filters (Mapping & text)
- Desynchronization Markers Management
- Ignore & Justification

WP6: Text Connectors
- Import Text with unified API
- Document Generation Update Synchronization Markers

WP2: Text Semantic Parsing & Text Generator
- Semantic Annotation
- Semantic Analysis
- Text Generation

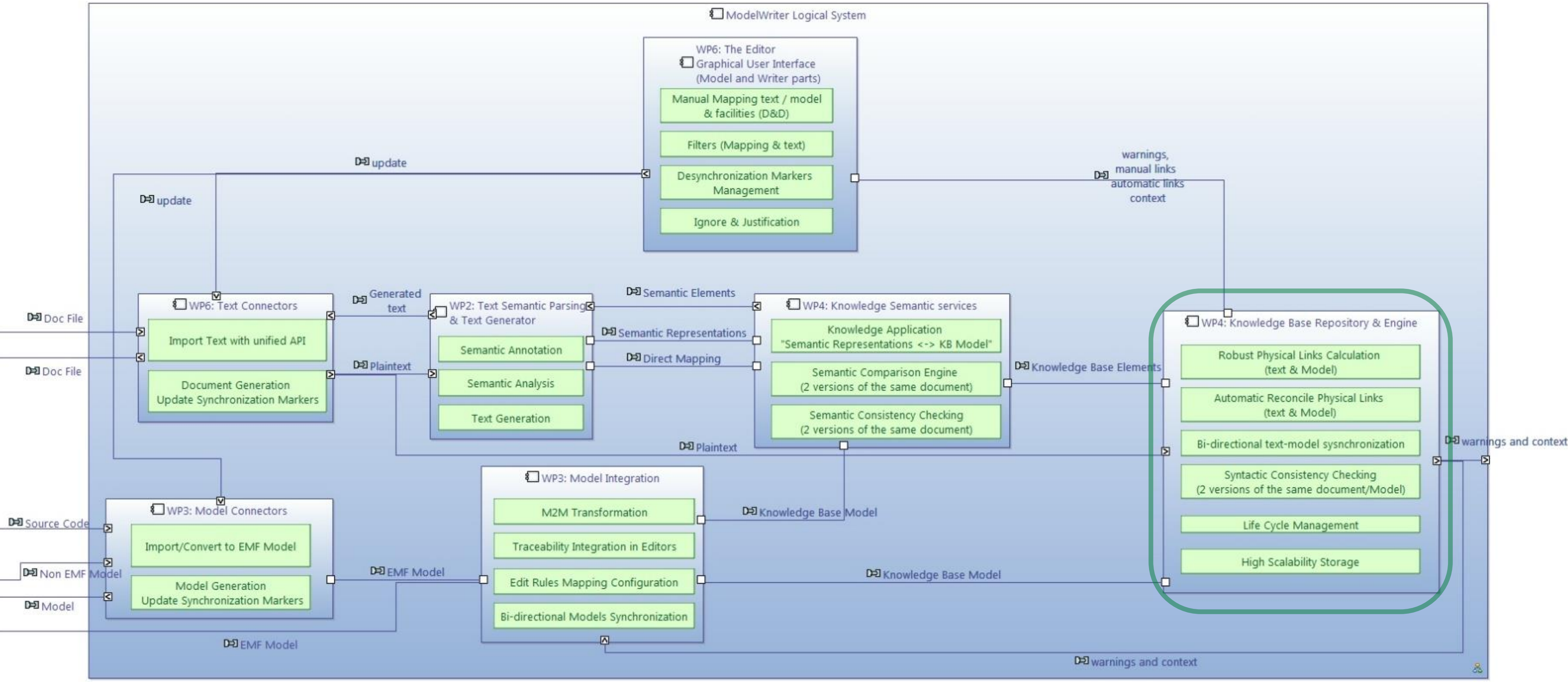
WP4: Knowledge Semantic services
- Knowledge Application "Semantic Representations <-> KB Model"
- Semantic Comparison Engine (2 versions of the same document)
- Semantic Consistency Checking (2 versions of the same document)

WP4: Knowledge Base Repository & Engine
- Robust Physical Links Calculation (text & Model)
- Automatic Reconcile Physical Links (text & Model)
- Bi-directional text-model sysnchronization
- Syntactic Consistency Checking (2 versions of the same document/Model)
- Life Cycle Management
- High Scalability Storage

WP3: Model Connectors
- Import/Convert to EMF Model
- Model Generation Update Synchronization Markers

WP3: Model Integration
- M2M Transformation
- Traceability Integration in Editors
- Edit Rules Mapping Configuration
- Bi-directional Models Synchronization

WP6
Knowledge base repository and engine



ModelWriter Logical System

WP6: The Editor
Graphical User Interface
(Model and Writer parts)

- Manual Mapping text / model & facilities (D&D)
- Filters (Mapping & text)
- Desynchronization Markers Management
- Ignore & Justification

WP6: Text Connectors

- Import Text with unified API
- Document Generation Update Synchronization Markers

WP2: Text Semantic Parsing & Text Generator

- Semantic Annotation
- Semantic Analysis
- Text Generation

WP4: Knowledge Semantic services

- Knowledge Application "Semantic Representations <-> KB Model"
- Semantic Comparison Engine (2 versions of the same document)
- Semantic Consistency Checking (2 versions of the same document)

WP4: Knowledge Base Repository & Engine

- Robust Physical Links Calculation (text & Model)
- Automatic Reconcile Physical Links (text & Model)
- Bi-directional text-model sysnchronization
- Syntactic Consistency Checking (2 versions of the same document/Model)
- Life Cycle Management
- High Scalability Storage

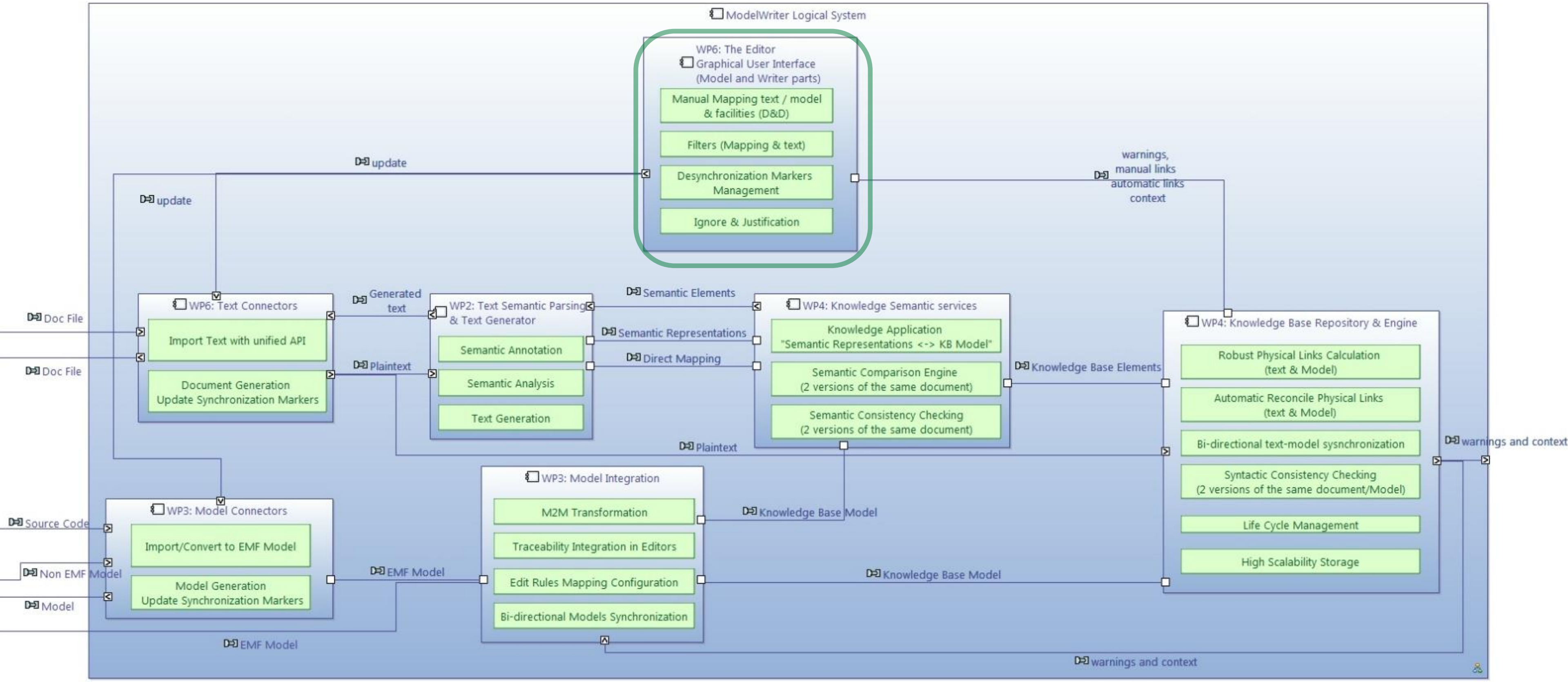
WP3: Model Connectors

- Import/Convert to EMF Model
- Model Generation Update Synchronization Markers

WP3: Model Integration

- M2M Transformation
- Traceability Integration in Editors
- Edit Rules Mapping Configuration
- Bi-directional Models Synchronization

WP6
Editors and GUI (Model and Writer parts)



ModelWriter Logical System

WP6: The Editor
Graphical User Interface
(Model and Writer parts)

- Manual Mapping text / model & facilities (D&D)
- Filters (Mapping & text)
- Desynchronization Markers Management
- Ignore & Justification

update

update

warnings, manual links automatic links context

WP6: Text Connectors

- Import Text with unified API
- Document Generation Update Synchronization Markers

Doc File

Doc File

Generated text

Plaintext

WP2: Text Semantic Parsing & Text Generator

- Semantic Annotation
- Semantic Analysis
- Text Generation

Semantic Elements

Semantic Representations

Direct Mapping

WP4: Knowledge Semantic services

- Knowledge Application "Semantic Representations <-> KB Model"
- Semantic Comparison Engine (2 versions of the same document)
- Semantic Consistency Checking (2 versions of the same document)

Knowledge Base Elements

WP4: Knowledge Base Repository & Engine

- Robust Physical Links Calculation (text & Model)
- Automatic Reconcile Physical Links (text & Model)
- Bi-directional text-model sysnchronization
- Syntactic Consistency Checking (2 versions of the same document/Model)
- Life Cycle Management
- High Scalability Storage

warnings and context

Plaintext

WP3: Model Integration

- M2M Transformation
- Traceability Integration in Editors
- Edit Rules Mapping Configuration
- Bi-directional Models Synchronization

Knowledge Base Model

Knowledge Base Model

WP3: Model Connectors

- Import/Convert to EMF Model
- Model Generation Update Synchronization Markers

Source Code

Non EMF Model

Model

EMF Model

EMF Model

warnings and context

WP6
Integration

Source code

- Github repository
- Checkstyle and code templates
- Target platforms

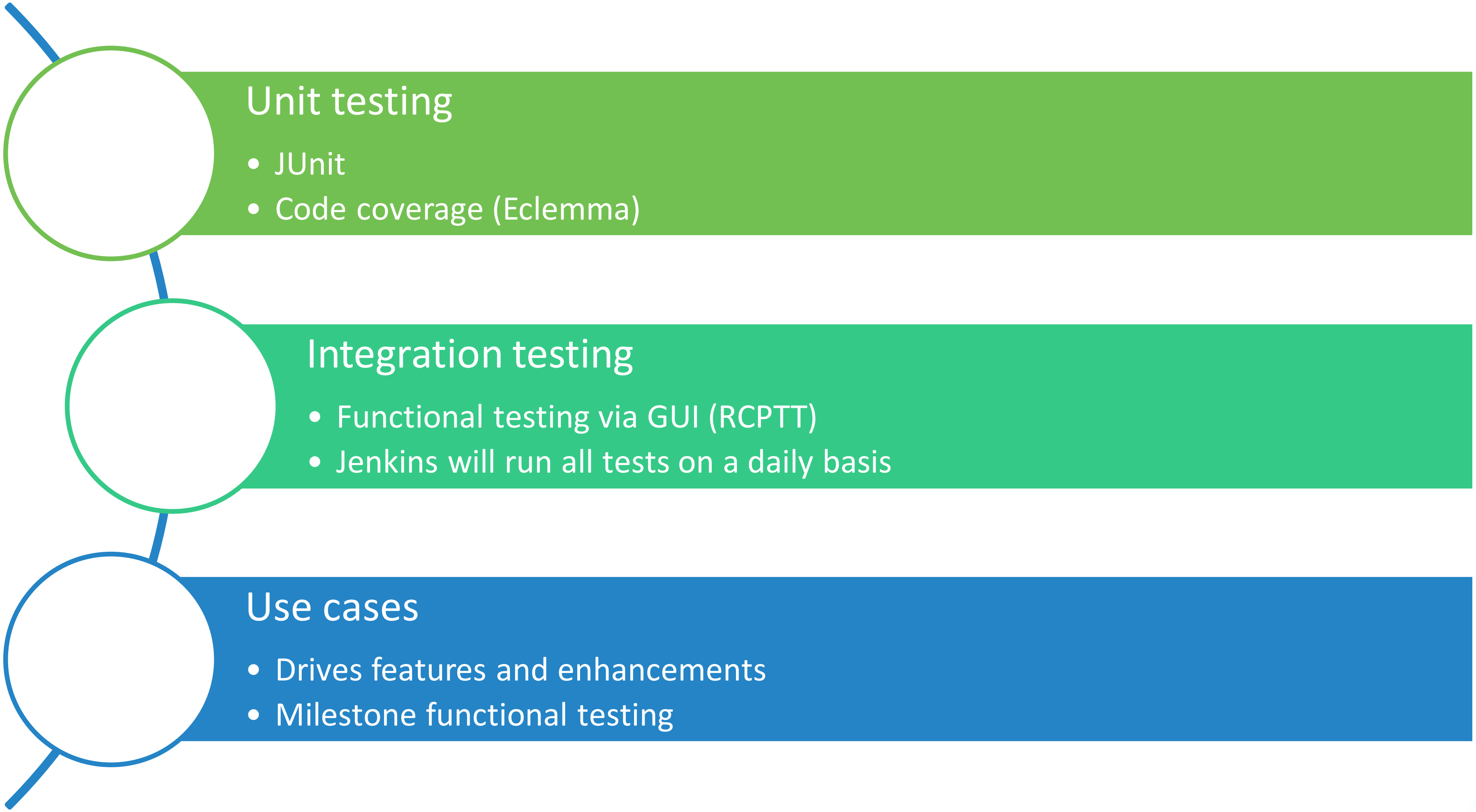
Continuous integration

- 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

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.

ITEA3

5 Demonstrations

Ferhat Erata (UNIT, ModelWriter Project Leader)

Dr. Mariem Mahfoudh (CNRS/LORIA)

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