ITEA Office
High Tech Campus 69 - 3
5656 AG Eindhoven
The Netherlands

T + 31 88 003 6136
E info@itea3.org
W www.itea3.org

ITEA 3 is a EUREKA strategic ICT cluster programme

D4.1.1
Knowledge Base Design document

ModelWriter

Text & Model-Synchronized Document Engineering Platform

Work Package: WP4
Task: T4.1 – Knowledge Base Design

Edited by:

Erhan Mengusoglu <erhanmengusoglu@mantis.com.tr> (Mantis)
…

Date: 02-Jun-2015
Version: 1.0.0

Document History

Version	Author(s)	Date	Remarks
0.1.0	Ferhat Erata Moharram Challenger	30-Apr-2015	Draft
1.0.0	Erhan Mengusoglu Yvan Lussaud	<date>	Initial Release
	Mariem Mahfoudh		
	Yvan Lussaud Anne Monceaux	29-Apr-2016	Document review
	Yvan Lussaud	27-Jun-2017	Change to reflect the code base

Table of Contents

Contents

1. Introduction

This deliverable provides basic design principles for the knowledge base which serves as the repository for metamodels.

[redacted] Goal of the Knowledge Base in ModelWriter

The Knowledge base is a master piece of the architecture of the ModelWriter product.
The Knowledge Base has the following functions:

- Found the links between text and models based on semantic relations
- Annotate texts (writer part)
- manage the links between synchronized artefacts(text and models)
- manage …
- Enrich models (ontologies) based on semantic parsing

[redacted] Acronyms

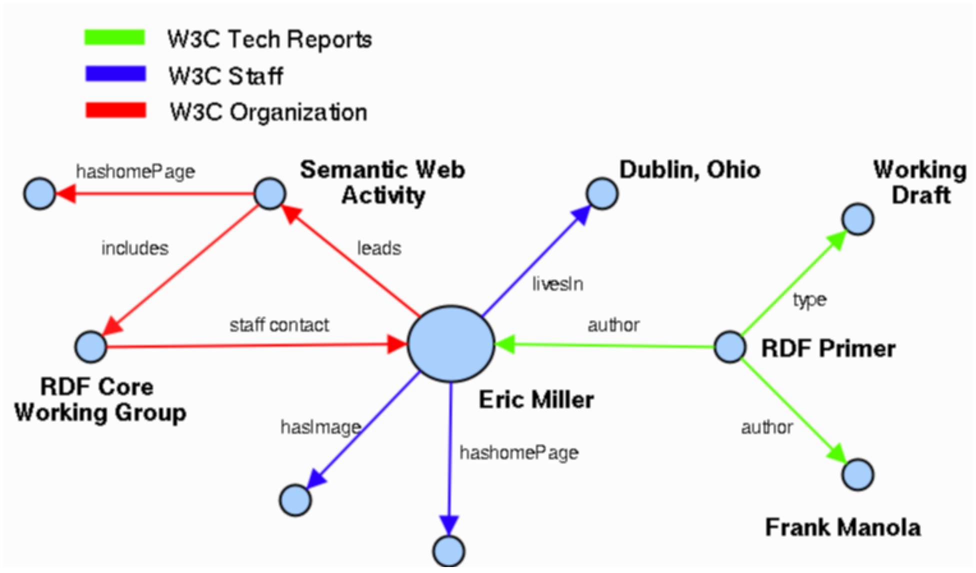
Abbreviation	Definition
RDF	Resource Description Framework
WP	Work Package
UC	Use Case

2. Definition of knowledge base and sample elements

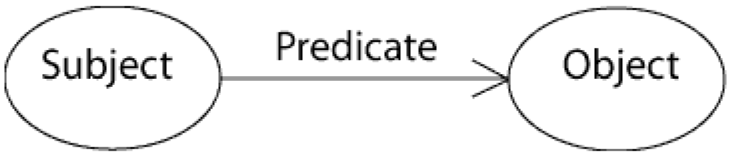
Knowledge is defined as “Facts, information, and skills acquired through experience or education; the theoretical or practical understanding of a subject” by the Oxford dictionary. In the context of the project we will take the part “theoretical or practical understanding of a subject” from this definition. In digital environment, knowledge is represented as a network of semantic definition for a particular subject using a semantic web approach.

Semantic web is originally an approach defined by W3 Consortium for creating digitally readable structures for web pages on the internet. This well-defined methodology for representing the data on web pages, later on, found to be useful for representing knowledge in different domains like biology, banking, astronomy etc.

An example semantic web is provided below:



In the project, model elements need to comply the notation of semantic web usually described as Resource Description Framework (RDF) documents. W3C describe RDF structures as “the underlying structure of any expression in RDF is a collection of triples, each consisting of a subject, a predicate and an object”.



In this notation, direction of the arc between subject and object is significant.

ModelWriter will use RDF as the meta-model for knowledge-base. By imposing the model elements to comply with RDF notation we will have standardized representation of text documents as models.

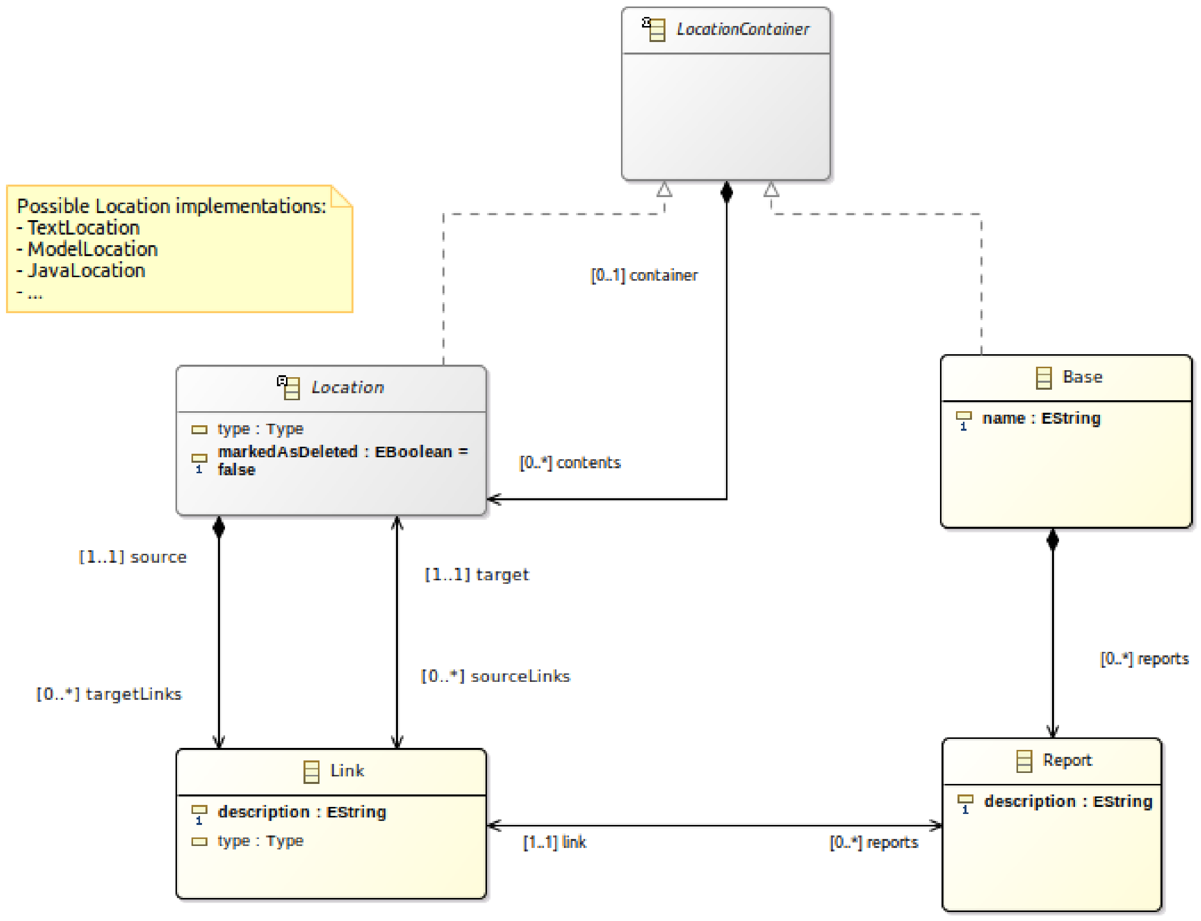
3. Definition of knowledge base to manage Model to Text synchronization

[redacted]The Knowledge Base Data Structure

The knowledge base data structure is one of those user “hidden-models” (or system models). It will be generated by ModelWriter behind the scene based on links between user inputs: text documents and models.
The Data structure model itself is not meant to be shown, and is derived from:

1. The text obtained from the “Writer” part of Modelwriter.
2. The models obtained from the “Model” part of Modelwriter.

The class diagram of the knowledge base data structure representation is given in the next figure.



In the next section we describe in detail every concept of this representation.

3.1.1. The Data Structure Description

This section describes concepts of the knowledge base from a data perspective. It will show all static data needed by each concepts.

3.1.2. The Data Structure Description

1.1.1.1. The class “Base”

The main class of this representation is “Base”. It contains all root locations a root location represents a referenced resource (e.g. a document, a model, a java class, etc.).
The Java interface “ILocation”.
A location shall have a name and shall also have enough data to reference the original artifact.
A location can represent a text location, Model location, java location, etc… In order to achieve this this class should be sub classed for specific needs.
Each location holds its own specific useful information e.g. a text location must hold all references needed to locate the concerned text part.
Each location might be contained by another location. Only locations which are linked together with links and locations which have at least one linked child location are stored in a tree structure.
Each location can be the source and the target of several links.
A location can contains reports (see 1.1.1.3) regarding the status of links (see 1.1.1.2).

1.1.1.2. The class “Link”

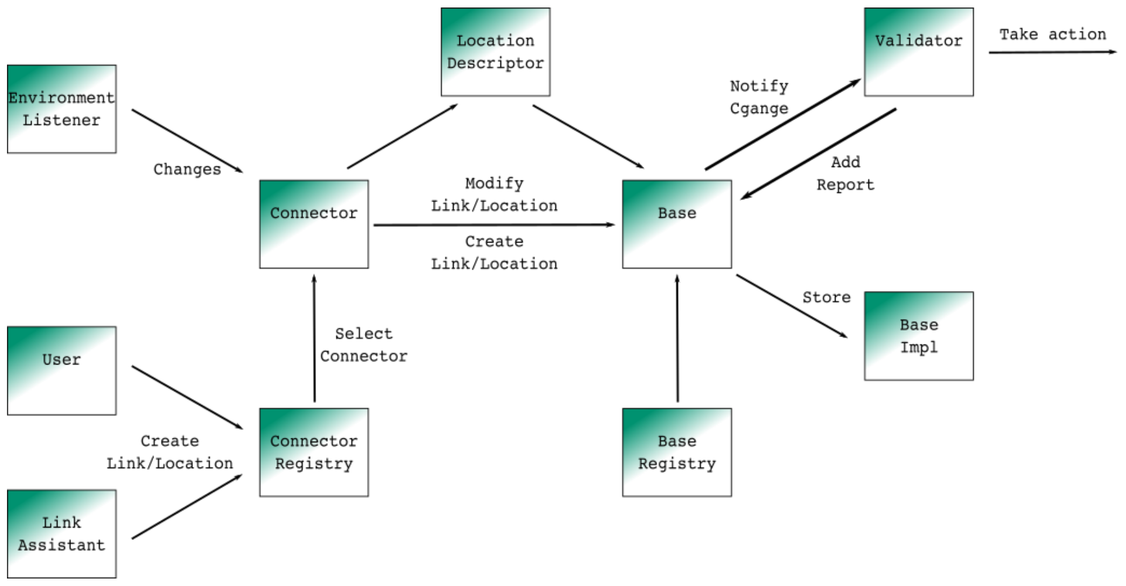
A link represents a mapping between a source location and a target location. The source location is the holder.
A link shall have a description explaining the purpose of the link. It is only used for human needs.
A link shall have a source location and a target location.

1.1.1.3. The class “Report”

A report is stored in a base (see 3.1.1) and keep track of status changes for a link (see 1.1.1.2). For the moment this class is not sub classed but it might be useful to have specific implementation for automated processing of reports.
A report shall have a description for human interactions.
A report shall have a reference to a link (see 1.1.1.2).

[redacted] The Knowledge Base Interactions

This section describes all the interactions in the knowledge base and their related constraints. The possible interactions concern the concepts of the data structure (location and link).
The last sub-section is dedicated to the notification system. The following figure shows an overview of interactions with the knowledge base.



In order to simplify the overview, interactions based on listeners are not represented here. The location descriptor is a technical artefact needed when we need to manipulate a location that is not created in base yet. For simplification it can be seen as a location.

3.2.1. Locations related interactions

1. *New location creation*: There is no specific constraint related to the creation of a new location in the knowledge base.
2. *Mark the location as changed*: All the source links and the target links of the concerned location will be marked as changed. This leads to the creation of a report on each source and tagret links of the location.
3. *Move the location*: This concerns the modification of technical information leading to localize the concerned location. In this case, there is no semantic change; the connector component must be able to recalculate the new technical information to localize the concerned location.
4. *Mark the location as deleted*: All the source links and the target links of the concerned location will be marked as deleted by creating a new report. The location itself is marked deleted with a flag to prevent further link creation.
5. *Delete a location*: The deletion of a location is only possible if it has no source links, no target links and no contained location. It can be performed as soon as all condition are met when the location was already marked as deleted.

3.2.2. Reports related interactions

4. *New report creation*: A report is created by hand, but most likely by a validator listening to the environment or the base. They are no preconditions to report creation.
5. *Report deletion*: The report deletion can be made at any time. This interaction is domain specific and further validation should be done by the user or the specific tooling deployed in the environment.

5.1.1. Links related interactions

1. *New link creation*: A link can be created by referencing a source location and a target location. A new link has no reports by default. There is no other specific constraint related to the creation of a new link in the knowledge base.
   We identify 4 types of links between Text and Model:

   1. Exact matching: identified using String matching. Ex: Attach (text element) IsSameAs http://airbus-group/opd-function#Attach (ontology concept)

   2. Morphology matching: identified using lemmatization and Stanford CoreNLP. Ex: Attached IsMorphologySimilarTo http://airbus-group/opd-function#Attach

   3. Semantic matching: identified based on the ontology and SKOS labels. Ex: Fixation isSysnonymTo http://airbus-group.installsys/component#AttachmentPoint

   4. UserLink: identified by user

2. *Edit a link*:
   a. Reconnect a link to another source location or to another target location.
   b. Reverse a link by switching its source and its target. This is a special case of the previous interaction (reconnect a link).
   c. Mark as valid after a location change or deletion. There is no specific constraint related to this interaction.

The edition of the links should be checking based on ontology's axioms and properties. For instance if a text element and an ontology concepts are semantically disjoint, then they cannot be synchronized. Ex: rigid Component **cannot be synchronized with** http://airbus-group.installsys/component#FlexibleComponent.

3. *Link deletion*: A link can only be deleted when there are no reports left referencing it. This ensure all synchronization has be done and validated before removing the link.

5.1.2. Notification system

All interaction leading to a modification of the data structure state shall notify the knowledge base listeners (the reconciler, the synchronizer, the logger, etc.).

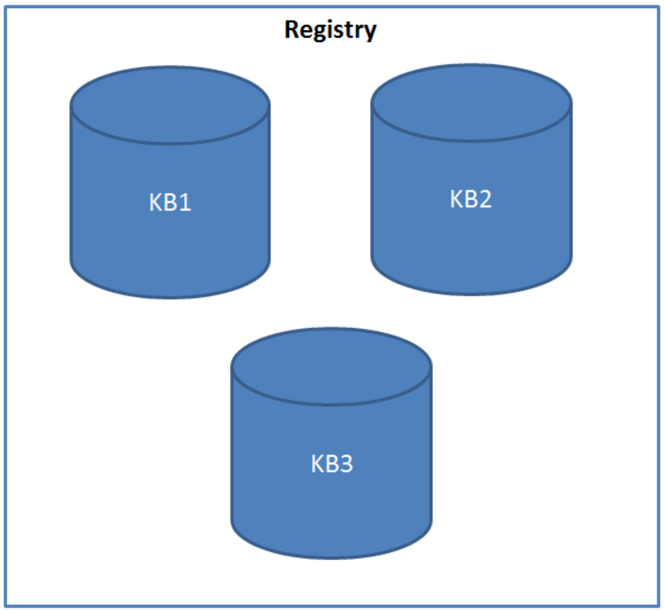
- A listener can be registered to track notifications from a location. The listener will be able to track all the notified change of a location or of one of its contents.
- A listener can be registered to track notifications from a link. The listener will be able to track all the notified change of the link status.

The notification system might allow easy filtering notifications (listen only the creation of links notifications, etc.)

[redacted]Multiple Knowledge Bases Management

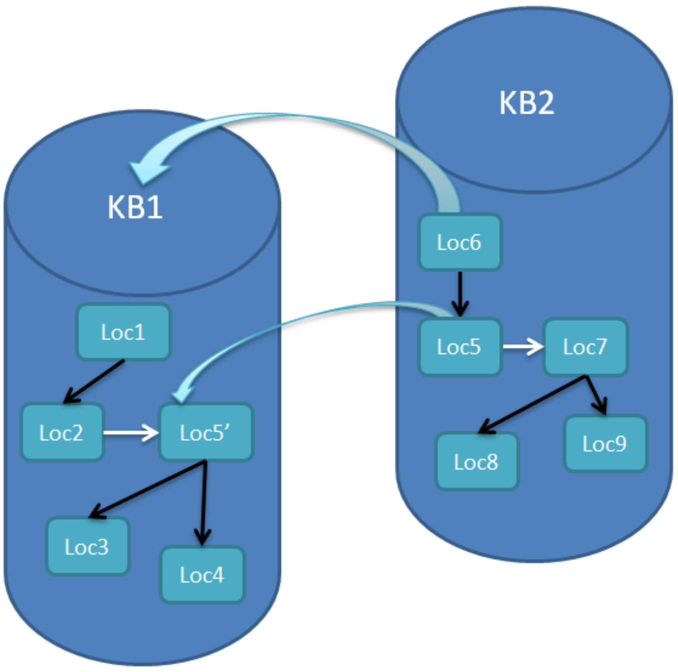
5.2.1. Registry

The Knowledge base might need to contact another knowledge base, knowing only the link, but not how to contact it. A registry is needed to provide a service and access the connected knowledge bases.
It will be useful to store links created in the editor.



5.2.2. Collaboration

A knowledge base ***KB1*** collaborates in another knowledge base ***KB2*** by offering the access to its locations and links content. The location ***Loc6*** in ***KB2*** represents the ***KB1***. The location ***Loc5*** in ***KB2*** represents the location ***Loc5'*** of ***KB1***. From the ***KB2*** point of view ***KB1*** is like any other resource. It is accessed via a connector dedicated to knowledge bases management. The connector will listen to notification from ***Loc5'*** in ***KB1*** and forwards notifications to the ***KB2***. For instance if ***Loc5'*** is marked as changed in ***KB1***, the connector will mark ***Loc5*** as changed after receiving the change notification in ***KB2***.


KB1
KB2
Loc1
Loc2
Loc5'
Loc3
Loc4
Loc6
Loc5
Loc7
Loc8
Loc9

6. Use of Knowledge Base in ModelWriter use cases

Knowledge base usage scenarios for each use case to be provided here.

[redacted]UC-FR1 – Synchronization between models and documentation (OBEO – Sirius Product)

In this use case we will mainly rely on the model to text synchronization of the knowledge base. The semantic annotation module (LORIA) will be an improvement of the ModelWriter user experience by providing suggestions in the creation the links and also in regard of modification synchronization.
The ontologies are mainly used to detect the semantic correspondences between texts elements and model concepts.
The idea here is to ensure the reliability of the link structure and check the semantic analysis on a known scope. This will prepare us for UC-FR2.

[redacted]UC-FR2 – Enterprise Architecture (OBEO – SmartEA Product)

In this use case we will use the model to text synchronization of the knowledge base and the semantic analysis. The goal is to rely on the semantic analysis to provide advanced feature in SmartEA. With this use case we will check scalability of the knowledge base.

[redacted]UC-FR3 – Synchronization of regulation documentation with a design rule repository (AIRBUS GROUP)

In this use case we will rather focus on the semantic annotation modules; the scenario being to use an OWL file that represents the domain knowledge (Model part) to create links with text. Then the synchronization mechanism based on the KB would be used.

4.4. UC-FR4 - Ontology enrichment (model) from documentations (LORIA – Airbus Group)

In this use case, we will target the conception and the implementation of a tool that evolve an ontology from documentations. The idea consists to parsing documentations such that text can be automatically mapped to formal models. Then, using this formal models to enrich ontologies.
Among the applications, we consider the technical documents of Airbus company as textual data (writer part) and also the ontologies of Airbus as models.

7. Representation of ontological structures in the knowledge base

Ontological structures are represented as RDF documents in the knowledge base.

8. Conclusion

This deliverable will serve as a reference document for designing and implementing model to text and text to model transformations. Bases for synchronizing models with texts and vice versa are also provided in this document.

References

[1] Wang, Xiao Hang, et al. "Ontology based context modeling and reasoning using OWL." Pervasive Computing and Communications Workshops, 2004. Proceedings of the Second IEEE Annual Conference on. Ieee, 2004..

[2] http://www.cs.uu.nl/docs/vakken/b3ii/Intelligente%20Interactie%20literatuur/College%205.%20Context%20Awareness%20en%20Ubiquitous%20Computing%20(Dignum)/Ontology%20for%20contexts%20(verplicht).pdf

Appendixes

N/A