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

D4.2.1 Interface Control Document (ICD)

ModelWriter

Text & Model-Synchronized Document Engineering Platform

Project number: ITEA 2 13028
Work Package: WP4 Knowledge Base Design and Implementation
Task: T4.2 - API of the Knowledge Base

Edited by:

Ferhat Erata <ferhat.erata@unitbilisim.com> (UNIT)
Moharram Challenger <moharram.challenger@unitbilisim.com> (UNIT)
Geylani Kardaş geylani.kardas@ege.edu.tr (KoçSistem)

Date: 07-June-2015
Document version: 1.0.0

Document History

Version	Author(s)	Date	Remarks
0.5.0	Ferhat Erata Moharram Challenger	07-June-2015	Draft
1.0.0	Mehmet Onat Geylani Kardas	09-Sep-2015	Providing the content including the interface and description

Table of Contents

1. Introduction

Role of the deliverable

This document provides the Interface Control Document (ICD), which specifies the API for accessing & manipulating the Knowledge Base.

The List of Technical Work Packages

UC Code	Requirements derived from
WP2	Semantic Parsing and Generation of Documents and Documents Components
WP3	Model to/from Knowledge Base (synchronization mechanism)
WP4	Knowledge Base Design and Implementation
WP6	Architecture, Integration and Evaluation

Structure of the document

This document is organized as follows:

- Chapter 1 introduces the document.
- Chapter 2 the interface.
- Chapter 3 concludes the document.

Terms, abbreviations and definitions

Abbreviation	Definition
RDF	Resource Description Framework
WP	Work Package
UC	Use Case
ICD	Interface Control Document

2. Interface Control Document (ICD)

```java
package synalp.commons.input.knowledgeBase;

import java.io.IOException;
import java.util.Set;

import com.hp.hpl.jena.ontology.DatatypeProperty;
import com.hp.hpl.jena.ontology.Individual;
import com.hp.hpl.jena.ontology.ObjectProperty;
import com.hp.hpl.jena.ontology.OntClass;
import com.hp.hpl.jena.rdf.model.Resource;
import com.hp.hpl.jena.util.iterator.ExtendedIterator;

public interface IOntologyAnalysis {

  // Method that provides the list of the ontology's classes
  /**
   * @return a Set of OntClass(Interface that represents an ontology node characterising a class
   *         description)
   */
  public abstract Set<OntClass> getClasses();

  // Method that creates a text from the label skos definition
  /**
   * @param fileTextFromKB that is text from Knowledge Base
   */
  public abstract void CreateTextFromDefinition(String fileTextFromKB) throws IOException;

  // Method that provides the list of the ontology's datatypesPoperties
  /**
   * @return an ExtendedIterator of DatatypeProperty(Interface that encapsulates the class of
   *         properties whose range values are datatype values)
   */
```

```
public abstract ExtendedIterator<DatatypeProperty> getDatatypeProperties();

// Method that provides the list of the ontology's objectPoperties
/**
 * @return an ExtendedIterator of ObjectProperty(Interface encapsulating properties whose range
 *         values are restricted to individuals)
 */
public abstract ExtendedIterator<ObjectProperty> getObjectProperties();

// Method that provides the list of the ontology's individuals
/**
 * @return a Set of Individual(Interface that encapsulates an individual in an ontology, sometimes
 *         referred to as a fact or assertion, or a member of the a-box. In order to be recognised
 *         as an individual, rather than a generic resource, at least one rdf:type statement,
 *         referring to a known class, must be present in the model)
 */
public abstract Set<Individual> getIndividuals();

// Method that provides the list of all ontology's concepts
/**
 * @return a Set of Resource(An RDF Resource)
 */
public abstract Set<Resource> getOntoConcepts();

// Method that provides the resources corresponding to a word
/**
 * @param word which will be linked.
 * @return an OntClass(Interface that represents an ontology node characterising a class
 *         description)
 */
public abstract OntClass getResource(String word);

// Method that checks if two classes are disjoint or not
/**
 * @param c1 that is OntClass (Interface that represents an ontology node characterising a class
     *         description)
     * @param c2 that is OntClass (Interface that represents an ontology node characterising a class
     *         description)
     * @return true or false
     */
    public abstract boolean isDisjoint(OntClass c1, OntClass c2);

}
```

3. Conclusion and way forward

This document provides the Interface Control Document (ICD), which specifies the API for accessing & manipulating the Knowledge Base.

In the second year of the implementation of these interfaces will be realized and integrated in the project.

References

N/A