SEVENTH FRAMEWORK PROGRAMME

INFORMATION AND COMMUNICATION TECHNOLOGIES

Project:

Accessibility Assessment Simulation Environment for New Applications Design and Development
(ACCESSIBLE, Grant Agreement No. 224145)

Deliverable number and title:

D 4.3 - A set of guidelines for the validation and integration of the implemented tools and methodologies

Lead beneficiary:	FFCUL
WP. no, title and activity type	WP4 – **Services and Content Knowledge Infrastructure** (RTD)
Contributing Task (s)	T4.4, T4.2, T4.3, T4.1
Dissemination level	PU - Public
Delivery date	April 2011
Status	Update
File name and size	"D4.3_updated_version_ACCESSIBLE_FFCUL_WP4-FD04_2011_v01.doc ", 290 Kb

(*This page has been left empty in purpose.*
The same stands after each main section.)

Authors List

Leading Author (*Editor*)				
Surname		*Initials*	*Beneficiary Name (Short Name*	*Contact email*
Lopes		R.	FFCUL	rlopes@di.fc.ul.pt
Co-authors (*In alphabetic order*)				
#	*Surname*	*Initials*	*Beneficiary Name (Short Name*	*Contact email*
1	Tzovaras	D.	CERTH/ITI	Dimitrios.Tzovaras@iti.gr
2	Votis	K.	CERTH/ITI	kvotis@iti.gr
3	Carriço	L.	FFCUL	lmc@di.fc.ul.pt
4	Bandeira	R.	FFCUL	rbandeira@lasige.di.fc.ul.pt
(*add lines as appropriate*)				

Peer Reviewers List

#	*Surname*	*Initials*	*Contact email*
1	Kehagias	D.K.	diok@iti.gr
2	Grudeva	P.G.	petia@marie-curie-bg.org

History Table

Date	*Comments*
September 2010	Final version delivered to EC
February 2011	Updated version
April 2011	Updated version with new comments, finalisation and delivered to EC

Executive Summary

This is the updated version of the Deliverable 4.3 that was initially issued in September 2010. The current Deliverable adopts the SMART methodology (as it was requested by the EC reviewers during the 2nd review) for the envisaged requirements upon which all the ACCESSIBLE WP4 outcomes will be based.

The integration of the ACCESSIBLE Content and Knowledge Infrastructure (e.g., the ACCESSIBLE ontology) is crucial for the configurability and personalisation characteristics of all the accessibility evaluation and simulation tools created within the context of the ACCESSIBLE project. These provide the brain for all the knowledge modelled into the ACCESSIBLE ontology, as specified in the ACCESSIBLE Harmonized Methodology.

All requirements for the ACCESSIBLE knowledge infrastructure are discussed in the light of their corresponding developments, thus providing a validation of all the methodologies and approaches taken in the course of Work Package 4. More specifically, the meta models will be validated regarding their semantics, through their coverage of different accessibility subjects. The rules inference engine and accompanying will be validated against a battery of tests

Furthermore, this deliverable documents the updates and advances on the ACCESSIBLE Inference Engine, and provides example configurations and clients for the integration facilities provided by this component.

Table of contents

List of figures

List of abbreviations and acronyms

(*in alphabetic order*)

JVM	Java Virtual Machine
OWL	Web Ontology Language
RDF	Resource Description Framework
SOAP	Simple Object Access Protocol
SPARQL	SPARQL Query Language for RDF
SWRL	Semantic Web Rules Language
WCAG2.0	Web Content Accessibility Guidelines 2.0

1 Introduction

The ACCESSIBLE Content and Knowledge Infrastructure (e.g., the ACCESSIBLE ontology) is a crucial component in the configurability and personalisation characteristics for all the tools created within the context of the ACCESSIBLE project, as specified by the ACCESSIBLE Harmonized Methodology.

This deliverable presents the results from the ACCESSIBLE Content and Knowledge Infrastructure, as thoroughly detailed in previous deliverables, D4.1 (A set of formalisms and taxonomies for accessibility assessment procedures and their inherent meta-models) and D4.2 (A software package containing a set of modelling tools, rules inference engine, and the rules graphical editor).

This report details several achievements and measurable objectives for the inference engine and associated tools and ontologies from Work Package 4. It provides a base ground upon which ACCESSIBLE tools developers can build their tools on. This way, all tests, goals, and metrics defined for these tools can be focused at Work Package 6, Pilot Plans, and Work Package 7, Pilot Demonstrations, since tools are the main results delivered by the ACCESSIBLE Project, targeted at developers and designers.

Therefore, the goal of this deliverable is centred on ensuring that the ACCESSIBLE Ontology and associated tools have been designed and implemented according to the goals and requirements for the ACCESSIBLE Content and Knowledge Infrastructure, as delineated in Work Package 2 Deliverable D2.2 (User needs and System Requirements Specification).

Furthermore, we present a new version of the ACCESSIBLE Inference Engine, hereinafter referred as OntologyQuery. This new version tackles the improvement of the Inference Engine detailed in Deliverable D4.2, as well as how its inner architecture can be leveraged towards its integration into other services.

The next Sections of this deliverable present an analysis of each requirement and how it has been fulfilled within Work Package 4 and associated Work Packages (e.g., WP3), and will also detail on the particular aspects of the inference engine.

2 Testing and Validation

In Deliverable D2.2 (User Needs and System Requirements Specification), a set of requirements were devised for the ACCESSIBLE Ontological Knowledge Resource, i.e., all the components specified and implemented in the context of Work Package 4. In the light of these requirements, we present next how they have been accomplished.

Each requirement is specified in a **SMART** manner (*Specific, Measurable, Attainable, Relevant and Timebound*). This allows for a precise definition of all user needs and system requirements as defined in D2.2. According to the SMART definition the term *Specific* refers to a more specific description that will explain what exactly is required and is addressed by "***Specific Description***". The term *Measurable* refers to the test that need to be performed in order to verify that this requirement has been met and it is addressed by "***Measurable***". The term *Attainable* refers to the constraints for each requirements and is addressed by "***Attainable***". The term *Relevant* refers to the relevance of the requirement and whether the requirement was easily achieved. It is addressed by "***Relevant***". Finally, the term *Time-bound* refers to the specific time, when a requirement has been met or is scheduled to be met in the future. It is addressed by "***Time-bound***".

2.1 Generic functional requirements

G-REQ3-1*: The ontology will support the OWL Web Ontology Language.*

Specific Description: The ACCESSIBLE ontology was specified using Semantic Web technologies, including RDF, RDF-S, and OWL.

An example excerpt of the ACCESSIBLE ontology is presented below, where RDF, RDF-S, and OWL namespaces appear:

```
<owl:Class rdf:ID="Application"/>
  <owl:ObjectProperty rdf:ID="User_linksTo_Approach">
    <owl:inverseOf>
      <owl:ObjectProperty rdf:ID="Approach_linksTo_User"/>
    </owl:inverseOf>
    <rdfs:range rdf:resource="#Approach"/>
    <rdfs:domain rdf:resource="#User"/>
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="Device_belongsTo_Disability">
    <rdfs:domain rdf:resource="#Device"/>
    <rdfs:range rdf:resource="#Disability"/>
    <owl:inverseOf>
      <owl:ObjectProperty rdf:ID="Disability_has_Device"/>
    </owl:inverseOf>
```

```
</owl:ObjectProperty>
```

Measurable: The ontology should be validated by the OWL validator, provided by Manchester University[1], in order to ensure that it is a valid OWL document. Also the inference engine will validate the correctness of the implemented ontologies.

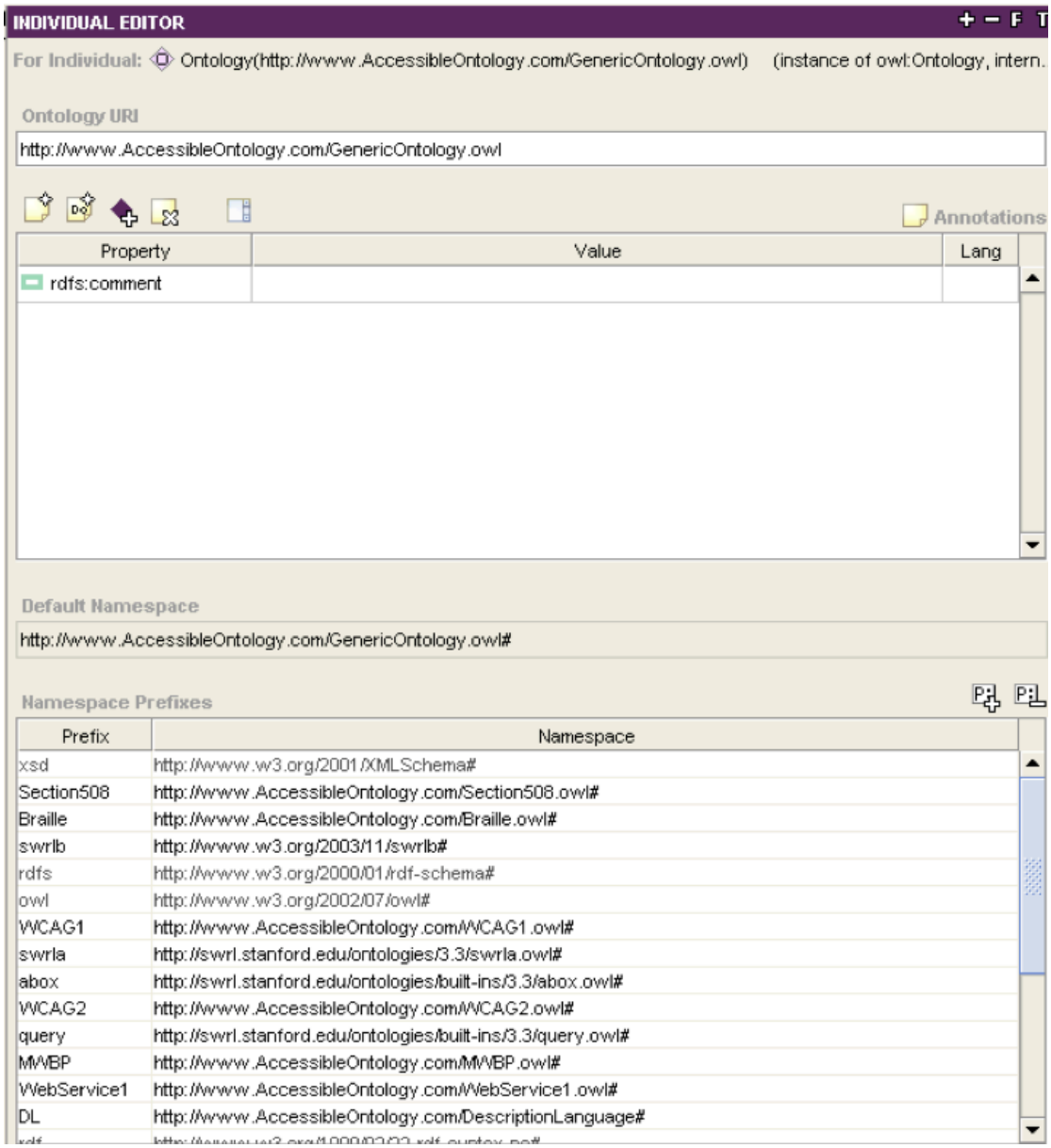
Attainable: OWL Web Ontology Language is already the most widely used language for describing ontologies, therefore, it was used to describe the ontologies developed within the premises of ACCESSIBLE. An OWL parser is needed for the access of the ACCESSIBLE ontologies.

Relevant: Within the availability of ACCESSIBLE resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is met throughout the development of the different versions of the ACCESSIBLE ontologies in the OWL format.

Furthermore, we present below a renderisation of the ACCESSIBLE ontology within the Protégé ontology modelling tool, detailing the namespaces within the Generic Ontology (including RDFS and OWL namespaces):

[1] http://owl.cs.manchester.ac.uk/validator/


INDIVIDUAL EDITOR
For Individual: Ontology(http://www.AccessibleOntology.com/GenericOntology.owl) (instance of owl:Ontology, intern.
Ontology URI
http://www.AccessibleOntology.com/GenericOntology.owl
Annotations
Property
Value
Lang
rdfs:comment
Default Namespace
http://www.AccessibleOntology.com/GenericOntology.owl#
Namespace Prefixes
Prefix
Namespace
xsd http://www.w3.org/2001/XMLSchema#
Section508 http://www.AccessibleOntology.com/Section508.owl#
Braille http://www.AccessibleOntology.com/Braille.owl#
swrlb http://www.w3.org/2003/11/swrlb#
rdfs http://www.w3.org/2000/01/rdf-schema#
owl http://www.w3.org/2002/07/owl#
WCAG1 http://www.AccessibleOntology.com/WCAG1.owl#
swrla http://swrl.stanford.edu/ontologies/3.3/swrla.owl#
abox http://swrl.stanford.edu/ontologies/built-ins/3.3/abox.owl#
WCAG2 http://www.AccessibleOntology.com/WCAG2.owl#
query http://swrl.stanford.edu/ontologies/built-ins/3.3/query.owl#
MWBP http://www.AccessibleOntology.com/MWBP.owl#
WebService1 http://www.AccessibleOntology.com/WebService1.owl#
DL http://www.AccessibleOntology.com/DescriptionLanguage#

***G-REQ3-2**: The ontology shall support the SWRL (a Semantic Web Rule Language) rules, which form of an implication between an antecedent (body) and consequent (head).*

Specific Description: The ACCESSIBLE ontology supports SWRL rules, in order to specify general production rules to establish interconnections between RDF triples.

Below we present an excerpt of a SWRL rule specified in the Generic Ontology:

```
<swrl:Imp rdf:ID="Rule_GuidelineToOutputResult">
    <swrl:head>
      <swrl:AtomList>
        <rdf:rest rdf:resource="http://www.w3.org/1999/02/22-rdf-
syntax-ns#nil"/>
        <rdf:first>
          <swrl:IndividualPropertyAtom>
            <swrl:propertyPredicate
rdf:resource="#Guideline_linksTo_OutputResult"/>
            <swrl:argument1>
              <swrl:Variable rdf:ID="z"/>
            </swrl:argument1>
            <swrl:argument2>
              <swrl:Variable rdf:ID="e"/>
            </swrl:argument2>
          </swrl:IndividualPropertyAtom>
        </rdf:first>
      </swrl:AtomList>
    </swrl:head>
    <swrl:body>
      <swrl:AtomList>
        <rdf:rest>
          <swrl:AtomList>
            <rdf:first>
              <swrl:IndividualPropertyAtom>
                <swrl:argument2>
                  <swrl:Variable rdf:ID="d"/>
                </swrl:argument2>
```

Measurable: The correctness of the developed SWRL rules within the ontology were achieved through the implemented inference engine. SWRL rules were executed and the corresponding properties were filled appropriately.

Attainable: The Rule Engine for the Java Platform, namely Jess, was used, as a Protégé plug-in for the successful execution of the rules.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is met throughout the different versions of the ACCESSIBLE generic and domain ontologies.

***G-REQ3-3**: The ontology shall support SPARQL queries that consist of triplets, in order to narrow the information space of accessibility assessment according to specified semantics of usage scenarios.*

Specific Description: The ACCESSIBLE ontology is queryable through SPARQL, since it has been defined with Semantic Web technologies.

An example SPARQL query to retrieve all impairments specified in the Generic Ontology is presented below:

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>
SELECT ?impairment
WHERE {
    ?impairment rdf:type acc:Impairment
}
```

Measurable: The execution of the developed SPARQL rules has been achieved through the integrated inference engine into the assessment tools.

Attainable: The import of the domain ontologies into the generic ontology is needed for the successful execution of the SPARQL queries.

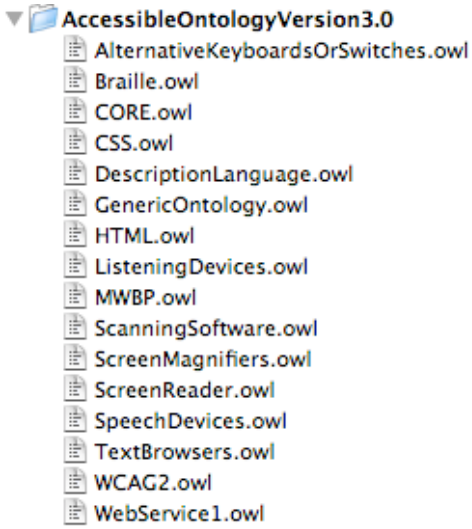
Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is met throughout the different versions of the ACCESSIBLE ontologies.

***G-REQ3-4**: In order to preserve the integrity and the maintenance of the ontology, the ACCESSIBLE ontology based knowledge resource shall include generic and domain-specific ontologies.*

Specific Description: The ACCESSIBLE ontology has been separated into several sub-ontologies, the Generic Ontology, and different Domain-Specific Ontologies for each technological domain covered in the ACCESSIBLE project (Web, Mobile Web, Web services, and SDL, mobile devices, etc.).

Below we present the ontology breakdown designed for the ACCESSIBLE Ontologies:



Measurable: The domain-specific ontologies were imported using Protégé and it was tested that all the classes and the properties are fully functional.

Attainable: Protégé already supports the development of a generic ontology and specific-domain ontologies. Thus, no constraints exist for this requirement.

Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is met throughout the different versions of the ontology, from the first one, namely version 2.0 since the latest, namely version 5.1.

Furthermore, the inclusion of the Domain-Specific Ontologies within the Generic Ontology is made through the inclusion of namespaces and associated Semantic Web predicates.

***G-REQ3-5**: The ACCESSIBLE ontology shall contain a generic version that describes the general structure of accessibility attributes of guidelines, users, devices, and applications.*

Specific Description: The Generic Ontology provides structural elements to describe accessibility assessment domains, as presented next, in an excerpt from the ACCESSIBLE Ontology:

```
<owl:Class rdf:ID="Checkpoint"/>
  <owl:Class rdf:ID="Capability"/>
  <owl:Class rdf:ID="OutputResult"/>
  <owl:Class rdf:ID="Test"/>
  <owl:Class rdf:ID="Device"/>
  <owl:Class rdf:ID="Standard"/>
  <owl:Class rdf:ID="Approach"/>
  <owl:Class rdf:ID="User"/>
  <owl:Class rdf:ID="Guideline"/>
  <owl:Class rdf:ID="Impairment"/>
  <owl:Class rdf:ID="FunctionalLimitation"/>
  <owl:Class rdf:ID="Disability"/>
  <owl:Class rdf:ID="Technique"/>
  <owl:Class rdf:ID="Application"/>
  <owl:ObjectProperty rdf:ID="User_linksTo_Approach">
    <owl:inverseOf>
      <owl:ObjectProperty rdf:ID="Approach_linksTo_User"/>
    </owl:inverseOf>
    <rdfs:range rdf:resource="#Approach"/>
    <rdfs:domain rdf:resource="#User"/>
  </owl:ObjectProperty>
```

Measurable: The most widely used guidelines, devices and applications were collected and they were compared with the generic ontology's ability to describe them.

Attainable:Mapping of domain ontologies into ACCESSIBLE generic ontology.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is met throughout the different versions of the ACCESSIBLE ontologies.

***G-REQ3-6**: The domain ontologies shall include different characteristics and instances of accessibility standards, disabilities, functional limitations, devices and application domains.*

Specific Description: Each Domain-Specific Ontology describes standards and/or guidelines for its corresponding technological domain (e.g., WCAG 2.0 for Web technologies), and maps these standards and guidelines into functional limitations or disabilities. These mappings reflect the work devised in Work Package 3, i.e., the ACCESSIBLE Harmonised Methodology.

As an example, this factor was achieved within the WCAG 2.0 ontology by specializing the generic terminology into WCAG 2.0 specific terms, thus bridging together both terminologies successfully:

```
<owl:Class rdf:ID="SuccessCriterion">
    <rdfs:subClassOf
rdf:resource="http://www.AccessibleOntology.com/GenericOntology.owl#C
heckpoint"/>
  </owl:Class>
  <owl:Class rdf:ID="Technique">
    <rdfs:subClassOf
rdf:resource="http://www.AccessibleOntology.com/GenericOntology.owl#T
echnique"/>
  </owl:Class>
  <owl:Class rdf:ID="Guideline">
    <rdfs:subClassOf
rdf:resource="http://www.AccessibleOntology.com/GenericOntology.owl#G
uideline"/>
  </owl:Class>
  <owl:Class rdf:ID="Test">
    <rdfs:subClassOf
rdf:resource="http://www.AccessibleOntology.com/GenericOntology.owl#T
est"/>
  </owl:Class>
  <owl:Class rdf:ID="Approach">
    <rdfs:subClassOf
rdf:resource="http://www.AccessibleOntology.com/GenericOntology.owl#A
pproach"/>
  </owl:Class>
```

Measurable: The most widely used standards, disabilities, functional limitations, devices and application domains were collected and they were compared with the generic ontology's ability to describe them.

Attainable: Due to the fact a great variety of resources (e.g. guidelines, devices and applications) exist, it was decided to focus on the most widely used accessibility standards, devices and applications.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is met throughout the different versions of the developed ACCESSIBLE ontologies.

Here we can perceive the terminology bridge between both ontologies, e.g., by mapping the generic term SuccessCriterion into the WCAG-specific term Checkpoint.

Regarding the specificities of ontologies for each domain, this factor was also successfully achieved, by mapping terminology and evaluation guidelines' structures

into proper Semantic Web concepts, such as Classes and corresponding instances.

One example detailed below concerns the Checkpoint STYLE_SHEETS_SUPPORT from the W3C Mobile Web Best Practices 1.0, and corresponding technique:

```
<j.0:Checkpoint rdf:ID="Checkpoint_48">
    <Checkpoint_has_Technique>
      <j.0:Technique rdf:ID="Technique_48">
        <hasId rdf:datatype="xs#int">48</hasId>
        <hasName
rdf:datatype="xs#string">STYLE_SHEETS_SUPPORT</hasName>
        <Technique_belongsTo_Checkpoint
rdf:resource="#Checkpoint_48"/>
      </j.0:Technique>
    </Checkpoint_has_Technique>
    <hasDescription rdf:datatype="xs#string">Organize documents so
that if necessary they may be read without style
sheets</hasDescription>
    <hasName rdf:datatype="xs#string">Style sheets support</hasName>
    <hasId rdf:datatype="xs#int">48</hasId>
```

Regarding functional limitations, the ontologies properly define these concepts according to the ACCESSIBLE Harmonised Methodology (which, in turn, is based on ICF), and map these into the different accessibility guidelines. Below we present an example for WCAG 2.0 and how the Success Criterion 1.2.3 maps into the Conductive Hearing Loss disability:

```
<FunctionalLimitation_belongsTo_Disability>
      <Disability rdf:ID="Conductive_Hearing_Loss">
        <Disability_belongsTo_Checkpoint>
          <rdf:Description
rdf:about="http://www.AccessibleOntology.com/WCAG2.owl#SuccessCriteri
on_1.2.3">
            <Checkpoint_has_Disability
rdf:resource="#Deaf_Blindness"/>
          </rdf:Description>
        </Disability_belongsTo_Checkpoint>
        <Disability_belongsTo_Checkpoint>
          <rdf:Description
rdf:about="http://www.AccessibleOntology.com/WCAG2.owl#SuccessCriteri
on_1.2.4">
            <Checkpoint_has_Disability
rdf:resource="#Deaf_Blindness"/>
          </rdf:Description>
```

```
        </Disability_belongsTo_Checkpoint>
        <Disability_belongsTo_Checkpoint>
          <rdf:Description
rdf:about="http://www.AccessibleOntology.com/WCAG2.owl#SuccessCriteri
on_1.2.2">
            <Checkpoint_has_Disability
rdf:resource="#Deaf_Blindness"/>
          </rdf:Description>
        </Disability_belongsTo_Checkpoint>
        <Disability_has_FunctionalLimitation>
          <FunctionalLimitation rdf:ID="Sound_Discrimination">
            <hasName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >Sound discrimination</hasName>
            <hasDescription
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
            >Sensory functions relating to sensing the presence of
sound involving the differentiation of ground and binaural synthesis,
separation and blending. </hasDescription>
            <FunctionalLimitation_belongsTo_Disability
rdf:resource="#Conductive_Hearing_Loss"/>
```

***G-REQ3-7**: All the domain ontologies shall comply with the generic ontology.*

Specific Description: Each Domain-Specific Ontology follows the OWL structures (e.g., Classes, Properties, etc.) defined in the Generic Ontology, by defining all technology-specific concepts as corresponding OWL instances or subclasses.

We present below a small subset of the Description Languages Ontology, where a specific checkpoint and corresponding guideline association follow the Generic Ontology structures, but specify the domain concepts as instances of the generic classes:

```
<j.0:Checkpoint rdf:ID="Checkpoint_17">
    <hasName rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >DL Checkpoint_17</hasName>
    <Checkpoint_belongsTo_Guideline>
      <j.0:Guideline rdf:ID="Guideline_17">
        <hasName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >DL Guideline 17</hasName>
        <Guideline_has_Checkpoint rdf:resource="#Checkpoint_17"/>
        <hasId rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
```

```
        >17</hasId>
      </j.0:Guideline>
    </Checkpoint_belongsTo_Guideline>
    <hasPriority rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
    >1</hasPriority>
    <hasDescription
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Usage of comboboxes instead of textboxes helping the user to
choose a value from old or proposed values must be used where
possible. </hasDescription>
```

Measurable: All the domain ontologies were imported using Protégé and it was checked that no deviations exist between the structure of domain ontologies and generic ontology, concerning the classes and the properties.

Attainable: Linkage of the structure of domain ontologies into the structure of the genric ontology.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is met throughout the different versions of the ontology, from the first one, namely version 2.0 since the latest, namely version 5.1.

***G-REQ3-8**: The ontology shall include widely used web content accessibility guidelines.*

Specific Description: The ACCESSIBLE ontology includes WCAG 2.0, the Mobile Web Best Practices (MWBP) provided by W3C, the defined guidelines for Web services and Description Languages.

Measurable: The most widely used web content accessibility guidelines were collected and they were compared with the guidelines that are described by the ACCESSIBLE ontology.

Attainable: To follow the structure of the WCAG 2.0 accessbility guidelines for the implementation of the ACCESSIBLE domain ontologies of the accessibility guidelines.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is met throughout the different versions of the ACCESSIBLE ontologies.

***G-REQ3-9**: The ontology shall include the most well known disabilities that people face and for which accessibility guidelines provide clues on how they should be supported in software applications.*

Specific Description: The ACCESSIBLE ontology specifies a thorough set of disabilities based on the ICF categorisation, as well as its mappings into accessibility guidelines, based on the Barrier Walkthrough method. These instances of the ACCESSIBLE ontology were devised according to the information specified in the ACCESSIBLE Harmonised Methodology.

A small sample of the specified disabilities within the ACCESSIBLE Ontology is presented below:

```
<Disability rdf:ID="Extreme_Light_Sensitivity"/>
<Disability rdf:ID="Rett_Syndrome"/>
<Disability rdf:ID="Absent_Limb_Or_Reduced_Limb_Function" />
```

Measurable: The most well known disabilities were collected, based on the International Classification of Functioning, Disability and Health (ICF), which is provided by the World Health Organization (WHO), and they were compared with the guidelines that are described by the ACCESSIBLE ontology.

Attainable: A great variety of disabilities exist, but ACCESSIBLE addresses only those that could be supported by the selected accessibility guidelines.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is met throughout the different versions of the ACCESSIBLE ontologies.

***G-REQ3-10**: The ontology shall include a variety of assistive devices, which are used from people with disabilities.*

Specific Description: The ACCESSIBLE ontology specifies a set of assistive devices, and their relationship to disabilities and functional limitations. These are particularly reflected in the ACCESSIBLE simulation tools. A small example of such specification within the ontology is presented below, where the specification of a Braille display is formalised in RDF/OWL:

```
<owl:Class rdf:ID="Braille">
    <rdfs:subClassOf
rdf:resource="http://www.AccessibleOntology.com/GenericOntology.owl#D
evice"/>
  </owl:Class>
```

```
  <Braille rdf:ID="Alva_BC640">
    <j.0:hasName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >Alva BC640</j.0:hasName>
    <j.0:hasDescription
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
    >The ALVA BC640 combines powerful features with a compact and
lightweight design. High quality Optelec Braille cells and easy-to-
operate keys allow for effortless reading and smart navigation. The
optional Braille Audio Feature Pack makes the ALVA BC640 uniquely
versatile. The ergonomically designed Braille input keys and
integrated high quality audio speakers makes you operate your ALVA
BC640 efficiently and comfortably. Upon request the ALVA BC640 can
even be equipped with internal memory, allowing you to store your
documents or host your preferred screenreader on-
board.</j.0:hasDescription>
  </Braille>
```

Measurable: The most well known devices were collected, based on ISO documents, as well as other AT web sites (e.g. ETNA) that are oriented in assistive devices and they were compared with the devices that are described by the ACCESSIBLE domain ontologies.

Attainable: A great variety of assistive devices exist, but ACCESSIBLE addresses the most widely used and those that are used by people who suffer from disabilities that are already described by the ACCESSIBLE ontology.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is met throughout the different versions of the ACCESSIBLE ontologies.

***G-REQ3-11**: The ontology shall incorporate different types of applications (such as web services, HTML) that can be verified for their accessibility status.*

Specific Description: The ACCESSIBLE ontology includes Domain-Specific Ontologies that cover different application domains, and their reflection into accessibility assessment guidelines, as detailed before.

Measurable: The most well known applications were collected, based on the pilot applications of the ACCESSIBLE project. These applications can be accessed through the ACCESSIBLE domain ontologies.

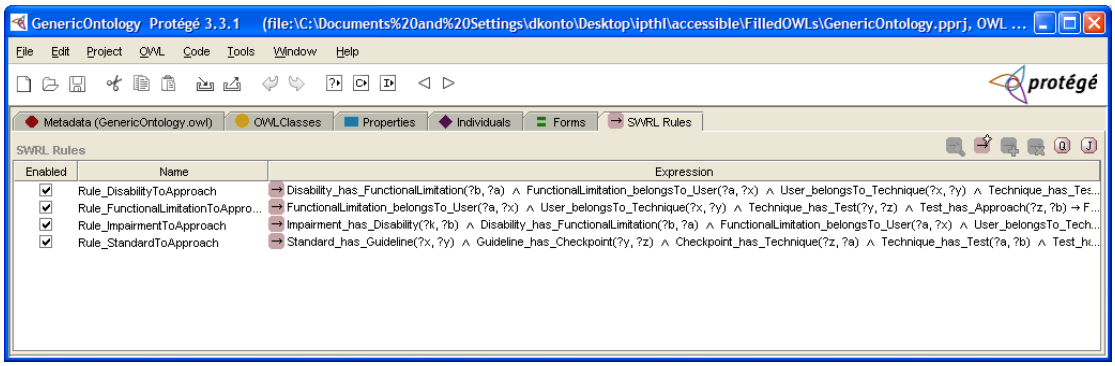
Attainable: A great variety of applications exist, but ACCESSIBLE addresses only those that can be supported by the accessibility guidelines

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is met throughout the different versions of the ontology, from the first one, namely version 2.0 since the latest, namely version 5.1.

***G-REQ3-12**: For the implementation of the ontologies, the Protégé ontology editor will be used as well as the SWRL supporting tool.*

Specific Description: As presented in Deliverable D4.1, A set of formalisms and taxonomies for accessibility assessment procedures and their inherent meta-models, the Protégé ontology editor was used to define all OWL ontological resources and associated languages. A screenshot of such edition process is presented below:



Measurable: The ontology was developed using the Protégé ontology editor and it was fully functional.

Attainable: Protégé is a free, open source ontology editor. Moreover, it is stable and widely used for the development of ontologies. Therefore, it was selected in order to develop the ontologies within the premises of ACCESSIBLE. The same stands for the SWRL tool. Thus, no constraints exist for this requirement.

Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is met throughout the different versions of the ontology, from the first one, namely version 2.0 since the latest, namely version 5.1.

***G-REQ3-13**: Adequate semantic description shall be provided from all the characteristics that are included in the generic as well as in the domain ontologies.*

Specific Description: All OWL classes and properties in the Generic Ontology, as well as in all Domain-Specific Ontologies, reflected the naming schemes recurrently used in accessibility guidelines (e.g., Guideline, Checkpoint, Technique, etc.) As detailed previously, the naming schemes are compliant with the expected terminology. Furthermore, all checkpoints and associated instances of accessibility guidelines provide a textual description of their goal, in order to be more readable by users that will interact directly or indirectly with the ontologies, as exemplified below:

```
<Guideline_has_Checkpoint>
      <j.0:Checkpoint rdf:ID="Checkpoint_I6.3">
        <hasId rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >34</hasId>
        <Checkpoint_belongsTo_Guideline
rdf:resource="#Guideline_I06"/>
        <hasPriority
rdf:datatype="http://www.w3.org/2001/XMLSchema#int"
        >1</hasPriority>
        <hasParent rdf:resource="#Checkpoint_I6"/>
        <hasDescription
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >The inputs and outputs of operations should have the minimum
possible level of abstraction </hasDescription>
        <hasName
rdf:datatype="http://www.w3.org/2001/XMLSchema#string"
        >WebService1 Checkpoint_I6.3</hasName>
      </j.0:Checkpoint>
    </Guideline_has_Checkpoint>
```

Measurable: A number of properties were added in order to describe all the characteristics that are included in the generic and the domain-specific ontologies.

Attainable: Fully attainable.

Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is met throughout the different versions of the ACCESSIBLE ontologies.

***G-REQ3-14**: The execution of queries and rules shall be supported through the integration of an open source inference engine that shall support the usage of SPARQL as well as SWRL.*

Specific Description: The ACCESSIBLE Inference Engine provides support to the integration of the ACCESSIBLE Ontology into all technology evaluators. Furthermore, the Jena and Pellet open source Semantic Web technology libraries provide the ground basis for the mentioned integrations. By using Jena and Pellet, support is provided to the SPARQL and SWRL languages, as well as core Semantic Web languages, such as RDF, RDF-S, and OWL.

More details about this integration has been specified in deliverable D4.2 (A software package containing a set of modelling tools, rules inference engine, and the rules graphical editor) and further explained in subsequent sections of this deliverable.

Measurable: The correctness of the developed SWRL and SPARQL rules were tested by executing the rules and checking that the corresponding properties were filled appropriately.

Attainable: Fully attainable.

Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is met throughout the different versions of the ACCESSIBLE assessment tools and their integration with the inference engine and the ACCESSIBLE ontologies.

2.2 Performance requirements

***P-REQ3-1**: Response times in communication of the ACCESSIBLE knowledge resource with the assessment system shall be in line with those specified in the communication protocol selected.*

Specific Description: The ACCESSIBLE Inference Engine is defined as both an embeddable library and SOAP-based Web service, in order to allow different integration scenarios into technology evaluators. The selection of its usage is made according to the limitations imposed by required response times.

Measurable: The latency on communication between the ACCESSIBLE ontology and accessibility assessment tools was measured and the results were satisfying.

Attainable: Fully attainable.

Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is already met.

***P-REQ3-2**: In the same way, the inference engine in the complex rules and queries execution shall also take into account the performance.*

Specific Description: The average loading time of all ontologies devised (the Generic Ontology and all Domain-Specific Ontologies) is around 2.5 seconds (more details on performance can be found in section 6).

Measurable: The loading time of the generic ontology as well as of all the domain-specific ontologies was measured and the results were satisfying.

Attainable: Fully attainable.

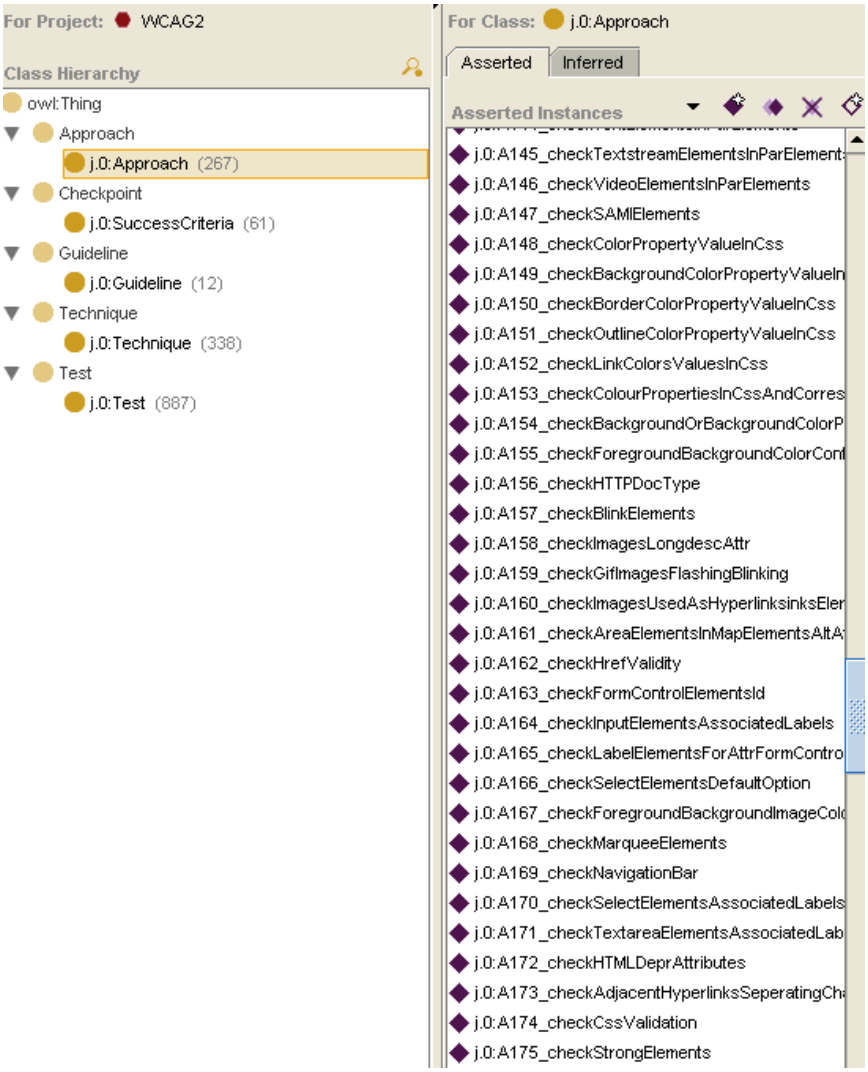
Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

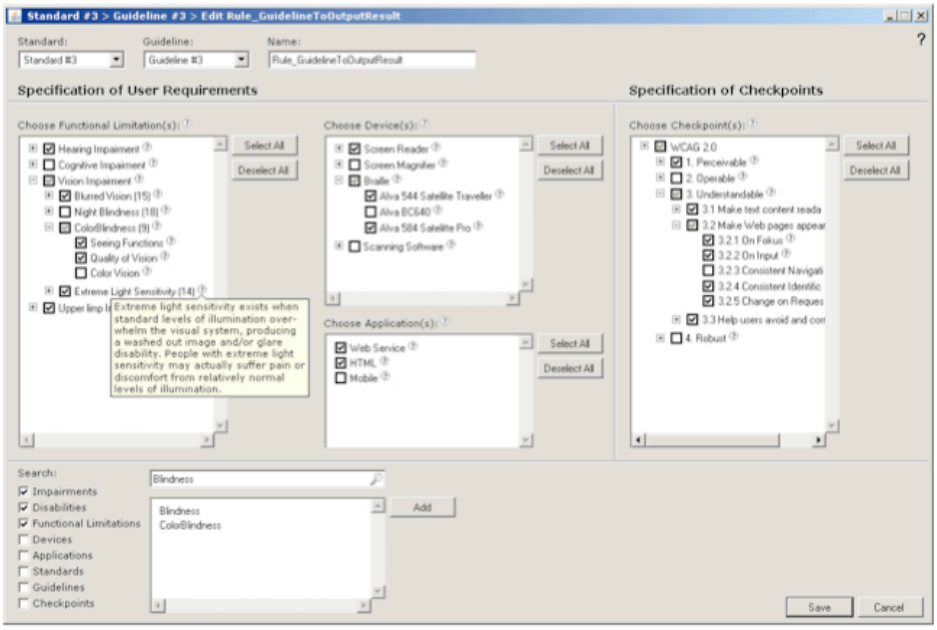
Time-bound: This requirement is already met.

2.3 Operational requirements

***O-REQ3-1**: The user shall be able to navigate through the ACCESSIBLE loaded ontology.*

Specific Description: The ACCESSIBLE ontology is navigable in different ways, including: Protégé, the ACCESSIBLE Rules User Interface, as well as technology-specific evaluators and the ACCESSIBLE Assessment Portal. Such examples are presented below:





Measurable: The ACCESSIBLE ontology was presented to different users and their capability to navigate throughout the ontology concepts was tested. Also users can access the ACCESSIBLE ontologies through the implemented assessment tools, the DIAS and the ACCESSIBLE portal.

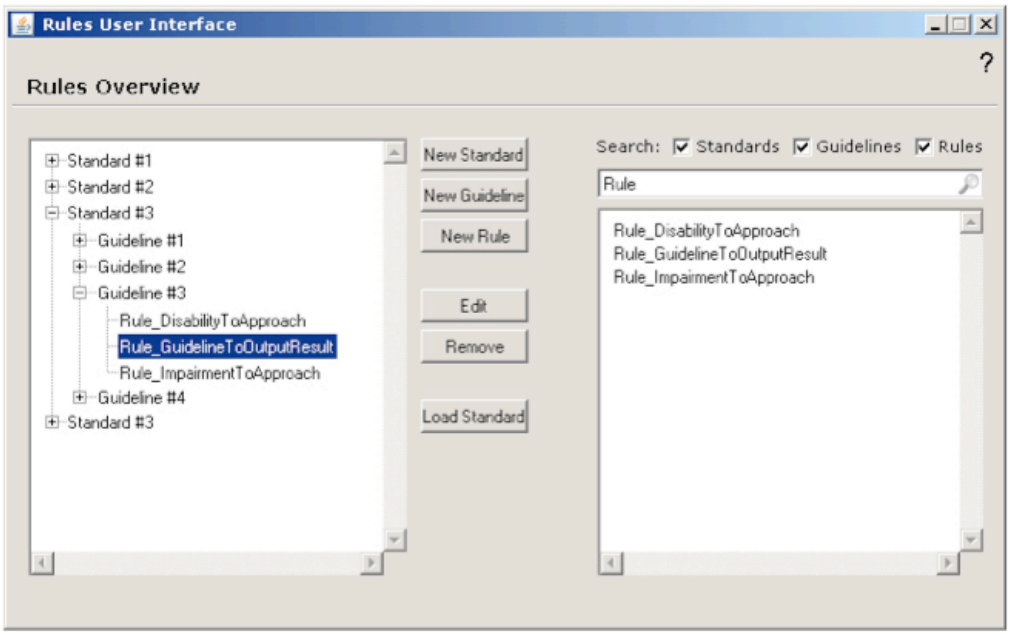
Attainable: The ACCESSIBLE ontology should be loaded using a tool that is able to support the structure of the ontology, namely the existence of the generic ontology and of domain-specific ontologies.

Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is already met through the GUI of the implemented tools.

***O-REQ3-2**: The user shall be able to search though the ontology in classes, properties and instances.*

Specific Description: The ACCESSIBLE ontology includes standardised terminology and associated meta-information that affords searchability. Below we present a screenshot of the ACCESSIBLE Rules User Interface depicting this situation:



Measurable: The ACCESSIBLE ontologies can be accessed by different users through the implemented rules Editor and the GUI of the implemented ACCESSIBLE tools.

Attainable: The ACCESSIBLE ontology should be loaded using a tool that is able to support the structure of the ontology, namely the existence of the generic ontology and of domain-specific ontologies.

Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is already met.

***O-REQ3-3**: When displaying ontologies to the user via GUI, elements must be ordered according to the part of the ontology being displayed.*

Specific Description: The ACCESSIBLE ontology is presented in technology-specific evaluators and related GUIs, according to the associated accessibility assessment contexts.

Measurable: The ACCESSIBLE ontology was presented via GUI and the ordering was compared with the associated accessibility assessment contexts.

Attainable: Fully attainable.

Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is already met.

***O-REQ3-4**: Developers or other users that have access to the ontology shall be familiar with the basic ideas of OWL language.*

Specific Description: The ACCESSIBLE ontology extensibility requires a minimum level of conceptualisations of OWL languages, which is leveraged through the ACCESSIBLE Rules User Interface.

Measurable: The ACCESSIBLE ontology was presented to users and their capability to understand the basic concepts of the ontology was tested.

Attainable: Due to the fact that the ACCESSIBLE ontology was presented using the ACCESSIBLE Rules User Interface, no constraints exist for this requirement.

Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is already met by the ACCESSIBLE Rules User Interface

***O-REQ3-5**: Developers or other users that wish to add new terms (such as a new accessibility guideline) to the ontology shall use the independent domain ontologies and comply with the generic ontology attributes.*

Specific Description: The ACCESSIBLE ontology provides normalised terminology that can be leveraged to implement technology-specific ontologies. The application of the Generic Ontology into the four Domain-Specific Ontologies implemented in the ACCESSIBLE ontology provides further support for this assertion.

Measurable: When a new domain-specific ontology is added to the ACCESSIBLE ontology, it should comply with the structure of the generic ontology.

Attainable: The constraints for this requirement refer to the compliance of the new domain-specific ontologies to the structure (concerning classes and properties) of the generic ontology.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is already achieved, due to the fact that different domain-specific ontologies were developed, but it will be also achieved when an external user will add a new domain-ontology to the ACCESSIBLE ontology.

2.4 Reliability requirements

***R-REQ3-1**: The ontology-based resource shall provide answers just for the knowledge embedded in ontologies (both general and domain-specific). All knowledge not represented will be regarded as unknown, not as false (i.e. open-world assumption).*

Specific Description: The Semantic Web technologies used in the definition of the ACCESSIBLE ontology (e.g., RDF, RDF-S, OWL) provide support for this assertion.

Measurable: The performance of the assessment tool that are using the ACCESSIBLE ontology was satisfying and they were able to provide to the users all the knowledge that is included in the ontology.

Attainable: For the knowledge included in the ontology no constraints exist.

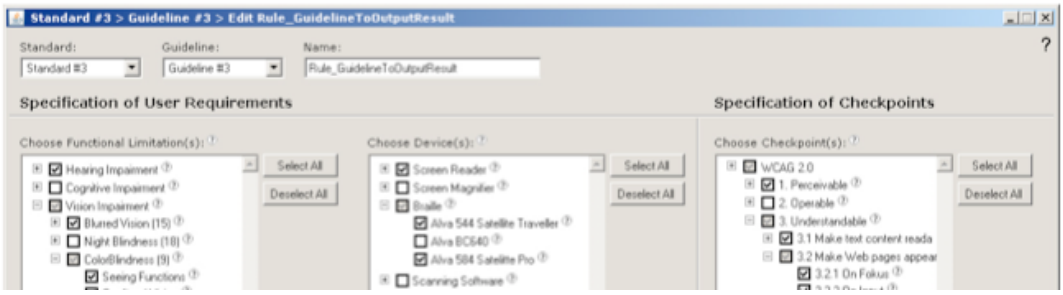
Relevant: Within the availability of resources and knowledge, this requirement was easily achieved.

Time-bound: This requirement is already met by the latest version of ontology, namely version 5.1

2.5 Maintainability & Interoperability requirements

***M-REQ3-1**: Developers shall be able to specify new SWRL rules and/or SPARQL queries through a special purpose GUI.*

Specific Description: The ACCESSIBLE Rules User Interface implements these features. Below we present a small example of the GUI and its application into the definition of a new rule.



Measurable: The ACCESSIBLE Rules User Interface was presented to users and their capability to express new SWRL rules and SPARQL queries within the ACCESSIBLE ontology was evaluated.

Attainable: The new SWRL rules as well as the SPARQL queries should be valid and comply with the structure of the ACCESSIBLE ontology.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is already met by the ACCESSIBLE Rules User Interface.

***M-REQ3-2**: All ontologies should be developed in a way that knowledge can be interchanged between different Operating Systems and different knowledge management systems. For this, the ontological resource will be developed with standardised and open technologies and languages, including RDF, OWL, SWRL, and SPARQL.*

Specific Description: All technologies used in the ACCESSIBLE ontology are runtime-agnostic (i.e., operating system, KMSs, etc.)

Measurable: The ACCESSIBLE ontology was tested in different OS/KMS settings.

Attainable: Due to the structure of the ACCESSIBLE ontology, it should be loaded using a tool that is able to support the structure of the ontology, namely the existence of the generic ontology and of domain-specific ontologies.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is already met.

***M-REQ3-3**: The knowledge inference engine must be interoperable and cope with open technologies. For this, it will be developed using open source knowledge software components developed in portable languages (e.g. Java).*

Specific Description: All the libraries and software developed in the context of Work Package 4 are open-source and implemented using open-source and cross-platform languages and technologies. Java, Groovy, and JVM provide support for the implementation, and Jena/Pellet provide open-source inference mechanisms for Semantic Web technologies and languages.

Measurable: The knowledge inference engine was integrated in different tools and no requirement of closed-source software libraries was required.

Attainable: Fully attainable.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is already met.

M-REQ3-4: *In this same way, the ACCESSIBLE knowledge resource and the developed inference engine as well as the rules editor shall be enough documented in order to be understandable for users external to the development.*

Specific Description: This deliverable, D4.3, as well as D4.2, provide documentation to support the extension and integration of the ACCESSIBLE Knowledge Resources into external software and accessibility evaluation processes.

Measurable: The documentation was adequate in order to describe the inference engine and the rules editor.

Attainable: Fully attainable.

Relevant: Within the availability of resources and knowledge, this requirement was achieved.

Time-bound: This requirement is already met.

3 OntologyQuery Server

This Section presents the details of the OntologyQuery Server component. Its goal lays on providing a web service SOAP compliant server for querying a chosen OWL2-DL compliant ontology (in our case i.e. the ACCESSIBLE Generic Ontology or single .owl ontologies).

3.1 Introduction and Context

The ACCESSIBLE Ontological Knowledge – according to the D4.1 deliverable - "acts as a registry of knowledge for all the other components of the ACCESSIBLE Project. All ACCESSIBLE components access the knowledge base by interacting with the ontological knowledge resource through available interfaces." Ontology in both ACCESSIBLE components designed and that are being designed is the essential core and interacts through external libraries/components or ACCESSIBLE built-in functionalities. All these ways of interacting have in common the SPARQL query language to access the accessible knowledge. A big amount of SPARQL queries are used and will be used to accomplish the goal expected then.

For the reasons explained above, a simple and straightforward tool for querying the ontology through SPARQL is a boost to the development of new queries. Every new SPARQL query can be checked and debugged, concentrating the attention on whether the given output is what we expect.

3.2 Requirements

Here are explained the main functional and non-functional requirements that have lead to the development of this component.

3.2.1 Functional Requirements

Function: Start the server

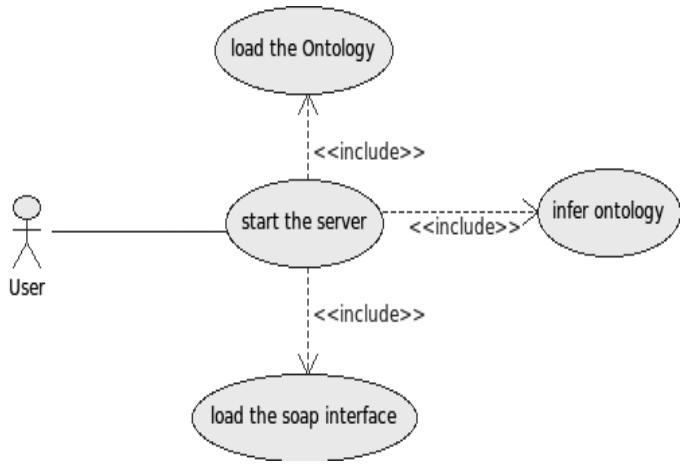


Figure 1. **Starting the OntologyQuery Server**

Starting conditions: This functionality occurs when the server component is loaded by another component (i.e. a general component that load both client and server part)

Normal flow of the functionality:

The server tries to load and infer the ontology indicated, using the alternative local file indicated into the locationmapper file. [E1]

Finally, it tries to set up the SOAP interface [E2]

Exceptional flow:

E1. In case some file needed is missing or it's impossible to continue loading the ontology, the component stops loading and generates an exception.

E2. In case some error occurs when it tries to set the soap interface up, the component stops working and generates an exception.

Function: Query the ontology loaded

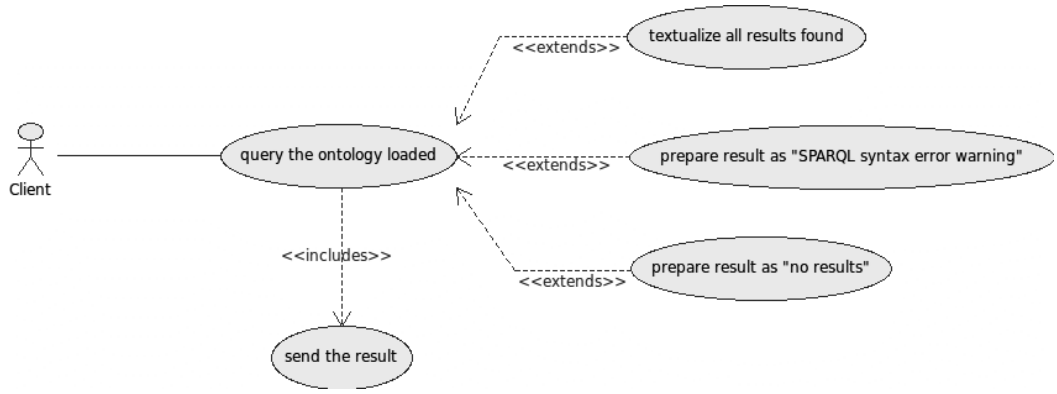


Figure 2. **Querying the Ontology**

Starting conditions: This functionality occurs when the QueryOntology client or a generic SOAP client sends a query to the server address using an operation provided by the server.

Normal flow of the functionality:

A SOAP Message with the request is sent by the client

Query the SPARQL query on the ontology loaded when the server was opened. [E1][E2]

Parses all the results and prepares a textual row for each result [E3]

Send the result through SOAP protocol to the client.

Exceptional flow:

E1. In case the operation doesn't exist, a SOAP error message will be sent to the client

E2. In case the SPARQL query has some error, a SOAP warning message will be sent to the client as result.

E3. If the SPARQL query has zero result, a SOAP message telling there are zero result will be sent.

3.2.2 Non-functional requirements

Besides the main function requirements presented in the previous Section, we have defined a set of non-functional requirements that must be taken into account in the architecture and implementation decisions for the OntologyQuery Server, as follows:

NF1	This component must be as portable as possible
NF2	The component must be able to load every OWL2-DL ontology
NF3	The component must be able to make inference through the Pellet reasoner
NF4	The SPARQL queries must be done on the inferred ontology
NF5	It must be possible to specify as a parameter a path for a location-mapper file used to load alternative local files, in case the specified ontology whether imports or refers to other files from the Web that are impossible to fetch (i.e., no connections or services unavailable)
NF6	It's mandatory to specify as a parameter the complete URI that will be used from this component to create an access point to the services.

3.3 OntologyQuery Server Component

The component's architecture was devised according to the constraints and requirements presented in the previous Sections. This architecture is presented below:

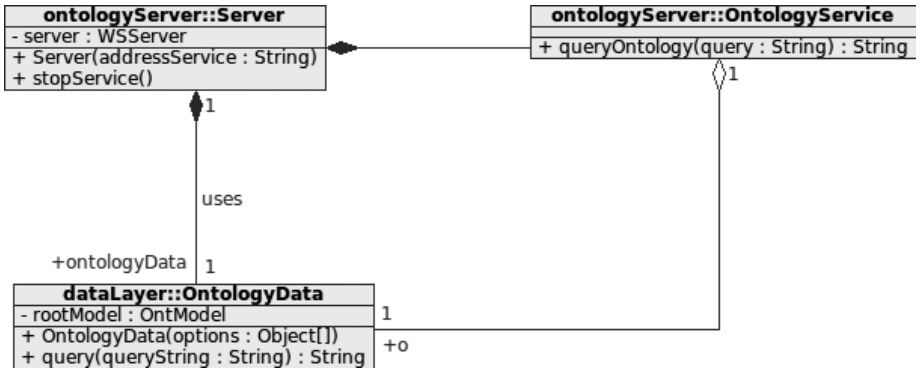


Figure 3. **OntologyQuery Server Architecture**

The components have been divided into two packages: a business layer named ontologyServer and a dataLayer. Further detailing these packages, this component comprises the following entities:

- **dataLayer::ontologyData:** it manages the actions related to the ACCESSIBLE Ontology. This class enables loading, inferring and querying the different aspects of the ACCESSIBLE Ontology.

- **OntologyServer::OntologyService:** this Bean-like entity is used for creating a SOAP service through GroovyWS library (c.f. the implementation Section below). Every public method declared creates a new service available.

- **OntologyServer:** thanks to the WSServer type property and the OntologyService type class, it offers a SOAP service for querying the ontology.

3.4 Implementation

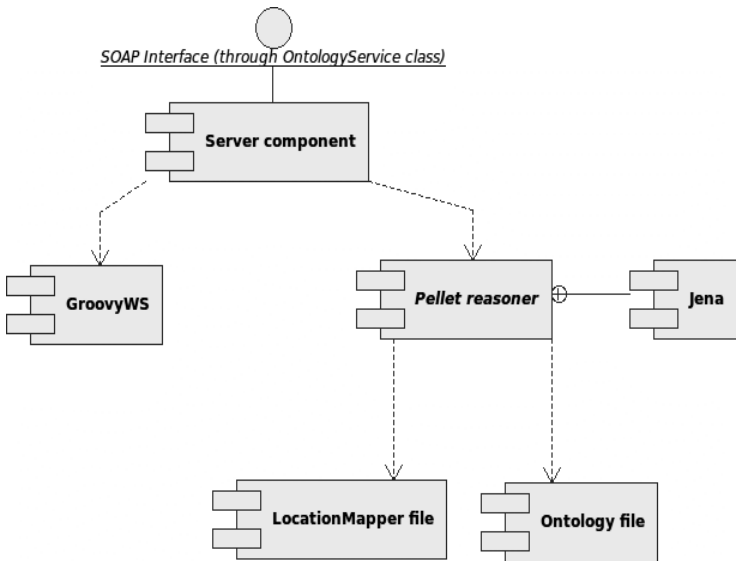


Figure 4. **OntologyQuery Server Implementation**

The Server component is the core of the program. It has to load and manage the ontology, infers it, performs SPARQL queries and provides a SOAP interface. To accomplish these tasks are being used these external components and libraries:

- **Groovy programming language:** Groovy is an open-source, agile programming language (akin to Ruby, Python, etc.) that sits on top of the JVM, providing a set of high abstractions and concepts – both at the language and libraries levels – to facilitate readability and flexibility of implementing software components;

- **JVM (Java Virtual Machine) technologies and libraries:** the foundation of the implementation of the Rules Inference Engine is the JVM and related

libraries. We opted for this technology due to its pervasiveness (available in all relevant execution platforms) and openness (licensed as open-source). Furthermore, the software libraries that are made available right from the start by the JVM provide a lower entry barrier in the development of software components;

- **Pellet:** Pellet3 is an open-source, state-of-the-art OWL 2 reasoner, made available as a Java library;

- **GroovyWS:** All the Web services have been realized with GroovyWS4, an external library of the Groovy ecosystem, which provides a simple way to implement and use Web Services.

4 OntologyQuery Client

This Section presents the details of the OntologyQuery Client component. Its goal lays on providing a program that can query the ACCESSIBLE ontology through the services offered from the server component previously documented.

4.1 Introduction and Context

This component creates a simple-to-use and straightforward graphical interface to use the server component. It realizes the missing presentation layer in the previous component, by interacting with the web service realized. This division has been made to modularize and make possible future uses by third parts.

4.2 Requirements

Here are explained the main functional and non-functional requirements that have lead to the development of this component.

4.2.1Functional Requirements

Function: Query the SPARQL string

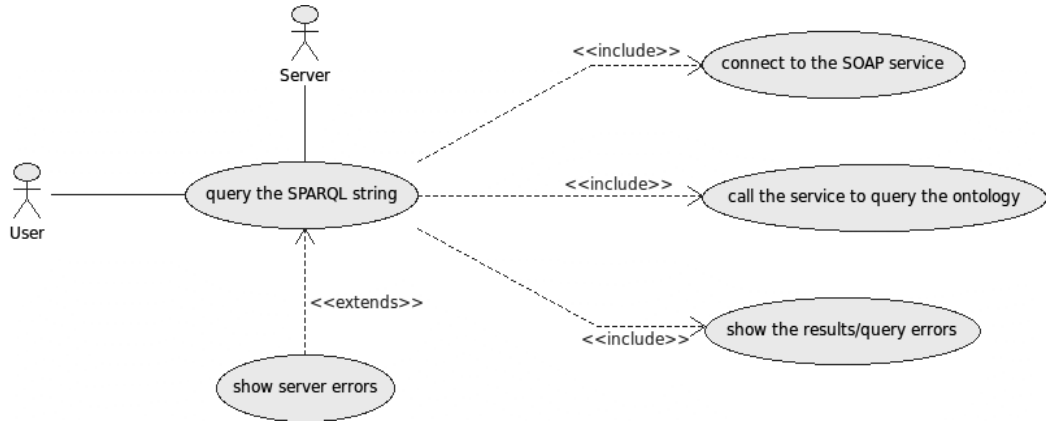


Figure 5. **OntologyQuery Client Use Case**

Starting conditions: This functionality occurs when the user click on “query”.
Normal flow of the functionality:
The client tries to create a connection with the server, by using the URI address specified at the loading of the program as a parameter [E1]
The client sends the query request with the string written by the user using the graphical interface;
The result is showed in the text field;
Exceptional flow:
E1. In case the URI is unavailable or the server doesn't permit the operation required, a popup error message appears.

4.3 OntologyQuery Client Component

The component's architecture was devised according to the constraints and requirements presented in the previous Sections. This architecture is presented below:

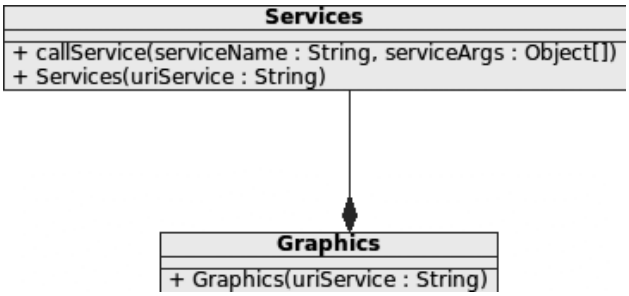


Figure 6. **OntologyQuery Client Architecture**

- **Services:** it is the entity designed to connect the graphical interface to the Web service offered by the server component.
- **Graphics:** it is the entity designed to build the graphic interface.

4.4 Implementation

The Client component can be seen as the Presentation Layer of this tool. It has to interact graphically with the end users and the business layer performed by the server component, infers it, performs SPARQL queries and provides a SOAP interface. To accomplish these tasks are being used these external components and libraries:

- **Groovy programming language:** Groovy is an open-source, agile programming language (akin to Ruby, Python, etc.) that sits on top of the JVM, providing a set of high abstractions and concepts – both at the language and libraries levels – to facilitate readability and flexibility of implementing software components;

- **JVM (Java Virtual Machine) technologies and libraries:** the foundation of the implementation of the Rules Inference Engine is the JVM and related libraries. We opted for this technology due to its pervasiveness (available in all relevant execution platforms) and openness (licensed as open-source). Furthermore, the software libraries that are made available right from the start by the JVM provide a lower entry barrier in the development of software components.

5 OntologyQuery General

This small component aims at loading the two main components as if they were bound together. This allows the quick and simple use of the client and server features, especially for testing and integration purposes.

5.1 Requirements

Here are explained the main functional and non-functional requirements that have lead to the development of this component.

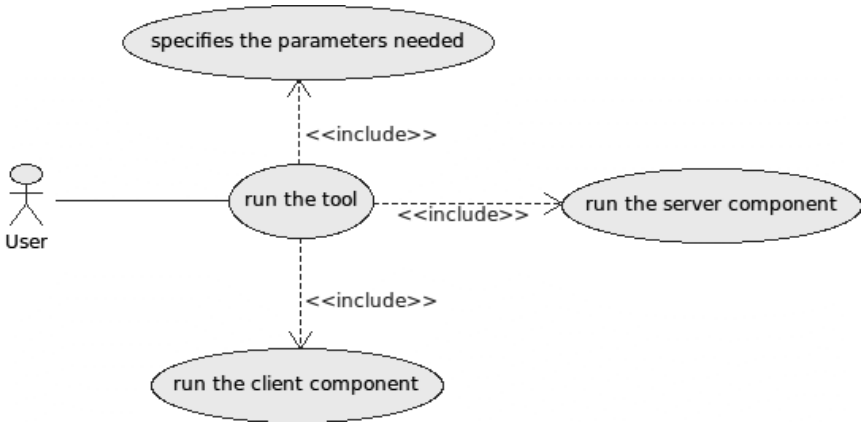


Figure 7. **OntologyQuery Client Implementation**

Starting conditions: This functionality occurs when the user tries to open the tool.

Normal flow of the functionality:

1. The user opens the program through the shell script execute.sh that will open the tool with the default parameters. Otherwise the user can run the tool with personalized parameters by typing

   java -jar "ontologyQuery.jar" <server-address> <path-ontology> <path-location-mapper-file>

   where the server-address will be the complete URI that will be used to create an access point to the services (e.g., http://localhost:6980/ontologyService), the path-ontology will be the path used to load the ontology, and the path-location-mapper-file will be the path used to load alternative local files, in case the specified ontology <path-ontology> whether imports or refers to other files from the web that are impossible to fetch (no connections or services unavailable) [E1];

2. The program will execute the server and client component individually by passing the parameters needed.

Exceptional flow:

E1. In case some parameter is missing, a textual warning message will be displayed and the program will be terminated.

5.2 General Component

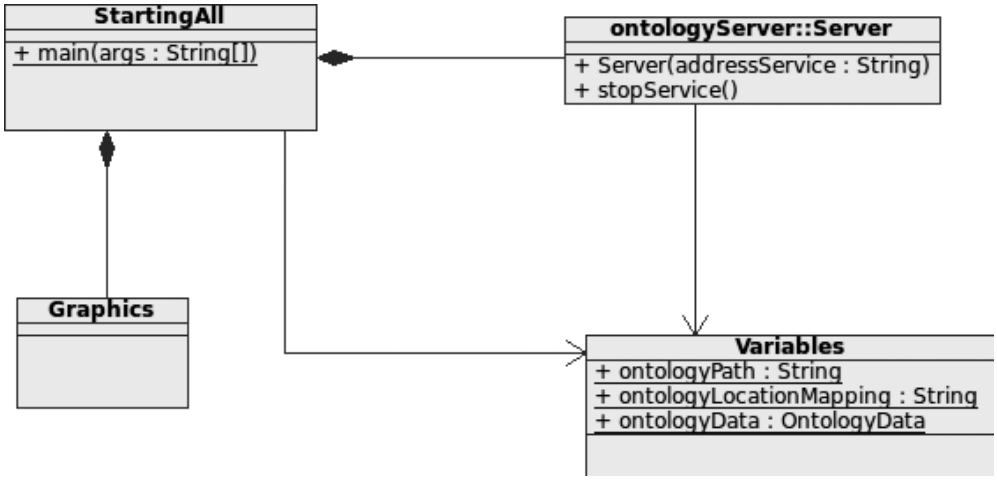


Figure 8. **OntologyQuery General Implementation**

- **Variables:** entity used for providing information to all the components of the tool.
- **StartingAll:** entity used to manage the execution of the two components and to set the necessary environment variables. This is the delegated class to run the tool thanks to its public static void main method.

6 Testing and validation

This part intends to give a guideline for evaluation and validation of methods and tools that have been implemented in WP4 (the Rules inference engine, the rules editor, ACCESSIBLE ontologies). It particularly aims to define measurement goals and for each of them, an associated set of tests and criteria. The purpose of this part is to provide a starting point for the evaluation of the ACCESSIBLE tools across the Pilot phases (WP6, WP7) where the implemented ACCESSIBLE tools (that are integrate the WP4 tools) will be examined and validated.
In fact, the next step, detailed in a future deliverable of the Workpackage 6 (D6.1b Pilot plans) will develop the evaluation procedure for the pilot applications and will determine the optimal evaluation methodology to properly carry out the system's performance test of the ACCESSIBLE assessment and simulation tools.
Thus, our main scope here is to test and validate the envisaged WP4 tools while deliverable D6.1b will define measurement goals and criteria for the evaluation of the ACCESSIBLE assessment and simulation tools.

6.1 Measurement goals and Criteria

Considering the WP4 ACCESSIBLE objective(s) and goals, as described in the work plan of WP4, we can then define a set of measurement goals (focused on performance and usability issues), each one containing a set of criteria used for measurement.
During Pilots we aim to measure:

- WP4 Tool Performance
- Usability of WP4 Tools and acceptance by users

For each of these measurement goals, we provide a table containing a corresponding set of criteria. Considering [Evans 87], [IEEE90], [Nyberg et.al.94] and [Lompré01], we also present a definition for every criterion.

6.1.1 WP4 Tools' Performance

6.1.1.0 System (software) Performance and Quality

Criteria	Definition
Interoperability	Examining interoperability issues between WP4 tools and other WP5 implementations
Reliability	The ability of a system or component to perform its required functions under stated conditions for a specified period of time
Testability	The degree to which a system or component facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met
Functional scope	The range or scope to which a system component is capable of being applied
Efficiency	The degree to which a system or component performs its designated functions with minimum consumption of resources.

Criteria	Definition
Robustness	The degree to which a system or component can function correctly in the presence of invalid inputs or stressful environment conditions

6.1.1.1 Interface Quality

Criteria	Definition
Homogeneity	Stability of the design choices (presentation of the information, the same procedure leads to the same results)
User-friendly	The degree to which a system or component contains enough information to explain its objectives and properties
Explicit control	The means allowing the user to control the effects of orders (to identify the active options in the menu, to be able to stop an impression)
Error management (handling)	The function that identifies and responds to user errors to maintain normal or to correct its operations
Conciseness	The degree to which the interface has no excessive information present. Reduction of the activities of perception and memorisation (icons, default options)

6.1.2 Usability of the WP4 Tools

6.1.2.0 Semantic aspects of querying

Criteria	Definition
Simplicity	The degree to which a system or component has a design and implementation that is straightforward and easy to understand
Efficiency	The degree to which a system or component performs its designated search functions with minimum consumption of resources

6.1.2.1 Results of interaction

Criteria	Definition
Significance	The quality of the results according to the user request.
Understandability	The degree to which the presentation of the results is clear to the user
Latency	The length of time it takes to respond to an event

6.2 Performance evaluation of ACCESSIBLE ontologies

The ACCESSIBLE generic and domain ontologies were designed with the open source ontology editor Protégé. In the initial implementation of the system we had not many real data available, so we could only test the basic structure of the ontologies. For that reason basic SWRL tests have been performed by running methods for creating Jena model in RDB and initialising knowledge from OWL files for detecting conflicts in the ontologies.

After the finalization of the ACCESSIBLE ontologies the usage of Logical and rule-based approaches to ontology validation and quality evaluation has been performed though the implemented inference engine.

As it was presented in previous section the average loading time of all ontologies devised (the Generic Ontology and all Domain-Specific Ontologies) is around 2.5 seconds (more details on performance can be found in section 6). We present below the timings (in nanoseconds) for loading each ontology (domain as well as the generic ontology), showing 10 loading attempts and corresponding average times:

GenericOntology.owl

Loading Attempt	Time (nanoseconds)
1	3915200283
2	1468300961
3	427083843
4	355854921
5	326917680
6	309591132
7	309558353
8	307069046
9	306600343
10	320505899
Average	804668246

ScreenMagnifiers.owl

Loading Attempt	Time (nanoseconds)
1	48579793
2	37021491
3	19861643
4	33945027
5	23667397
6	28766675
7	19085709
8	14782317
9	28655402
10	9874991
Average	26424045

DescriptionLanguage.owl

Loading Attempt	Time (nanoseconds)
1	799082597

2	357082555
3	220911797
4	310817936
5	144726907
6	113738619
7	197657327
8	83962668
9	69255347
10	73824047
Average	237105980

ScreenReader.owl

Loading Attempt	Time (nanoseconds)
1	39400967
2	14095323
3	17253040
4	17961108
5	12426096
6	21067709
7	26604650
8	10318821
9	10236815
10	10214364
Average	17957889

HTML.owl

Loading Attempt	Time (nanoseconds)
1	113884381
2	153794780
3	58265469
4	120397320
5	49297494
6	84575086
7	57722596
8	35101296
9	28820612
10	46555585
Average	74841462

SpeechDevices.owl

Loading Attempt	Time (nanoseconds)
1	36251357
2	30667282
3	30628313
4	29097952
5	23184601
6	21430202
7	10803512
8	11094028
9	10254899
10	17767296
Average	22117944

DescriptionLanguage.owl

Loading Attempt	Time (nanoseconds)
1	799082597
2	357082555
3	220911797
4	310817936
5	144726907
6	113738619
7	197657327
8	83962668
9	69255347
10	73824047
Average	237105980

TextBrowsers.owl

Loading Attempt	Time (nanoseconds)
1	95392025
2	31523274
3	11193822
4	23333898
5	17582455
6	29371559
7	23008788
8	13892517
9	15517752
10	8224217
Average	26904031

ListeningDevices.owl

Loading Attempt	Time (nanoseconds)
1	33192042
2	17932715
3	23292745
4	18480933
5	9870311
6	9307585
7	18173742
8	8963847
9	8298268
10	11822250
Average	15933444

WCAG2.owl

Loading Attempt	Time (nanoseconds)
1	1652583795
2	1372793924
3	411352289
4	404863320
5	413322760
6	336031265
7	292876576
8	287841391

9	291531739
10	284716903
Average	574791396

MWBP.owl

Loading Attempt	Time (nanoseconds)
1	457519084
2	193839505
3	257726457
4	102464581
5	111618873
6	117309487
7	163601832
8	139156146
9	131697985
10	80564911
Average	175549886

WebService1.owl

Loading Attempt	Time (nanoseconds)
1	866943518
2	727619047
3	296845505
4	334775115
5	257375931
6	145288971
7	122994905
8	130304228
9	132303315
10	180124837
Average	319457537

ScanningSoftware.owl

Loading Attempt	Time (nanoseconds)
1	32057065
2	11372929
3	15563411
4	10639942
5	18426601
6	11435835
7	8509125
8	8376951
9	15478975
10	8619982
Average	14048082

6.3 Performance Evaluation of Inference Engine on Context Information (ACCESSIBLE ontologies)

Speed is a significant factor in the implementations of rule-based systems, and many inference engines slow dramatically as the size of the problem increases.

The implemented inference engine though the integration of Jena and Pellet open source Semantic Web technology libraries provide the ground basis for supporting the SPARQL and SWRL languages. For the performance evaluation on information, we placed our focus on scalability and subsequent performance issue which was the important issue for the successfully implementation. Specifically, in evaluating the performance of query processing, we considered 9 sets of SPARQL queries that were generated through the ACCESSIBLE ontologies.

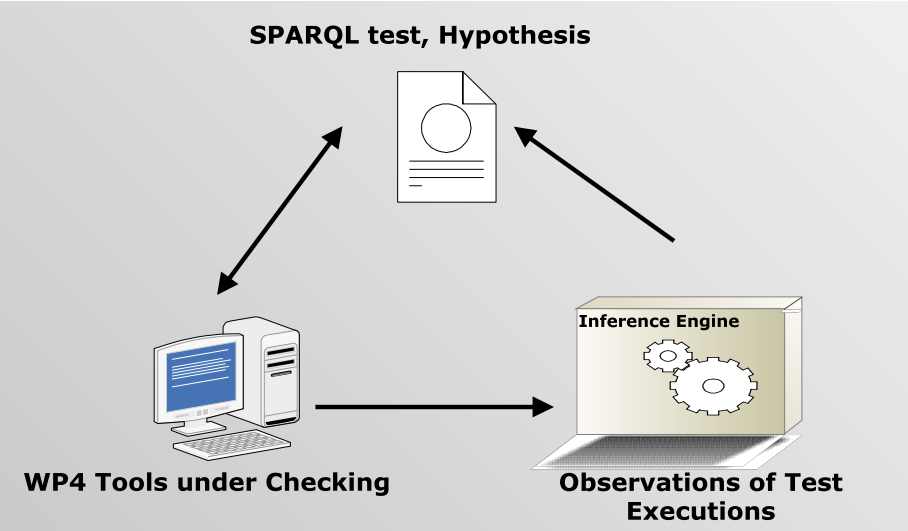


Figure 1 Using the Inference engine to assess test set adequacy and performance

In order to evaluate handling of context information, SPARQL is used as follows (Sparql example for the selection of impairments):

```
SELECT *
WHERE {
        ?impairment rdf:type acc:Impairment ; acc:hasName ?name
}
```

The following sets of SPARQL queries were selected for the performance evaluation:

(1) Get applications query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>

SELECT *
WHERE {
        ?application rdf:type acc:Application ; acc:hasName ?name
}
```

(2) Get approaches and links query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>
```

```
SELECT *
WHERE
{
                ?approach rdf:type acc:Approach ; acc:hasDescription ?description .
        OPTIONAL { ?approach acc:Approach_linksTo_Standard ?standard . }
        OPTIONAL { ?approach acc:Approach_belongsTo_Checkpoint ?checkpoint . }
        OPTIONAL { ?approach acc:Approach_linksTo_Impairment ?impairment . }
        OPTIONAL { ?approach acc:Approach_linksTo_Disability ?disability . }
        OPTIONAL { ?approach acc:Approach_belongsTo_Test ?test . }
  OPTIONAL { ?approach acc:Approach_has_OutputResult ?outputResult }
}
ORDER BY ?approach
```

(3) Get Techniques query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>

SELECT *
WHERE {
        ?technique rdf:type acc:Technique ; acc:hasName ?name ;
acc:Technique_belongsTo_Checkpoint ?checkpoint .
    OPTIONAL { ?technique acc:areAllTestsImplemented ?areAllTestsImplemented }
}
ORDER BY ?technique
```

(4) Get Standards *query*

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>

SELECT *
WHERE {
        ?standard rdf:type acc:Standard ; acc:hasName ?name
}
```

(5) Get Personas query

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>

SELECT *
WHERE {
        ?persona rdf:type acc:User ; acc:hasName ?name ; acc:hasAge ?age ; acc:hasMaritalStatus
?maritalStatus ; acc:hasJob ?job ; acc:hasLocation ?location ; acc:User_has_Disability ?disability ;
acc:Meet ?meet ; acc:TechnologyUsage ?technologyUsage
}
```

(6) Get Functional limitations

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>

SELECT *
WHERE
{
                ?functionalLimitation rdf:type acc:FunctionalLimitation; acc:hasName ?name .
```

```
        OPTIONAL { ?functionalLimitation acc:FunctionalLimitation_belongsTo_Disability
?disability }
}
ORDER BY ?functionalLimitation
```

(7) Get Disabilities

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>

SELECT *
WHERE
{
                ?disability rdf:type acc:Disability ; acc:hasName ?name .
        OPTIONAL { ?disability acc:Disability_belongsTo_Impairment ?impairment }
}
ORDER BY ?disability
```

(8) Get Devices

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>

SELECT *
WHERE {
        ?device rdf:type acc:Device ; acc:hasName ?name
}
```

(9) Get Checkpoints

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX acc: <http://www.AccessibleOntology.com/GenericOntology.owl#>

SELECT *
WHERE {
        ?checkpoint rdf:type acc:Checkpoint ; acc:hasName ?name ; acc:hasPriority ?priority
}
```

We select query response time as the performance measure. The summary of response time is listed in the following Table 1.

Query #	Inference Engine Response Time (ms)
1	10,20
2	3043,21
3	2502,20
4	2225,34
5	250,00
6	520,30
7	35,70
8	221,30
9	980,50

Table 1 Summary of query response time

7 Conclusions

In this deliverable we discussed the compliance with the requirements for all components and knowledge defined at Work Package 4. Each requirement was faced with statements regarding each one of the components and knowledge infrastructure, from the ACCESSIBLE Inference Engine, the ACCESSIBLE Rules User Interface, as well as the ACCESSIBLE Ontology.

All of these requirements were successfully met which results on the correctness of the concepts present within the ACCESSIBLE Ontology, and their integration into the ACCESSIBLE Inference Engine. The technologies and languages used for the definition of these components and knowledge were based on existing – and thoroughly disseminated – standards, which affords also the main properties required for these components: their application and integration into different accessibility assessment and evaluation tools (such as those created within the scope of the ACCESSIBLE project), and their extensibility into new application domains, new accessibility standards, etc.

This deliverable also presented the advancements on the ACCESSIBLE Inference Engine, regarding their optimisation and integration values towards its inclusion into the ACCESSIBLE Architecture and corresponding assessment and simulation tools. The new version of the ACCESSIBLE Inference Engine was presented, along the side of configurations and example client features, in order to afford a seamless integration of the Inference Engine services into different assessment tools.

The execution of the Inference Engine, whether if integrated as a Web service or a library within accessibility assessment tools, provides the required guidance for the expedite evaluation of accessibility in different application domains, and, consequently, provide evidence of the compliance with the elicited requirements for the ACCESSIBLE Content and Knowledge Infrastructure – the overarching goal of Work Package 4.

8 References

[Evans 87] Evans, Michael W. & Marciniak, John. *Software Quality Assurance and Management.* New York, NY: John Wiley & Sons, Inc., 1987.

[IEEE 90] Institute of Electrical and Electronics Engineers. *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries.* New York, NY: 1990

[Nyberg et.al.94] Eric H. Nyberg, Teruko Mitamura & Jaime G. Carbonell. *Evaluation Metrics for Knowledge Based Machine Translation*, Proceedings of COLING-94, Kyoto, Japan, August 5-9, 1994

[Lompré 01] Nicole Lompré. Démarche pour l'évaluation ergonomique. 2001