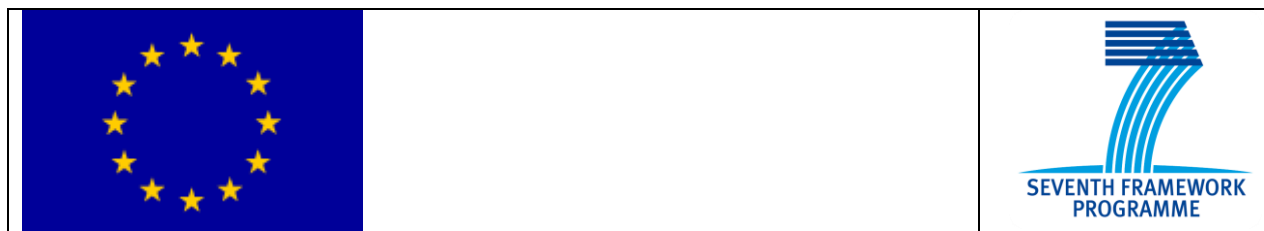


**Seventh Framework Programme
ICT-2009-6.4
Information and Communication Technology**



**Tagging Tool based on a Semantic Discovery
Framework**



Project ID: 247893

Deliverable D3.1.3d

Version 1.1.5

**TaToo Semantic Tagging Tools and Services Specifications
V3**

Annex of D3.1.3 Semantic Service Environment and Framework Architecture V3

Document Control Page			
Title	TaToo Semantic Tagging Tools and Services Specifications V3		
Creator	cismet GmbH (CIS)		
Editor	Pascal Dihé		
Description	This Annex to D3.1.3 contains the functional specifications of the TaToo Semantic Tagging Tools and Services.		
Publisher	TaToo Consortium		
Contributors	all		
Type	Text		
Format	MS Word		
Language	EN-GB		
Creation date	09.05.2011		
Version number	1.1.5		
Version date	01.06.2012		
Last modified by	Pascal Dihé		
Rights	Copyright "TaToo Consortium". During the drafting process, access is generally limited to the TaToo Partners.		
Audience	<input type="checkbox"/> internal <input checked="" type="checkbox"/> public <input type="checkbox"/> restricted, access granted to:		
Review status	<table border="0" style="width: 100%;"> <tr> <td style="vertical-align: top;"> <input type="checkbox"/> Draft <input checked="" type="checkbox"/> WP Leader accepted <input type="checkbox"/> PCO quality controlled <input type="checkbox"/> Co-ordinator accepted </td> <td style="vertical-align: top; padding-left: 20px;"> Where applicable: <input type="checkbox"/> Accepted by the GA <input type="checkbox"/> Accepted by the GA as public document </td> </tr> </table>	<input type="checkbox"/> Draft <input checked="" type="checkbox"/> WP Leader accepted <input type="checkbox"/> PCO quality controlled <input type="checkbox"/> Co-ordinator accepted	Where applicable: <input type="checkbox"/> Accepted by the GA <input type="checkbox"/> Accepted by the GA as public document
<input type="checkbox"/> Draft <input checked="" type="checkbox"/> WP Leader accepted <input type="checkbox"/> PCO quality controlled <input type="checkbox"/> Co-ordinator accepted	Where applicable: <input type="checkbox"/> Accepted by the GA <input type="checkbox"/> Accepted by the GA as public document		
Action requested	<input type="checkbox"/> to be revised by Partners involved in the preparation of the Project Deliverable <input type="checkbox"/> to be revised by all TaToo Partners <input type="checkbox"/> for approval of the WP Leader <input checked="" type="checkbox"/> for approval of the PCO (Quality Manager) <input checked="" type="checkbox"/> for approval of the Project Co-ordinator <input type="checkbox"/> for approval of the General Assembly		
Requested deadline	08/05/2012		

Revision history

Version	Date	Modified by	Comments
0.1	09/05/2011	PDi	1 st draft version, copied specifications from D3.2.1
0.2	26/05/2011	PDi	+ Schema Mapping & RDF Tagger
0.3	27.07.2001	SNe	+ Evalaution Service + Processor
0.4	30.07.2011	PDi	Tagging Updates
0.8	31.07.2011	PDi	Finalised for QA
0.9	08.08.2011	GSc, TPa	QA
1.0	08.08.2011	PDi	Corrections after QA
1.1	02.04.2012	PDi	Updates for V3
1.1.1	20.05.2012	LPe	+ TPZ Contributions
1.1.2	30.05.2012	BBo	+ AIT Contributions
1.1.3	31.05.2012	SNe	+ Added specifications of the TaToo Linked Data components - IDSIA
1.1.4	31.05.2012	PDi	+ integration and updates, conclusion, etc.
1.1.5	01.06.2012	PDi	Version ready for QA



Copyright © 2012, TaToo Consortium

The TaToo Consortium (www.tatoo-project.eu) grants third parties the right to use and distribute all or parts of this document, provided that the TaToo project and the document are properly referenced.

THIS DOCUMENT IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS DOCUMENT, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Table of Contents

1. Management summary	11
1.1. Purpose of this document.....	12
1.2. Intended audience.....	12
1.3. Structure of the document.....	12
2. Referenced standards	13
3. User Components	15
3.1. Simple Tagging Portlet.....	15
3.1.1 Overview and outline.....	15
3.1.1.1 Nature of the component	15
3.1.1.2 Role and scope of the component.....	15
3.1.2 Context.....	15
3.1.2.1 Relation to technical requirements	16
3.1.2.2 Relations to standards.....	16
3.1.2.3 Relations to other TaToo components.....	17
3.1.2.4 Relations to information models	17
3.1.3 Specification.....	17
3.1.3.1 Objectives.....	17
3.1.3.2 Target users	18
3.1.3.3 Functional requirements	18
3.1.3.4 Mock up.....	18
3.1.3.5 Interaction with other TaToo components	19
3.2. Advanced Tagging Portlet.....	20
3.2.1 Overview and outline.....	20
3.2.1.1 Nature of the component	20
3.2.1.2 Role and scope of the component.....	20
3.2.2 Context.....	20
3.2.2.1 Relation to technical requirements	20
3.2.2.2 Relations to standards.....	21
3.2.2.3 Relations to other TaToo components.....	22
3.2.2.4 Relations to information models	22
3.2.3 Specification.....	22
3.2.3.1 Objectives.....	22
3.2.3.2 Target users	23
3.2.3.3 Functional requirements	23
3.2.3.4 Mock up.....	23
3.2.3.5 Interaction with other TaToo components	25
3.3. Tags Editing Portlet.....	25

3.3.1	Overview and outline.....	25
3.3.1.1	Nature of the component	25
3.3.1.2	Role and scope of the component	25
3.3.2	Context.....	26
3.3.2.1	Relation to technical requirements	26
3.3.2.2	Relations to standards	27
3.3.2.3	Relations to other TaToo components.....	27
3.3.2.4	Relations to information models	27
3.3.3	Specification.....	27
3.3.3.1	Objectives.....	28
3.3.3.2	Target users	28
3.3.3.3	Functional requirements	28
3.3.3.4	Mock up.....	28
3.3.3.5	Interaction with other TaToo components	29
3.4.	Geotagging Portlet	30
3.4.1	Overview and outline.....	30
3.4.1.1	Nature of the component	30
3.4.1.2	Role and scope of the component	30
3.4.2	Context.....	31
3.4.2.1	Relation to technical requirements	31
3.4.2.2	Relations to standards	32
3.4.2.3	Relations to other TaToo components.....	32
3.4.2.4	Relations to information models	32
3.4.3	Specification.....	33
3.4.3.1	Objectives.....	33
3.4.3.2	Target users	33
3.4.3.3	Functional requirements	33
3.4.3.4	Mock up.....	33
3.4.3.5	Interaction with other TaToo components	35
3.5.	Linking Portlet	35
1.1.1	Overview and outline.....	35
1.1.1.1	Nature of the component	35
1.1.1.2	Role and scope of the component	35
1.1.2	Context.....	35
1.1.2.1	Relations to standards	36
1.1.2.2	Relations to other TaToo components.....	36
1.1.2.3	Relations to information models	36
1.1.3	Specification.....	37
1.1.3.1	Objectives.....	37
1.1.3.2	Target users	37

1.1.3.3	Functional requirements	37
1.1.3.4	Interaction with other TaToo components	38
4.	Public Services	39
4.1.	Tagging Service	39
4.1.1	Overview and Outline	39
4.1.1.1	Role and Scope of the Service	39
4.1.1.2	Service specification summary	40
4.1.2	Context.....	40
4.1.2.1	Relation to technical requirements	41
4.1.2.2	Relations to standards.....	41
4.1.2.3	Relations to other TaToo components.....	42
4.1.2.4	Relations to information models	42
4.1.3	Specification of the Tagging Service Interface	42
4.1.3.1	Specification of the addAnnotationToResource operation	43
4.1.3.2	Specification of the addAnnotationsToResources operation	44
4.1.3.3	Specification of the getAnnotationsOfResource operation	46
4.1.3.4	Specification of the getAnnotationsOfResources operation.....	47
4.1.3.5	Specification of the removeAnnotation operation	48
4.1.3.6	Specification of the removeAnnotations operation.....	50
4.1.3.7	Specification of the updateAnnotation operation	51
4.1.3.8	Specification of the updateAnnotations operation.....	53
1.2.	Evaluation Service.....	54
1.2.1	Overview and Outline.....	54
1.2.1.1	Role and Scope of the Service	54
1.2.1.2	Service specification summary	55
1.2.2	Context.....	55
1.2.2.1	Relations to standards.....	55
1.2.2.2	Relations to other TaToo components.....	55
1.2.2.3	Relations to information models	55
1.2.3	Specification of the Evaluation Service Interface	56
1.2.3.1	Specification of the addResourceEvaluation operation	56
1.2.3.2	Specification of the addAnnotationEvaluation operation.....	57
1.2.3.3	Specification of the addTagEvaluation operation.....	59
1.2.3.4	Specification of the getResourceEvaluations operation.....	60
1.2.3.5	Specification of the getAnnotationEvaluations operation.....	61
1.2.3.6	Specification of the getTagEvaluations operation.....	62
1.3.	Linking Service	63
1.3.1	Overview and Outline.....	63
1.3.1.1	Role and Scope of the Service	63
1.3.1.2	Service specification summary	64

1.3.2	Context.....	64
1.3.2.1	Relations to standards.....	64
1.3.2.2	Relations to other TaToo components.....	64
1.3.2.3	Relations to information models	65
1.3.3	Specification of the Linking Service Interface.....	65
1.3.3.1	Specification of the addLinks operation	65
1.3.3.2	Specification of the addSimilarityLinks operation.....	67
1.3.3.3	Specification of the getLinkedResources operation.....	68
1.3.3.4	Specification of the getLinkedResourcesByGivenRelationship operation 70	
1.3.3.5	Specification of the getSimilarResources operation	71
2.	Core Components	73
2.1.	Schema Mapping Component.....	73
2.1.1	Overview and outline.....	73
2.1.1.1	Nature of the component	73
2.1.1.2	Role and scope of the component	73
2.1.2	Context.....	74
2.1.2.1	Relation to technical requirements	74
2.1.2.2	Relations to standards.....	74
2.1.2.3	Relations to other TaToo components.....	75
2.1.2.4	Relations to information models	75
2.1.3	Specification.....	75
2.1.3.1	Objectives.....	75
2.1.3.2	Target users	75
2.1.3.3	Functional requirements	75
2.1.3.4	Interaction with other TaToo components	75
2.2.	Filtering Component.....	76
2.2.1	Overview and outline.....	76
2.2.1.1	Nature of the component	76
2.2.1.2	Role and scope of the component	76
2.2.2	Context.....	76
2.2.2.1	Relation to technical requirements	76
2.2.2.2	Relations to standards.....	77
2.2.2.3	Relations to other TaToo Components.....	77
2.2.2.4	Relations to information models	77
2.2.3	Specification.....	77
2.2.3.1	Objectives.....	77
2.2.3.2	Target users	77
2.2.3.3	Functional requirements	78
2.2.3.4	Interaction with other TaToo components	78

2.3.	RDF Tagger Component	78
2.3.1	Overview and outline	78
2.3.1.1	Nature of the component	78
2.3.1.2	Role and scope of the component	78
2.3.2	Context.....	79
2.3.2.1	Relation to technical requirements	79
2.3.2.2	Relations to standards	79
2.3.2.3	Relations to other TaToo components.....	79
2.3.2.4	Relations to information models	80
2.3.3	Specification.....	80
2.3.3.1	Objectives.....	80
2.3.3.2	Target users	80
2.3.3.3	Functional requirements	80
2.3.3.4	Interaction with other TaToo components	80
2.4.	Evaluation Processor	81
2.4.1	Overview and Outline.....	81
2.4.1.1	Nature of the component	81
2.4.1.2	Role and Scope of the component	81
2.4.2	Context.....	82
2.4.2.1	Relation to technical requirements	82
2.4.2.2	Relations to standards	82
2.4.2.3	Relations to other TaToo components.....	82
2.4.2.4	Relations to information models	83
2.4.3	Specification.....	83
2.4.3.1	Objectives.....	83
2.4.3.2	Target users	83
2.4.3.3	Functional requirements	83
2.4.3.4	Interaction with other TaToo components	84
2.5.	Linking Processor.....	84
2.5.1	Overview and Outline.....	84
2.5.1.1	Nature of the component	84
2.5.1.2	Role and Scope of the component	84
2.5.2	Context.....	85
2.5.2.1	Relations to standards	85
2.5.2.2	Relations to other TaToo components.....	85
2.5.2.3	Relations to information models	86
2.5.3	Specification.....	86
2.5.3.1	Objectives.....	86
2.5.3.2	Target users	86
2.5.3.3	Functional requirements	86

2.5.3.4 Interaction with other TaToo components	86
3. Conclusions	87
4. Acknowledgements	88
5. References	89

Index of Figures

Figure 3-1: Simple Tagging Portlet Mockup	19
Figure 3-2: Advanced Tagging Portlet Mockup	24
Figure 3-3: Tags Editing Portlet Mockup	29
Figure 3-4: Geotagging Portlet Mockup	34

Index of Tables

Table 3-1: Tagging Portlet technical requirements	16
Table 3-2: Tagging Portlet technical requirements	21
Table 3-3: Tagging Portlet technical requirements	27
Table 3-4: Geotagging Portlet technical requirements	32
Table 4-5: Tagging Service technical requirements	41
Table 4-6: Tagging Service Interface	42
Table 4-7: Specification of the addAnnotationToResource Operation	44
Table 4-8: Specification of the addAnnotationsToResources Operation	46
Table 4-9: Specification of the getAnnotationsOfResource Operation	47
Table 4-10: Specification of the getAnnotationsOfResources Operation	48
Table 4-11: Specification of the removeAnnotation Operation	49
Table 4-12: Specification of the removeAnnotations Operation	51
Table 4-13: Specification of the updateAnnotation Operation	52
Table 4-14: Specification of the updateAnnotations Operation	54
Table 4-15. Linking Service Interface	65
Table 4-16. Specification of the addLinks operation	67
Table 4-17. Specification of the addSimilarityLinks operation	68
Table 4-18. Specification of the getLinkedResources operation	69
Table 4-19. Specification of the getLinkedResourcesByGivenRelationship operation	71
Table 4-20. Specification of the getSimilarResources operation	72
Table 2-21: Schema Mapping Component technical requirements	74
Table 2-22: Filtering Component technical requirements	77
Table 2-23: RDF Tagger Component technical requirements	79
Table 2-24: Evaluation Processor technical requirements	82

1. Management summary

The present document has been produced by the consortium of the European Project FP7-247893 Tagging Tool based on a Semantic Discovery Framework (TaToo) and corresponds to Annex 2 of the deliverable D3.1.3 – Semantic Service Environment and Framework Architecture V3 (TaToo-D313, 2012).

In this annex the focus is on the tagging, evaluation and linking components, which are described in detail together with a functional specification, intended to be used by software developers. The deliverable, as the final version of three iteration cycles, provides the functional specifications of the:

- Simple Tagging Portlet
- Advanced Tagging Portlet
- Tags Editing Portlet
- Geotagging Portlet
- Linking Portlet
- Tagging Service
- RDF Tagger Component
- Schema Mapping Component
- Filtering Component
- Evaluation Service
- Evaluation Processor
- Linking Service
- Linking Processor

Presented components are specified using common specification templates worked out in the context of WP3 - Specification; see TaToo-D313a, 2012 and TaToo-D313b, 2012.

The achievements of the third version and the main improvements compared to the second version of the Semantic Tagging Tools and Services Specifications are:

- Specification of new Tagging Portlets: Simple Tagging Portlet, Advanced Tagging Portlet, Tags Editing Portlet, Geotagging Portlet.
- Deep Tagging, providing in-depth and more complex tagging functionality to the end user
- Geotagging, providing tagging functionality related to NUTS and GeoNames ontologies
- Update of the Specification Tagging Service
- Update of the Specification of the Filtering Component
- Update of the Specification of the RDF Tagger Component
- Specification of component in the context of linked data: Linking Portlet, Linking Service and Linking Processor

1.1. Purpose of this document

The goal of this document is to provide the final functional (implementation independent) specifications of TaToo tagging, evaluation and linking components and services according to the guidelines provided in D3.1.3 – TaToo Semantic Service Environment and Framework Architecture V3 (TaToo - D313, 2012) while considering the functional and non-functional requirements identified in D2.3.3 – Requirements Document (TaToo - D233, 2011), the experience from the second implementation and design phase which took place in WP4 – Implementation and feedback collected from the validation scenarios.

1.2. Intended audience

The target readers of this document are individuals interested in the TaToo Project, especially in the tagging functionality, as well as Work Package and Task Leaders of the TaToo project (WP4 - Implementation, WP5 – Validation Scenarios) involved in the implementation of the related TaToo Services and Tools.

1.3. Structure of the document

In the following an overview of the document structure and the relationships between the different chapters is given.

- **Chapter 1** consists of this executive summary, it offers an overview and explains the overall purpose of this document.
- **Chapter 2** provides definitions for technology standards referred in subsequent chapters.
- **Chapter 3** contains the final specifications of tagging user components for the third iteration.
- **Chapter 4** contains the final specification of tagging public services for the third iteration.
- **Chapter 5** contains the final specification of tagging, evaluation and linking processors (core components) for the third iteration.
- **Chapter 6** summarizes the results of the work performed so far in the scope of tagging, evaluation and linking.
- **Chapter 7** recognizes that research was funded by the European Community.
- **Chapter 8** lists the references and bibliography used in writing this document.

2. Referenced standards

With the aim of improving document readability, the description of the standards referred across sections describing components have been compiled here.

GeoNames

GeoNames is not a standard but a geographical database available and accessible through various Web services, under a Creative Commons attribution license. It contains over 10,000,000 geographical names corresponding to over 7,500,000 unique features. All features are categorized into one out of nine feature classes and further subcategorized into one out of 645 feature codes. Beyond names of places in various languages, data stored include latitude, longitude, elevation, population, administrative subdivision and postal codes. All coordinates use the World Geodetic System 1984 (WGS84). (<https://en.wikipedia.org/wiki/Geonames>)

Java Portlet Specification (JSR 168 and 268)

Portlets are defined as Java-based Web components, managed by a portlet container, which processes requests and generates dynamic content. Portals use portlets as pluggable user interface components that provide a presentation layer to information systems. The Java Portlet Specification achieves interoperability among portlets and portals by defining the APIs for portlets and by standardizing the rules for preferences, user data, portlet requests and responses, deployment, packaging, and security.

NUTS

The Nomenclature of Territorial Units for Statistics or Nomenclature of Units for Territorial Statistic (NUTS) is a geocode standard for referencing the subdivisions of countries for statistical purposes. The standard is developed and regulated by the European Union, and thus only covers the member states of the EU in detail. (<https://en.wikipedia.org/wiki/NUTS>)

OWL

OWL is a W3C recommendation for a language for defining ontologies performed as a vocabulary extension of RDF. The specification of OWL defines two subsets identified as relevant to the implementers. These subsets are OWL-Lite (for a basic implementation) and OWL-DL (providing the same expressiveness of description logics). However, there are also other well-known subsets defined by different implementers of OWL.

RDF

RDF is a W3C recommendation to establish a standard model for data interchange on the Web. RDF has features that facilitate data merging even if the underlying schemas differ, and it specifically supports the evolution of schemas over time without requiring all the data consumers to be changed.

WSDL

The Web Service Description Language, in version 2.0, is a W3C recommendation for describing Web services. According to the W3C definition, WSDL provides a model and an XML format for describing Web services. WSDL enables one to separate the description of the abstract functionality offered by a service from concrete details of a service description such as “how” and “where” that functionality is offered. This specification defines a language for describing the abstract functionality of a service as well as a framework for describing the concrete details of a service description. It also defines the conformance criteria for documents in this language. Finally, WSDL also describes extensions for message exchange patterns, operation safety, operation styles and binding extensions for SOAP and HTTP.

REST

Although not yet a W3C Recommendation, REST based Web services have had a good adoption since its inception in 2000, being included in the agenda of the W3C working group dedicated to Web services. The Representational State Transfer (REST) is a style of software architecture for distributed hypermedia systems such as the World Wide Web.

The key concept in REST is the existence of resources (sources of specific information), each of which is referenced with a global identifier (e.g., a URI in HTTP). In order to manipulate these resources, components of the network (user agents and origin servers) communicate via a standardized interface (e.g., HTTP) and exchange representations of these resources (the actual documents conveying the information). A REST based web service is a simple web service implemented using HTTP and the principles of REST. It is a collection of resources, with three defined aspects: the base URI for the web service, the MIME type of the data supported by the web service and the set of operations supported by the web service using HTTP methods (e.g., POST, GET, PUT or DELETE).

SPARQL

SPARQL is the query language for RDF repositories recommended by the W3C. SPARQL allows users to query on different RDF repositories using a common language. SPARQL queries describe patterns of graphs that are matched against a repository. The result of a SPARQL query can be a result set or a RDF graph. SPARQL allows performing set operations over its results (union, intersection, etc).

XSLT

XSLT is a language for transforming XML documents into other XML documents by the W3C. It “is a declarative, XML-based language used for the transformation of XML documents into other XML documents. The original document is not changed; rather, a new document is created based on the content of an existing one.” (<http://en.wikipedia.org/wiki/XSLT>).

3. User Components

This chapter describes in details the Tagging, Evaluation and Linking tools.

3.1. Simple Tagging Portlet

This section presents a detailed description and specification for the Simple Tagging Portlet. It provides a component overview followed by the description of the relationships between the component and its context.

3.1.1 Overview and outline

This section provides an overview of the component describing the nature and the role of the component.

3.1.1.1 Nature of the component

The Simple Tagging Portlet is a TaToo User Component. It is part of the TaToo Web Portal and provides basic tagging functionality to the end user, with the Tagging Services to access the TaToo Business tier. Portlets are defined as Web components, managed by a portlet container, that process requests and generate dynamic content. Portlets are configurable through different portlets to offer the presentation layer to the end user.

3.1.1.2 Role and scope of the component

The Simple Tagging Portlet allows a user to tag information about a discovered or already known resource, where a resource could be a data source, a service, a Web page, etc. The user is prompted by the portlet with a category and topic selection panel to choose terms from an ontology to create tags for a resource or a set of resources. When the user adds tags, the Simple Tagging Portlet contacts the TaToo Tagging Service to update the information related to the resource in the TaToo Framework Repository (e.g. meta-information repository).

The Simple Tagging Portlet displays to the TaToo Web Portal user the result of the tagging operation whether it is successful or unsuccessful.

3.1.2 Context

This section describes the relationships between the component and its context, including technical requirements, other TaToo Components, etc.

3.1.2.1 Relation to technical requirements

The Simple Tagging Portlet component addresses the following technical requirements as specified in D2.3.3 – Requirements Document V3 (TaToo-D233). The technical requirements are mapped to the concrete functional requirements specified in chapter 3.2.3.3.

Requirements ID and Name	Scope	Functional Requirement	Comments
TR.TAGGING.010 Meta-information on third party resources	S/I	2, 3, 4	
TR.TAGGING.020 Access to tags (TaToos)	S/I	5	
TR.TAGGING.030 Postponed Tagging / Tagging of known resources	S/I	n/a	Requirement on the Tagging Service..
TR.TAGGING.050 Tagging Client	S/I	all	The portlet itself is a client.
TR.TAGGING.060 Semantic Tags	S/I	all	
TR.TAGGING.070 Storing of Tags	S/I	all	
TR.TAGGING.080 Ontology Tagging	S/I	all	
TR.TAGGING.090 Sharing of Tags (TaToos)	S/I	5	

Table 3-1: Tagging Portlet technical requirements

3.1.2.2 Relations to standards

The Simple Tagging Portlet is specified and implemented following the Java Portlet Specification 168 and 268. A brief description of the Java Portlet Specification can be found in chapter 2, Referenced standards.

The Simple Tagging Portlet processes the annotations provided by the user as selected from an ontology. Available ontologies are encoded using OWL. A brief description of OWL can be found in chapter 2, Referenced standards.

The Simple Tagging Portlet returns the tags selected from the user to the Tagging Services in XML format. A brief description of XML can be found in chapter 2, Referenced standards.

3.1.2.3 Relations to other TaToo components

The Simple Tagging Portlet depends on the TaToo Tagging Service to process the tagged information and update the meta-information related to the particular resource in the TaToo Repository.

The Simple Tagging Portlet depends on the Ontology Manager Service to retrieve a list of terms belonging to a specific ontology.

The Simple Tagging Portlet depends on the Search Portlet for the discovery of the resources to be tagged, if already existing.

3.1.2.4 Relations to information models

The Simple Tagging Portlet should be able to offer the tagging functionality to the user no matter what used ontology. The ontology should serve the user providing terms to tag the resources.

3.1.3 Specification

This section provides the functional specification of the Simple Tagging Portlet, including objectives, mock ups and requirements.

3.1.3.1 Objectives

The objective of the Simple Tagging Portlet is to give the user a graphical interface that allows to:

- add tags, intended as a semantically enriched tags, to a resource;
- add tags to more than one resource matching a search query;
- hide semantic information (i.e. triple statement) from the user interface;
- display tags already associated to a resource.

3.1.3.2 Target users

Simple Tagging Portlet users are those interested in adding tags to already know resources or as resulting from a discovery process without being familiar with semantics. In the former case the users provide the URIs identifying the resources they already know; this is generally the case where the user is the resource owner. In the latter case the user searches resources through a query (through the Search Portlet) and tags those resources they consider relevant.

3.1.3.3 Functional requirements

The functional requirements of the Simple Tagging Portlet are:

1. **Category Selection.** Allows the user to select a category that corresponds to an Ontology Class from an ontology prompted by the portlet;
2. **Topic Selection.** Allows the user to select a topic, that is one of the Ontology individuals pertaining to the selected class, or category;
3. **Tagging.** Adds the term to the meta-information (annotation instance) associated to the resource;
4. **Retrieve Tags.** Allows the user to access already known resource and update existing tags;
5. **Tagging Result.** Informs the user about the result of the tagging operation.

3.1.3.4 Mock up

In this section a mock up of the Simple Tagging Portlet is presented (see Figure 3-2). The Simple Tagging Portlet is contained within the TaToo Web Portal, and accessed through it. From the top to the bottom, the portlet presents two working areas that the user can access to perform the operations of tagging provided by the TaToo Framework.

In the first area the user has the option to add a new resource providing its URI through a text field, or browse the resources discovered by the Search Portlet. For the selected Resource, it is possible to retrieve the already provided annotations stored in the TaToo Knowledge Base. Annotations will be presented in the table under the Resource combo box.

In the second area, the user can select its domain to provide Annotations using terms from his environment. The annotation is structured in three parts: Resource URI, Property and Value, where Property and Value are specified selecting a term from an ontology (respectively predicate and object in the RDF triple), but in the user interface of the Simple Tagging Portlet, these information are hidden and provided in a different way. The user can select a category and then a topic. Topic will be the object of the triple statements, while the predicate is inferred by using MERM Ontology properties.

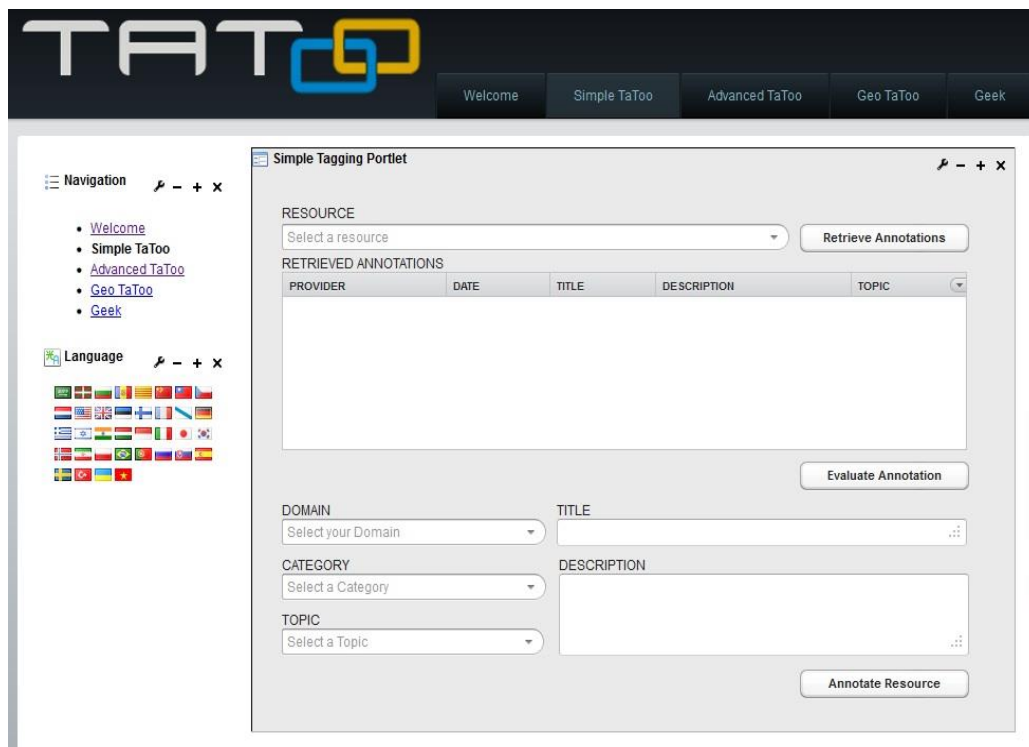


Figure 3-1: Simple Tagging Portlet Mockup

The user is supported in the selection of category and topic terms by auto-complete combo boxes.

3.1.3.5 Interaction with other TaToo components

The Simple Tagging Portlet interacts with the Ontology Manager Service to retrieve the terms from the ontology that the user wants to use to annotate the resource. It also interacts with the Search Portlet to get discovered resources as resulting from a user's search process.

The Simple Tagging Portlet interacts with the Tagging Service to:

- annotate semantically enriched meta-information to the selected resource;
- retrieve annotations of a known resource;

The Simple Tagging Portlet interacts with the Ontology Manager Service to:

- retrieve terms from the ontology of a specific domain.

3.2. Advanced Tagging Portlet

This section presents a detailed description and specification for the Advanced Tagging Portlet. It provides a component overview followed by the description of the relationships between the component and its context.

3.2.1 Overview and outline

This section provides an overview of the component describing the nature and the role of the component.

3.2.1.1 Nature of the component

The Advanced Tagging Portlet is a TaToo User Component. It is part of the TaToo Web Portal and provides advanced and in-depth tagging functionality to the end user, with the Tagging Services to access the TaToo Business tier. Portlets are defined as Web components, managed by a portlet container, that process requests and generate dynamic content.

3.2.1.2 Role and scope of the component

The Advanced Tagging Portlet allows a user to tag information about a discovered or already known resource, where a resource could be a data source, a service, a Web page, etc. The user is prompted by the portlet with a selection panel to choose terms from an ontology to create tags for a resource or a set of resources. When the user adds tags, the Advanced Tagging Portlet contacts the TaToo Tagging Service to update the information related to the resource in the TaToo Framework Repository (e.g. meta-information repository).

The Advanced Tagging Portlet displays to the TaToo Web Portal user the result of the tagging operation whether it is successful or unsuccessful.

3.2.2 Context

This section describes the relationships between the component and its context, including technical requirements, other TaToo Components, etc.

3.2.2.1 Relation to technical requirements

The Advanced Tagging Portlet component addresses the following technical requirements as specified in D2.3.3 – Requirements Document V3 (TaToo-D233). The technical requirements are mapped to the concrete functional requirements specified in chapter 3.2.3.3.

Requirements ID and Name	Scope	Functional Requirement	Comments
TR.TAGGING.010 Meta-information on third party resources	S/I	2, 3, 4	
TR.TAGGING.020 Access to tags (TaToos)	S/I	5	
TR.TAGGING.030 Postponed Tagging / Tagging of known resources	S/I	n/a	Requirement on the Tagging Service..
TR.TAGGING.050 Tagging Client	S/I	all	The portlet itself is a client.
TR.TAGGING.060 Semantic Tags	S/I	all	
TR.TAGGING.070 Storing of Tags	S/I	all	
TR.TAGGING.080 Ontology Tagging	S/I	all	

Table 3-2: Tagging Portlet technical requirements

3.2.2.2 Relations to standards

The Advanced Tagging Portlet must be specified and implemented following the Java Portlet Specification 168 and 268. A brief description of the Java Portlet Specification can be found in chapter 2, Referenced standards.

The Advanced Tagging Portlet processes the annotations provided by the user as selected from an ontology. Available ontologies are encoded using OWL. A brief description of OWL can be found in chapter 2, Referenced standards.

The Advanced Tagging Portlet returns the tags selected from the user to the Tagging Services in XML format. A brief description of XML can be found in chapter 2, Referenced standards.

3.2.2.3 Relations to other TaToo components

The Advanced Tagging Portlet depends on the TaToo Tagging Service to process the tagged information and update the meta-information related to the particular resource in the TaToo Repository.

The Advanced Tagging Portlet depends on the Ontology Manager Service to retrieve a list of terms belonging to a specific ontology.

The Advanced Tagging Portlet depends on the Search Portlet for the discovery of the resources to be tagged, if already existing.

3.2.2.4 Relations to information models

The Advanced Tagging Portlet is able to offer the tagging functionality to the user no matter what used ontology. The ontology should serve the user providing terms to tag the resources. In this process the portlet can take advantage of the MERM ontology.

More information about ontologies and MERM can be found in D3.1.3 – TaToo Semantic Service Environment and Framework Architecture V3, Chapter 6.4 - Minimum Environmental Resource Model.

3.2.3 Specification

This section provides the functional specification of the Tagging Portlet, including objectives, mock ups and requirements.

3.2.3.1 Objectives

The objective of the Advanced Tagging Portlet is to give the user a graphical interface that allows to:

- add annotations, intended as a semantically enriched tags, to a resource;
- add tags to more than one resource matching a search query;
- display resource meta-information;
- display tags already associated to a resource.

3.2.3.2 Target users

Advanced Tagging Portlet users are Semantics aware users interested in adding annotations to a new resource or as resulting from a discovery process. In the former case the users provide the URI identifying the resource. In the latter case the user searches resources through a query (through the Search Portlet) and tags those resources they consider relevant.

3.2.3.3 Functional requirements

The functional requirements for the Advanced Tagging Portlet are:

1. Terms Selection. Allows the user to select a term from an ontology prompted by the portlet;
2. Tagging. Adds the term to the meta-information associated to the resource;
3. Multiple Tagging. Allows to tag multiple resources at the same time with the same terms;
4. Multiple Resources Tagging. Allows the user to tag at the same time with different terms multiple resources;
5. Retrieve Tags. Allows the user to access already known resource and update existing tags;
6. Tagging Result. Informs the user about the result of the tagging operation.

3.2.3.4 Mock up

In this section a mock up of the Advanced Tagging Portlet is presented (see Figure 3-2). The Advanced Tagging Portlet is contained within the TaToo Web Portal, and accessed through it. From the top to bottom, three working areas are identified that the user can access to perform the operations of tagging provided by the TaToo Framework.

In the area the user has the option to add a new resource providing its URI through a text field. At the bottom part, a list of provided / discovered resources is shown. For each resource the user can either retrieve meta-information concerning the resource itself, or retrieve meta-information of previously stored annotations.

In the second area, the user can browse retrieved annotations; in particular the user can evaluate annotations, triggering the Evaluation Portlet deployed on the TaToo Web Portal, or have more details on the annotation, that is the formal triple statement.

In the third area below, the user can actually compose the annotation that will be associated to the selected resource. The user can select the domain to restrict the terms to its environment, and then select a property and a value from the selected Domain Ontology. Moreover, the user can select the type of annotation that he is providing, to perform a deep tagging, and provide a title and description for the annotation.

Advanced Tagging Portlet

RESOURCE

Add a resource Add Resource

Retrieve Annotations Resource Details

RETRIEVED ANNOTATIONS

PROVIDER	DATE	TITLE	DESCRIPTION	TYPE

Evaluate Annotation Annotation Details

DOMAIN TITLE

Select a Domain [Text Field]

PROPERTY DESCRIPTION

Select a Property [Text Field]

VALUE [Radio Buttons]

Select a concept from your Domain [Text Field]

Web Page
 Web Service
 Software
 Timeseries

ANNOTATION DETAILS

PROPERTY	VALUE

Add Geolocation Annotate Resource

Figure 3-2: Advanced Tagging Portlet Mockup

Finally, the user can double check the annotations composed before sending them to the Tagging Service, in order to store them in the TaToo Knowledge Base.

3.2.3.5 Interaction with other TaToo components

The Advanced Tagging Portlet interacts with the Search Portlet to get discovered resources as resulting from a user's search process. It also interacts with the Evaluation Portlet providing the Annotation URI to be evaluated.

The Tagging Portlet interacts with the Tagging Service to:

- annotate semantically enriched meta-information to the selected resource;
- add a common set of annotations to more than one resource;
- add various annotations to more than one resource;
- retrieve annotations of a known resource;
- retrieve meta-information of a resource;

With the Ontology Manager Service to:

- retrieve terms from a Domain Ontology;
- retrieve properties from a Domain Ontology;
- Perform custom SPARQL queries to retrieve set of Ontology concepts or individuals.

3.3. Tags Editing Portlet

This section presents a detailed description and specification for the Tags Editing Portlet. It provides a component overview followed by the description of the relationships between the component and its context.

3.3.1 Overview and outline

This section provides an overview of the component describing the nature and the role of the component.

3.3.1.1 Nature of the component

The Tags Editing Portlet is a TaToo User Component. It is part of the TaToo Web Portal, allows the end user to edit or delete annotations that he has already provided, with the Tagging Services to access the TaToo Business tier. Portlets are defined as Web components, managed by a portlet container, that process requests and generate dynamic content. Portals are configurable through different portlets to offer the presentation layer to the end user.

3.3.1.2 Role and scope of the component

The Tags Editing Portlet allows a user to browse annotations associated to already known or just discovered resources, where a resource could be a data source, a service, a Web page, etc. The

user is prompted by the portlet with the list of resources that he can select to retrieve associated annotations. The user is then able to either delete the selected annotation or edit it, providing a new property and value from the related selections panels. When the user edits annotations, the Tags Editing Portlet contacts the TaToo Tagging Service to update the information related to the resource in the TaToo Framework Repository. Otherwise if the user deletes the annotation the Tagging Service will trigger the entry delete in the TaToo Knowledge Base.

The Tags Editing Portlet displays to the TaToo Web Portal user the result of the edit or delete operation whether it is successful or unsuccessful.

3.3.2 Context

This section describes the relationships between the component and its context, including technical requirements, other TaToo Components, etc.

3.3.2.1 Relation to technical requirements

The Simple Tagging Portlet component addresses the following technical requirements as specified in D2.3.3 – Requirements Document V3 (TaToo-D233). The technical requirements are mapped to the concrete functional requirements specified in chapter 3.2.3.3.

Requirements ID and Name	Scope	Functional Requirement	Comments
TR.TAGGING.010 Meta-information on third party resources	S/I	2, 3, 4	
TR.TAGGING.020 Access to tags (TaToos)	S/I	5	
TR.TAGGING.030 Postponed Tagging / Tagging of known resources	S/I	n/a	Requirement on the Tagging Service.
TR.TAGGING.050 Tagging Client	S/I	all	The portlet itself is a client.
TR.TAGGING.060 Semantic Tags	S/I	all	
TR.TAGGING.070 Storing of Tags	S/I	all	
TR.TAGGING.080	S/I	all	

Ontology Tagging			
TR.TAGGING.090 Sharing of Tags (TaToos)	S/I	5	
TR.TAGGING.100 Editing of Tags (TaToos)	S/I	all	

Table 3-3: Tagging Portlet technical requirements

3.3.2.2 Relations to standards

The Tags Editing Portlet is specified and implemented following the Java Portlet Specification 168 and 268. A brief description of the Java Portlet Specification can be found in chapter 2, Referenced standards.

The Tags Editing Portlet processes the annotations provided by the user as selected from an Ontology. Available ontologies are encoded using OWL. A brief description of OWL can be found in chapter 2, Referenced standards.

The Tags Editing Portlet returns the tags selected from the user to the Tagging Services in XML format. A brief description of XML can be found in chapter 2, Referenced standards.

3.3.2.3 Relations to other TaToo components

The Tags Editing Portlet depends on the TaToo Tagging Service to process the tagged information and update or delete the meta-information related to the particular resource in the TaToo Repository.

The Tags Editing Portlet depends on the Search Portlet for the discovery of the resources which annotations will be modified or deleted.

3.3.2.4 Relations to information models

The Tags Editing Portlet should be able to offer the tagging functionality to the user no matter what used ontology. The ontology should serve the user providing terms to tag the resources.

3.3.3 Specification

This section provides the functional specification of the Tags Editing Portlet, including objectives, mock ups and requirements.

3.3.3.1 Objectives

The objective of the Tags Editing Portlet is to give the user a graphical interface that allows to:

- modify existing annotations, intended as a semantically enriched tags, already associated to a resource;
- delete existing annotations, intended as a semantically enriched tags, already associated to a resource;
- hide semantic information (i.e. triple statement) from the user interface;
- display tags already associated to a resource.

3.3.3.2 Target users

Tags Editing Portlet users are advanced user interested in editing or deleting their own annotations already provided. The users will select the resource among the list of known or discovered resources and will either trigger the delete or the edit operation. In the latter case the user will be prompted with tagging functionality combo boxes.

3.3.3.3 Functional requirements

The functional requirements for the Tags Editing Portlet are:

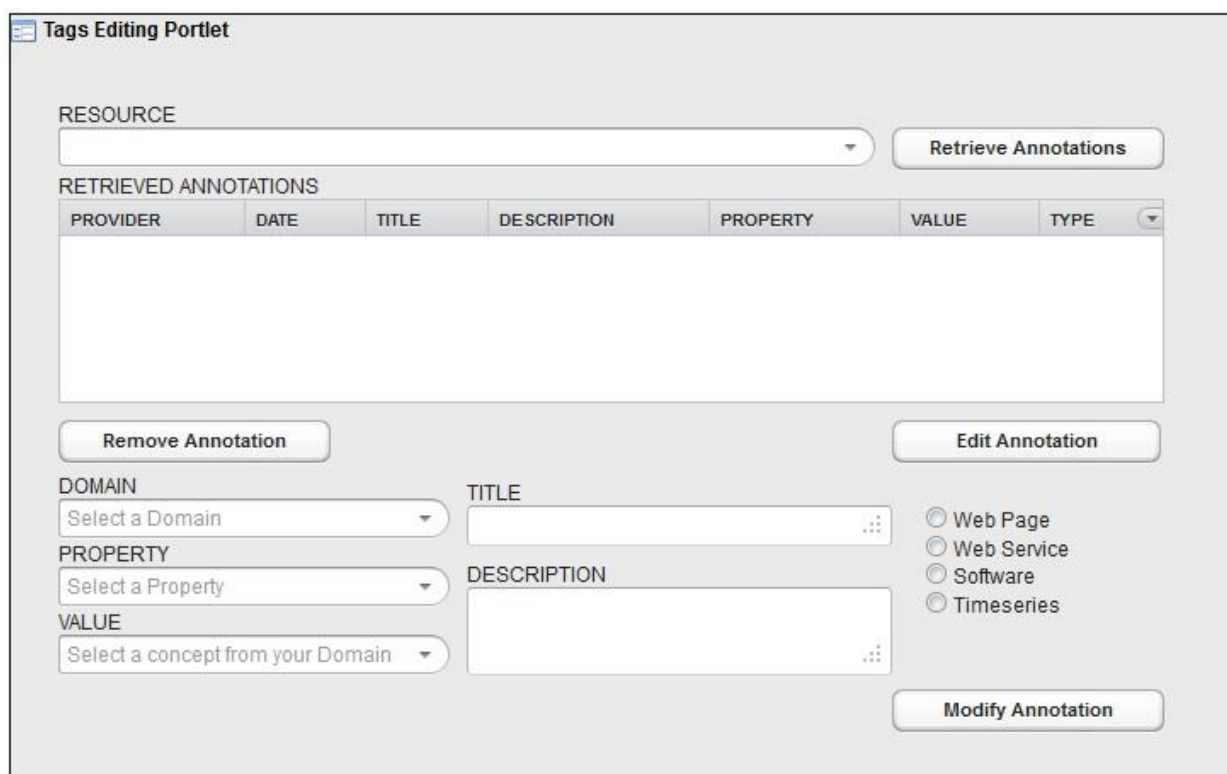
1. Retrieve annotations. Allows the user to select a discovered or already known resource and retrieve previously associated meta-information;
2. Edit Tagging. Allows the user to associate meta-information to a resource, modifying existing triple statement;
3. Delete Tagging. Allows the user to select an annotation and remove it from the TaToo Knowledge Base repository;
4. Operation Result. Informs the user about the result of the edit or delete operation.

3.3.3.4 Mock up

In this section a mock up of the Tags Editing Portlet is presented (see Figure 3-3). The Tags Editing Portlet is contained within the TaToo Web Portal, and accessed through it. From the top to the bottom, the portlet presents two working areas that the user can access to perform the operations of delete or tags editing provided by the TaToo Framework.

In the first area the user has the option to add a new resource providing its URI through a text field, or browse the resources discovered by the Search Portlet. For the selected Resource, it is possible to retrieve the already provided annotations stored in the TaToo Knowledge Base repository. Annotations will be presented in the table under the Resource combo box.

In the second area, the user can browse retrieved annotations; in particular the user can delete annotations, triggering the remove annotation operation provided by the Tagging Service, or edit annotations. If the user selects the edit option, the combo boxes pertaining to tagging will become available from the initial disabled status. The user can then update the selected annotation by composing a new triple statement by selecting the domain ontology, and then property and value terms. It is also possible to update title and description, and the type of annotation provided. The user is then allowed to select modify annotation to confirm the operation.



The mockup shows a 'Tags Editing Portlet' interface. At the top, there is a 'RESOURCE' dropdown menu and a 'Retrieve Annotations' button. Below this is a table titled 'RETRIEVED ANNOTATIONS' with columns: PROVIDER, DATE, TITLE, DESCRIPTION, PROPERTY, VALUE, and TYPE. The table is currently empty. Below the table are two buttons: 'Remove Annotation' and 'Edit Annotation'. Under the 'Edit Annotation' button, there are several input fields: 'DOMAIN' (dropdown), 'PROPERTY' (dropdown), 'VALUE' (dropdown), 'TITLE' (text input), and 'DESCRIPTION' (text input). To the right of these fields are four radio buttons for annotation types: 'Web Page', 'Web Service', 'Software', and 'Timeseries'. At the bottom right, there is a 'Modify Annotation' button.

Figure 3-3: Tags Editing Portlet Mockup

The User is supported in the selection of category and topic terms by auto-complete combo boxes.

3.3.3.5 Interaction with other TaToo components

The Tags Editing Portlet interacts with the Ontology Manager Service to retrieve the terms from the ontology that the user wants to use to annotate the resource. It also interacts with the Search Portlet to get discovered resources as resulting from a user's search process.

The Tags Editing Portlet interacts with the Tagging Service to:

- edit semantically enriched meta-information associated to the selected resource;

- delete meta-information associated to selected resource;
- retrieve annotations of a known resource;

3.4. Geotagging Portlet

This section presents a detailed description and specification for the Geotagging Portlet. It provides a component overview followed by the description of the relationships between the component and its context.

3.4.1 Overview and outline

This section provides an overview of the component describing the nature and the role of the component.

3.4.1.1 Nature of the component

The Geotagging Portlet is a TaToo User Component. It is part of the TaToo Web Portal and provides advanced geotagging functionality to the end user.

3.4.1.2 Role and scope of the component

The Geotagging Portlet allows a user to add geographical information to a discovered or already known resource, whereby a resource could be a data source, a service, a Web page, etc. The Geotagging Portlet does not make direct use of geographical coordinates when assigning a location to a resource. Instead, it allows tagging resources with GeoNames Features from the GeoNames Ontology or NUTS regions which are related to a specific geographic point or polygon.

The user can select a point or polygon on a map to create geotags for a resource. When the user adds a geotag, the Geotagging Portlet contacts the TaToo Tagging Service to update the information related to the resource in the TaToo Framework Repository (knowledgebase). The Geotagging Portlet supports also the visualisation of geotags already associated with a resource.

Geotags in the context of the Geotagging Portlet refer to ontology instances of NUTS regions or GeoNames features stored in the TaToo Framework Repository. Conceptually, NUTS regions or GeoNames features are related to an annotation instance through the *locatedIn* object property. The annotation is then associated with the resource. It is therefore also possible to assign multiple locations (e.g. city, a region and a country) to a single resource.

The Geotagging Portlet displays to the TaToo Web Portal user the result of the geotagging operation whether it is successful or unsuccessful.

3.4.2 Context

This section describes the relationships between the component and its context, including technical requirements, other TaToo Components, etc.

3.4.2.1 Relation to technical requirements

The Geotagging Portlet addresses the following technical requirements as specified in D2.3.3 – Requirements Document V3 (TaToo-D233). The technical requirements are mapped to the concrete functional requirements specified in chapter 3.2.3.3.

Requirements ID and Name	Scope	Functional Requirement	Comments
TR.TAGGING.010 Meta-information on third party resources	S/I	3, 4	
TR.TAGGING.020 Access to tags (TaToos)	S/I	6	
TR.TAGGING.030 Postponed Tagging / Tagging of known resources	S/I	n/a	Requirement on the Tagging Service..
TR.TAGGING.050 Tagging Client	S/I	all	The portlet itself is a client.
TR.TAGGING.060 Semantic Tags	S/I	all	
TR.TAGGING.070 Storing of Tags	S/I	all	
TR.TAGGING.080 Ontology Supported Tagging	S/I	all	GeoNames, NUTS and MERM Ontologies supported
TR.DISCOVERY.060	S/I	6	

Geospatial and time related search			
TR.TAGGING.120 (Automatic) geo-tagging of data sources	S/I	1,2,3,4	

Table 3-4: Geotagging Portlet technical requirements

3.4.2.2 Relations to standards

The Geotagging Portlet must be specified and implemented following the Java Portlet Specification 168 and 268. A brief description of the Java Portlet Specification can be found in chapter 2, Referenced standards.

The Geotagging Portlet supports geotags provided by the user as selected from the GeoNames and the NUTS ontologies. Available ontologies are encoded using OWL. A brief description of OWL and NUTS can be found in chapter 2, Referenced standards.

The Geotagging Portlet returns the geotags made by the user to the Tagging Services in XML format. A brief description of XML can be found in chapter 2, Referenced standards.

3.4.2.3 Relations to other TaToo components

The Geotagging Portlet depends on the TaToo Tagging Service to process the tagged information and update the meta-information related to the particular resource in the TaToo Repository.

The Geotagging Portlet supports the creation and visualisation of geotags only, therefore the Geotagging Portlet depends on the Simple Tagging, Advanced Tagging and Search Portlet for the selection of the resources to be tagged.

The Geotagging Portlet depends on third party services to search for GeoNames features or NUTS regions given the geographical location (point or polygon) as input (gazetteer services).

3.4.2.4 Relations to information models

The Geotagging Portlet offers the geotagging functionality to the user according to the MERM, the GeoNames and the NUTS ontologies. Those ontologies serve the user providing GeoNames features or NUTS regions instances to geotag the resources. In this process the portlet takes advantage of the MERM to create annotations and of the GeoNames and NUTS ontologies to create the actual geotags. The Geotagging Portlet does not support domain ontologies. To tag resources with domain concepts, the Simple Tagging Portlet, the Advanced Tagging Portlet or any other Tagging User Component may be used.

More information about ontologies and MERM can be found in D3.1.3 – TaToo Semantic Service Environment and Framework Architecture V3, Chapter 6.4 - Minimum Environmental Resource Model. A brief description of NUTS and GeoNames can be found in chapter 2, Referenced standards.

3.4.3 Specification

This section provides the functional specification of the Geotagging Portlet, including objectives, mock ups and requirements.

3.4.3.1 Objectives

The objective of the Geotagging Portlet is to give the user a graphical interface that allows to:

- add geotags, referring to a point location represented by instances of the GeoNames ontology, to a resource ;
- add geotags, referring to a region represented by instances of the NUTS ontology, to a resource ;
- display geotags already associated to a resource on a map.

3.4.3.2 Target users

Geotagging Portlet users are users that are interested in geospatial resources, specifically resources that are related to a certain region, city, etc.

3.4.3.3 Functional requirements

The functional requirements for the Geotagging Portlet are:

1. Point Selection. Allows the user to select a point in a map;
2. Polygon Selection. Allows a user to select a polygon (e.g. a bounding box) in a map;
3. Geotagging with GeoNames. Allows the user to geotag a resource with a GeoNames feature identified by the previously selected point;
4. Geotagging with NUTS. Allows the user to geotag a resource with a NUTS region identified by the previously selected polygon;
5. Geotagging with multiple locations. Allows the user to geotag a single resource with multiple locations (NUTS and GeoNames)
6. View resource location. Allows the user to show the location(s) of the resource on a map.

3.4.3.4 Mock up

In this section a mock up of the Geotagging Portlet is presented (see Figure 3-4).

The Geotagging Portlet consists of a map panel and information panel. The map panel is used for two purposes: Firstly it shows the location(s) of a resource on the map, if the appropriate geotags (GeoNames or NUTS instances) are related to the resource. Secondly it allows the user to interactively make a point or a polygon selection by either clicking on the map or by drawing a polygon (e.g. a bounding box). The point or polygon is then translated by dedicated services to either GeoNames feature or NUTS regions.

The second area, the information panel, is used to display meta-information about the selected NUTS region or GeoNames Feature. It can also be used to select a specific feature or region if the coordinate selection made by the users is ambiguous and covers several regions or features. The user can then select the appropriate region or feature from a bounding box. Depending on the map technology used to implement the Geotagging Portlet (GoogleMaps, OpenLayers), the information panel can also be realised as popup.

Finally, the user can click on the Annotate button to assign a new geotag to the resource. This can be repeated multiple times, so the user is able to assign multiple geotags to a single resource.



Figure 3-4: Geotagging Portlet Mockup

3.4.3.5 Interaction with other TaToo components

The Geotagging Portlet interacts with the TaToo Tagging Service to store and retrieve annotations which contain information on the geographic location of the resource. It furthermore interacts with one or more gazetteer services which translate geographical coordinates selected by the user in the map to GeoNames features or NUTS regions. It interacts also with the User Access Manager to authenticate the user that is using the Portlet

3.5. Linking Portlet

This section presents a detailed description and specification for the Linking Portlet. It provides a component overview followed by the description of the relationships between the component and its context.

1.1.1 Overview and outline

This section provides an overview of the component describing the nature and the role of the component.

1.1.1.1 Nature of the component

The Linking Portlet is a TaToo User Component, which is part of the TaToo Web Portal. It enables end users to access the TaToo Linked Data functionalities provided by the TaToo Business tier.

1.1.1.2 Role and scope of the component

The Linking Portlet provides UI elements that enable TaToo users to benefit from the introduced Linked Data functionalities. In particular, these UI elements are responsible for adding typed links according to the Linked Data principles between resources annotated by the TaToo System.

The Linking Portlet is reachable in both TaToo tagging and TaToo discovery use cases. This means that a user is able to link resources to other resources at the time she/he annotates the resource as well as browses search results (discovered resources). To satisfy both use cases, but not to replicate the same UI elements, the TaToo system provides the Linking Portlet that is independent from the discovery and tagging portlets but is invoked by them.

1.1.2 Context

This section describes the relationships between the component and its context, including technologies and standards, other TaToo Components, etc.

1.1.2.1 Relations to standards

The Linking Portlet is specified and implemented following the Java Portlet Specification 168 and 268. A brief description of the Java Portlet Specification can be found in chapter 2, Referenced standards.

1.1.2.2 Relations to other TaToo components

The Linking Portlet depends on the Linking Service and its functionalities (operations). The Linking Service serves the portlet with all necessary information (e.g. link types) from the TaToo Knowledgebase for specifying links between resources. Moreover, it receives link specifications from the portlet and invokes corresponding operations of the TaToo business tier to create and store the links into TaToo Knowledgebase.

1.1.2.3 Relations to information models

The TaToo Linked Data are created according to the Linked Data principles. TaToo resources, that is, resources that are annotated and evaluated by the TaToo approach, are uniquely identified and can be easily linked according to the Linked Data principles. Considering that the TaToo approach has already provided formalisms (i.e. MERM) and mechanisms (i.e. tagging operations) for handling and storing resource URIs, the key issue of the TaToo Linked Data approach is the selection of the link types that are used for linking the TaToo resources.

The TaToo Framework is going to support a predefined set of the link types (i.e. properties). That set of link types will include both upper-level link types and domain-level link types. The upper-level link types include object properties for standardised or well-known upper level ontologies. Some candidates for the TaToo upper-level link types include:

1. `rdfs:seeAlso` - A related resource that provides further information about the subject resource.
2. `rdfs:isDefinedBy` - A related resource that represents a definition of the subject resource.
3. `rdfs:member` - A related resource that is a member of the subject resource.
4. `owl:sameAs` - URI references of the subject and object resources refer to the same thing.
5. `owl:differentFrom` - Subject and object resources are different.
6. `dcterms:hasPart` - A related resource that is included either physically or logically in the subject resource.

7. `dcterms:isPartOf` - A related resource in which the subject resource is physically or logically included.
8. `dcterms:isReferencedBy` - A related resource that references, cites, or otherwise points to the subject resource.
9. `dcterms:isReplacedBy` - A related resource that supplants, displaces, or supersedes the subject resource.
10. `dcterms:isRequiredBy` - A related resource that requires the subject resource to support its function, delivery, or coherence.

Besides the upper-level link types, the predefined set of the TaToo link types will contain some domain-level link types which will be selected from the TaToo domain ontologies (i.e., MU, AIT, and JRC).

1.1.3 Specification

This section provides the functional specification of the Linking Portlet, including objectives, target users, and requirements.

1.1.3.1 Objectives

The objective of the Linking Portlet is to give the user a graphical interface that allows them to set explicit, typed links between resources managed by the TaToo System.

1.1.3.2 Target users

Users of the Linking Portlet are TaToo users that are interested in the integration/linking of related TaToo resources. By having linked resources, it would be able to discover/access them not only by searching but also navigating across the resources following the added links.

1.1.3.3 Functional requirements

The functional requirements for the Linking Portlet are:

- Selection of resources to be linked
- Selection of a link type (property)



1.1.3.4 Interaction with other TaToo components

The Linking Portlet interacts with the TaToo Tagging and Discovery Portlets from which it gets information (i.e. URIs) about resources to be linked. Moreover, the portlet interacts with the TaToo Linking Service from which it gets a list of TaToo supported link types and to which it sends information about links to be created and stored into TaToo Knowledgebase. Finally, it interacts with the User Access Manager to authenticate the user that is using the Portlet.

4. Public Services

This chapter includes the specification of the Tagging and evaluation services.

4.1. Tagging Service

This section defines the functional specification of the Tagging Service and is based on the Template for the Specification of TaToo Service.

In the current version, the interface of the tagging service was cleaned by deprecated V1 operations (addTag, ...). Those operations were replaced in V2 by operations that are more aligned to the MERM ontology. V2 and V3 clients (portlets and validation scenario specific user components) were adapted to the new V2 tagging operations; therefore the old V1 operations could be safely removed. Furthermore, the retrieval of ontologies is in V3 handled by a dedicated Ontology Manger Service, therefore operations for the retrieval of ontologies (MERM ontology and domain ontologies) were also removed from the interface of the tagging services.

Major updates were performed on the definition of tagging related data types (resource, annotation, etc.) to support deep tagging (tagging with complex annotation types, such as time series and service annotations) and to harmonise data types between the tagging and the discovery service.

4.1.1 Overview and Outline

This section provides an overview of the component describing the Nature and the Role of the component.

4.1.1.1 Role and Scope of the Service

The Tagging Service is a TaToo Public Service that allows User Components, in particular the various Tagging Portlets and validation scenario specific User Components, to access the tagging functionality offered by the TaToo System and exposed through the public interface of this service. A Tagging Service receives tagging requests from the different User Components to:

- associate tags with resources;
- access tags associated with a resource;
- update tags associated with the resource;
- delete tags;

Additional functionalities of the Tagging Service include:

- apply a filter on the list tags retrieved (e.g. filter by annotation provider, date annotated, etc.)

- access tags in different languages (if available)

In V1 the functionality of the Tagging Service was limited to the creation of new tags and the read-only access to existing tags. In V2 support for editing and deleting was added. In V3 better support for advanced (deep) tags was added, which means to support all possibly types of the MERM Annotation class. Tagging is limited to semantic tagging which means choosing from a number of terms of the selected ontology and the supported formats for ontologies and tags are limited to RDF and OWL. The access to ontologies is realised by a dedicated Ontology Manager Service.

The Tagging Service is the service responsible for all the operations related to the tagging of resources. It interacts with the tagging User Components that demand updating of tags for a resource and the Clearinghouse to access to the knowledgebase, but also to retrieve ontology information if needed. For the specification of the Clearinghouse, please refer to D3.1.3 - Semantic Service Environment and Framework Architecture V3 (TaToo-D313), section 7.3.1 - Clearinghouse.

The Tagging Service is supposed to be an interoperable and standalone Web service, in particular a Web Service W3C compliant taking advantage of widely adopted standards such as XML, WSDL, and SOAP.

4.1.1.2 Service specification summary

The service specification of Tagging is comprised of the Tagging Interface, that includes all operations related to the tagging functionality. The Tagging Interface contains the following operations:

- addAnnotationToResource, associates a single annotation with exactly one resource (1 to 1).
- addAnnotationsToResources, associates different annotations with at least one resource (n to m).
- getAnnotationsOfResource, retrieves all annotations of a single resource.
- getAnnotationsOfResources, retrieves all annotations of at least one resources
- removeAnnotation, deletes a specific annotation
- removeAnnotations, deletes several annotations
- updateAnnotation, edits a specific annotation
- updateAnnotations, edits several annotations

4.1.2 Context

This section describes the relationships between the service and its context, including technical requirements, other TaToo components, etc.

4.1.2.1 Relation to technical requirements

The Tagging Service addresses the following technical requirements as specified in D2.3.3 – “Requirements Document V3”.

Requirements ID and Name	Scope	Fulfilment
TR.TAGGING.010 Meta-information on third party resources	S/I	by the addAnnotationToResource and addAnnotationToResources operation.
TR.TAGGING.020 Access to tags (TaToos)	S/I	by the getAnnotationsOfResource and getAnnotationsOfResources operation.
TR.TAGGING.030 Postponed Tagging / Tagging of known resources	S/I	by providing the Tagging Interface and implementing it as web service.
TR.TAGGING.040 Tagging Service	S/I	see above.
TR.TAGGING.060 Semantic Tags	S/I	by supporting tags that are terms from an ontology .
TR.TAGGING.080 Ontology supported tagging	S/I	see above.
TR.TAGGING.100 Editing of Tags (TaToos)	S/I	by the edit and delete annotations operations

Table 4-5: Tagging Service technical requirements

4.1.2.2 Relations to standards

The Tagging Service is a Web Service that follows W3C recommendations. The Tagging Service could be either a RESTful Web Service, exposing a uniform set of “stateless” operations, or an arbitrary Web Service exposing a WSDL Interface that contains an arbitrary set of operations. A brief description of WSDL and REST can be found in chapter 3, Referenced standards.

4.1.2.3 Relations to other TaToo components

The TaToo Tagging Service is accessible via its interface from Tagging Tools, specifically the Tagging Portlet or other portlets and third parties applications.

The Tagging Service interacts with the TaToo Clearinghouse to access the TaToo Core Components to store the meta-information related to the resource and / or process the meta-information via a TaToo Tagging Processor components (e.g. Filtering Component).

4.1.2.4 Relations to information models

The Tagging Service operations process information that relate to the MERM and other ontologies of a certain domain.

More information about ontologies and MERM can be found in D3.1.3 - Semantic Service Environment and Framework Architecture V3 (TaToo-D313), chapter 6.4 - Minimum Environmental Resource Model.

4.1.3 Specification of the Tagging Service Interface

This section provides the functional specification of the Tagging Service, including the Service Interface and its operations.

The objective of the Tagging Service is to provide an interoperable, Web available interface as an entry point for the TaToo Tagging operations, which are described in the following table:

Operation Name	Description
addAnnotationToResource	Optional convenience operation that associates exactly one annotation with exactly one resource and sends it to the Clearinghouse.
addAnnotationsToResources	Mandatory operation that adds 1-n annotations to 1-m resources and sends it to the Clearinghouse.
getAnnotationsOfResource	Optional convenience operation that retrieves all annotations associated to exactly one resource from the TaToo Framework Repository.
getAnnotationsOfResources	Mandatory operation that retrieves 1-n annotations associated to 1-m resources from the TaToo Framework Repository.
editAnnotation	Optional operation that edits a specific annotation
removeAnnotations	Mandatory operation that deletes all specified annotations

Table 4-6: Tagging Service Interface

4.1.3.1 Specification of the `addAnnotationToResource` operation

The optional *addAnnotationToResource* operation is responsible for the tagging of exactly one resource, identified by its URI, and thus allows the establishment of a 1-1 relationship between one annotation and one resource. If different annotations shall be associated with different resources at once, the *addAnnotationsToResources* operation must be used instead. The *addAnnotationToResource* operation is an optional convenience operation whose functionality is covered completely by the *addAnnotationsToResources* operation.

To achieve the tagging functionality the operation is supported by an ontology provided by the Clearinghouse via the Ontology Manager Service. In a Tagging User Component (e.g. the Advanced Tagging Portlet), the user will select the terms to compose the tag that will be associated to the resources and stored in the TaToo Framework Repository.

A request to perform the *addAnnotationToResource* operation shall include the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is

String `addAnnotationToResource(Resource resource, Annotation annotation, locale locale)` throws `InvalidParameterValue`, `MissingParameterValue`, `ResourcesNotFound`, `TaTooInternalError`, `MalformedAnnotation`

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Optional			
Receives	Name	Type	Use	Description
	resource	Resource	Mandatory	The resource to be annotated
	annotation	Annotation	Mandatory	The annotation to be added
	locale	String	Optional	The language of the user, default: ‘en-GB’
Returns	Type		Description	
	String		Status message.	
Throws	Type		Cause	

	ResourcesNotFound	The resources identified by the provided URIs could not be found.
	TaTooInternalError	A problem occurred during the processing of the request by some of the TaToo Core Components, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.
	InvalidParameterValue	Operation request contains an invalid parameter value. Returns the name of the parameter with invalid value.
	MissingParameterValue	Operation request does not provide values of some of the mandatory parameters. Returns the name of the missing parameter.
	MalformedAnnotation	The type or the format of the Annotation is not supported by the service.

Table 4-7: Specification of the addAnnotationToResource Operation

4.1.3.2 Specification of the addAnnotationsToResources operation

The *addAnnotationsToResources* operation is used to add *n* annotations to exactly *n* resources. The annotations and resources have to be ordered in order to ensure a proper assignment. If the resource already exists, the resource meta-information will not be added. The operation returns among some status information the URIs of the newly created annotations. If exactly one annotation shall be added to one resource, the convenience operation *addAnnotationToResource* can be used instead.

If a new resource shall be added to the semantic repository, all properties of the resource shall be provided. If an annotation shall be added to an existing resource, only the property URI of the resource has to be provided.

The locale parameter is used to identify the language of literals, e.g. currently title and description of the resources and annotations and the labels of topics. If the locale parameter is not provided, the default locale en-GB will be used.

To achieve the tagging functionality the operation will be supported by an ontology provided by the Clearinghouse via the Ontology Manager Service. In a Tagging User Component (e.g. the Tagging Portlet), the user will select the terms to compose the tags that will be associated to the resources and stored in the TaToo Framework Repository.

A request to perform the *addTags* operation shall include the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is

String addAnnotationsToResources(List<Resource> resources, List<Annotation> annotations, String locale) throws MissingParameterValue, InvalidParameterValue, TaTooInternalError, MalformedAnnotation

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	resources	List <Resource>	Mandatory	The resources to be annotated
	annotations	List <Annotation>	Mandatory	Ordered list of annotations
	locale	String	Optional	Language of the user, default: ‘en-GB’
Returns	Type		Description	
	String		Status message.	
Throws	Type		Cause	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	
	TaTooInternalError		A problem occurred during the processing of the request by some of the TaToo Core Components, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.	
	InvalidParameterValue		Operation request contains an invalid parameter value. Returns the name of the parameter with invalid value.	

	MissingParameterValue	Operation request does not provide values of some of the mandatory parameters. Returns the name of the missing parameter.
	MalformedAnnotation	The type or the format of the Annotation is not supported by the service.

Table 4-8: Specification of the addAnnotationsToResources Operation

4.1.3.3 Specification of the getAnnotationsOfResource operation

The optional *getAnnotationsOfResource* operation is responsible for the retrieval of all annotations associated to a resource given the resource URI as input. An optional filter condition can be specified to narrow the number of annotations. The *getAnnotationsOfResource* operation is an optional convenience operation whose functionality is covered completely by the *getAnnotationsOfResources* operation.

The Tagging Service will contact the Clearinghouse with the URI of the resource and will receive all the meta-information. The optional filter condition is evaluated by a dedicated Tagging Processor.

A request to perform the *getAnnotationsOfResource* operation shall include the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is **List<Annotation> getAnnotationsOfResource(String resourceURI, String filterOptions, String locale) throws MissingParameterValue, InvalidParameterValue, TaTooInternalError, ResourcesNotFound**

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Optional			
Receives	Name	Type	Use	Description
	resourceURI	String	Mandatory	Resources URI
	filterOptions	String	Optional	Filter options
	locale	String	Optional	Language of the user, default: ‘en-GB’

Returns	Type	Description
	List <Annotation>	List of annotations
Throws	Type	Cause
	ResourcesNotFound	The resources identified by the provided URIs could not be found.
	TaTooInternalError	A problem occurred during the processing of the request by some of the TaToo Core Components, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.
	InvalidParameterValue	Operation request contains an invalid parameter value. Returns the name of the parameter with invalid value.
	MissingParameterValue	Operation request does not provide value of some of the mandatory parameters. Returns the name of the missing parameter.

Table 4-9: Specification of the `getAnnotationsOfResource` Operation

4.1.3.4 Specification of the `getAnnotationsOfResources` operation

The mandatory `getAnnotationsOfResources` operation is responsible for the retrieval of annotations associated to resources given the resource URIs as input. This operation supports the retrieval of a multiple annotations of multiple resources. An optional filter condition can be specified to narrow the number of annotations. If only the annotations of one specific resource need to be retrieved, the convenience operation `getAnnotationsOfResource` can be used instead.

The operation returns a map with a URI of a resource as key and a list of annotations associated with the resource that match the optional filter condition as value.

A request to perform the `getAnnotationsOfResources` operation shall include the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is

Map<URI, List<Annotation>> getAnnotationsOfResources(List<String> resourceURIs, String filterOptions, String locale) throws MissingParameterValue, InvalidParameterValue, TaTooInternalError, ResourcesNotFound

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	resources	List <Resource>	Mandatory	List of resources
	filterOptions	String	Optional	Optional filter condition
	locale	String	Optional	Language of the user, default: 'en-GB'
Returns	Type		Description	
	Map<URI, List<Annotation>>		Resource annotation mappings	
Throws	Type		Cause	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	
	TaTooInternalError		A problem occurred during the processing of the request by some of the TaToo Core Components, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.	
	InvalidParameterValue		Operation request contains an invalid parameter value. Returns the name of the parameter with invalid value.	

Table 4-10: Specification of the getAnnotationsOfResources Operation

4.1.3.5 Specification of the removeAnnotation operation

The optional *removeAnnotation* operation is responsible for the removal of a single annotation identified by the annotation URI. If the user identified by the parameter foafOnlineAccount is not the owner (provider) of the annotation, the operation will fail and a NotAuthorised exception is thrown. The *removeAnnotation* operation is an optional convenience operation whose functionality is covered completely by the *removeAnnotations* operation.

A request to perform the removeAnnotation operation shall include the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter.

Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is

String removeAnnotation(Annotation annotation, String foafOnlineAccount) throws MissingParameterValue, InvalidParameterValue, TaTooInternalError, NotAuthorised

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Optional			
Receives	Name	Type	Use	Description
	annotation	Annotation	Mandatory	Annotation to be removed
	foafOnlineAccount	String	Mandatory	Owner of the annotation
Returns	Type		Description	
	void			
Throws	Type		Cause	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	
	TaTooInternalError		A problem occurred during the processing of the request by some of the TaToo Core Components, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.	
	InvalidParameterValue		Operation request contains an invalid parameter value. Returns the name of the parameter with invalid value.	
	MissingParameterValue		Operation request does not provide value of some of the mandatory parameters. Returns the name of the missing parameter.	
	NotAuthorised		The user performing the operation is not authorised to do so.	

Table 4-11: Specification of the removeAnnotation Operation

4.1.3.6 Specification of the `removeAnnotations` operation

The optional `removeAnnotations` operation is responsible for the removal of more than one annotation of a resource, identified by a list of annotation URIs. This operation supports the removal of multiple annotations. If the user identified by the parameter `foafOnlineAccount` is not the owner (provider) of all provided annotations, the operation will fail and a `NotAuthorised` exception is thrown. If exactly one annotation shall be deleted, the convenience operation `removeAnnotation` can be used instead. 4.1.1.1

A request to perform the `removeAnnotations` operation shall include the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is

String `removeAnnotations(List<Annotation> annotations, String foafOnlineAccount)`
throws `MissingParameterValue, InvalidParameterValue, TaTooInternalError, NotAuthorised`

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	annotations	List<Annotation>	Mandatory	Annotations to be removed
	foafOnlineAccount	String	Mandatory	Owner of the annotation
Returns	Type		Description	
	void			
Throws	Type		Cause	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	
	TaTooInternalError		A problem occurred during the processing of the request by some of the TaToo Core Components, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.	

	InvalidParameterValue	Operation request contains an invalid parameter value. Returns the name of the parameter with invalid value.
	MissingParameterValue	Operation request does not provide value of some of the mandatory parameters. Returns the name of the missing parameter.
	NotAuthorised	The user performing the operation is not authorised to do so.

Table 4-12: Specification of the removeAnnotations Operation

4.1.3.7 Specification of the updateAnnotation operation

The optional *updateAnnotation* operation is responsible for the editing of a specific annotation associated to a resource. This operation supports the editing of a single annotation of a single resource. The provided annotation object must contain a valid resource URI as well as a valid Annotation URI, otherwise the operation will fail. Technically, editing of annotations is realised as removal and an update, thus the URI of the edited annotation will change.

If the user identified by the parameter foafOnlineAccount is not the owner (provider) of the provided annotation, the operation will fail and a NotAuthorised exception is thrown. The *updateAnnotation* operation is an optional convenience operation whose functionality is covered completely by the *updateAnnotations* operation.

A request to perform the *updateAnnotation* operation shall include the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is

String updateAnnotation (Annotation annotation) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Optional			
Receives	Name	Type	Use	Description

	annotation	Annotation	Mandatory	updated Annotation
	Locale	String	Optional	Identifier of the user locale
	foafOnlineAccount	String	Mandatory	owner of the annotation
Returns	Type		Description	
	String		Status indicator and unique id of the updated annotation	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	
	MissingParameterValue		Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	
	TaTooInternalError		A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.	
	NotAuthorised		The user performing the operation is not authorised to do so.	

Table 4-13: Specification of the updateAnnotation Operation

4.1.3.8 Specification of the `updateAnnotations` operation

The optional `updateAnnotations` operation is responsible for the editing of several annotations at once. This operation supports the editing of multiple annotations associated to a multiple resource. The provided annotation objects must contain a valid resource URI as well as a valid Annotation URI, otherwise the operation will fail. Technically, editing of annotations is realised as removal and an update, thus the URI of the edited annotation will change.

If the user identified by the parameter `foafOnlineAccount` is not the owner (provider) of all provided annotation, the operation will fail and a `NotAuthorised` exception is thrown. If exactly one annotation shall be edited, the convenience operation `updateAnnotation` can be used instead.

A request to perform the `updateAnnotations` operation shall include the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is

String `updateAnnotations` (List<Annotation> annotations) throws `InvalidParameterValue`, `MissingParameterValue`, `ResourcesNotFound`, `TaTooInternalError`

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Optional			
Receives	Name	Type	Use	Description
	annotations	List<Annotation>	Mandatory	List of updated Annotation
	Locale	String	Optional	Identifier of the user locale
	foafOnlineAccount	String	Mandatory	owner of all annotations
Returns	Type		Description	
	String		Status indicator and unique ids of the updated annotations	
Throws	Type		Cause	

	InvalidParameterValue	Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.
	MissingParameterValue	Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.
	ResourcesNotFound	The resources identified by the provided URIs could not be found.
	TaTooInternalError	A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.

Table 4-14: Specification of the updateAnnotations Operation

1.2. Evaluation Service

The Evaluation Service is a service of the TaToo system that belongs to the TaToo Public Services Tier. It provides functionalities for a community-based evaluation of the TaToo resources and resource annotations. The service's operations are supposed to be invoked by the TaToo evaluation portlet, which belongs to the TaToo portal, as well as any other client application/tool that employs the TaToo services and provides a user support for the resource and annotation evaluation.

1.2.1 Overview and Outline

This section provides an overview of the component describing the nature and the role of the component.

1.2.1.1 Role and Scope of the Service

The Evaluation Service exposes publicly accessible TaToo evaluation functionality to the respective user components.

The Evaluation Service interacts with the Evaluation Portlet from which it receives the evaluation request and the Clearinghouse service through which its sends the request to the TaToo business layer, that is, to the Evaluation Processor.

The Evaluation Service is supposed to be an interoperable and standalone Web service developed according to widely adopted standards such as XML, WSDL, and SOAP.

1.2.1.2 Service specification summary

The interface of the Evaluation Service provides the following operations:

- *addResourceEvaluation*, adds an evaluation to a resource;
- *addAnnotationEvaluation*, adds an evaluation to a resource annotation;
- *addTagEvaluation*, adds an evaluation to an annotation tag
- *getResourceEvaluations*, retrieves all evaluations of a given resource;
- *getAnnotationEvaluations*, retrieves all evaluations of a given annotation;
- *getTagEvaluations*, retrieves all evaluations of a given tag;

Please note that the *addTagEvaluation* and *getTagEvaluations* operations are optional, since such a fine granularity of evaluations is currently not needed nor considered during the discovery process.

1.2.2 Context

This section describes the relationships between the Evaluation Service and its context, including applied standards, relationships with other TaToo components and the TaToo resource model.

1.2.2.1 Relations to standards

The Evaluation Service will be a Web service, whose design follows W3C recommendations. It will be a SOAP-based Web Service exposing a WSDL interface that provides the given set of operations. A brief description of WSDL and REST can be found in chapter 3, referenced standards.

1.2.2.2 Relations to other TaToo components

The Evaluation Service is accessible via its interface from Evaluation Tools, specifically the Evaluation Portlet or other portlets and third party applications.

The service interacts with the Clearinghouse to access the TaToo Business Tier functionalities (i.e., Evaluation Processor) that are responsible to generate and store resource, annotation and tag evaluations to the TaToo RDF repository. Moreover, the service also interacts with the Clearinghouse service to retrieve evaluations of the resources, annotations, and tags.

1.2.2.3 Relations to information models

The Evaluation Service's request must contain information necessary to generate the TaToo resource, annotation and tag evaluations as they are defined in MERM. The MERM ontology provides concepts and properties that define all of the three types of the TaToo evaluations.

More information about MERM can be found in D3.1.1 - Semantic Service Environment and Framework Architecture, chapter 6.4 - Minimum Environmental Resource Model.

1.2.3 Specification of the Evaluation Service Interface

This section provides the functional specification of the Evaluation Service, including the service's interface and its operations. Table 4-2 provides a list of the service's operations.

Operation Name	Description
addResourceEvaluation	Mandatory operation that adds an evaluation to the resource.
addAnnotationEvaluation	Mandatory operation that adds an evaluation to the resource annotation.
addTagEvaluation	Optional operation that adds an evaluation to the annotation tag.
getResourceEvaluations	Mandatory operation that returns a list of the resource's evaluations.
getAnnotationEvaluations	Mandatory operation that returns a list of the annotation's evaluations.
getTagEvaluations	Optional operation that returns a list of the tag's evaluations.

Table 4-2: Tagging Service Interface

1.2.3.1 Specification of the addResourceEvaluation operation

This operation calls its counterpart *addResourceEvaluation* operation of the Clearinghouse service and forwards the evaluation request to it.

A request to perform the *addResourceEvaluation* operation includes the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the "Description" shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is:

String addResourceEvaluation (resourceURI, evaluatorID, evaluationCriterion, evaluationValue, languageID) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable
Preconditions	None

Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	resourceURI	String	Mandatory	URI of the resource
	evaluatorID	String	Mandatory	ID of the evaluator
	evaluationCriterion	String	Mandatory	Evaluation criteria
	evaluationValue	int	Mandatory	Evaluation value
languageID	String	Optional	Language	
Returns	Type		Description	
	String		Status indicator and the URI of the newly added evaluation	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	
	MissingParameterValue		Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	
	TaTooInternalError		A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.	

Table 4-21: Specification of the addResourceEvaluation Operation

1.2.3.2 Specification of the addAnnotationEvaluation operation

This operation calls its counterpart *addAnnotationEvaluation* operation of the Clearinghouse and forwards the evaluation request to it.

The signature of the operation is:

String addAnnotationEvaluation(annotationURI, evaluatorID, evaluationCriterion, evaluationValue, languageID) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	annotationURI	String	Mandatory	URI of the annotation
	evaluatorID	String	Mandatory	ID of the evaluator
	evaluationCriterion	String	Mandatory	Evaluation criteria
	evaluationValue	int	Mandatory	Evaluation value
Returns	Type		Description	
	String		Status indicator and the URI of the newly added evaluation	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	
	MissingParameterValue		Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	
TaTooInternalError		A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.		

Table 4-22: Specification of the addAnnotationEvaluation Operation

1.2.3.3 Specification of the addTagEvaluation operation

This optional operation calls its counterpart *addTagEvaluation* operation of the Clearinghouse (if available) and forwards the evaluation request to it.

The signature of the operation is:

String addTagEvaluation(annotationURI, tagProperty, tagValue, evaluatorID, evaluationCriterion, evaluationValue, languageID) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Optional			
Receives	Name	Type	Use	Description
	annotationURI	String	Mandatory	URI of the annotation
	tagProperty	String	Mandatory	Tag's property
	tagValue	String	Mandatory	Tag's value
	evaluatorID	String	Mandatory	ID of the evaluator
	evaluationCriterion	String	Mandatory	Evaluation criteria
	evaluationValue	int	Mandatory	Evaluation value
languageID	String	Optional	Language	
Returns	Type		Description	
	String		Status indicator and the URI of the newly added evaluation	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	
	MissingParameterValue		Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	

	TaTooInternalError	A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.
--	--------------------	---

Table 4-23: Specification of the addTagEvaluation Operation

1.2.3.4 Specification of the getResourceEvaluations operation

This operation calls its counterpart *getResourceEvaluations* operation of the Clearinghouse and forwards the evaluation request to it.

The signature of the operation is:

List<Evaluation> getResourceEvaluations(resourceURI) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	resourceURI	String	Mandatory	URI of the resource
Returns	Type		Description	
	List<Evaluation>		List of evaluations in a form of pairs (evaluation criterion, evaluation value)	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	
	MissingParameterValue		Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.	

	ResourcesNotFound	The resources identified by the provided URIs could not be found.
	TaTooInternalError	A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.

Table 4-24: Specification of the getResourceEvaluation Operation

1.2.3.5 Specification of the getAnnotationEvaluations operation

This operation calls its counterpart *getAnnotationEvaluations* operation of the Clearinghouse and forwards the evaluation request to it.

The signature of the operation is:

List<Evaluation> getAnnotationEvaluations (annotationURI) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	annotationURI	String	Mandatory	URI of the annotation
Returns	Type		Description	
	List<Evaluation>		List of evaluations in a form of pairs (evaluation criterion, evaluation value)	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	

	MissingParameterValue	Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.
	ResourcesNotFound	The resources identified by the provided URIs could not be found.
	TaTooInternalError	A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.

Table 4-25: Specification of the getAnnotationEvaluations Operation

1.2.3.6 Specification of the getTagEvaluations operation

This optional operation calls its counterpart *addTagEvaluations* operation of the Clearinghouse (if available) and forwards the evaluation request to it.

The signature of the operation is:

List<Evaluation> getTagEvaluations(annotationURI, tagProperty, tagValue) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Optional			
Receives	Name	Type	Use	Description
	annotationURI	String	Mandatory	URI of the annotation
	tagProperty	String	Mandatory	Tag's property
	tagValue	String	Mandatory	Tag's value
Returns	Type		Description	
	List<Evaluation>		List of evaluations in a form of pairs (evaluation criterion, evaluation value)	
Throws	Type		Cause	

	InvalidParameterValue	Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.
	MissingParameterValue	Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.
	ResourcesNotFound	The resources identified by the provided URIs could not be found.
	TaTooInternalError	A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.

Table 4-26: Specification of the getTagEvaluations Operation

1.3. Linking Service

The TaToo Linking Service is a service of the TaToo system that belongs to the TaToo Public Services Tier. It provides a public access to the TaToo Linking Data functionalities implemented by the TaToo Linking Processor that is a part of the TaToo Business Tier. The service's operations are supposed to be invoked by the TaToo Linking Portlet, which belongs to the TaToo portal, as well as any other client application/tool that employs the TaToo services and provides a user support for the TaToo Linked Data approach.

1.3.1 Overview and Outline

This section provides an overview of the component describing the nature and the role of the component.

1.3.1.1 Role and Scope of the Service

The Linking Service exposes publicly accessible TaToo Linked Data functionality to the respective user components.

The Linking Service interacts with the Linking Portlet from which it receives a request for resource linking and the Clearinghouse service through which its sends the request to the TaToo business layer, that is, to the Linking Processor.

The Linking Service is supposed to be an interoperable and standalone Web service developed according to widely adopted standards such as XML, WSDL, and SOAP.

1.3.1.2 Service specification summary

The interface of the Linking Service provides the following operations:

- `addLinks`, creates links between a given resource and a list of resources related to it. The links' type is determined by a given link property;
- `addSimilarityLinks`, creates similarity links between a given resource and a list of resources similar to it. The links' type is determined by the TaToo resource similarity property defined in MERM;
- `getLinkedResources`, retrieves all resources linked to a given resource regardless of the link types;
- `getLinkedResourcesByGivenRelationship`, retrieves all resources linked to a given resource by links of a given link type;
- `getSimilarResources`, retrieves all resources similar to a given resource;

1.3.2 Context

This section describes the relationships between the Linking Service and its context, including applied standards, relationships with other TaToo components and the TaToo resource model.

1.3.2.1 Relations to standards

The Linking Service will be a Web service, whose design follows W3C recommendations. It will be a SOAP-based Web Service exposing a WSDL interface that provides the given set of operations. A brief description of WSDL and REST can be found in chapter 3, referenced standards.

1.3.2.2 Relations to other TaToo components

The Linking Service is accessible via its interface from the TaToo Linking Portlet, Tagging Portlet and Discovery Portlets as well as any other third party applications that benefits from the TaToo Linked Data functionalities.

The service interacts with the Clearinghouse service to access the TaToo business layer functionalities (i.e., Linking Processor) that are responsible to create and store the link representations to the TaToo Knowledgebase. Moreover, the service also interacts with the Clearinghouse to retrieve information about linked resources, for which again the Clearinghouse invokes corresponding operations of the Linking Processor.

1.3.2.3 Relations to information models

The Linking Service’s request must contain information necessary to generate links between the resources such as the resources’ URIs and the link types. The TaToo Linked Data approach supports a predefined set of the link types (i.e., properties) including both upper-level link types and domain-level link types. We already discussed the link types supported in TaToo (see Section 3.5.2.3), so that it will not be presented here again.

1.3.3 Specification of the Linking Service Interface

This section provides the functional specification of the Linking Service, including the service’s interface and its operations. Table 4-11 provides a list of the service’s operations.

Operation Name	Description
addLinks	Mandatory operation that creates links between a given resource and a list of resources related to it. The links’ type is determined by a given link property.
addSimilarityLinks	Mandatory operation that creates similarity links between a given resource and a list of resources similar to it. The links’ type is determined by the TaToo resource similarity property defined in MERM.
getLinkedResources	Mandatory operation that retrieves all resources linked to a given resource regardless of the link types.
getLinkedResourcesByGivenRelationship	Mandatory operation that retrieves all resources linked to a given resource by links of a given link type.
getSimilarResources	Mandatory operation that retrieves all resources similar to a given resource.

Table 4-15. Linking Service Interface

1.3.3.1 Specification of the addLinks operation

This operation calls its counterpart *addLinks* operation of the Clearinghouse service and forwards the linking request to it.

A request to perform the *addLinks* operation includes the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is:

String addLinks (resourceURI, LinkPropertyURI, resourceURIs) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	resourceURI	String	Mandatory	URI of the resource to be linked to a given list of resources.
	LinkPropertyURI	String	Mandatory	URI of the link property that defines the type of the link to be added.
	resourceURIs	List<String>	Mandatory	List of resource URIs to be linked to the given resource, which is specified as the first argument of the operation.
Returns	Type		Description	
	String		Status indicator	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	

	MissingParameterValue	Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.
	ResourcesNotFound	The resources identified by the provided URIs could not be found.
	TaTooInternalError	A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.

Table 4-16. Specification of the addLinks operation

1.3.3.2 Specification of the addSimilarityLinks operation

This operation calls its counterpart *addSimilarityLinks* operation of the Clearinghouse service and forwards the linking request to it.

A request to perform the *addSimilarityLinks* operation includes the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is:

String addSimilarityLinks (resourceURI, resourceURIs) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description

	resourceURI	String	Mandatory	URI of the resource to be linked by the TaToo resource similarity links to a given list of resources.
	resourceURIs	List<String>	Mandatory	List of resource URIs to be linked by the TaToo resource similarity links to the given resource, which is specified as the first argument of the operation.
Returns	Type		Description	
	String		Status indicator	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	
	MissingParameterValue		Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	
	TaTooInternalError		A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.	

Table 4-17. Specification of the addSimilarityLinks operation

1.3.3.3 Specification of the getLinkedResources operation

This operation calls its counterpart *getLinkedResources* operation of the Clearinghouse service and forwards the linking request to it.

A request to perform the *getLinkedResources* operation includes the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional

[mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is:

String getLinkedResources (resourceURI) throws InvalidParameterValue, MissingParameterValue, ResourcesNotFound, TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	resourceURI	String	Mandatory	URI of the resource whose linked resources should be retrieved.
Returns	Type		Description	
	String		Status indicator	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	
	MissingParameterValue		Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.	
	ResourcesNotFound		The resources identified by the provided URIs could not be found.	
	TaTooInternalError		A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.	

Table 4-18. Specification of the getLinkedResources operation

1.3.3.4 Specification of the `getLinkedResourcesByGivenRelationship` operation

This operation calls its counterpart `getLinkedResourcesByGivenRelationship` operation of the Clearinghouse service and forwards the linking request to it.

A request to perform the `getLinkedResourcesByGivenRelationship` operation includes the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is:

String `getLinkedResourcesByGivenRelationship` (**resourceURI**, **LinkPropertyURI**)
 throws **InvalidParameterValue**, **MissingParameterValue**, **ResourcesNotFound**,
TaTooInternalError

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	resourceURI	String	Mandatory	URI of the resource whose linked resources, by the given link type, should be retrieved.
	LinkPropertyURI	String	Mandatory	URI of the link type.
Returns	Type		Description	
	String		Status indicator	
Throws	Type		Cause	
	InvalidParameterValue		Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.	
	MissingParameterValue		Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.	

	ResourcesNotFound	The resources identified by the provided URIs could not be found.
	TaTooInternalError	A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.

Table 4-19. Specification of the `getLinkedResourcesByGivenRelationship` operation

1.3.3.5 Specification of the `getSimilarResources` operation

This operation calls its counterpart *getSimilarResources* operation of the Clearinghouse service and forwards the linking request to it.

A request to perform the *getSimilarResources* operation includes the parameters listed and defined in the table below. This table also specifies the data type (Type), the obligation [optional | mandatory] (Use) and a short description (Description) of each listed parameter. Furthermore the “Description” shall state the consequences for service instances if the correspondent parameter is optional and omitted.

The signature of the operation is:

String `getSimilarResources` (resourceURI) throws `InvalidParameterValue`, `MissingParameterValue`, `ResourcesNotFound`, `TaTooInternalError`

Overrides	Not applicable			
Preconditions	None			
Post conditions	None			
Use	Mandatory			
Receives	Name	Type	Use	Description
	resourceURI	String	Mandatory	URI of the resource whose similar resources should be retrieved.
Returns	Type		Description	
	String		Status indicator	
Throws	Type		Cause	

	InvalidParameterValue	Operation request contains an invalid parameter value (e.g. out of range, malformed document, etc.). Returns the name of the parameter with invalid value.
	MissingParameterValue	Operation request either does not include a parameter value or a empty list or map. Returns the name of the missing parameter.
	ResourcesNotFound	The resources identified by the provided URIs could not be found.
	TaTooInternalError	A problem occurred during the processing of the request by a TaToo Core Component, e.g. a specific processor. Returns the internal exception thrown by the respective Core Component.

Table 4-20. Specification of the getSimilarResources operation

2. Core Components

The Tagging Processors are a set of TaToo Core Components taking part in the tagging process and supporting functionality to the Clearinghouse. In principle these components can provide a wide range of functionality. The concrete identification of these components has been achieved for the first implementation phase resulting in the identification of two components: a component for converting formats / schemas (The Schema Mapping component) and a component for supporting the filtering of tags (the Filtering Component). For the second implementation phase, the RDF Tagger Component and Evaluation Processor were identified. For the third period the Linking Processor, a component for the establishment of links between resources according to linked data principles has been defined.

2.1. Schema Mapping Component

This section defines the functional specification of the Schema Mapping Component and is based on the Template for the Specification of TaToo Components (TaToo-D313b). It includes an overview of the role of the component in the landscape of TaToo, a description of the relations of the component with its context and a specification of the objectives and functionality of the component.

2.1.1 Overview and outline

This section provides an overview of the component describing the Nature and the Role of the component.

2.1.1.1 Nature of the component

The Schema Mapping Component is a special kind of a Tagging Processor, and thus belongs to the TaToo Core Component Building Block.

The Schema Mapping Component supports the mapping from one xml-schema to another where the mapping rules are described in XSLT.

2.1.1.2 Role and scope of the component

The Schema Mapping Component is a component that can be used for the mapping of different meta-information schemas in the TaToo tagging process. It provides functionality that is related to the mapping of meta-information from a source into a target schema. We consider a schema mapping to be “the definition of an automated transformation of each instance of a data structure A into an instance of a data structure B that preserves the intended meaning of the original information” (Doerr, 2004). Thus, during the mapping process the semantics of the information are preserved.

The Schema Mapping Component is a generic component that can map between different schemas as long as:

- the information to be mapped is encoded in XML or an XML-based dialect (e.g. RDF/XML);
- the target and source schemas are described in the XML-Schema Language; and
- an XSL document containing the schema mapping rules exists.

The description of the schema mapping is required as an input, whereby the client is the responsible for providing the appropriate mapping rules. It is outside the scope of the Schema Mapping Component to automatically derive a mapping between two schemas.

The usability of the Schema Mapping Component in the TaToo tagging process depends both on the compliance of the input schema to TaToo’s Minimum Environmental Resource Model (MERM) as well as on the quality of the mapping rules. Due to the limited expressiveness of XSLT tags encoded as simple triples or property-value lists without any information about the structure of annotations are therefore are not supported by the Schema Mapping Component.

2.1.2 Context

This section describes the relationships between the component and its context, including technical requirements, other TaToo Components, etc.

2.1.2.1 Relation to technical requirements

The Schema Mapping Component addresses the following technical requirements as specified in D2.3.3 – Requirements Document V3 (TaToo-D233). The technical requirements are mapped to the concrete functional requirements specified in chapter 2.1.3.3.

Requirements ID and Name	Scope	Functional Requirement	Comments
TR.TAGGING.000 - Tagging means	S/I	1	currently no special requirements regarding schema mapping exist.

Table 2-21: Schema Mapping Component technical requirements

2.1.2.2 Relations to standards

The Schema Mapping Component uses XSLT for the schema transformation and thus supports various xml-dialects like, for example, RDF. A brief description of XSLT and RDF can be found in chapter 2.

2.1.2.3 Relations to other TaToo components

The Schema Mapping Component can be invoked by the Tagging Service through the Clearinghouse to perform a mapping (tag conversion). It can only be used to perform a transformation; it does not store annotations as ontology instances in the knowledgebase.

The Schema Mapping Component is currently not used during the tagging process, since a conversion to the RDF/XML is not necessary.

2.1.2.4 Relations to information models

The Schema Mapping Component processes information that relate to a certain meta-information schema that is encoded in an xml-based dialect as for example RDF.

2.1.3 Specification

This section includes the functional specification of the component, including objectives, functional requirements, etc.

2.1.3.1 Objectives

The objective of the Schema Mapping Component is to map meta-information (tags) from a source into a target schema. It takes meta-information (e.g. an RDF document) in XML format and a description of the mapping from the source to the target schema in the XSLT language as input and returns the meta-information in the target schema.

2.1.3.2 Target users

The Schema Mapping Component is intended to be used by the Clearinghouse to perform requests done by the TaToo Tagging Service.

2.1.3.3 Functional requirements

The functional requirement for the Schema Mapping Component is:

1. Schema Mapping: Map from one schema into another.

2.1.3.4 Interaction with other TaToo components

The Schema Mapping Component will interact with the Clearinghouse to perform requests by the Tagging Service.

2.2. Filtering Component

This section defines the functional specification of the Filtering Component and is based on the Template for the Specification of TaToo Components. It includes an overview of the role of the component in the landscape of TaToo, a description of the relations of the component with its context and a specification of the objectives and functionality of the component.

2.2.1 Overview and outline

This section provides an overview of the component describing the Nature and the Role of the component.

2.2.1.1 Nature of the component

The Filtering Component is a special kind of a Tagging Processor, and thus belongs to the TaToo Core Component Building Block.

The Filtering Component is implemented in V3 of the TaToo Framework. Its aim is to filter resources, users, and annotations by extending SPARQL queries generated by the RDF Tagger Component.

2.2.1.2 Role and scope of the component

The Filtering Component is the component responsible to provide filtering capabilities for user components.

- The Filtering Components main functionality is a Tag Filter, which is able to filter tags by specific tag values.

2.2.2 Context

This section describes the relationships between the component and its context, including technical requirements, other TaToo Components, etc.

2.2.2.1 Relation to technical requirements

The Filtering Component addresses the following technical requirements as specified in D2.3.2 – “Requirements Document V3”. The technical requirements are mapped to the concrete functional requirements specified in chapter 2.2.3.3.

Requirements ID and Name	Scope	Functional Requirement	Comments

TR.TAGGING.000 - Tagging means	S/I	1	currently no special requirements regarding filtering exist.
--------------------------------	-----	---	--

Table 2-22: Filtering Component technical requirements

2.2.2.2 Relations to standards

The Filtering Component uses RDF for the description of tags. A brief description of RDF can be found in chapter 3.

2.2.2.3 Relations to other TaToo Components

The Filtering Component can be invoked by the Tagging Service through the Clearinghouse to perform filtering. It depends on the Semantic Processor to retrieve the meta-information (tags in RDF-Format) from the knowledge base. For the specification of the Semantic Processor, please refer to D3.1.3 - Semantic Service Environment and Framework Architecture V3 (TaToo-D313), section 7.3.12 - Semantic Processor.

2.2.2.4 Relations to information models

The Filtering Component processes information that relate to a certain meta-information schema that is encoded in an xml-based dialect as for example RDF.

2.2.3 Specification

This section includes the functional specification of the component, including objectives, functional requirements, etc.

2.2.3.1 Objectives

The objective of the Filtering Component is to perform filtering operations requested by User Components. It takes meta-information (e.g. an RDF document) in XML format and user input to perform certain operations.

2.2.3.2 Target users

The Filtering Component is intended to be used by the Clearinghouse to perform requests done by the TaToo Tagging Service and by the Semantic Processor to provide tagging information for semantic processing.

2.2.3.3 Functional requirements

The functional requirement for the Filtering Component is: to filter tags by specific tag values.

2.2.3.4 Interaction with other TaToo components

The Filtering Component will interact with the TaToo Semantic Processor to retrieve meta-information and with the Clearinghouse to perform requests by the Tagging Service.

2.3. RDF Tagger Component

This section defines the functional specification of the RDF Tagger Component and is based on the Template for the Specification of TaToo Components (TaToo-D313b). It includes an overview of the role of the component in the landscape of TaToo, a description of the relations of the component with its context and a specification of the objectives and functionality of the component.

2.3.1 Overview and outline

This section provides an overview of the component describing the Nature and the Role of the component.

2.3.1.1 Nature of the component

The RDF Tagger is a special kind of a Tagging Processor, and thus belongs to the TaToo Core Component Building Block.

The RDF Tagger Component is the tagging processor responsible for storing annotations as well as resources in the TaToo knowledgebase represented by the Semantic Processor in a format that is compliant to the MERM ontology.

2.3.1.2 Role and scope of the component

The RDF Tagger Component implements the actual business logic of the Tagging Service. Thus it provides in principle the same functionality as the Tagging Service. This includes currently functionality to

- associate annotations with resources,
- retrieve annotations associated with resources
- edit annotations associated with resources
- delete annotations associated with resources

The RDF Tagger is furthermore responsible to save annotation and resource objects received from the Tagging Service (through the Clearinghouse) as instances of the MERM ontology and to store them as RDF in the knowledgebase. It performs also a validity and plausibility checks of the annotations.

2.3.2 Context

This section describes the relationships between the component and its context, including technical requirements, other TaToo Components, etc.

2.3.2.1 Relation to technical requirements

The RDF Tagger Component addresses the following technical requirements as specified in D2.3.3 – Requirements Document V3 (TaToo-D233). The technical requirements are mapped to the concrete functional requirements specified in chapter 2.1.3.3.

Requirements ID and Name	Scope	Functional Requirement	Comments
TR.TAGGING.020 Access to tags (TaToos)	S/I	1	By interfacing with the Semantic Processor.
TR.TAGGING.060 Semantic Tags	S/I	1, 2	By encoding annotations tags as instances of the MERM ontology.
TR.TAGGING.080 Ontology supported tagging	S/I	1, 2	see above.
TR.TAGGING.100 Editing of Tags (TaToos)	S/I	2	Optional in V2 depending on the concept for updating / deleting annotations.

Table 2-23: RDF Tagger Component technical requirements

2.3.2.2 Relations to standards

The RDF Tagger Component has no immediate relation to standards2.

2.3.2.3 Relations to other TaToo components

The RDF Tagger Component can be invoked by the Tagging Service through the Clearinghouse to store and retrieve tags from the knowledgebase. For this purpose, it interacts with the Semantic Processor.

2.3.2.4 Relations to information models

The RDF Tagger Component stores tags as instances of the MERM Ontology (resource and annotation instances).

2.3.3 Specification

This section includes the functional specification of the component, including objectives, functional requirements, etc.

2.3.3.1 Objectives

The objective of the RDF Tagger Component is to store and retrieve tags that are encoded as ontology instances of the MERM ontology.

2.3.3.2 Target users

The RDF Tagger Component is intended to be used by the Clearinghouse to perform requests initiated by the Tagging Service.

2.3.3.3 Functional requirements

The functional requirements for the RDF Tagger Component are:

1. Convert and store tags: store tags retrieved from the tagging service as MERM compliant ontology instances in the TaToo knowledgebase (Semantic Processor). Depending on the format / encoding of tags used by the tagging service, a conversion might be needed, e.g. from property / value lists to RDF.
2. Retrieve and convert tags: retrieve tags from the TaToo knowledgebase. Depending on the format / encoding of tags used by the tagging service, a conversion might be needed, e.g. from RDF served by the knowledgebase to property / value lists.

2.3.3.4 Interaction with other TaToo components

The RDF Tagger Component will interact with the Clearinghouse to perform requests by the Tagging Service and interact with the Semantic Repository to store and retrieve RDF triples.

It furthermore interacts with the Filtering Component, by exchanging SPARQL queries and filter conditions.

2.4. Evaluation Processor

The TaToo Evaluation Processor is a component of the TaToo business layer. It is responsible for generating the TaToo evaluations based on the evaluation information received as part of the evaluation request.

The processor receives the evaluation request from the TaToo evaluation service and generates the TaToo evaluations according to the TaToo evaluation schema. The TaToo evaluation schema is defined as a part of the MERM ontology.

The communication between the TaToo Evaluation Service and Processor is not direct. It is realised through the Clearinghouse service that acts as a single entry point to the TaToo business layer.

2.4.1 Overview and Outline

This section provides an overview of the component describing the nature and the role of the component.

2.4.1.1 Nature of the component

The Evaluation Processor is currently the default Processor for the Evaluation Service, and thus belongs to the TaToo Core Component Building Block.

The Evaluation Processor is supposed to be an interoperable and standalone Web service developed according to widely adopted standards such as XML, WSDL, and SOAP.

2.4.1.2 Role and Scope of the component

The Evaluation Processor exposes operations that are accessible to the respective user components through the Clearinghouse.

The Evaluation Processor interacts with the Clearinghouse from which it receives the evaluation request and to which it sends the response of the invoked operation. Moreover, the processor has access to the TaToo RDF repository into which it stores generated evaluations as well as retrieves requested evaluations.

The Evaluation Processor implements the actual business logic of the Evaluation Service. Thus it provides in principle the same functionality as the Evaluation Service. This includes currently the following operations:

- *addResourceEvaluation*, generates and adds an evaluation to a resource;
- *addAnnotationEvaluation*, generates and adds an evaluation to a resource annotation;
- *addTagEvaluation*, generates and adds an evaluation to an annotation tag

- *getResourceEvaluations*, retrieves all evaluations of a given resource;
- *getAnnotationEvaluations*, retrieves all evaluations of a given annotation;
- *getTagEvaluations*, retrieves all evaluations of a given tag;

2.4.2 Context

This section describes the relationships between the Evaluation Processor and its context, including applied standards, relationships with other TaToo components and the TaToo resource model.

2.4.2.1 Relation to technical requirements

The Evaluation Processor addresses the following technical requirements as specified in D2.3.3 – Requirements Document V3 (TaToo-D233). The technical requirements are mapped to the concrete functional requirements specified in chapter 2.1.3.3.

Requirements ID and Name	Scope	Functional Requirement	Comments
TR.ARCH.050 - Evaluation of resources	S/I	1, 4	By providing the possibility to add evaluations to resources.
TR.REPR.000 - Evaluation of annotations	S/I	2, 5	By providing the possibility to add evaluations to annotations.
TR.DAQ.010 - Ranking Indicators	S/I	1, 2, 3, 4, 5, 6	By providing specific evaluations for rankings.
TR.TAGGING.140 - Tagging of Tags	S/I	3,6	By providing the possibility to add evaluations to tags.

Table 2-24: Evaluation Processor technical requirements

2.4.2.2 Relations to standards

The Evaluation Processor will be a Web service, whose design follows W3C recommendations. It will be a SOAP-based Web Service exposing a WSDL interface that provides the given set of operations. A brief description of WSDL and REST can be found in chapter 2, referenced standards.

2.4.2.3 Relations to other TaToo components

The same as the other components of the TaToo business layer, the Evaluation Processor is accessible via the Clearinghouse service. The service uses the API of the TaToo RDF repository

to store the evaluations to it. It also uses the SPARQL endpoint of the repository to query the TaToo knowledge base for requested evaluations.

2.4.2.4 Relations to information models

The evaluation request sent to the Evaluation Processor must contain information necessary to generate valid TaToo evaluations (i.e., resource evaluations, annotation evaluations, and tag evaluations). This information is defined in the MERM ontology, which provides concepts and properties that define all of the three types of the TaToo evaluations.

More information about MERM can be found in D3.1.3 - Semantic Service Environment and Framework Architecture V3 (TaToo-D313), chapter 6.4 - Minimum Environmental Resource Model.

2.4.3 Specification

This section includes the functional specification of the component, including objectives, functional requirements, etc.

2.4.3.1 Objectives

The objective of the Evaluation Processor is to generate the TaToo evaluations according to the TaToo evaluation schema and to store them in the TaToo knowledgebase. The TaToo evaluation schema is defined as a part of the MERM ontology.

2.4.3.2 Target users

The Evaluation Processor is intended to be used by the Clearinghouse to perform requests initiated by the Evaluation Service.

2.4.3.3 Functional requirements

The functional requirements for the Evaluation Processor are:

1. Add resource evaluation: generate and add an evaluation to a whole resource, store the evaluation in the knowledgebase.
2. Add annotation evaluation: generate and add an evaluation to an annotation, store the evaluation in the knowledgebase.
3. Add tag evaluation: generate and add an evaluation to a single tag, store the evaluation in the knowledgebase.

4. Get resource evaluations: retrieves all evaluations of a given resource.
5. Get annotation evaluations: retrieve all evaluations of a given annotation.
6. Get tag evaluations: retrieve all evaluations of a given tag.

2.4.3.4 Interaction with other TaToo components

The Evaluation Processor will interact with the Clearinghouse to perform requests by the Evaluation Service and interact with the Semantic Repository to store and retrieve evaluations.

2.5. Linking Processor

The Linking Processor is a component of the TaToo business layer. It is responsible for generating links between TaToo resources based on a linking request and the information received from the TaToo Linking Service.

The communication between the TaToo Linking Processor and Service is not direct. It is realised through the Clearinghouse service that acts as a single entry point to the TaToo business tier.

2.5.1 Overview and Outline

This section provides an overview of the component describing the nature and the role of the component.

2.5.1.1 Nature of the component

The Linking Processor is supposed to be an interoperable and standalone Web service developed according to widely adopted standards such as XML, WSDL, and SOAP.

2.5.1.2 Role and Scope of the component

The Linking Processor exposes operations that are accessible to the respective user components through the Clearinghouse and the Linking Service.

The Linking Processor interacts with the Clearinghouse from which it receives the linking request and to which it sends the response of the invoked operation. Moreover, the processor has access to the TaToo RDF repository into which it stores generated links (i.e., RDF link representations).

The Linking Processor implements the actual business logic of the Linking Service. Accordingly, it provides the corresponding functionality to those of the Linking Service. It currently includes currently the following operations:

- `addLinks`, creates links between a given resource and a list of resources related to it. The links' type is determined by a given link property;
- `addSimilarityLinks`, creates similarity links between a given resource and a list of resources similar to it. The links' type is determined by the TaToo resource similarity property defined in MERM;
- `getLinkedResources`, retrieves all resources linked to a given resource regardless of the link types;
- `getLinkedResourcesByGivenRelationship`, retrieves all resources linked to a given resource by links of a given link type;
- `getSimilarResources`, retrieves all resources similar to a given resource;

The signatures of these operations are completely the same as the signatures of the corresponding operations of the linking service so that we do not provide them here. For the signatures of the operations we refer the reader to Sections 4.3.3.1 – 4.3.3.5.

2.5.2 Context

This section describes the relationships between the Linking Processor and its context, including applied standards, relationships with other TaToo components and the TaToo resource model.

2.5.2.1 Relations to standards

The Linking Processor will be a Web service, whose design follows W3C recommendations. It will be a SOAP-based Web Service exposing a WSDL interface that provides the given set of operations. A brief description of WSDL and REST can be found in chapter 2, referenced standards.

2.5.2.2 Relations to other TaToo components

The same as the other components of the TaToo business layer, the Linking Processor is accessible via the Clearinghouse service. The service uses the API of the TaToo RDF repository to store the RDF representations of the generated links into it. It also uses the SPARQL endpoint of the repository to query the TaToo knowledge base for available/supported link types (properties).

2.5.2.3 Relations to information models

The Linking Processor's request must contain information necessary to generate links between the resources such as the resources' URIs and the link types. The TaToo Linked Data approach supports a predefined set of the link types (i.e., properties) including both upper-level link types and domain-level link types. We already discussed the link types supported in TaToo (see Section 3.5.2.3), so that it will not be presented here again.

2.5.3 Specification

This section includes the functional specification of the component, including objectives, functional requirements, etc.

2.5.3.1 Objectives

The objective of the Linking Processor is to generate the links between TaToo resources according to the given link types and to store them in the TaToo KB.

2.5.3.2 Target users

The Linking Processor is invoked by the Clearinghouse service to perform requests initiated by the Linking Service, which in turn is invoked by the Linking Portlet.

2.5.3.3 Functional requirements

The functional requirements for the Linking Processor correspond to the operations it implements:

1. Creating RDF links that establish an explicit relationship between TaToo resources determined by specified link (relationship) types.
2. Creating RDF links that establish explicit similarity links.
3. Storing created RDF links into TaToo KB (i.e., RDF repository).
4. Retrieving all resources linked to a given TaToo resource.
5. Retrieving all resources linked to a given TaToo resource by a given link type.
6. Retrieving all resources similar to a given TaToo resource.

2.5.3.4 Interaction with other TaToo components

The Linking Processor will interact with the Clearinghouse to perform requests by the Linking Service and interact with the TaToo RDF Repository to store generated links and retrieve available link types.

3. Conclusions

The objective of this third and final version of TaToo Semantic Tagging Tools and Services Specifications was the specification of the components and the functionality to be implemented during the third iteration.

Components taking part in the provisioning of tagging functionality have been refined and enhanced. New user interfaces specifications for tagging have been provided to cover all possible aspects of user experience while annotating a resource. In particular, a Simple Tagging portlet that hides semantic information (i.e. triple statement) from the user interface for basic users that aren't familiar with semantics, an Advanced Tagging portlet for expert users, a Geotagging portlet to associate annotations to resources by selecting point or areas on a map and a Tags Editing portlet to edit or delete users' own annotations have been specified. Tagging Service interface was integrated with more operations and refined data types in order to provide the deep tagging functionality. Evaluation functionality has been consolidated with respect to second version.

Furthermore, three components were specified for adding typed links according to the Linked Data principles between resources annotated by the TaToo System.

In the scope of the final iteration of the Tagging Tools and Services functional specification, the whole Tagging chain within the TaToo Framework has been refined from the architecture point in order of view to provide: deep tagging, geo tagging, different types of tagging in relation to the type of user, enriched meta-information of resources and annotations.

4. Acknowledgements

“The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under Grant Agreement Number 247893.”

5. References

- TaToo DOW, 2011** TaToo Consortium: Annex I (Description of Work) of the TaToo Grant Agreement Nr. 247893, 2009.
- TaToo-D233, 2011** Božić B., Schimak G.: Requirements document – V2, Deliverable 2.3.3 of TaToo Project, Public Document, 2011.
- TaToo-D313, 2012** Dihé P., TaToo Semantic Service Environment and Framework Architecture – V2, Deliverable 3.1.3 of TaToo Project, Public Document, 2012.
- TaToo-D313a, 2012** Dihé P., TaToo Service Specification Template, Annex of Deliverable 3.1.3 TaToo Semantic Service Environment and Framework Architecture – V3 of TaToo Project, Public Document, 2012.
- TaToo-D313b, 2012** Dihé P., TaToo Component Specification Template, Annex of Deliverable 3.1.3 TaToo Semantic Service Environment and Framework Architecture – V3 of TaToo Project, Public Document, 2012.