Web Service: annotateService

Description: The service performs semantic annotation of textual documents provided as plain text or as XML.

Target Namespace: http://annotate.wp61.lttfll.eu

Output format: XML

Operations

1. annotate

   Annotate document (method POST)

   Input: type annotate
   - content type string the content of the input document. In case it's bigger in size, place it in the request body instead (POST)
   - lang type string the language of the document to be processed
   - pipe type int 0, 1 or 2 for pipe 0.1, pipe 0.2 or pipe 0.3 respectively

   Output: type annotateResponse
   - return type string Semantically annotated XML document
**Web Service: CoocurrenceRetrieverImpl**

**Target Namespace:**  http://coocurrence.relatedtags.wp6.ltfl.eu  
**Output format:** Every operation outputs xml by default, but can also generate json by setting the optional response parameter to json

---

### Operations

1. **getAsymmetricResourceCoocurrence**
   
   **Input:**  
   - type `getAsymmetricResourceCoocurrence`  
   - `tagString` - optional, nillable; type `string`

   **Output:**  
   - type `getAsymmetricResourceCoocurrenceResponse`  
   - `return` - optional, nillable; type `anyType`

2. **getJaccardResourceCoocurrence**
   
   **Input:**  
   - type `getJaccardResourceCoocurrence`  
   - `tagString` - optional, nillable; type `string`

   **Output:**  
   - type `getJaccardResourceCoocurrenceResponse`  
   - `return` - optional, nillable; type `anyType`

3. **getJaccardUserCoocurrence**
   
   **Input:**  
   - type `getJaccardUserCoocurrence`  
   - `tagString` - optional, nillable; type `string`  
   - `numResources` - optional; type `int`  
   - `numUsers` - optional; type `int`

   **Output:**  
   - type `getJaccardUserCoocurrenceResponse`  
   - `return` - optional, nillable; type `anyType`

4. **getUserAsymmetricCoocurrence**
   
   **Input:**  
   - type `getUserAsymmetricCoocurrence`  
   - `tagString` - optional, nillable; type `string`  
   - `numResources` - optional; type `int`  
   - `numUsers` - optional; type `int`  
   - type `getUserAsymmetricCoocurrenceResponse`
5. **retrieveTags**

**Input:** type `retrieveTags`
- `tag` - optional, nillable; type `string`

**Output:** type `retrieveTagsResponse`
- `return` - optional, nillable; type `string`
Web Service: CrawlUserAccount

Description: This service adds a user to be crawled

Target Namespace: http://useraddcrawler.wp62b.ltlfll.eu

Output format: XML

Operations

1. adduser

   Adds a new user account to be crawled

   Input: type adduser
   - crawler type string the name of the crawler - there’s a dedicated crawler type for each community (twitter, youtube, ...)
   - username type string The username for which we start crawling.
   - depth - optional, nillable; type int depth - the number of levels on which we monitor peers. The default is 2.
Web Service: DBpediaWebService

Description: Web service which provides various convenience methods for accessing DBpedia datasets

Target Namespace: http://dbpedia.relatedtags.wp6.ltfll.eu

Output format: Every operation outputs xml by default, but can also generate json by setting the optional response parameter to json

----------------------------------------

Operations

1. existsInDBpedia

Check whether a tag exists in dbpedia

   Input: type existsInDBpedia
   ■ tag - optional, nillable; type string Name of a tag

   Output: type existsInDBpediaResponse
   ■ return - optional; type boolean True when the tag exists as a lexicalisation in dbpedia

----------------------------------------

2. getAlternativeLexicalisations

Retrieve alternative lexicalisations for a tag

   Input: type getAlternativeLexicalisations
   ■ tag - optional, nillable; type string Name of a tag

   Output: type getAlternativeLexicalisationsResponse
   ■ return - optional, unbounded, nillable; type string List of lexicalisations

----------------------------------------
3. **getCommonAncestors**  
Retrieve shared dbpedia categories given two tags

**Input:** type `getCommonAncestors`  
- tag1 - optional, nillable; type `string` Name of a tag  
- tag2 - optional, nillable; type `string` Name of a tag  

**Output:** type `getCommonAncestorsResponse`  
- return - optional, unbounded, nillable; type `string` List of dbpedia category names

4. **getConceptsURIs**  
Retrieve possible concept URIs for a tag

**Input:** type `getConceptsURIs`  
- tag - optional, nillable; type `string` Name of a tag  

**Output:** type `getConceptsURIsResponse`  
- return - optional, unbounded, nillable; type `string` List of concept URIs

5. **getLT4eLConceptsURIs**  
Retrieve lt4el concept URIs given a specific tag

**Input:** type `getLT4eLConceptsURIs`  
- tag - optional, nillable; type `string` Name of a tag  

**Output:** type `getLT4eLConceptsURIsResponse`  
- return - optional, unbounded, nillable; type `string` List of concept URIs

6. **getProperties**  
Retrieve dbpedia properties which hold between two tags

**Input:** type `getProperties`  
- tag1 - optional; type `string` Name of a tag  
- tag2 - optional; type `string` Name of a tag  
- context - optional; type `string` Context identifier of an ontology  
- lang - optional; type `string` Language tag (en,nl,it...)  

**Output:** type `getPropertiesResponse`  
- return - optional, unbounded, nillable; type `string` List of property URIs

7. **getPropertiesBetweenConcepts**  
Retrieve dbpedia properties which hold between two concepts

**Input:** type `getPropertiesBetweenConcepts`  
- concept1 - optional; type `string` URI of a concept  
- concept2 - optional; type `string` URI of a concept
8. **isInLT4eLOntology**

Check whether the tag is present as a lexicalisation in the lt4el ontology

**Input:** type `isInLT4eLOntology`

- `tag` - optional, nillable; type `string` Name of a tag

**Output:** type `isInLT4eLOntologyResponse`

- `return` - optional; type `boolean` True when the tag exists as a lexicalisation in the lt4el ontology

9. **isSynonym**

Check whether two tags are synonyms

**Input:** type `isSynonym`

- `tag1` - optional, nillable; type `string` Name of a tag
- `tag2` - optional, nillable; type `string` Name of a tag

**Output:** type `isSynonymResponse`

- `return` - optional; type `boolean` True when the two tags can be resolved to concepts which are synonyms

---

This page was generated by [wsdl-viewer.xsl](http://tomi.vanek.sk)
Web Service: Definition

**Description:** A Webservice to retrieve a definition for a concept or keyword

**Target Namespace:** http://definition.ontology.wp6.ltfll.eu

**Output format:** Every operation outputs xml by default, but can also generate json by setting the optional `response` parameter to `json`

---

**Operations**

1. **getDefinition**
   
   Method to retrieve a concept given a single keyword

   **Input:** type `getDefinition`
   - `term` type `string` A keyword or some term for which a definition exists
   - `lang` type `string` Language tag (i.e. 'en','nl',....)
   - `context` type `string` Full URI to an ontology identifier

   **Output:** type `getDefinitionResponse`
   - `return` - optional, unbounded, nillable; type `ArrayOfString` Link to a readmore page + Text of the requested definition

2. **getDefinitionByConcept**
   
   Method to extract a definition given the full URI of a concept

   **Input:** type `getDefinitionByConcept`
   - `concept` type `string` Full URI to a concept of some ontology that is loaded
   - `lang` type `string` Language tag (i.e. 'en','nl',....)
   - `context` type `string` Full URI to an ontology identifier

   **Output:** type `getDefinitionByConceptResponse`
   - `return` - optional, unbounded, nillable; type `string` A link to a readmore page along with a raw string with the requested definition and the lexicalisation

---

This page was generated by wsdl-viewer.xsl (http://tomi.vanek.sk)
Web Service: DeliciousTagSearch

Description: A Webservice to retrieve resources from Delicious

Target Namespace: http://external.search.wp6.ltfl.eu

Output format: Every operation outputs xml by default, but can also generate json by setting the optional response parameter to json
Operations

1. **search**

   Search for delicious resources using keyword based search

   **Input:** type `search`
   - tag - optional, nillable; type `string` Query based on a list of space separated terms
   - resultCount - optional; type `int` The maximum number of resources that should be returned

   **Output:** type `searchResponse`
   - return - optional, unbounded, nillable; type `ArrayOfString` Search results with the title, url and tags associated with the resource

2. **searchConcept**

   Search for delicious resources using concept lexicalisations

   **Input:** type `searchConcept`
   - concept - optional, nillable; type `string` concept type string Full URI to a concept of some ontology that is loaded
   - context - optional, nillable; type `string` context type string Full URI to an ontology identifier
   - lang - optional, nillable; type `string` lang type string Language tag (i.e. 'en','nl',...)
   - resultCount - optional; type `int` The maximum number of resources that should be returned

   **Output:** type `searchConceptResponse`
   - return - optional, unbounded, nillable; type `ArrayOfString` Search results with the title, url and tags associated with the resource

Web Service: DifficultyFeedback

**Description:** A web service to manage user judgements of resources and estimate familiarity scores for new ones

**Target Namespace:** http://service.difficulty.wp6.ltfll.eu

**Output format:** Every operation outputs xml by default, but can also generate json by setting the optional `response` parameter to `json`

---

**Operations**

1. **addFeedback**

   Add a new familiarity score for a new resource for a specific user

   **Input:**
   - `addFeedback`
     - `url` - optional, nillable; type `string` The URL of the resource
     - `userURI` - optional, nillable; type `string` A full URI that matches a user (for example a SIOC:UserAccount)
     - `difficulty` - optional; type `int` An amount of familiarity of the user with the resource as expressed through a real value in the range of [0.0, 1.0]

2. **estimateDifficulty**

   Estimate the level of familiarity of a new resource for an existing resource

   **Input:**
   - `estimateDifficulty`
     - `url` - optional, nillable; type `string` The URL of the resource
     - `userURI` - optional, nillable; type `string` A full URI that matches a user (for example a SIOC:UserAccount)

   **Output:**
   - `estimateDifficultyResponse`
     - `return` - optional; type `double` An estimate of the familiarity of the resource of a the given user

3. **getConcepts**

   Acquire a list of concepts that match a certain familiarity interval

   **Input:**
   - `getConcepts`
     - `userURI` - optional, nillable; type `string` A full URI that matches a user (for example a SIOC:UserAccount)
Output:

- minimum - optional; type double Minimum level of familiarity required (between 0.0 and 1.0)
- maximum - optional; type double Maximum level of familiarity allowed (between 0.0 and 1.0)
- numWords - optional; type int Number of words to use to describe a concept/topic
Web Service: DisambiguateTaggingService

Description: Web service to disambiguate a set of terms by linking them to their respective concepts from a reference ontology

Target Namespace: http://disambiguation.wp6.ltfl.eu

Output format: Every operation outputs xml by default, but can also generate json by setting the optional response parameter to json

Operations

1. mapTermsToConcepts

   Resolves a list of terms to concepts by disambiguating them

   Input: type mapTermsToConcepts
      - queryString - optional, nillable; type string A space separated list of terms which need to be disambiguated

   Output: type mapTermsToConceptsResponse
      - return - optional, nillable; type anyType A list of term, concept pairs
Web Service: LoggingService

Description: This service logs actions performed by the user
Target Namespace: http://logging.wp62b.ltl.eu
Output format: XML

Operations

1. log

   Adds a new user account to be crawled

   Input: type log
   - ip type string the ip where the request is made
   - username - optional, nillable; type string The username of the person performing the action
   - url type string the service called together with the list of parameters
   - actiontype type string the type of request. Can be (search, mouseover, clickOnResult)
Web Service: OntologyDocumentSearch

**Description:** Web service to search a social media datasets using a domain ontology

**Target Namespace:** http://search.ontology.wp6.ltfl.lue

**Output format:** Every operation outputs xml by default, but can also generate json by setting the optional response parameter to json.
Operations

1. **getDocuments**
   Retrieve a list of documents by providing a single concept

   **Input:** type `getDocuments`
   - concept - nillable; type `string` A concept URI
   - lang - nillable; type `string` Language tag (i.e. 'en','nl',...)
   - contextURI - nillable; type `string` Full URI to an ontology identifier
   - sortingMethod - nillable; type `string` Name of the document sorting method (recent, popular, specific)

   **Output:** type `getDocumentsResponse`
   - return - optional, unbounded; type `ArrayOfArrayOfString` A list of document URI, title pairs

2. **getDocumentsByKeywords**
   Retrieve a list of documents by providing a search query

   **Input:** type `getDocumentsByKeywords`
   - keywordString - optional, nillable; type `string` A space separated list of search terms
   - languagetag - optional, nillable; type `string` Language tag (i.e. 'en','nl',...)
   - sortingmethod - optional, nillable; type `string` Name of the document sorting method (recent, popular, specific)
   - queryexpansion - optional, nillable; type `string` Whether to apply query expansion (1=enabled, 0=disabled)
   - contextString - optional, nillable; type `string` Full URI to an ontology identifier

   **Output:** type `getDocumentsByKeywordsResponse`
   - return - optional, unbounded, nillable; type `ArrayOfArrayOfString` A list of document URI, title pairs

---

This page was generated by wsdl-viewer.xsl ([http://tomi.vanek.sk](http://tomi.vanek.sk))
Web Service: OntologyManagementService

**Description:** Web service which provides various convenience methods for accessing DBpedia datasets

**Target Namespace:** http://query.utils.ontology.wp6.ltfll.eu

**Output format:** Every operation outputs xml by default, but can also generate json by setting the optional response parameter to json

---

**Operations**

1. **addNewProperty**
   
   Add a new property between two concepts
   
   **Input:**
   type addNewProperty
   - subject - optional, nillable; type string Full URI of a concept
   - predicate - optional, nillable; type string Full URI of property
   - object - optional, nillable; type string Full URI of a concept
   - context - optional, nillable; type string Full URI to an ontology identifier
   
   **Output:**
   type addNewPropertyResponse
   - return - optional; type boolean True when modification succeeds and false otherwise

2. **addNewResource**
   
   Add a new resource to an ontology
   
   **Input:**
   type addNewResource
   - subject - optional, nillable; type string Full URI of a concept
   - predicate - optional, nillable; type string Full URI of property
   - object - optional, nillable; type string Full URI of a concept
   - context - optional, nillable; type string Full URI to an ontology identifier
   
   **Output:**
   type addNewResourceResponse
   - return - optional; type boolean True when modification succeeds and false otherwise

3. **addOntology**
   
   Load an ontology
4. **exportOntology**

Export the ontology in RDF/XML

**Input:** type `exportOntology`
- context - optional, nillable; type `string` Full URI to an ontology identifier

**Output:** type `exportOntologyResponse`
- return - optional, nillable; type `string` Raw RDF/XML of ontology

5. **getAllLexicalisations**

Retrieve all the lexicalisations in some ontology

**Input:** type `getAllLexicalisations`
- context - optional, nillable; type `string` Full URI to an ontology in RDF+XML format

**Output:** type `getAllLexicalisationsResponse`
- return - optional, nillable; type `anyType` List of lexicalisations

6. **getDistance**

Calculate the number of steps needed to reach another concept from some other concept

**Input:** type `getDistance`
- concept1 - optional, nillable; type `string` Full URI of a concept
- concept2 - optional, nillable; type `string` Full URI of a concept
- context - optional, nillable; type `string` Full URI to an ontology identifier

**Output:** type `getDistanceResponse`
- return - optional; type `int` Number of steps in ontology (properties)

7. **getLexicalisationForConcept**

Retrieve all possible lexicalisations for a specific concept

**Input:** type `getLexicalisationForConcept`
- concept - optional, nillable; type `string` Full URI of a concept
- lang - optional, nillable; type `string` Language tag (i.e. 'en','nl',...)
- context - optional, nillable; type `string` Full URI to an ontology identifier

**Output:**
identifier
- only_single - optional; type boolean Whether to only return a single lexicalisation

Output: type getLexicalisationForConceptResponse
- return - optional, nillable; type anyType List of lexicalisations

8. getNumberOfClasses

Calculate the number of classes in some ontology

Input: type getNumberOfClasses
- context - optional, nillable; type string Full URI to an ontology
type getNumberOfClassesResponse
- return - optional, nillable; type string Number of classes in ontology identified by context

9. getProperties

Retrieve a list of ontological properties which hold between two tags (automatically resolve them to concepts)

Input: type getProperties
- concept1 - optional, nillable; type string Full URI of a concept
- concept2 - optional, nillable; type string Full URI of a concept
- context - optional, nillable; type string Full URI to an ontology
type getPropertiesResponse
- return - optional, nillable; type Set
- empty - optional; type boolean List of properties

10. inHierarchy

Check whether concept2 is a subclass of concept1 at some level

Input: type inHierarchy
- concept1 - optional, nillable; type string Full URI of a concept
- concept2 - optional, nillable; type string Full URI of a concept
- context - optional, nillable; type string Full URI to an ontology
type inHierarchyResponse
- return - optional, nillable; type Set
- empty - optional; type boolean
11. **loadOntology**

Retrieve and store an external ontology locally and make it available at some context

**Input:**

- `type loadOntology`
  - `context` - optional, nillable; `type string` Full URI to an ontology identifier

12. **queryStringToConcepts**

Associate a list of terms with the most likely concepts through disambiguation

**Input:**

- `type queryStringToConcepts`
  - `query` - optional, nillable; `type string`
  - `lang` - optional, nillable; `type string` Language tag (i.e. 'en','nl',...)
  - `context` - optional, nillable; `type string` Full URI to an ontology identifier

**Output:**

- `type queryStringToConceptsResponse`
  - `return` - optional, nillable; `type anyType` A list of concepts

13. **removeResource**

Remove a resource from an ontology

**Input:**

- `type removeResource`
  - `resource` - optional, nillable; `type string` Full URI of a concept
  - `context` - optional, nillable; `type string` Full URI to an ontology identifier

14. **resourceInOntology**

Check whether a resource is present in the ontology

**Input:**

- `type resourceInOntology`
  - `resource` - optional, nillable; `type string` Full URI of a concept
  - `context` - optional, nillable; `type string` Full URI to an ontology identifier

**Output:**

- `type resourceInOntologyResponse`
  - `return` - optional; `type boolean` True when the resource is present in the ontology and false otherwise

15. **sparqlQuery**

Execute a sparql query

**Input:**

- `type sparqlQuery`
  - `query` - optional, nillable; `type string`

**Output:**

- `type sparqlQueryResponse`
  - `return` - optional, nillable; `type ResultSet`
  - `resourceModel` - optional, nillable; `type Model`
    - `closed` - optional; `type boolean`
- empty - optional; type boolean
- lock - optional, nillable; type Lock
- reificationStyle - optional, nillable; type ReificationStyle
- resultVars - optional, nillable; type anyType
- rowNumber - optional; type int
**Web Service: pathfindService**

**Description:** Calculate a social path between two users

**Target Namespace:** http://pathfind.wp62b.ltfl.eu

**Output format:** XML

---

**Operations**

1. **pathfind**

   This method returns a path between two users. This is used in the graphical interface to show how two users are related.

   **Input:** type *pathfind*
   - username1 type *string* The username of a user on a social service
   - username2 type *string* The username of a user on a social service

   **Output:** type *getpathfindResponse*
   - return type *string* List of users that connects the two input users
Web Service: SearchBibsonomy

**Description:** Web service to search on Bibsonomy

**Target Namespace:** http://external.search.wp6.ltlfl.eu

**Output format:** Every operation outputs xml by default, but can also generate json by setting the optional `response` parameter to `json`

---

**Operations**

1. **search**

   Search for Bibsonomy resources using keyword based search

   **Input:** type `search`
   
   - query - optional, nillable; type `string` Query based on a list of space separated terms
   
   - resultCount - optional; type `int` The maximum number of resources that should be returned

   **Output:** type `searchResponse`
   
   - return - optional, unbounded, nillable; type `ArrayOfString` Search results with the title, url and tags associated with the resource
   
   - array - optional, unbounded, nillable; type `string`
Web Service: SearchEngineRelationRetriever

Description: A Webservice to extract relations from text

Target Namespace: http://relations.wp6.ltfl.eu

Output format: Every operation outputs xml by default, but can also generate json by setting the optional response parameter to json

Operations

1. SentenceStringsToWords
   Covert a sentence to a list of words

   Input: type SentenceStringsToWords
   - listSents - optional, nillable; type anyType A list of sentences

   Output: type SentenceStringsToWordsResponse
   - return - optional, nillable; type anyType A list of lists of words

2. getFragments
   Retrieve a list of sentence fragments

   Input: type getFragments
   - term1 - optional, nillable; type string A term that should occur in a sentence
   - term2 - optional, nillable; type string A term that should occur in a sentence
   - resultCount - optional; type int The number of sentence fragments that should be returned

   Output: type getFragmentsResponse
   - return - optional, unbounded, nillable; type string A list of sentence fragments that contain both specified input terms

3. getParsedSentences
   Retrieve a list of sentences where for each sentence its dependency tree is generated using the Stanford NLP parser

   Input: type getParsedSentences
   - term1 - optional, nillable; type string A term that should occur
4. **getSentences**

Retrieve a list of sentences from a search engine

**Input:** type `getSentences`
- `term1` - optional, nillable; type `string` A term that should occur in a sentence
- `term2` - optional, nillable; type `string` A term that should occur in a sentence
- `resultCount` - optional; type `int` The number of parsed sentences that should be returned

**Output:** type `getSentencesResponse`
- `return` - optional, nillable; type `String` An XML representation of the dependency tree for that sentence

5. **getText**

Retrieve the text from a specific URL and strip it of HTML-markup if present

**Input:** type `getText`
- `sourceUrlString` - optional, nillable; type `string` The URL to a text or HTML-document

**Output:** type `getTextResponse`
- `return` - optional, nillable; type `String` The text that the documents contains stripped of HTML-markup
Web Service: searchService

**Description:** The service exposes functionalities for searching in document annotations based on lexical terms and ontology concept URIs. The query expressions can be enriched by domain ontology expansion.

**Target Namespace:** http://semanticsearch.wp61.lifl.eu

**Output format:** XML

---

**Operations**

1. **resource**

   Retrieve a document by URI

   **Input:**
   - **id** type `string` the URI of the desired document
   - **format** type `string` (rdf|xml) retrieves either the original XML document or its annotation description in RDF
   - **compress** type `string` (y|n) - whether to compress the data

   **Output:**
   - **return** type `string` RDF or XML representation of a document

2. **search**

   Semantic Search

   **Input:**
   - **q** type `string` The query expression, whitespace separated set of terms interpreted as conjuncts (terms containing whitespaces should be placed in double quotes).
   - **lang** - optional, nullable; type `string` the language of the input terms. Defaults to English
   - **noexpand** - optional, nullable; type `boolean` if provided with value 'true' or 'y', query expansion is disabled and only exact matches are returned
   - **out** - optional, nullable; type `boolean` formatting of the result. 'html' value produces browsable web page while any other value results in a simple list of URIs (targeted to machine interaction)

   **Output:**
3. **snippet**

Generate a snippet from a document high-lighting certain annotations

**Input:** type `snippet`
- id type `string` document URI
- concepts type `string` annotations to be highlighted
- limit type `int` truncate the result if too long (number of hits)

**Output:** type `snippetResponse`
- return type `string` XML representation of the snippet

---

4. **stats**

Generate a snippet from a document high-lighting certain annotations

**Input:** type `stats`
- ids type `string` list of URIs of the documents

**Output:** type `statsResponse`
- return type `string` XML representation of the statistics - a table with format of the rows: concept URI, Term, Number of occurrences

---

This page was generated by wsdl-viewer.xsl ([http://tomi.vanek.sk](http://tomi.vanek.sk))
Web Service: SearchSlideshare

**Description:** Web service to search for resources on slideshare

**Target Namespace:** http://external.search.wp6.ltfll.eu

**Output format:** Every operation outputs xml by default, but can also generate json by setting the optional `response` parameter to `json`

---

**Operations**

1. **search**

   Search for Slideshare resources using keyword based search

   **Input:** type `search`
   - `query` - optional, nillable; type `string` A list of space separated terms
   - `resultCount` - optional; type `int` The maximum number of resources that should be returned

   **Output:** type `searchResponse`
   - `return` - optional, unbounded, nillable; type `ArrayOfString` Search results with the title, url and tags associated with the resource
     - `array` - optional, unbounded, nillable; type `string`
Web Service: SearchYoutube

Description: Web service to search Youtube resources using tags
Target Namespace: http://external.search.wp6.ltfl.eu
Output format: Every operation outputs xml by default, but can also generate json by setting the optional response parameter to json

Operations

1. search

Search for Youtube resources using keyword based search

Input: type search
- query - optional, nillable; type string A space separated list of terms
- resultCount - optional; type int Maximum number of search results to return

Output: type searchResponse
- return - optional, unbounded, nillable; type ArrayOfString
  - array - optional, unbounded, nillable; type string
    Search results with the title, url and tags associated with the resource

2. smartSearch

Search for Youtube resources using a keyword based search combined with concept filtering

Input: type smartSearch
- query - optional, nillable; type string A space separated list of terms
- context - optional, nillable; type string Full URI to an ontology identifier
- lang - optional, nillable; type string Language tag (i.e. 'en', 'nl',...)
- resultCount - optional; type int Maximum number of search results to return

Output:
type `smartSearchResponse`
- return - optional, unbounded, nillable; type `ArrayOfString` Search results with the title, url and tags associated with the resource
  - array - optional, unbounded, nillable; type `string`
Web Service:

socialSearchRecommendationService

  Description: Search and recommend resources based on social data
  Target Namespace: http://socialSearchRecommendation.wp62b.ltfl.eu
  Output format: XML

Operations

1. socialRecommendation

  Uses the recommendation algorithm to offer interesting resources that would be of interest for the user

  Input: type socialRecommendation
  - username type string the username with which the user is authenticated in the system
  - password type string the password with which the user is authenticated in the system

  Output: type socialRecommendationResponse
  - return type string list of resources in an xml format

2. socialSearch

  returns a list of resources or users computed using the FolkRank algorithm

  Input: type socialSearch
  - type string represents the type of resource searched for. For resource search the type should be "resource". When searching for users this should be "users"
  - format type string represents the format in which the response is returned to the user. Its values can be "graphML" or "XMLList". If the "graphML" value is used the service will return an XML document containing the description of users and resources relevant for the query and also of the relations between them. This document will be used by the visualization widget. In case the "XMLList" parameter is submitted the result is an XML document containing a list of users/resources ordered according their relevance for the search query.
  - username type string represents the username of the user
who performs the search
- password type string the password associated with the username
- numberOfResults type string represents the maximum number of results needed to be returned.
- listOfTags type string the tags searched in the query

Output: type socialSearchResponse
- return type string Either a list of resources formatted as a graphML file or a list of resources formatted as an XMLList that contains a list of resources, the users that contributed and the tags used to tag it.
Web Service: usersimilarityService

**Description:** Compute the similarity of two users

**Target Namespace:** http://usersimilarity.wp62b.ltfl.eu

**Output format:** XML

---

**Operations**

1. **usersimilarity**

   computes the similarity of two users

   **Input:** type `userSimilarity`
   - `username1 type string` The username of a user on a social service
   - `username2 type string` The username of a user on a social service

   **Output:** type `getUserSimilarityResponse`
   - return type `float` float between 0 and 1 corresponding to the user similarity

---

This page was generated by wsdl-viewer.xsl ([http://tomi.vanek.sk](http://tomi.vanek.sk))
Web Service: VisualizationService

Target Namespace: http://visualization.wp6.ltfll.eu

Output format: Every operation outputs xml by default, but can also generate json by setting the optional response parameter to json

Operations

1. createGraph

Create a graph given a seed concept

Input: type createGraph
- context - optional, nillable; type string Full URI to an ontology identifier
- seedConcept - optional, nillable; type string Seed concept from which the graph should be extended
- lang - optional, nillable; type string Language tag (i.e. 'en', 'nl', ...)
- filterWeakPredicates - optional, nillable; type string A boolean that indicates whether ltfll:related relations in the ontology should be filtered or not
- depth - optional; type int How many steps away from the seed concepts should the ontology be expanded

Output: type createGraphResponse
- return - optional, nillable; type string a JSON fragment which describes the generated graph

2. createGraphFromDifficultWords

Create a graph that includes concepts with low familiarity scores

Input: type createGraphFromDifficultWords
- difficultWords - optional, nillable; type string A space separated list of terms to include in the graph with low familiarity scores
- difficultConcepts - optional, nillable; type string Comma separated list of concepts to include in the graph with low familiarity scores
- context - optional, nillable; type string Full URI to an ontology identifier
- lang - optional, nillable; type string Language tag (i.e. 'en', 'nl', ...)
- filterWeakPredicates - optional, nillable; type string A boolean
that indicates whether ltfll:related relations in the ontology should be filtered or not
- depth - optional; type int How many steps away from the seed concepts should the ontology be expanded
- minimum - optional; type double Minimum level of familiarity required (between 0.0 and 1.0)
- maximum - optional; type double Maximum level of familiarity allowed (between 0.0 and 1.0)
- userURI - optional, nillable; type string A full URI that matches a user (for example a SIOC:UserAccount)
- numWords - optional; type int Number of words to use to describe a concept/topic

Output: type createGraphFromDifficultWordsResponse
- return - optional, nillable; type string a JSON fragment which describes the generated graph

3. createGraphFromQuery
Create a graph from a space separated list of terms

Input: type createGraphFromQuery
- queryString - optional, nillable; type string A list of space separated terms
- context - optional, nillable; type string Full URI to an ontology identifier
- lang - optional, nillable; type string Language tag (i.e. 'en','nl',...)
- filterWeakPredicates - optional, nillable; type string A boolean that indicates whether ltfll:related relations in the ontology should be filtered or not
- depth - optional; type int How many steps away from the seed concepts should the ontology be expanded

Output: type createGraphFromQueryResponse
- return - optional, nillable; type string a JSON fragment which describes the generated graph

This page was generated by wsdl-viewer.xsl (http://tomi.vanek.sk)