KP-LAB Knowledge Practices Laboratory – Specifications for the Knowledge Matchmaker (V.2.0), the Knowledge Synthesizer (V.1.0) and the Analytical and Knowledge Mining Services (V.1.0)

Jan Paralic, Karol Furdik, Peter Bednar, Frantisek Babic, Jozeph Wagner, Marek Schmidt, Pavel Smrz, Nicolas Spyratos, Ekaterina Simonenko, Vassilis Christophides, et al.

27490

KP-LAB

Knowledge Practices Laboratory

Integrated Project

Information Society Technologies

D5.6: Specifications for the Knowledge Matchmaker (V.2.0), the Knowledge Synthesizer (V.1.0) and the Analytical and Knowledge Mining Services (V.1.0)

Due date of deliverable: **31/01/09**
Actual submission date:

Start date of project: 1.2.2006 Duration: 60 Months

Organisation legal name of lead contractor for this deliverable:
Technical University of Kosice (TUK)

Final Draft

Project co-funded by the European Commission within the Sixth Framework Programme (2002-2006)		
Dissemination Level		
PU	Public	✓
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

Contributor(s):	Jan Paralic	TUK	Jan.Paralic@tuke.sk
	Karol Furdik	TUK	kfurdik@stonline.sk
	Peter Bednar	TUK	Peter.Bednar@tuke.sk
	Frantisek Babic	TUK	frantisek.babic@tuke.sk
	Jozef Wagner	TUK	jozef.wagner@gmail.com
	Marek Schmidt	UEP	fregaham@gmail.com
	Pavel Smrz	UEP	Pavel.Smrz@gmail.com
	Nicolas Spyratos	LRI-ORSAY	Nicolas.Spyratos@lri.fr
	Ekaterina Simonenko	LRI-ORSAY	Ekaterina.Simonenko@lri.fr
	Vassilis Christophides	ICS-FORTH	christop@ics.forth.gr
	Giorgos Flouris	ICS-FORTH	fgeo@ics.forth.gr
	Dimitris Kotzinos	ICS-FORTH	kotzino@ics.forth.gr
	Yannis Rousakis	ICS-FORTH	rousakis@ics.forth.gr
Editor(s):	Karol Furdik	TUK	kfurdik@stonline.sk
Partner(s):	TUK, UEP, LRI-ORSAY, ICS-FORTH		
Work Package:	WP5 – Semantic Web Knowledge Middleware		
Nature of the deliverable:	Report		
Internal reviewers:	Merja Bauters (METROPOLIA), Marina Scapola (DIBE)		
Review documentation:	*http://www.kp-lab.org/intranet/work-packages/wp5/result/deliverable-5.6/*		

Version history

Version	Date	Editors	Description
0.1	30/10/2008	Karol Furdik,	Initialization document, tasks and responsibilities.
0.11	09/11/2008	Frantisek Babic, Jozef Wagner	Content related to the Analytical and Knowledge Mining Services.
0.12	27/11/2008	Giorgos Flouris	Content related to the Knowledge Synthesizer.
0.13	05/12/2008	Marek Schmidt, Pavel Smrz	Content related to the Information extraction services part.
0.2	09/12/2008	Karol Furdik	Content related to the Comprehension service. Integration of all the inputs.
0.21	15/12/2008	Peter Bednar	Context related to the "search similar" service.
0.22	16/12/2008	Frantisek Babic, Jozef Wagner	Updates of the Analytical and Knowledge Mining Services part.
0.23	17/12/2008	Jan Paralic	Review, comments, rewriting some parts of document.
0.24	17/12/2008	Ekaterina Simonenko	Input to the Analysis of logs service (as a part of AKMS).
0.25	19/12/2008	Nicolas Spyratos	Updates to the Analysis of logs service (as a part of AKMS).
0.3	19/12/2008	Karol Furdik	Integration of inputs, preparation of a consolidated draft.
0.31	05/01/2009	Frantisek Babic	Input to the Analytical and Knowledge Mining Services part.
0.32	06/01/2009	Marek Schmidt, Pavel Smrz	Input to the Information extraction services part.
0.33	11/01/2009	Karol Furdik	Integration of inputs, updates of Comprehension service description.
0.34	12/01/2009	Giorgos Flouris	Upgrade of specification part of the Synthesizer and some minor changes
0.35	12/01/2009	Jozef Wagner	Upgrade of the Recommendation services part
0.4	13/01/2009	Jan Paralic	Consolidating the document based on partners' inputs, completing some missing parts
0.41	16/01/2009	Karol Furdik	Checking, minor updates of the Motivating scenarios part.
0.5	17/01/2009	Jan Paralic	Final draft, ready for internal review
0.6	03/02/2009	Karol Furdik, Jan Paralic	Revised parts about Knowledge Matchmaker and AKMS, reflecting remarks made by reviewers

0.7	05/02/2009	Giorgos Flouris	Revised parts about Knowledge Synthesizer, reflecting remarks made by reviewers
1.0	05/02/2009	Jan Paralic	Final version, covering all revisions made by partners after internal review

Executive summary

This deliverable presents specifications of three components responsible for advanced manipulation with the knowledge stored in the KP-Lab Semantic Web Knowledge Middleware (SWKM). It starts with motivating scenarios defined within various Working Knots (WKs), extracting relevant functional requirements and mapping them on the high-level requirements, of particular driving objectives and user tasks (described in deliverable [D2.4]).

The first component is Knowledge Matchmaker (V2.0), which utilizes various text mining, information extraction, and heuristic methods for advanced access to and manipulation with shared knowledge artefacts according to the explicit meaning of artefacts expressed by their textual content, as well as metadata, including semantic tagging. This second version presents a set of completely new services supporting miscellaneous functionalities such as support for semantic tagging process, search for similar artefacts, information extraction capabilities, as well as recommendation services.

Next two components are completely new. The Knowledge Synthesizer (V1.0) can be used to combine information found in multiple sources; this feature is necessary to allow automated merging of the conceptualizations modeled in independently edited conceptualizations.

The Analytical and Knowledge Mining Services (V1.0) provide means for analyzing participation and activities within past or running knowledge creation processes, as well as for support of knowledge evolution analysis (e.g. via identification of critical patterns in selected knowledge creation processes).

Table of Contents

TABLE OF CONTENTS **6**

1 INTRODUCTION **7**

2 REQUIREMENTS **9**

2.1 MOTIVATING SCENARIOS 9
2.1.1 Collaborative work with knowledge artefacts *10*
2.1.2 Advanced notification and recommendation *13*
2.1.3 Merging of knowledge artefacts (conceptualizations) *13*
2.1.4 Analysis of knowledge creation processes *14*
2.2 RELATION OF HIGH-LEVEL REQUIREMENTS, DRIVING OBJECTIVES AND USER TASKS TO THE NEEDS OF THE SERVICES PROVIDED BY SWKM 16
2.2.1 Collaborative maintenance of semantically tagged artefacts *16*
2.2.2 Notifications and recommendations during the collaborative knowledge creation processes *17*
2.2.3 Merging of multiple knowledge artefacts (conceptualizations) *18*
2.2.4 Analyzing the knowledge creation processes *19*
2.2.5 Summary of the Requirements *20*

3 FUNCTIONAL AND ARCHITECTURAL DESIGN **21**

3.1 KNOWLEDGE MATCHMAKER (V.2.0) 21
3.1.1 Comprehension Service *21*
3.1.2 Information Extraction Service *25*
3.1.3 Recommendation Service *26*
3.2 KNOWLEDGE SYNTHESIZER (V.1.0) 27
3.2.1 The Process of Merging in the Related Literature *30*
3.3 ANALYTICAL AND KNOWLEDGE MINING SERVICES (V.1.0) 32
3.3.1 Services for support of participation and activity analysis of the knowledge creation processes *33*
3.3.2 Tools for support of visual analysis of logs *34*
3.3.3 Services for support of knowledge evolution analysis *37*

4 CONCLUSIONS AND FUTURE WORK **41**

BIBLIOGRAPHY **42**

1 Introduction

Knowledge Matchmaker (V2.0), Knowledge Synthesizer (V1.0), and Analytical and Knowledge Mining Services (V1.0) are the middleware modules of KP-Lab system, proposed to support advanced collaborative and semantic-based manipulation of shared knowledge artefacts that should enable the emerging of knowledge practices. These modules provide a set of services that extend the basic SWKM functionality for accessing and manipulating the ontology data towards the utilization within the end-user KP-Lab tools.

Knowledge Matchmaker (V2.0) contains a set of services that support a collaborative work within a group of participants involved in a knowledge creation process. By utilizing the text mining, information extraction, and various heuristic methods, it enables advanced access to and manipulation of shared knowledge artefacts according to the explicit meaning of the artefacts expressed by their textual content, as well as metadata, including semantic tags. The coordination of collaborative work and social awareness between the group members are supported by advanced notification and recommendation services, enabling automatic notification on modifications of artefacts and/or particular actions provided by the participants. Particular services supporting these functions were identified for the Knowledge Matchmaker (V2.0) as follows:

- The *Comprehension Service* provides advanced classification, analysis, and consistency check of semantically tagged knowledge artefacts. The heuristic rules and frequency-based text analysis mechanisms are employed to support meaningful collaborative work with shared knowledge artefacts as aiding to acquire consistency in semantic tags, search and retrieval facilities, and collaborative maintenance of various source materials and produced artefacts.
- The *Information Extraction Service* enables to support the user with semantically tagged artefacts enabling her to search for related ones based on these semantic descriptions. It also identifies entities and relations directly in the content of the artefacts and allows reviewing suggested annotations produced by the service and improving the extraction models based on the user feedback.
- The *Recommendation Service* offers a possibility for a user, as a member of a group aiming at collaborative knowledge creation, to subscribe for updates concerning a specific knowledge artefact and/or for a specific periodicity of notification.

Knowledge Synthesizer (V1.0) provides capabilities to analyze, integrate, and merge the conceptualizations of knowledge artefacts expressed by the visual modeling languages of different domains. The analysis of different models and explanations that are produced by users includes, for example, uncovering similar (and dissimilar) explanations or models, and identifying groups (or people) whose theories, models, or explanations are very different (or very similar) to each other.

Analytical and Knowledge Mining Services (AKMS) provide supporting services for analysis of knowledge creation processes in two different ways. First, AKMS provides services for analyzing participation and activities within past or ongoing

processes. Secondly, AKMS supports also the knowledge evolution analysis providing e.g. means for manual identification of critical patterns in knowledge creation processes. Once defined, the AKMS serves with the possibility of proactive identification of known critical patterns in selected ongoing processes.

The specification of these middleware modules and their respective services is based on the analysis of requirements formulated in [D2.4] cooperatively by pedagogical and professional experts and researchers as well as technical developers of the KP-Lab project. The motivating scenarios and identified high-level requirements are presented in the following chapter. Based on the analysis of the required functionality, a design of the inner architecture, functionality, and interface of particular services for the middleware modules is proposed and specified in detail in the chapter 3. Finally, conclusions and suggestions for future work, including the proposals for implementation and integration of the designed services into the KP-Lab end-user tools, is presented in the chapter 4.

2 Requirements

This chapter contains a description of the motivating scenarios and high-level requirements that demonstrate the need and role of the respective SWKM modules and their services from an end-user perspective, as well as an employment and functional integration of these services into the whole KP-Lab system.

The chapter starts with descriptions of motivating scenarios presented in the following section 2.1 that are based on the outcomes of the Working Knots (WKs), being a platform for collaborative design and discussion space that elicits and integrates the requirements of pedagogical users with the approaches and solutions proposed by technical partners of the KP-Lab project. Outline of the identified motivating scenarios and the relevant WKs where the specifications of scenarios were formed, are presented in Table 1.

Motivating scenario	**Working Knots referenced**
1. Collaborative work with artefacts	WK1 Shared Space and Common Tools
	WK2 Management and Analysis of Complex Knowledge Structures
	WK5 Document Centered Collaboration
2. Advanced notification and recommendation support	WK1 Shared Space and Common Tools
	WK6 Change Laboratory
3. Merging of knowledge artefacts (conceptualizations)	WK2 Management and Analysis of Complex Knowledge Structures
4. Analysis of processes in a Knowledge practices environment	WK3 Process Management and Analysis
	WK6 Change Laboratory

Table 1. *Motivating scenarios and respective Working knots used for the scenarios' specification*

As next, in section 2.2 particular functional requirements, which resulted from motivating scenarios, are extracted and categorized into smaller groups corresponding to new SWKM modules. Moreover, there is also presented a mapping of identified required functionalities to particular user tasks (UT), driving objectives (DO), and high-level requirements (HLR), as they were described and elaborated in the D2.4 deliverable [D2.4].

2.1 Motivating scenarios

The motivating scenarios identified as outcomes of the respective WKs and outlined in Table 1 are presented and described in more details in this section. The support for collaborative work with knowledge artefacts is proposed by means of functions for checking and maintenance of semantic tagging, semantic-based retrieval, advanced clustering and classification. The collaborative knowledge creation is supported by notification and recommendation functionality allowing personal, punctual, and

scheduled notification. Merging of knowledge artefacts proposes methods to support the model management, especially by means of parallel editing of artefact's conceptualizations. Finally, the functions for analysis of knowledge creation processes identify several information resources for monitoring and analysis of collaborative knowledge practices on a global level.

2.1.1 Collaborative work with knowledge artefacts

The scenarios for collaborative manipulation with knowledge artefacts aiming at creation of innovative knowledge practices were elaborated within the WK Management and Analysis of Complex Knowledge Structures, namely in the description of semantic tagging[1]. Actual implementation of these scenarios into the KP-Lab user tools was discussed in the WK Shared Space and Common Tools. The Semantic Tagging and Tag Vocabulary Editor tools are referenced in [D6.6] as main user-side tools dedicated to provide this functionality for users. In addition, the specification of usage scenarios for the Semantic wiki tool [SW_SUS], provided within the WK Document Centered Collaboration, inherently contains explicit as well as implicit references to the collaborative maintenance of shared knowledge artefacts by means of semantic tagging and manipulating tag vocabularies.

Knowledge practices environment enables users to work on shared knowledge objects in one place. It shall allow the participants of a knowledge creation process to perceive and handle shared materials, knowledge representations and respective processes in an integrated way, supporting a process of new and innovative knowledge creation.

The semantic tagging [D5.3], [Bauters07] is a method that enables to organize shared objects according to their explicitly expressed meaning and allows accessing the artefacts in mutual semantic relations. The meaning of artefacts is represented by a set of links, associations, with the concepts of a common and shared vocabulary – a simple light-weight domain ontology (e.g. vocabulary or taxonomy) stored and managed in the SWKM. This representation enables the users to share and access the artefacts semantically, via semantic search (provided in faceted form within the user interface) and similarity search (clustering). A combination of semantic tags with analysis of textual content of the artefacts allows users to classify the knowledge artefacts into pre-defined or ad-hoc created categories, identifying "semantically similar" artefacts, grouping of artefacts with the similar meaning into clusters, and extracting specific semantic relations between the artefacts.

The process of semantic tagging, if performed manually, requires additional efforts on the side of participants of knowledge creation processes. To increase the usefulness of tagging, the end-user KP-Lab tools (by invoking the services of the Knowledge Matchmaker) will provide helpful suggestions and tag recommendations based on analysis of artefact's content and/or analysis of the semantic tags of a given tag vocabulary. The tagging support is designed in a subtle manner, not to irritate the

[1] http://www.kp-lab.org/intranet/design-teams/wk-management-and-analysis-of-complex-knowledge-structures/semantic-tagging/annotating-knowledge-objects-with-semantic-tags/

users during the tagging process. Users can invoke the Knowledge Manager services and decide if the provided recommendations are useful and acceptable or not.

The support for semantic tagging procedure, as it was designed in previous version of the Knowledge Matchmaker [D5.3], required an extensive set of training data and that is why it was difficult to maintain and adapt properly on the changes of the space of shared knowledge artefacts. Current design is focused on the easy, transparent, and automatic support for semantic tagging (by advanced clustering and classification methods) as well as on the exploitation of the tagging in a Knowledge practices environment for better accessing, retrieving, and grouping the shared artefacts.

Proposed supporting functionality for the process of semantic tagging assumes the prerequisites as existence of a common vocabulary of tags and a full-text indexing of the textual content of shared artefacts (which is already supported by the existing search services [Search]). The following basic operations can then be identified to support the collaborative semantic tagging and semantic maintenance of shared artefacts:

- Assistance in the process of semantic tagging. Recommendation of tags that semantically match with the textual content of the artefacts.
- Consistency check of the semantic tags. Evaluation of homogeneity, similarities, and differences between the semantic tags inserted by different users.
- Maintenance of the tag vocabulary. Proposal for adding / modification / removal of a semantic tag from/to vocabulary, which can then be accepted or cancelled by users' choice.

The following partial scenarios can be considered for each of the above mentioned operations.

Assistance during the process of semantic tagging. The users can obtain recommendations of tags suitable for semantic description of a particular artefact (despite the artefact is already tagged or not). Knowledge Matchmaker provides services to analyze the textual content of the artefact, identify key terms in the text, suggest corresponding tags, and analyze the similarities of already tagged artefacts. Using information extraction capabilities, the service automatically recommends tag synonyms used by other users thus making the tagging consistent between different users and supporting collaboration. As mentioned before, this recommendation functionality is designed in a subtle way and is purely optional. Users do not have to accept any of the recommendations provided by the service, even invocation of the assistance service in the user-side tool during the tagging process is not obligatory. However, accepting the (some of) recommendations may help to keep the structure of semantic tags consistent, minimizing deviations of the meaning of particular semantic tags, and consequently ease the manipulation with knowledge artefacts as meaningful semantically described pieces of information.

Recommendations can be based on several different vocabularies / tag structures that are provided, for example, in multiple commenting threads [Bauters07]. Finally, users can be notified about new / modified / newly tagged artefacts related to his/her interests (specified by tags, similarity of tags, or a similarity of content from specific artefacts). This functionality can be useful to keep the consistency of tagging in a

collaborative environment, and can also help if the set of tagged artefacts is large and difficult to maintain manually.

Checking consistency of semantic tags. By re-classifying the set of already tagged knowledge artefacts, users can check whether the artefacts are classified homogeneously and consistently. Knowledge Matchmaker can compare and quantify a similarity of tags inserted by different users [Bauters07] and can provide recommendations for changes in tags for particular artefact, for example a suggestion to add or remove some tags for given artefact. This functionality can help to keep the semantic tagging consistent in a single domain, especially if several users perform the tagging in a collaborative environment.

Proposal of changes for semantic tag vocabulary. Based on the analysis of structural correlations between the tags of a given vocabulary, similarities of existing tagging and textual content of the artefacts, and the availability of full-text indexes, the KP-Lab system can propose keywords to be added into the vocabulary of semantic tags, or possibly also removed from the vocabulary [LocSca08]. Users can immediately see a temporary preview of the artefacts distributed according to the updated semantic tags. This feature can help to keep the consistency of tags; moreover, it provides a quick overview of main topics covered by the textual content of the knowledge artefacts. It can also be used as an initial step to create the vocabulary of tags from scratch [LocSca08].

Besides the semantic tagging, Knowledge Matchmaker can provide a support for extended searching and/or grouping of search results according to the semantic and textual properties of the knowledge artefacts. KP-Lab Search service [Search] already provides support for combining of the semantic search and free text search and user can arbitrary combine various search strategies based on the faceted search interface [D6.6, Search tool]. These search strategies will be extended with the following mechanisms provided by the Knowledge Matchmaker:

"Search similar" functionality. Users can select one or more knowledge objects and find similar knowledge objects. Similarity can be based on a) textual content b) metadata properties (i.e. author, creation date, etc.) or c) semantic annotations. This service will return the list of knowledge objects together with the similarity scores used to sort the search result.

Extension of search results, query expansion. Users can refine the search results by selecting some of the semantic tags (e.g. as attributes of some of the retrieved artefacts) and invoking a Knowledge Matchmaker service for re-classification and search extension. This service will then provide a set of artefacts which are not actually tagged by the selected tags, but which should belong to these tags according to the textual content.

Both above-mentioned search functions can increase the quality (recall) of the retrieval procedure, since they combine principles of semantic search and textual analysis.

Note that Search architecture is divided to user interface integrated in the end-user applications and back-up indexing and search services. This deliverable describes extensions to the search services only and the corresponding user interface changes will be specified in the ongoing versions of the [D6.6 Search Tool] deliverable.

2.1.2 Advanced notification and recommendation

When users work asynchronously on (often complex) knowledge creation processes, proper means for subscription and notification about relevant events in the running processes is one of the fundamental requirements. It may become more and more difficult to keep track of what is going on and to decide which occurrences are relevant to one's own activities. Consequently, users should have the possibility to decide on a personalized notification scheme, which allows them to specify the events they want to be informed about, together with means and timing of notification, by user-defined subscription criteria.

Notification mechanism designed and implemented within [D5.3] and [D5.4] respectively, was based on topics only. Various discussions and experiences of users within the working knots (WK Shared Space and Common Tools, WK Process Management and Analysis, and WK Change Laboratory mainly) lead to additional requirements, which have been described in form of use cases in the [D6.6] deliverable published very recently (for details see [D6.6-SSpUMT]).

Based on these requirements user should be able to select any type of content items (knowledge artefacts) within her/his actual content view and ask for being notified about changes relevant to them (so called *Punctual notification*). Moreover, user will be able to select also interval in which the notifications can be delivered to her/him in digested form, i.e. daily, weekly or monthly (so called *Scheduled notification*).

In both types of notifications the delivery channel may be either e-mail, RSS feed, mobile phone or a combination of them, as user prefers. All these settings are specified in the user's preferences, which are stored and can be changed whenever user needs [D6.6-SSpUMT].

In DoW3.1 there was also envisaged Community Formation service, which was based on the original topic-based notification idea. Since the requirements and needs for such a kind of service were not presented in any of the Working Knots, this service will not be designed and implemented in the upcoming version of Knowledge Matchmaker. It can be reconsidered later on, if such requirement appears in some Working Knot.

2.1.3 Merging of knowledge artefacts (conceptualizations)

Merging of knowledge artefacts (conceptualizations) is one of the operations that are necessary for *model management* [Ber03]. This is especially true in scenarios of *collaboration* (like collaborative knowledge creation practices), which often involve a parallel editing of conceptualizations created by the collaborating parties. In the context of the KP-Lab project [D6.6], two examples of recently developed tools for supporting collaborative modelling activities centred around conceptualizations

expressed as RDF/S KBs [D5.3] are the Collaborative Semantic Tagging [SemTag], in which learners collaboratively annotate various content items with semantic tags (i.e., vocabulary terms), and the Collaborative Semantic Modelling [ColMol], in which learners have additionally the possibility to structure the terms of their vocabularies using various semantic relationships such as “is_A” and “has_part”. These tools are developed to cover different needs and requirements of the learners [D2.4].

Collaborative modelling activities may take several forms. A comprehensive analysis and classification of the various dimensions characterizing collaborative modelling activities was presented in [NCLM06]; in this deliverable, we are interested in one of the possible dimensions, namely on scenarios of *asynchronous collaboration*, where the users work on different local copies which are afterwards committed and merged.

Consider for example, the case where two (or more) users are independently engaged in the development of a theory regarding the problem or phenomenon under investigation. This could be made, for example, using the Visual Model Editor tool [D6.6]. Following some period of independent work, the users may want at some point to combine their theories (models) in an effort to explicate the similarities and differences between the different conceptualizations. This is part of the process of trialogical learning in which different (or even competing) theories and suggestions are combined to produce an innovative outcome.

In this respect, support for merging different conceptualizations when the curator (if any) or the users decide to do so is required. In order to support this operation, it would be useful to have a tool that would allow the users to inspect the results of the merging before actually executing it.

Another feature of the merging process is that its output provides useful insights regarding the similarities between the information found in the various sources. This feature, combined with the information that can be extracted by comparing the various conceptualizations (using the Comparison Service, see [D5.3], [D5.4]), can prove valuable in the analysis of the different produced models or theories and the eventual understanding of their differences and similarities. Note here that the Comparison Service is focused on identifying the differences, whereas the merging (provided by Knowledge Synthesizer) focuses on explicating the similarities.

The need to support merging in the context of collaborative activities has arisen in the working knot “Managing and Analysing Complex Knowledge Structures” (MACKS) and has led to the decision that some advanced functions related to the merging of models should be added in the next version of the Visual Model Editor (VME) and Visual Modelling Language Editor (VMLE) [D6.6], [D2.4].

2.1.4 Analysis of knowledge creation processes

Knowledge creation type of processes, both in educational as well as professional settings, frequently contain some predefined goals and are based on collaboration between all included participants, using relevant resources and useful tools. Whole process is monitored and all the performed actions and modifications (events) are

stored into various repositories to provide additional and important information about completed and/or still ongoing processes. The summarized information is a very useful tool for maintaining the knowledge creation processes and analyzing them. Under summarized information we mean here various aggregations of available data, e.g. number of participants involved and number of actions performed by each of them; number of content items used / changes made / versions produced; number of annotations defined / assigned / changed; number of comments added; number of to-do items created / fulfilled or not fulfilled; number of chats, meetings, links, etc.

Requirements for such kind of functionality have been discussed in relevant Working Knots, in particular in WK Shared Space and Common Tools, WK Process Management and Analysis but also in WK Change Laboratory. Discussions in these WK's brought list of user requirements and expectations that were used for creation of this technical specification. Analytical features that will be described in details below will be integrated as part of KP-environment, and additionally can also be utilised in M2T and ASDT.

Such kind of various aggregated information, which can be provided and by end user tools presented e.g. in a form of different graphs, can be useful for different purposes, e.g. for identification of division of work, identification of most active persons, identification of well collaborating group of people. Similarly, other types of objects may be put in the center of the analysis, e.g. different types of objects of activity, or a combination of objects and subjects leading to some advanced social network analysis facilities. For these purposes there are already specified some services within the data export tool for analysis [D6.6-DEAT], which serves as a separate channel of information gained from various KP-Lab repositories in order to produce selected data for its analysis within specialized third party analytical and/or network visualization tools.

Another approach to the analysis of knowledge creation processes is to consider the processes as a series of different actions in a chronological order, possibly with different levels of granularity, where some subsets of them may have crucial importance. Such carefully (manually) selected subsets of actions will be called *critical patterns*. These patterns usually lead to some critical moments in a knowledge creation process, which can mean, for example, a significant progress, discovery of new knowledge/approach, or in opposite they may indicate non-success of a particular process or its immature finish. Such kind of patterns may also conceptually represent interesting knowledge practices emerged within particular knowledge process – either being positive (something like best practice), or negative (worst practices).

In such a way particular critical pattern from one process (i.e. particular sequence of selected events) can be manually selected (in a suitable user interface) by the user and stored as a new type of the knowledge object. Other users then can visualize patterns and use pattern-matching service to find similar patterns in the historical or actual data. Notification service can be integrated with the pattern-matching service to check current processes and to notify the users about the relevant patterns identified in the running process.

2.2 Relation of high-level requirements, driving objectives and user tasks to the needs of the services provided by SWKM

Identification of motivating scenarios for the advanced semantic-based support of collaborative knowledge creation processes, as it was presented in section 2.1, enables addressing the functional requirements, as another step towards the specification of particular modules and components providing the required functionality of the KP-Lab system middleware. In particular, the relevant user tasks, driving objects, and high-level requirements, as proposed and elaborated in the D2.4 deliverable [D2.4], are identified and further described in this section to scaffold the functionalities identified in the above-presented motivating scenarios.

2.2.1 Collaborative maintenance of semantically tagged artefacts

The above-mentioned scenarios for collaborative work with shared knowledge artefacts (see section 2.1.1) imply a set of high-level functional requirements, defined in [D2.4] to perform by comprehension, classification, and partly also by clustering services. Particular activities for supporting the collaborative manipulation with artefacts can be divided into three groups, where each group addresses a set of specific high-level functional requirements.

1. *Grouping and advanced classification of artefacts*

- HLR1.1: "Users can create structure and share various artefacts in one place", which is part of DO1: "Users are provided with a collaborative environment where they can work on shared artefacts" and UT1: "Organizing shared artefacts and collaborative tools".
- HLR4.1: "Users can categorize, classify and cluster artefacts in different manners", which is part of DO4: "Users can describe the semantics of artefacts and their relations" and UT2: "Modifying the content of the shared artefacts individually and collaboratively".

Outlined high-level requirements cover the activities for advanced classification and clustering of artefacts, based on a combination of textual analysis and semantic tagging. The tag vocabularies can be extended by a lexicon of synonyms, rule based word transformations, stemming mechanisms and other linguistic resources. After the analysis of the textual content, the knowledge artefacts can be structured by classification and clustering procedures that will match the words and statements from analyzed texts with the linguistic resources (entries of the lexicon of synonyms, etc.). A structure of artefacts related to the semantic tags and/or a structure of semantically similar "chunks" of artefacts can be provided as an output of this procedure. The advantage of this approach is no requirement for availability of prior training set and a possibility to update the synonym lexicon in the case of need (especially if a standard solution as, for example, WordNet will be used).

2. *Support for semantic tagging*

- HLR4.2: "Users can use semantic descriptions to collaboratively work on the structure and meaning of artefacts as well as their relations", which is part of DO4 and UT2.

- HLR4.7: “Users are provided with functionality for suggestions of semantic descriptions for artefacts and suggestions for amendments to the vocabularies based on text-mining analysis”, which is part of DO4 and UT2.
- HLR7.6: “Users are able to semantically describe and analyze text-based artefacts (or document sections) according to the structure and content of the document”, which is part of DO7: “Users have the capability to create, use, edit and revise various kinds of text-based artefacts collaboratively and in a sustained manner” and UT2.

These high-level requirements include assistance activities during the process of semantic tagging as suggesting and recommendations of potentially suitable (semantic) tags based on content of the artefact, analysis of artefact’s textual content and identification of key terms, checking of semantic tag consistency, and various heuristic methods aiming at analysis and improvement of tag distribution.

The HLR4.2 requires the ability to describe relations between artefacts. The suitable representation for a semantic tag would thus be a triple (tagged artefact, relation, other resource representing other artefact or a meaning of the tag).

Moreover, as required by the HLR4.2, the designed service will propose changes and improvements of the semantic tag vocabulary, by proposing keywords (extracted from the texts of analyzed artefacts or obtained by quantitative analysis of existing tagging structure) to be added into or possibly also removed from the vocabulary.

3. *Search and semantic-based retrieval*

- HLR1.4: “Users can search artefacts within and outside the shared environment using full text, metadata or domain ontologies”, which is part of DO1 and UT1.
- HLR1.5: “Users can create and work with selected subsets of artefacts (e.g. a user might select all content items relevant for a certain task at hand)”, which is part of DO1 and UT1.
- HLR8.6: “Users can search the content and metadata using full text and/or semantic metadata search for planning and reflecting on activities”, which is part of DO8: “Users can plan, organize and manage tasks collaboratively” and UT3: “Management and organization of collaborative work processes”.

Users will be able to select one or more knowledge objects of activities and find similar objects. Similarity can be based on a) text content b) metadata properties (i.e. author, creation date, etc.) or c) semantic annotations. Moreover, users can refine the search results by selecting some of the semantic tags (e.g. as attributes of some of the retrieved artefacts) and invoking a Knowledge Matchmaker service for re-classification and search extension.

2.2.2 Notifications and recommendations during the collaborative knowledge creation processes

In order to allow smooth collaboration within the KP-Lab environment, users must be able to follow and react on events and changes relevant to their own tasks and

obligations. Especially when cooperation takes place asynchronously, over longer periods of time and tasks are complex, it becomes more and more difficult to keep track of what is going on and to decide which occurrences are relevant to one's own activities. These activities are relevant to the following high-level requirements [D2.4]:

- HLR8.4: "Users are provided with advanced awareness affordances and can request notifications of users' interactions (e.g. users can manage various awareness levels and collaboration rules)", which is part of DO8 and UT3.
- HLR5.5: "User are able to tailor and select what notifications to receive and in what frequency to their mobile device", which is part of DO5 "Users can contribute to shared work from situated but distant places" and UT1.

I.e. users will have the possibility to decide on a personalized notification scheme, which allows them to specify the events they want to be informed about and when, via user defined subscription criteria. Users' subscriptions are not only based on traditional criteria such as type or initiator of an event, or ID of a knowledge object, but also based on semantics of the event (e.g. regarding annotation of the knowledge artefact in question, or change in the conceptual model used). User will also be able to select also interval in which the notifications can be delivered to her/him, i.e. daily, weekly or monthly and select suitable delivery channel (e-mail, RSS feed, mobile phone or a combination of them), as user prefers.

2.2.3 Merging of multiple knowledge artefacts (conceptualizations)

As already mentioned, it is helpful, in the context of trialogical learning, to allow users or knowledge workers to combine information from two or more different information sources. These sources often have the form of conceptualizations of the same phenomenon under investigation, which are created by different people (or groups), in which case their merging would return the combined knowledge of the group. Unlike the Comparison Service [D5.3], this process is used to uncover the similarities (rather than the differences) between the various conceptualizations.

The merging process should return a conceptualization that consists of the common information found in the sources. The user should be able to select the sources that he will use for the merging and should have enough flexibility in order to be able to see the information found in at least one of the conceptualizations, or the information shared by all conceptualizations. This activity is related to the following high-level requirements, driving objectives and user tasks from [D2.4]:

- HLR4.5: "Users are able to compare and integrate different knowledge representations/visual models", which is part of DO4: "Users can describe the semantics of artefacts and their relations" and UT2: "Modifying the content of the shared artefacts individually and collaboratively". Note that, as already mentioned, the comparison of knowledge representations/visual models is supported through the Comparison Service, whereas the integration of knowledge representations/visual models is supported through the Knowledge Synthesizer Service.
- HLR6.3: "Users can share and integrate different visual modeling languages, ontologies and vocabularies", which is part of DO6: "Provide users with possibilities to develop and use their own conceptual models" and

UT2: “Modifying the content of the shared artefacts individually and collaboratively”. In this respect, the Knowledge Synthesizer supports the integration of different visual modeling languages, ontologies and vocabularies .

Furthermore, the ability to merge conceptualizations would allow the user to perform a more thorough analysis on how the different groups view the phenomenon under investigation, e.g., by uncovering commonalities or differences in the conceptualizations, or by identifying groups (or people) whose perception is very different (or very close) to the average perception and so on.

2.2.4 Analyzing the knowledge creation processes

Analytical and Knowledge Mining Services aim to provide users (incl. researchers, teachers, students, tutors, mentors, experts, and knowledge workers) with **(1)** summarized information about the activities going on in a particular workspace, as well as **(2)** support discovery of interesting working / critical patterns indicating interesting knowledge practices.

For the first type of supporting functions **(1)**, the following high-level requirements are relevant [D2.4]:

- HLR9.2: “Users are provided with customized summaries about the knowledge objects available within the shared environment (e.g. users might request an overview of the tasks completed within the last 2 weeks or the interactions of people within a shared workspace)”, which is part of DO9 “Users are provided with history on content development and work process advancement” and UT3.
- HLR13.6: “Users are provided with summative information on performed actions (e.g. added comments, created tasks, modifications in metadata, background materials for decisions, etc.)”, which is part of DO13 and UT5 “Investigation and development of knowledge practices”.

Users will be able to retrieve summarized information about their own (or others’) behaviour in order to monitor and reflect on their own (or others’) working practices. Content and format might vary depending on users’ needs. These will be various aggregations of available data with respect to participants involved, type of actions performed, content items used, etc.

For the second type of supporting functions **(2)**, the following high-level requirements are relevant [D2.4]:

- HLR1.2: “Users are able to view the artefacts and their relations from different perspectives”, which is part of DO1 and UT1.
- HLR8.7 - Users are provided with a customized analysis of groups’ working processes (e.g. identification of typical sequences of actions or interesting rules), which is part of DO8 and UT3.
- HLR9.1 - Users can track the evolution and changes of knowledge objects and find out their authors and contributors (sequences of performed steps in time, incl. versioning), which is part of DO9 and UT3.

Information about the evolution of contents and work processes provides another, completely different means to monitor ongoing and learn from past knowledge

creation processes. This is also a way how to reflect on the community's practices and developing them. Towards that end, it is important to identify relevant actions that led to the advancement or evolution of a particular knowledge object (discussion contributions, comments, linked artefacts, or changed conceptual models etc.) or to an identified critical moment in the process. In these situations, semantic context of the relevant actions will be taken into account as one dimension of analysis. Analytical and Knowledge Mining Services will provide means to identify, describe and store such kind of critical patterns on one hand side and look for them (resulting e.g. in notifications) e.g. in running or other past processes.

2.2.5 Summary of the Requirements

Required functionality	**Description**	**Service that provides the functionality**
Checking of semantic tags' consistency	Users can check the consistency of semantic tags and obtain recommendations for possible/potential changes of tags of an artefact and/or of enhancements / modifications in the tag vocabulary.	Knowledge Matchmaker, Comprehension service
Checking of semantic tag vocabulary, proposal of changes	Users can check a structural consistency of a vocabulary of semantic tags, and obtain a proposal of changes for a given set of artefacts.	Knowledge Matchmaker, Comprehension service
Search / semantic-based retrieval	Users can select one or more knowledge objects and find similar objects based on text content metadata properties or semantic annotations. Users can refine the search results by selecting some of the semantic tags and invoking a service for re-classification and search extension.	Knowledge Matchmaker, Comprehension service, Search service [Search]
Suggestion of semantic descriptions	Users can identify entities and relations directly in the content of the artefacts and let the service produce these annotations in a standard representation (RDFa).	Knowledge Matchmaker, Information extraction service
Improve information extraction models	Users can review suggested annotations produced by the information extraction service and improve in such a way the extraction models based on the users' feedback.	Knowledge Matchmaker, Information extraction service
Merging of Conceptualizations	Users performing an automated merging of the information found in at least one, or all, of some pool of conceptualizations	Knowledge Synthesizer
Analysis of Different Conceptualizations	Users can identify the commonalties between conceptualizations created by different users, as an aid to the analysis of the users' understanding of the phenomenon under investigation.	Knowledge Synthesizer
Customized analyses of knowledge creation processes	Users can retrieve summarized information about the activities going on in a particular workspace from various perspectives.	Analytical and Knowledge Mining Services – event aggregation service
Identification, description and discovery of critical patterns	Users are provided with support for discovering of interesting working / critical patterns indicating interesting knowledge practices.	Analytical and Knowledge Mining Services – define pattern and matching services

Table 2. *Summary of the high-level functional requirements*

3 Functional and Architectural Design

3.1 Knowledge Matchmaker (V.2.0)

The second prototype of the Knowledge Matchmaker contains further enhancements of the clustering, categorisation, and notifications services as they were specified according to the motivating scenarios and high-level requirements for supporting the collaborative knowledge processes and work on semantically annotated (tagged) artefacts. In particular, it provides the services and methods for semantic tagging recommendations and consistency check, extended semantic-based retrieval capabilities, information extraction services, as well as the notification and recommendation services built on the History/Participation Awareness [HPA01] that support personalised punctual and scheduled notification during the collaborative knowledge creation.

3.1.1 Comprehension Service

The Comprehension service provides middleware functionality for collaborative work with knowledge artefacts, especially focusing on the support of semantic tagging and retrieval. Two main functional streams can be identified for the service, namely 1) consistency checking and support of the semantic tagging process, and 2) semantic search and retrieval. Text analysis capability, linguistic extensions of the vocabulary of semantic tags, and analysis of structural correlations in the semantic tagging are the common attributes of both functional streams.

The support for semantic tagging adapts methods of machine learning and text analysis, namely heuristics based on linguistic analysis and investigation of similarities and structural correlations of the semantic tags of artefacts (content items). The analysis of the tag structure for a given set of artefacts includes heuristic rules for assigning leaf tag nodes preferably to the inner tags of the taxonomy, suggestions for tag updates based on frequency analysis of co-occurrences of semantic tags on annotated artefacts, and investigation of the similarity of tag structures.

The Comprehension service encapsulates also the functionality for *advanced categorization*, based on analysis of texts and semantic tags of the knowledge artefacts. Linguistic analysis includes identification of key terms in the textual content of the artefacts and further matching with the entries of the lexicon of synonyms, linguistic rules, etc., to obtain suggestions of potentially suitable semantic tags. These proposed tags are logically merged with the existing tags (both semantic and free tags, while free tags are transformed to the semantic tags) that may already exist for the input artefacts. The tags in the resulting set are marked by a flag describing a recommended action in the semantic tag vocabulary (i.e. to add a new tag or to modify an existing tag – with a possibility for users to discard or modify the provided suggestions). The linguistic analysis is combined with the heuristic rules to provide the methods for consistency check of the tags and recommendation of the potentially suitable semantic tags for a given knowledge artefact.

The retrieval of artefacts, as shared knowledge objects, is also supported by the Comprehension service (see *search similar* functionality described below). It is based on the same principles and uses the same inner mechanisms as the tag consistency check. It analyses the textual content and properties (e.g. a textual description), metadata properties, and semantic tagging of a given artefact (or a set of artefacts) to retrieve the knowledge artefacts that are most similar, by means of all the analyzed information, to the given input artefact(s).

1. Consistency checking of the semantic tags

a) Checking the consistency of semantic tags for a given input artefact (or a set of artefacts). The method provides suggestions/proposals based on investigating structural relationships of semantic tags and annotated knowledge artefacts, using various heuristics and statistical algorithms. These algorithms, however, cannot give any relevant value judgments of the provided tag suggestions. Of course, it will be the user who finally decides if something is appropriate or not. Based on the implementation of the consistency checking in a user-side tool [D6.6] the user can accept, modify, or decline the suggested tags provided by the method as an output.

The proposed method for semantic tag checking enables to select a proper algorithm for evaluation of the tag structure. The algorithm is defined by the input mode as follows:

- *Heuristic rules* that are based on examining the structure of already tagged knowledge artefacts and comparing it with the structure of underlying tag vocabulary. Tag nodes are weighted according to the position and mutual relationships in the vocabulary hierarchy. The pre-defined heuristic rules then prioritize the leaf tag notes for tagging, recommend proportional distribution of tags and balanced tag-artefact structure. This mode can be useful if the tag vocabulary is complex and hierarchically organized (for example, from general to specific concepts).
- *Frequency of co-occurrences* of semantic tags on annotated knowledge artefacts. Algorithm employed in this mode is based on similarity of tag structures, which is combined with the similarity of texts. The method in this mode examines the vector of tags and terms (keywords) extracted from the textual content of an input artefact and compares it with the term and tag vectors of other artefacts in the shared space. The similarity of the vectors then enables to find the tags that should be added to or removed from the initial tag set of the input artefact, according to the structural correlations given by the statistical similarity of the vectors.
- *Combination* of the above algorithms. This mode is suitable if the implementation of the method in a user-side tool does not allow to enter the checking mode (e.g. in order not to disturb users with a selection of proper checking algorithm during the tagging process) and/or the structure of tags can not be examined to insert a specific checking mode automatically.

The signature of the method for semantic tag checking can be specified in the following form:

`SemanticTag[] checkTagging(URI artefactURI, String mode)`

input: artefactURI: URI of the artefact whose semantic tags will be checked;
mode: identifier of the algorithm used for consistency check of tagging: {"*heuristic rules*", "*frequency*", or "*combination*"}.

output: a set of semantic tags recommended for the input artefact as an update, according to the specified algorithm. Based on the implementation of the method on a user-side tool, the user can accept, modify, or decline the recommended tags.

Variant of the `checkTagging` method for checking a group of artefacts:

`SemanticTags[][] checkTagging(String[] artefactURIs, String mode)`

The proposed method for checking the consistency of semantic tags is primarily focused to support the collaborative work with knowledge artefacts, enabling to keep the structure of semantic tags consistent in a collaborative knowledge evolution environment. Since the method does not require any training data[2], it can also be considered as an effective technique for classification of already tagged artefacts into the space of concepts (tags, terms) of a given tag vocabulary. Using automated heuristic and statistical algorithms, it helps to organize the knowledge artefacts in the shared space in a systematic way. In addition, the suggested tags returned by the tag consistency checking method to the user enable to discover "hidden" relationships between an artefact and the concepts of tag vocabulary, and this way the method helps to build qualitatively new knowledge structures or views. The method is proposed to be implemented as an inherent module of the Semantic Tagging tool [D6.6] and can also be employed in the tools that enable a collaborative manipulation with shared knowledge artefacts of various types, e.g. Shared Space, Semantic Multimedia Annotation [D6.6], or Semantic wiki [SW_SUS].

b) Checking the vocabulary of semantic tags and proposal of changes. The method recommends actions as adding new tags, modification of tag names and/or of tag hierarchy, removal of particular tags, in a given tag vocabulary according to the analysis of textual content and tag structure of input knowledge artefacts. Algorithms employed in this method are similar as these used in the above-described method for tag consistency check. They include extraction of a vector of tags and key terms from the input artefacts and evaluation of their similarity to the tags from the vocabulary. Based on this comparison, the tags in the vocabulary are labeled by recommended action (i.e. add / modify / delete) and are returned as an output of the method. It is, however, necessary to emphasize that the method is non-destructive, since it provides only suggestions for changes. User can then decide if (or which of) the recommendations will be accepted and which will be discarded. Actual persistent modification of the tag vocabulary is not provided by this method; however, it can be performed by simple rewriting of the "old" tag vocabulary by the tag structure returned by this method.

[2] Here we are referring to the classification services designed for the first prototype of the Knowledge Matchmaker [D5.3], which was based on the text mining algorithms requiring quite extensive training data set.

The signature of the method for checking the tag vocabulary is proposed in the following form:

```
TagVocabulary[] checkTagVocabulary(String URI
TagVocabularyURI, String[] artefactURIs)
```

input: tagVocabularyURI: URI of the tag vocabulary to be checked and used for semantic tags of input artefacts, referenced by the second input parameter;
artefactURIs: URIs of semantically tagged artefacts - the semantic tags of these artefacts will be used as a reference for checking the tag vocabulary and proposal of changes.

output: the resulting tag vocabulary with the changes proposed according to the analysis of tagging of the input artefacts.

The described method for checking the tag vocabulary is inverse to the previously presented method of tags consistency checking. However, the purpose of this method is the same, i.e. to help users maintain the consistency of semantic tag structure and in such a way to support the collaborative work with knowledge artefacts. It is proposed to employ this method in a tool for design and management of semantic tag vocabulary, namely in the Tag Vocabulary Editor [D6.6]. As a support of the semantic tagging process, the method may also be used within the Semantic Tagging tool [D6.6] and/or in the user-side tools for collaborative work with semantically enriched artefacts, e.g. the Semantic wiki [SW_SUS].

2. Search similar

This functionality returns the list of knowledge objects (artefacts, content items, etc.) similar to the given knowledge object. Similarity is based on the vector document model and can be computed according to the textual properties, metadata properties (i.e. creator, creation date, etc.), or semantic annotations like tags from controlled vocabularies or comments. It is required that all properties included in the similarity measure will be indexed using the Search indexing service (see [Search] specification).

```
ObjectHit[] findSimilar(String URI artefactURI, String[]
fields)
```

input: artefactURI: URI of the artefact used for “similar like” query;
fields: the list of indexed fields that will be used to compute similarity. Each field corresponds to the semantic or textual property of the object of activity indexed in the search index (see [Search] specification).

output: the list of results of the “similar like” query sorted by similarity scores. ObjectHit contains reference to the similar object of activity and similarity score.

3.1.2 Information Extraction Service

The purpose of the Information Extraction Service is to support the user with semantically tagging artefacts and thus enabling to search for related ones based on these semantic descriptions. Machine learning techniques are utilized to allow semantic annotation of textual artefacts. The service identifies entities and relations directly in the content of the artefacts and produces these annotations in a standard representation (RDFa).

The important property for the service is the ability to adapt to new content. The envisioned functionality of the information extraction service will allow reviewing suggested annotations produced by the service and improve the extraction models based on the user feedback.

```
String initModel (String[] annotatedArtefactURIs, String
    ontologyURI, String[] settings)
```

Initialize the extraction model from a training set of annotated artefacts.

Input:

annotatedArtefactURIs: references to XML documents with manually annotated artefact textual contents (in RDFa format);
ontologyURI: an optional link to the relevant ontology. If provided, it allows the service to build better models by utilizing the hierarchy of concepts;
settings: implementation-specific settings.

Output:

URI of the new extraction model

```
String ie (String modelURI, String contentXML, String[]
    settings)
```

Input:

modelURI: a trained extraction model used for extraction;
contentXML: text content, optionally with embedded semantic annotations created manually, or by means of the previous call of this method with additional user feedback;
settings: additional flags, such as input/output format selection, mode (train-only, extraction-only).

Output:

Same format as input contentXML string with embedded new annotations in the specified format.

XML format specification

The XML format used in the information extraction service needs to represent semantic annotations embedded in the content and statements about these semantic

tags for training purposes. The format takes advantage of the RDFa annotations extended with additional attributes:

`feedback`
: Stores the "decision" of the user for the statement represented by this tag, thus providing the service with the user feedback for learning

  `“positive”`
  : the user acknowledged the statement as being correct;

  `“negative”`
  : the user acknowledged the statement as being incorrect;

  `“none”` (default)
  : the user did not provide a feedback for this annotation.

`confidence`
: value between 0.0 and 1.0 which denotes the confidence value of the extracted triple represented by the tag.

`extract`
: Presence of this attribute tells the information extraction service explicitly to extract information about the content of this element.

  `“classify”`
  : classify the content of this element, optionally can be used with the `“typeof”` RDFa tag to specify which particular set of classes is possible (based on the ontology);

  `“relations”`
  : Extract relations among the entity and other relevant entities;

  `“all”` (default)
  : classify the content of this element and extract relations.

3.1.3 Recommendation Service

Notification and recommendation services will cooperate with History/Participation Awareness (HPA) services designed within the WP4 [HPA01]. The events logged in HPA will be processes by Notification and recommendation service, matched with registered subscriptions and a respective user will be notified through various channels, based on user preference.

The process of registering a subscription is handled by registerSubscription method:

```
String registerSubscription (String userId, String subjectId, String subjectType, String objectId, String objectType, String actionType)
```

This method registers a new subscription and adds it into the list of users’ subscriptions (user is identified by its URI). If some parameter is null, it is not taken into account in the matching phase. If the method performs successfully, subscription identifier is returned. Service also provides a method for listing and removal of registered subscriptions:

```
String removeSubscription (String userId, String subscriptionId)
```

If subscriptionId is null, all subscriptions for specified users are removed.

```
Event[] getSubscriptions (String userId)
```

Returns all subscriptions registered for a particular user.

As a part of *Punctual Notification*, user can obtain URL of a RSS channel, where all notifications are immediately present.

```
String getRssUrl(String userId, String subscriptionId)
```

If subscriptionId is null, service returns URL to RSS feed which integrates all subscriptions for a given user. If subscriptionId parameter is present, the returned RSS feed contains only notification that matches to the particular subscription

In order to support also *Scheduled Notification*, i.e. the possibility for a user to subscribe for a specific periodicity of notification (e.g., to be notified once a day, once a week, or once a month), the following service is provided.

```
String registerEmail (String userId, String startTime, String endTime, String timeInterval, String emailAddress)
```

With these methods, user can register for scheduled notifications. In this method, the user can specify the necessary time constraints for scheduled notification. startTime and endTime parameters define a period in which the emails are sent. A null value in these parameters denotes an open interval. A timeInterval parameter specifies how often emails should be sent. With this service, the user can for example set a daily or weekly digest of notifications, which will be send to his email address.

3.2 Knowledge Synthesizer (V.1.0)

The Knowledge Synthesizer is responsible for combining different conceptualizations represented in the core Semantic Web language, namely RDF/S. We identify two different basic modes of operation, which produce different results, but are based on the same motivating idea and driving requirements. The two different modes determine whether all or any of the sources will be considered during the merging operation in order to identify the information at the output.

Let us consider a set of RDF KBs, say K_1, K_2, …, K_n, each corresponding to a different conceptualization of the same phenomenon under investigation. The role of the service is to provide a new RDF KB, say K, whose content (triples) is determined by the mode of operation as follows:

- In the case of UNION, the new KB K should contain the triples found in any of the n sources. Therefore, UNION should be used when we are interested in determining the information found in at least one of the sources.

- In the case of INTERSECTION, the new KB K should contain the triples found in all sources. Therefore, INTERSECTION should be used when we are interested in determining the information that all the sources agree upon.

In calculating the UNION (or INTERSECTION) of the KBs, the user may choose to consider the RDF KBs themselves (i.e., the explicit knowledge of the KB), or their closure (which includes both the explicit and inferred knowledge and is calculated by taking into account the transitivity of subsumption and instantiation relationships). This choice essentially determines whether the inferred knowledge in each of the sources will be considered in the computation of the result or not.

The major challenge faced by the service is that the set that occurs from the simple, set-theoretic union (or intersection) of the triples found in two or more RDF KBs (or their closure), is not necessarily an RDF KB itself, because it may contain invalidities. The resolution of such invalidities is the main problem that must be addressed by the service, as it is not a priori known what types of invalidities may be encountered, nor is it obvious how each such invalidity could, or should, be resolved. Additionally, the types of invalidities that we may encounter, as well as the methodologies that we should use to resolve them, are different in each of the modes of operation (UNION, INTERSECTION, and with or without taking into account the inferred knowledge). Despite the differences, this critical difficulty appears in all modes of operation, and our methodology to address it is common.

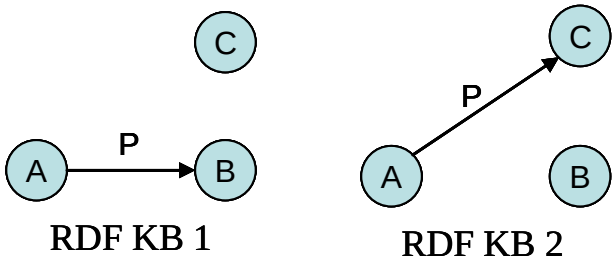

Figure 1. *Invalid UNION and INTERSECTION*

An example that illustrates this problem is shown in Figure 1, where we have two, almost identical RDF KBs; the only difference between these KBs is that the range of property P is different. These two particular KBs cannot be easily merged: UNION is problematic because P would have two different ranges, whereas INTERSECTION would be invalid because P would have no explicitly defined range. In both cases, we must make a decision as to which one should be the range of the property P in the resulting KB, so as to make the resulting KB valid.

The problem is similar to the one that has been addressed in the Change Impact Service (see [D5.3]). In that case, the straightforward deletion or addition of some triple(s) could cause invalidities, which should be resolved by means of side-effects (additional changes, which are in fact extra deletions and additions) that should be applied upon the KB. The determination of the side-effects was made using some kind of preference ordering that allowed us to determine the most plausible way to resolve some invalidity out of the various possible ones.

The same general method will be followed here as well: given some invalid KB (say K) that resulted as the union or intersection of two or more (valid) KBs (or their closure), we add or remove some knowledge (triples) to/from K in order to render it valid. Towards this end, we first identify the ways in which K is invalid (by determining which validity rules are invalidated). Then, we determine the various possible ways in which K can be rendered valid. Finally, we use some preference ordering that determines the most plausible (best) out of the different options for resolving the invalidity. In the example of Figure 1, the service would select either B or C as the range of the property, depending on the relative position of B and C in the subsumption hierarchy.

Thus, the general process followed by the Knowledge Synthesizer is as follows: first, we identify the triples that correspond to the set-theoretic union, or intersection, of the triples in the input conceptualizations, or their closure, depending on the mode of operation; then this temporary set of triples is fed to the component (of the Knowledge Synthesizer) that will identify and restore any possible invalidity that is found in that, temporary, KB. After restoring the invalidities (if any), the resulting KB is returned as the output of the service.

The use of the Knowledge Synthesizer guarantees that the result (output) will be a valid KB. In addition, the service will always return a KB that is "as close as possible" to the result of the set-theoretic union or intersection of the input RDF KBs (or their closure), where the notion of "proximity" between the temporary (possibly invalid) result and the final output is determined via the preference ordering.

At a more technical level, the signature of the Knowledge Synthesizer will be as follows:

```
String merge(String[][] nameGraphSpaceURI, String mode,
  String closure)
```

The Knowledge Synthesizer accepts in its input a collection of RDF KBs (`nameGraphSpaceURI`). Each of those RDF KBs will be represented by a set of URIs (`nameGraphSpaceURI[]`), each URI corresponding to a single namespace or named graph. Thus, each source RDF KB actually corresponds to a set of namespaces and/or named graphs (and determined by a set of URIs). In order to determine the triples belonging in said source RDF KB, we take the union of the triples in the namespaces and/or named graphs corresponding to the input URIs (`nameGraphSpaceURI[]`), as well as the triples in the namespaces/named graphs that depend on those namespaces/named graphs. Note that this kind of union will necessarily result to a valid KB, as it corresponds to namespaces/named graphs that are already stored, so they cannot contain conflicts.

In addition, the Knowledge Synthesizer takes in the input a number of parameters that determine the mode of operation. These parameters determine whether the operation of UNION or the operation of INTERSECTION will be executed (`mode`), as well as whether the inferred knowledge will be considered or not (`closure`).

The output of the service is a string containing the TRIG serialization of the (valid) RDF KB returned in the output of the service. This RDF KB can be imported in memory and manipulated using the Main Memory Model API (see [D5.4]), or stored in the persistent memory (database) for querying and updating.

Further details on the implementation of the service will appear in the upcoming deliverable D5.8: "Prototype of the Knowledge Matchmaker (V.2.0) and the Knowledge Synthesizer (V.1.0)", which is due on M42.

3.2.1 The Process of Merging in the Related Literature

The need for combining KBs and semantic information (in general) has been identified in several contexts. There are some theoretical works [Kon00], [Kon04], [KLM04], [KP05], coming from the area of *belief merging*, that describe several different operators for combining KBs. For instance, [Kon00] studies the properties of a number of different merging operators where a KB is a set of first-order formulae. Although very interesting theoretically, the practical exploitation of these results is quite distant, given that first-order logic is an undecidable language. Nonetheless, such approaches uncover the prevailing intuitions behind the process of merging, and describe an automatic, albeit non-practical, method to perform merging. Our approach is more practical-oriented and aims to produce a working prototype addressing the merging of RDF KBs.

There is also a rich literature on *ontology merging and integration*, that is quite relevant to our work. Ontology merging and integration deal with the fusion of the information found in two or more ontologies. There is a subtle difference between the two fields which is described in [FMK+08]. The former (ontology merging) refers to the combination of ontologies covering highly overlapping or identical domains; this process is used to fuse ontologies that contain information about the same subject into one large (and hopefully more accurate) ontology. The latter (ontology integration) refers to the composition (via reuse) of ontologies covering loosely related (i.e., similar) domains (subjects); this is mainly used when building a new ontology that covers all these subjects. Note that the terms merging and integration are often misused in the literature [FMK+08].

Our work on the Knowledge Synthesizer is closer to ontology merging, because the envisioned application scenarios of the Knowledge Synthesizer within KP-Lab are expected to require the merging of conceptualizations covering identical phenomena. Note however that ontology merging corresponds to the operation of UNION; to our knowledge there is no work in the literature dealing with the INTERSECTION operation. A thorough literature review of the two areas (ontology merging and integration) and a lot of pointers to relevant papers can be found in [FMK+08]. Furthermore, it should be emphasized that, to our knowledge, all existing works in ontology merging and integration use manual or semi-automatic approaches to resolve the conflicts that may appear during merging [FMK+08].

The most popular tools used for ontology merging are PROMPT [NM00], [NM03] and Chimaera [MFRW00]. These tools use a semi-automatic approach focused on

suggesting how elements from the source ontologies should be merged in the resulting ontology. The final choice relies on the ontology engineer. An interesting theoretical approach to ontology integration (also applicable to ontology merging) appears in [CGL02], which focuses on the formal definition of mappings between the resulting and the source ontologies and how these mappings can be exploited for query answering; another theoretical approach to ontology merging can be found in [BCM99].

Another frequent misuse of the terms ontology merging and integration is to refer to the fields that are related to *heterogeneity resolution*, such as ontology mapping, ontology matching etc (see [FMK+08]). It is true that, in the general case, establishing a proper mapping between the fused ontologies is critical towards their successful fusion. However, this is not the only challenge of the merging process: even if a perfect mapping is provided, the result of fusion is not at all clear, as was shown in the example of Figure 1 above.

Despite the importance of mappings in the general case of ontology merging, our work does not deal with the problem of establishing mappings between the different source ontologies. In the context of KP-Lab, we expect that the terminology used by the users will be common, so the need for a sophisticated mapping is significantly reduced. Instead, we assume a simple, default mapping, which is based on a string comparison of the local names of the URIs of the various elements in the two RDF KBs, as well as the versioning relationship between the namespaces/named graphs involved. This mapping is good enough for the expected usage scenarios, described in the respective section of this deliverable, because we expect the merged ontologies to be parallel ("fork") versions of the same ontology.

The fields of ontology merging and integration (which are the most relevant to the work on Knowledge Synthesizer), as well as the Knowledge Synthesizer itself, have strong ties with another ontology-related field, namely *ontology debugging* [FMK+08]. Ontology debugging is the field that deals with the resolution of invalidities in a given ontology [FMK+08]; therefore, it is the field that addresses the most difficult subproblem faced by the Knowledge Synthesizer service, namely, the determination of the actions to be taken in order to resolve the invalidities caused by the merging of the source RDF KBs. Details on ontology debugging and several pointers to the related literature can be found at [FMK+08].

The ontology debugging field is characterized by the fact that many approaches depend on manual input by the user to determine the proper way to resolve some invalidity [FMK+08]. Many researchers believe that the best thing an automated system can do is to propose alternative ways to repair an ontology, but it's up to a human expert to select the appropriate one to resolve the invalidity [SC03]. As a result, most approaches deal with the problem of *diagnosis*, i.e., determining the invalidities as well as the source(s) of each invalidity, leaving the problem of *repair*, i.e., the resolution of invalidities, to some human expert. In this respect, the tool's role is to provide, in a concise and user-friendly manner, all the necessary information that will help a human expert resolve the invalidity.

Nonetheless, there are certain tools that perform automated ontology debugging (e.g., [Kal06], [LPSV06], [MLBP06], [QP07]); such tools work on expressive logical models, such as *Description Logics* (DLs) [BCMGNPS02] and use tableaux-based methods to identify the invalidities, the source(s) of invalidities and the necessary actions for resolving the invalidities. However, these methods are not applicable in the RDF/S context, as they are based on a different logical setting.

3.3 Analytical and Knowledge Mining Services (V.1.0)

Analytical and Knowledge Mining Services (AKMS) will provide a set of analytical services based on various data analysis and data mining tasks performed on data stored in several KP-Lab repositories, e.g. log storage for History and Participation awareness (see T4.11).

AKMS will be implemented as integrated part of KP-Lab system, because this functionality requires interaction with end user tools that are developed under WP6 responsibility and platform services that are provided by WP4. The main interactions between are depicted on Figure 2.

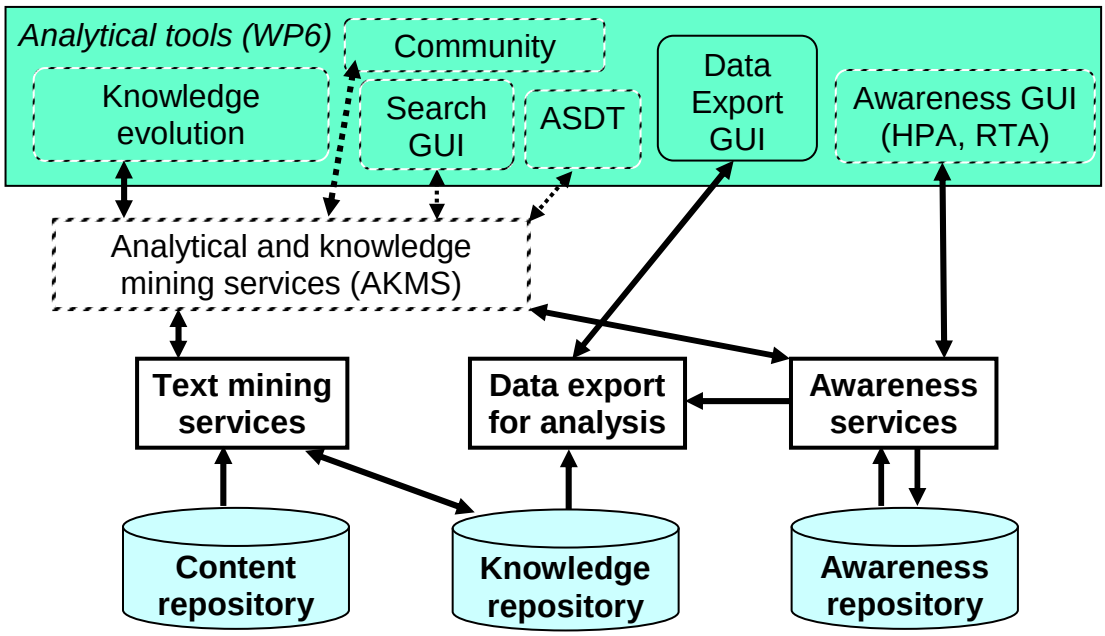

Figure 2. *Integration of AKMS services in whole KP-Lab System*

The main source of data for AKMS is Awareness repository that provides log storage for both types of awareness features that are implemented within KP-Lab project, i.e. Real time awareness (WP6) and History/Participation Awareness (WP4). Information stored in this repository describes actions, activities, changes and modifications performed by users in the KP-Lab environment. So this requires communication with all integrated parts of KP-environment, such as: the support tools e.g. preferences and setting: the common tools e.g. M2T and additional tools, e.g. SMAT or ASDT (for the functional view on KP-Lab tools see [D6.6]). This communication will be realized through client library at the Flex side as was agreed in WP6. This library monitors all

events in GUI and then sends them as a package into Awareness repository. List of proposed events for monitoring is part of HPA technical specification [HPA01] and actual list of supported actions can be found on official project wiki [HPA02].

AKMS features are specified according to the end-user application requirements for knowledge-intensive cooperation and reflection on users' knowledge practices in KP-Lab tools. Two main analytical perspectives will be implemented in the first version of AKMS:

- Services supporting participation and activity analysis of knowledge creation processes
- Services supporting knowledge evolution analysis

Each of these perspectives emerged from evolutionary discussions with end-user partners that will use relevant analytical features and results of analyses for their education or research purposes.

3.3.1 Services for support of participation and activity analysis of the knowledge creation processes

Within this part two different mechanisms supporting analytical features will be provided. The first one is aimed for any tool asking for specialized aggregated information that will further be processed by the tool (e.g. visualized or used for support of decisions etc.). The second mechanism (see subsection 3.3.2) supports envisaged stand-alone visualization tool with special support for analytical queries as they are known in data warehouses. But in this particular case the user will be guided in the process of formulating the analytical query and a suitable form of visualization of its results.

The first mechanism will be supported by the following web service.

```
String eventAggregationService (Query query,
List<AggregationFunction> aggregationFunctions,
Set<GroupBy> groupBy)
```

query parameter describes constrains which will be used for filtering of the events included in the aggregated view. Query object encapsulate the following constrains already specified for HPA:

- actionType - type of performed activity,
- objectID - URI of the Object of activity,
- subjectID - URI of the Actor,
- timeRange - time interval,
- filter - set of key value pairs which will be compared with events custom properties,
- excludeFilter - true of false, whether include or not events which does not have properties from filter present in them.

aggregationFunctions: specify the list of aggregation functions included in the view computed from the set of selected events.

- NumOfEvents - the number of events,

- NumOfSubjects - the number of unique subjects included in the result,
- NumOfObjects - the number of unique objects included in the result,
- TimeSpan - the date of the first event and date of the last event (starting and ending date).

groupBy: specify clause for the grouping of the result. It is possible to specify the following values:

- Subject - group results by subject,
- Object - group result by object,
- actionType - group result by type of the activity.

return value: XML of the result (scheme to be specified, see example for proposal).

Based on this basic general service some types of helpers will be implemented to provide concrete analytical requirements, see Example.

Example. This example will present aggregated view, which will select all users working on the Task1 object and for each user it will contains the number of updates and time span when the user updated this object:

```
group by Subject (subjectID), Query(objectID = Task1, actionType = update), aggregationFunctions = NumOfEvents, TimeSpan
```

Result:

```
<result>
  <row>
    <subject>User1</subject>
    <numOfEvents>10</numOfEvents>
    <startingDate>10-11-2008</startingDate>
    <endingDate>20-11-2008</endingDate>
  </row>
<row>
    <subject>User2</subject>
    <numOfEvents>2</numOfEvents>
    <startingDate>10-11-2008</startingDate>
    <endingDate>12-11-2008</endingDate>
  </row>
</result>
```

3.3.2 Tools for support of visual analysis of logs

The Visual Analysis of Logs Service (VALS) will provide functionalities supporting the "participation and activity analysis" perspective of AKMS. VALS will be designed to provide users with the following features:

- a user-friendly visual representation of the participation log, adapted to the formulation of analysis requests;

- the possibility for the user to easily formulate analysis requests for retrieving summaries about users activities;
- the possibility for the user to choose an appropriate mode for the presentation of the results: “histogram”, “pie-chart” etc.

User interaction steps are summarized in the following Table 3.

Authentication	The user connects to VALS by providing login and password (thus proving that he is allowed to explore the log).
Log presentation	The log from the HPA is presented to the user, schematically, in the form of a graph with clickable nodes (together with a “submit” button).
Query formulation	The user formulates an analysis query, visually, by clicking on nodes of the graph.
Selection of a result presentation	VALS shows a menu of available presentation modes (histogram, bar-chart etc.) for the user to select one.
Result exploration	VALS shows to the user the result of the query evaluation in the selected mode of presentation.

Table 3. *User interaction steps identified for visual analysis of logs*

Example. Suppose a user would like to visualize in a meaningful way for him the result of the following query:

```
Activeness (i.e. count of actions) by participant for
     Task-1.4 during the month of September 2008
```

The interaction steps between the user and VALS will be as follows:

1. The user connects to http://VALS.lri.fr.

2. If required by the user, the screen showing (which can look like Table 4.) the log with all events stored in HPA will be presented (this step is not necessary):

ID (group ID)	Time	Subject ID	Subject Type	Object ID	Object Type	Action
1 (1)	2008-07-31 18:30:07.0	http://www.kplab.org/systemmodel/TLO#Student_Mary	user	http://www.kp-lab.org/system-model/TLO#Task_4.3	task	modification
2 (2)	2008-07-31 18:56:30.0	http://www.kplab.org/systemmodel/TLO#Student_Paul	user	http://www.kp-lab.org/system-model/TLO#Task_1.4	task	modification
3 (3)	2008-08-01 00:52:54.0	http://www.kplab.org/systemmodel/TLO#Teacher_Frank	user	http://www.kp-lab.org/ontologies/ss#Note:_081114-1642-f2511634-a914-4dfe-8088-4070f8b4f53b	Content Item	cration

Table 4. *Example of preview of the selected part of the log (shortened).*

3. VALS transforms the log with all events stored in HPA into a schema (i.e. a graph) and shows it to the user (see Figure 3).

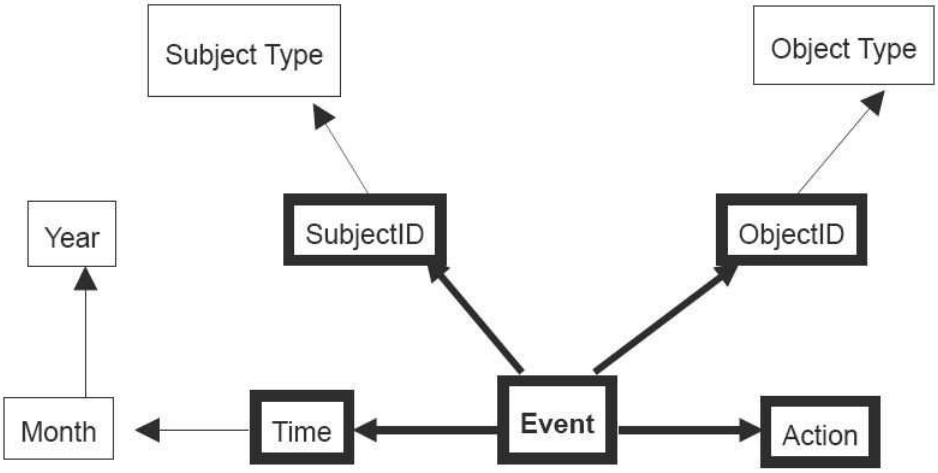

Figure 3. *Example of a schema automatically derived from the log data*

The user formulates his query, by clicking on the nodes (guided by VALS so that he chooses valid arguments for his query). In our example, the user will perform the following actions:

- click on **SubjectID** (thus indicating that he wants the Events grouped by participant)
- click on **ObjectID** (specifying "ObjectID = Task-1.4" so that only participants for that particular task be considered)
- click on **Month** (specifying "Month= September 2008" so that only participants for that particular month be considered)
- click on **Event** (since he wants every Event ID to be considered), then click on "COUNT" from the menu of operations proposed by VALS
- click on a presentation mode in a menu presented by VALS (line chart, pie chart etc.)
- click on the "Submit" button and VALS then returns the result in the presentation mode selected by the user (e.g. in form of a bar chart as shows Figure 4).

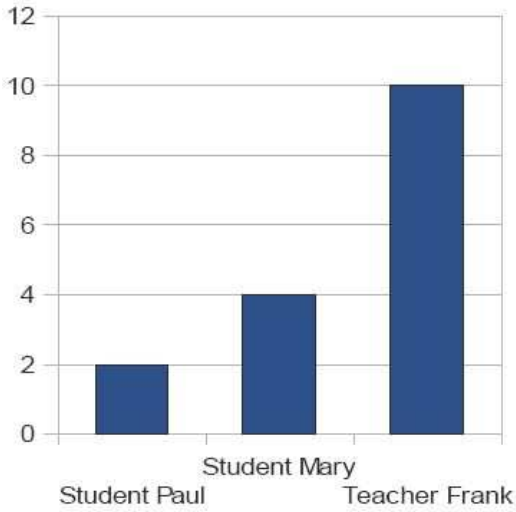

Figure 4. *Example of possible log data visualization in form of a bar chart*

Description of services

Log transformation:
TO DO: add signature & a short description
input: Relational table containing the log data
output: A schema presenting the log schematically, in the form of a graph

Query formation assistance:
TO DO: add signature & a short description
input: A set of clicks on a schema graph
output: An analytic SQL query ready for the evaluation

Result presentation:
TO DO: add signature & a short description
input: A query and a choice of presentation mode
output: The query result presented in the selected presentation mode

3.3.3 Services for support of knowledge evolution analysis

Let us start with an illustrative example of a simple knowledge creation process of this deliverable D5.6 (see Figure 5).

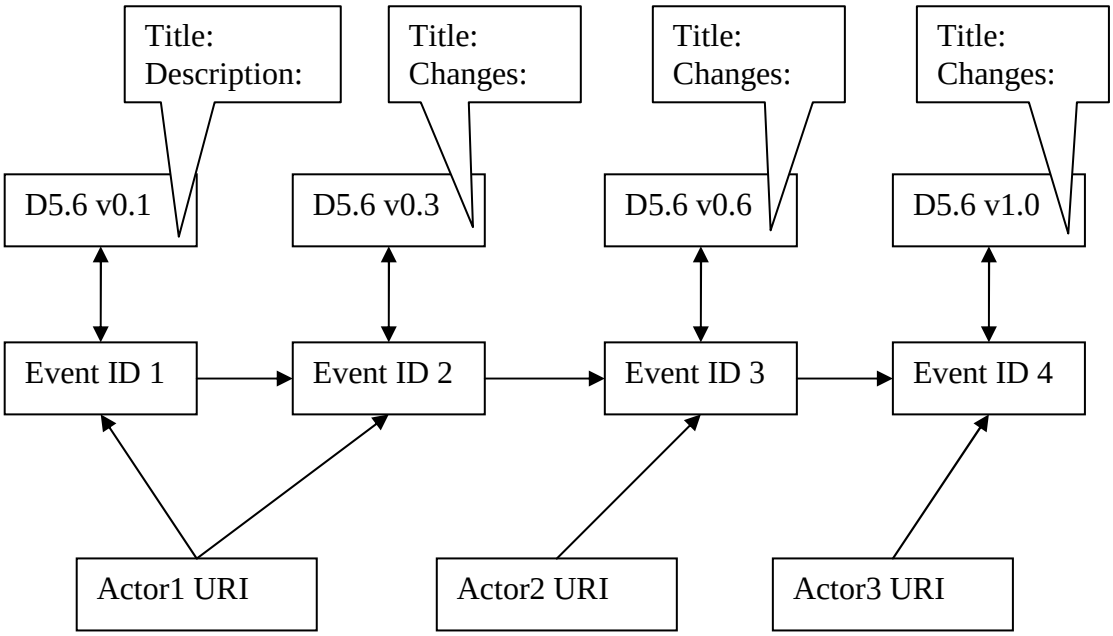

Figure 5. *Evolution process of D5.6 creation*

This particular process can be visualized as a series of actions in time (see Figure 5), e.g. based on particular versions of the document with relevant properties or linked to other types of knowledge object – chats, meetings, etc. In order to acquire all necessary data for such kind of visualization, combined access to all the repositories, i.e. to the Knowledge repository, to the Content repository, as well as to the Awareness repository, is needed. Information relevant to the performed events through timeline can be extracted from the Awareness repository and semantic information based on properties of relevant objects can be retrieved from the Knowledge repository.

One possibility to create this flow is:

1. User defines his/her interests – D5.6, from day1 (first draft) to day30 (final version) → evolution.
2. Each change relevant to this object (D5.6) represents one event in HPA (Awareness) repository – so list of events will be extracted.
3. Each event represents relevant version of the document – based on the URI, a relevant version can be found in the Content repository – so important aspect of this step is to save the ID of the content (versioning of the Content repository) as custom property in the log of events. It is relevant only for the objects with the content stored in the Content repository.
4. Based on the URI we can provide some information about each document version – properties such as title, description, and maybe, if it is possible, also a description of performed changes. This last information is possible in the situation when user makes a change in the document and writes a short notice describing this change, e.g. as some type of own tag. Then we can provide this type of own tag as the description of changes for relevant version (property Changes).

5. Based on the URI we can provide some supporting information for each version, e.g. assigned actors, chats, meetings, etc.

For these purposes we can use advanced versions of previous service with combination of relevant queries to the Knowledge repository based on user requirements – what users want to have in common description of relevant object version (in Figure 5 you can see property Title, Description and Changes)

Critical Patterns

The AKMS will provide for user's possibility to define critical patterns and based on proposed description the system will be able to discover similar types of patterns in historical or actual data. In the simplest form, pattern is a sequence of actions that lead to/caused a critical moment, as you can see on Figure 6.

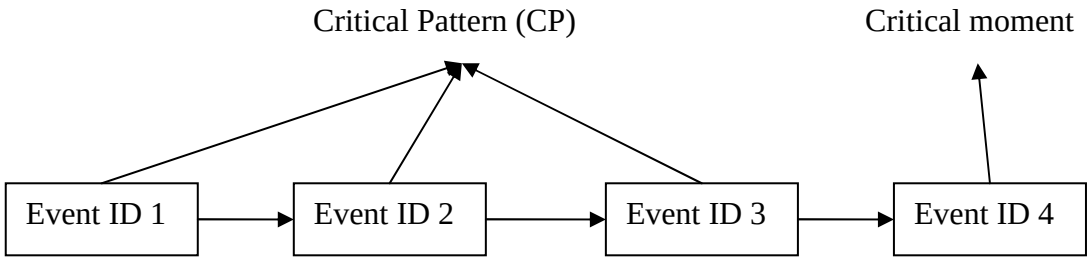

Figure 6. *Identification of critical moment and relevant critical pattern that caused it*

The critical pattern is defined as the sequence of events (i.e. events which preceded the critical moment and which have been identified as crucial on this path) and their relevant semantic properties that sufficiently identify/describe this particular type of critical pattern.

Particular critical pattern from one process can be manually selected by the user and can be stored as a new type of the knowledge object. Other users then can visualize patterns and use pattern-matching service to find similar patterns in the historical or actual data. Notification service can be integrated with the pattern-matching service to check current processes and to notify users about the relevant patterns identified in the ongoing processes (see Figure 7).

Internally, critical pattern can be represented as a sequence of tuples (events):

```
CP = <actionType, objectID, subjectID, TimeRange>1,  ...,
< actionType, objectID, subjectID, TimeRange>n
```

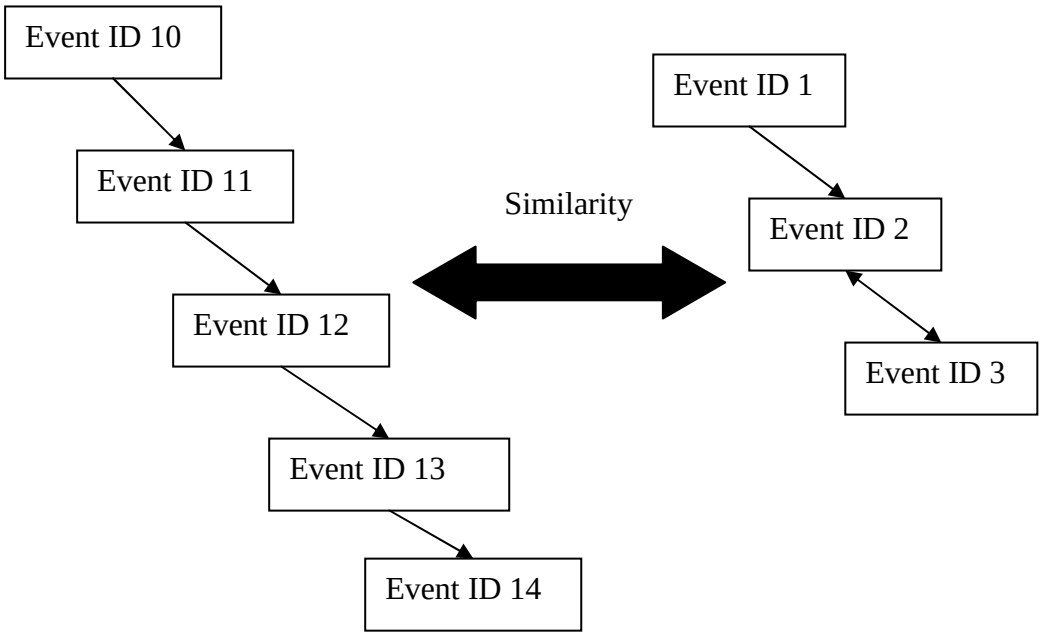

Figure 7. *Discovering defined pattern in historical data*

We can decompose the matching of such a sequences to the comparison of two tuples $<>_i$, $<>_j$, where we need to compare type of the action, object instance and (possibly) subject instance. Object instance can be arbitrary object of activity and pattern-matching service will reuse the Knowledge Matchmaker services to compute the similarity of two objects of activity. The subject is an agent (i.e. user) responsible for the action. Similarity of subjects can also be computed (if required) for example according to the groups to which the users belong to.

Possibility for future discussion is to have hierarchy of types of activities to be able to generalize the concept of critical patterns.

Patterns can be saved into the Knowledge repository, so they will be available for further usage (they will have its own URI; user can tag the critical pattern, etc.).

CriticalPattern - `events[]` – is a sequence of events (user actions) identified on the critical path leading to critical moment in a knowledge creation process.

```
URI definePatternService (CriticalPattern pattern)
```

This service returns URI of newly defined critical pattern stored in the Knowledge repository.

```
MatchingResult[] matchingService(URI pattern)
```

- score – similarity between the patterns
- CriticalPattern – new matched pattern from the history data

The result from the matching is a score that measures the level of similarity between defined and newly discovered critical patterns.

4 Conclusions and Future Work

This deliverable presented functional specification of three different SWKM modules. The first one is Knowledge Matchmaker (V2.0) and utilizes various text mining, information extraction, and heuristic methods for advanced access to and manipulation with shared knowledge artefacts according to the explicit meaning of artefacts expressed by their textual content, as well as metadata, including semantic tags. This second version presents a set of completely new services supporting miscellaneous functionalities derived from the motivating scenarios and mapped on the high-level requirements providing users intelligent tools for tag consistency checking, information extraction. Better support on search and notification will also be achieved.

Next two presented SWKM modules are completely new. The Knowledge Synthesizer (V1.0) can be used to combine information found in multiple sources; this feature is necessary to allow automated merging of the conceptualizations modeled in independently edited conceptualizations.

The Analytical and Knowledge Mining Services (V1.0) provide means for analyzing participation and activities within past or ongoing knowledge creation processes, as well as for support of knowledge evolution analysis (by means of manual identification of critical patterns and their later proactive identification in selected running processes).

Based on these specifications, proposed services will be implemented and delivered in form of stand-alone deliverables – software prototypes. In particular, Analytical and Knowledge Mining Services (V1.0) are due in M40 (deliverable D5.7); the Knowledge Matchmaker (V2.0) and the Knowledge Synthesizer (V1.0) are due in M42 (deliverable D5.8).

Bibliography

[Bauters07] Bauters, M. et al: Semantic tagging according to PBL vocabulary requirements - Questions remaining. Version 0.6 draft. KP-Lab internal deliverable. September 21, 2007.

[BCMGNPS02] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, P. Patel-Schneider (eds). The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press, 2002.

[Ber03] P.A. Bernstein. Applying Model Management to Classical Meta Data Problems. In Proceedings of the 1st Biennial Conference on Innovative Data Systems Research (CIDR-03), pages 209-220, 2003.

[BCM99] T. Bench-Capon, G. Malcolm. Formalizing Ontologies and Their Relations. In Proceedings of the 16th International Conference on Database and Expert Systems Applications (DEXA-99), pages 250-259, 1999.

[CGL02] D. Calvanese, G. De Giacomo, M. Lenzerini. A Framework for Ontology Integration. In I. Cruz, S. Decker, J. Euzenat, D. McGuinness (eds) The Emerging Semantic Web. Selected Papers from the First Semantic Web Working Symposium, IOS Press, pages 201-214, 2002.

[ColMol] End User Requirements for Collaborative Semantic Modelling. KP-Lab internal document, v.0.6, August 2007.

[D2.4] Driving Objectives and High-level Requirements for KP-Lab Technologies. KP-Lab project Deliverable D2.4, November 2008.

[D5.3] Specification of the SWKM Knowledge Evolution, Recommendation, and Mining services. KP-Lab project Deliverable D5.3, November 2007.

[D5.4] Prototype (V2.0) of the SWKM Knowledge Mediator, MatchMaker and Manager. KP-Lab project Deliverable D5.4, March 2008.

[D6.6] D6.6 M33 specification of end-user applications. KP-Lab project Deliverable D6.6, December 2008.

[D6.6-Search] D6.6 M33 specification of end-user applications – Search. Appendix of the KP-Lab project Deliverable D6.6, December 2008.

[D6.6-SSpUMT] D6.6 M33 specification of end-user applications – Shared Space and User Management Tool. Appendix of the KP-Lab project Deliverable D6.6, December 2008.

[D6.6-DEAT] D6.6 M33 specification of end-user applications – Data export analysis tool. Appendix of the KP-Lab project Deliverable D6.6, December 2008.

[FMK+08] G. Flouris, D. Manakanatas, H. Kondylakis, D. Plexousakis, G. Antoniou. Ontology Change: Classification and Survey. Knowledge Engineering Review (KER), 23(2), pages 117-152, 2008.

[HPA01] Technical specification for History/Participation awareness in Shared Space M24 specifications (M30 update for M32 prototype). http://www.kp-lab.org/intranet/design-teams/wk-shared-space-and-common-tools/awareness/participation-and-history-based-awareness/history_awareness_technical_v02.doc/view

[HPA02] Actual list of action types that are required by particular KP-Lab tools to be logged: http://kplab.evtek.fi:8080/wiki/Wiki.jsp?page=M30HistoryParticipationAwarenessListOfActionsForLogging

[Kal06] A. Kalyanpur. Debugging and Repair of OWL Ontologies. Doctoral Dissertation, University of Maryland, College Park. 2006.

[Kay 2007] Kay, J, K Yacef and P Reimann, Visualisations for team learning: small teams working on long-term projects. In C. Chinn, G. Erkens & S. Puntambekar (Eds.), Minds, mind, and society. Proceedings of the 6th International Conference on Computer-supported Collaborative Learning (CSCL 2007) 351-353, New Brunswick, NJ: International Society of the Learning Sciences.

[Kon00] S. Konieczny. On the Difference Between Merging Knowledge Bases and Combining them. In Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning (KR-00), pages 135-144, 2000.

[Kon04] S. Konieczny. Belief Base Merging as a Game. Journal of Applied Non-Classical Logics, 14(3):275-294, 2004.

[KLM04] S. Konieczny, J. Lang, P. Marquis. DA2 Merging Operators. Artificial Intelligence, 157(1-2):49-79, 2004.

[KP05] S. Konieczny, R.P. Perez. Propositional Belief Base Merging or How to Merge Beliefs/Goals Coming from Several Sources and Some Links with Social Choice Theory. European Journal of Operational Research, 160(3):785-802, 2005.

[LPSV06] S.C. Lam, J. Pan, D. Sleeman, W. Vasconcelos. A Fine-Grained Approach to Resolving Unsatisfiable Ontologies. In Proceedings of the 2006 IEEE/WIC/ACM International Conference on Web Intelligence (WI-06), 2006.

[LocSca08] Locoro, A., Scapolla, M.: Editing Semantic Tags Specifications Draft version for M27 release. Version 1.2 for M27. April 22, 2008. http://kplab.evtek.fi:8080/wiki/Wiki.jsp?page=M27EditingSemanticTagsSpecificationsDraftVersion

[MFRW00] D. McGuiness, R. Fikes, J. Rice, S. Wilder. An Environment for Merging and Testing Large Ontologies. In Proceedings of the 7th International Conference

on Principles of Knowledge Representation and Reasoning (KR-00), also available as Technical Report KSL-00-16, Knowledge Systems Laboratory, Stanford University, 2000.

[MLBP06] T. Meyer, K. Lee, R. Booth, J. Pan. Finding Maximally Satisfiable Terminologies for the Description Logic ALC. In Proceedings of the 21st National Conference on Artificial Intelligence (AAAI-06), pages 269-274, 2006.

[NCLM06] N. Noy, A. Chugh, W. Liu, M. Musen. A Framework for Ontology Evolution in Collaborative Environments. In Proceedings of the 5th International Semantic Web Conference (ISWC-06), 2006.

[NM00] N. Noy, M. Musen. Algorithm and Tool for Automated Ontology Merging and Alignment. In Proceedings of the 17th National Conference on Artificial Intelligence (AAAI-00), also available as SMI technical report SMI-2000-0831, 2000.

[NM03] N. Noy,.M. Musen. The PROMPT Suite: Interactive Tools for Ontology Merging and Mapping. In International Journal of Human-Computer Studies 59(6), pages 983-1024, 2003.

[QP07] G. Qi, J. Pan. A Stratification-based Approach for Inconsistency Handling in Description Logics. In Proceedings of the International Workshop on Ontology Dynamics (IWOD-07), pages 83-96, 2007.

[SC03] S. Schlobach, R. Cornet. Non-Standard Reasoning Services for the Debugging of Description Logic Terminologies. In Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI-03), 2003.

[Search] P. Bednar. Technical specification of Indexing and Search services. Version 0.5 draft, KP-Lab internal deliverable. September 30, 2007.

[SemTag] Specifications for Annotating Knowledge Objects with Semantic Tags. KP-Lab internal document, October 2007. Available at: http://www.kp-lab.org/intranet/design-teams/wk-management-and-analysis-of-complex-knowledge-structures/semantic-tagging/annotating-knowledge-objects-with-semantic-tags/AnnotatingObjectsWithSemanticTags-specifications-v1.doc/view

[SW_SUS] M. Bauters et al. WK5: Document Centered Collaboration. System usage scenarios for the Semantic wiki. KP-Lab deliverable. Revision 0.61. 23.11.2008