# Lifecycle Knowledge Management: Getting the Semantics Across in X-Media

Steffen Staab, Thomas Franz, Olaf Görlitz, Carsten Saathoff, Simon Schenk, and Sergej Sizov

ISWeb, University of Koblenz-Landau, Germany
{staab,franz,goerlitz,saathoff,sschenk,sizov}@uni-koblenz.de
http://isweb.uni-koblenz.de/

**Abstract.** Knowledge and information spanning multiple information sources, multiple media, multiple versions and multiple communities challenge the capabilities of existing knowledge and information management infrastructures by far — primarily in terms of intellectually exploiting the stored knowledge and information. In this paper we present some semantic web technologies of the EU integrated project X-Media that build bridges between the various information sources, the different media, the stations of knowledge management and the different communities. Core to this endeavour is the combination of information extraction with formal ontologies as well as with semantically lightweight folksonomies.

## 1 Management of Lifecycle Knowledge

Knowledge and information management faces larger challenges than ever, not in spite, but because of their successes. Spanning *multiple information sources*, *multiple media*, *multiple versions* and *multiple communities* challenges the capabilities of existing information management infrastructures by far — not primarily in terms of storage size and network bandwidth, but in terms of intellectually exploiting the stored knowledge. The rationale behind the growth of available resources lies in the possibility of digital recording and the objective of optimizing business processes wherever possible.

In the EU IST project "X-Media — Large scale knowledge sharing and reuse across media"[1] we approach these challenges for management of product lifecycle knowledge for complex industrial products (such as cars or jet engines) by targeting the

- Understanding of content from across multiple media formats and resources;
- Representation of dynamic and uncertain knowledge including provenance, change and process knowledge;
- Semantic user interaction tightly embedding paradigms of searching, retrieval, browsing, access and re-use into a knowledge sharing infrastructure.

[1] This work was funded by the X-Media project (www.x-media-project.org) sponsored by the European Commission as part of the Information Society Technologies (IST) programme under EC grant number IST-FP6-026978. The view of the authors does not necessarily represent the view of the X-Media project as a whole.

We target the management of lifecycle knowledge allowing for run-through knowledge extraction, capturing, representation, and re-use. In this paper we elaborate the scenario of problem analysis and resolution that we approach at the industrial use-cases partners of our project, Rolls Royce and Fiat. We highlight the semantic infrastructure technology that we estimate will help us to achieve our goals of product lifecycle management.

## 2 Scenario

The shortcomings of existing knowledge management systems become clear in situations that require complex analysis of the product lifecycle, such as resolution of non-trivial technical problems. What happens, when maintenance technicians or pilots perceive unusual sound or vibration of a jet engine? Usually, the relevant pieces of problem-related information are scattered across several systems, data repositories, and media types. Consequently, the systematic problem analysis requires expert knowledge and experience in different areas of the product lifecycle. The lack of flexible mechanisms for finding problems with similar provenance and identifying ways to resolve them in the past would force the manufacturer to ask through a number of experts or departments. It is not surprising that the resulting necessary consultations may delay the actual problem resolution by several weeks or months. What could improve the efficiency of this time-consuming procedure?

### 2.1 User Requirements

The complex problem analysis typically requires a flexible and integrated view on all aspects of the product lifecycle. Furthermore, the framework should allow the tight cooperation between manufacturer, product suppliers, and customers for flexible representation and sharing of their requirements and demands. This objective leads to following major user requirements.

*Integrated workflows.* The lifecycle of complex systems is documented in their individual history including maintenance reports, fault descriptions, records on operational conditions, etc. Every time the product (e.g. jet engine) is serviced or inspected, financial information is generated. If problems are found, pictures are taken, reports are written. It is clear that knowledge acquisition should be seamlessly integrated into all workflows of the product lifecycle.

*Tracking of provenance.* During lifecycle monitoring, much secondary knowledge is generated and exchanged regarding the sources of information, backgrounds and motivation of decisions made, or responsibilities for these decisions (e.g. what type of sensors was used, which maintenance operations preceded the anomaly, or why the technician decided to suspend the seemingly health engine). This provenance knowledge can be used for fault isolation, recalling common activities for frequent maintenance scenarios, framework accounting, or workflow optimization purposes.

*Knowledge sharing.* It could be expected that the continuously increasing amount of available information during the product lifecycle facilitates the product design and solving of observed problems. However, meaningful interpretation of this information typically requires a big portion of background knowledge, experience, and/or expert terminology that are not represented by the corresponding system at all. The knowledge exploitation framework should provide flexible mechanisms for knowledge sharing and interpretation.

*Ease of cross-media problem analysis.* Existing systems typically provide an isolated view on one type of media (e.g. sensor data, textual explanations, X-ray, thermal paint, or photography digital images). Each media type comes with domain-specific attributes and features that are crucial for understanding in the context of analyzed problem, such as recognition of deformations or surface erosion. Since particular issue attachments (email chains, textual problem reports, pictures of damaged parts) are usually stored separately, the cross-media analysis becomes the bottleneck of the problem resolution. The framework should provide systematic view on all relevant problem facets across different media types and data formats.

### 2.2 Technical Requirements

*Scalability.* The broad use of sensor networks allows fine-grained collection of physical parameters (temperature, sound, vibration, etc.) that characterize the operation of complex systems. The amount of data produced by modern monitoring tools is rapidly increasing. The records for each individual system describing its whole lifecycle can easily sum up to several Terabytes of information. An important requirement is therefore the high efficiency and scalability of interactive search/retrieval algorithms and expert tools.

*Flexible representation.* Monitoring physical properties is always incident with measurement errors that can be captured by averaging, smoothing, or providing tolerance/confidence intervals. Many sources of human knowledge, such as maintenance plans, user observations (e.g. pilot reports), or expert recommendations (e.g. problem resolution protocols), show some grade of uncertainty or are partially contradictive. Ideally, the framework should be able to cope with incomplete and uncertain knowledge and assist the expert user to navigate through the information space in a focused manner.

*Infrastructure.* An important aspect of the knowledge management framework is clearly the flexible architecture for sharing and re-use of expert knowledge. The framework should support collaborative knowledge organization beyond the borders of particular systems, expert groups, application scenarios, or media types. To this reason, it should offer to each expert user the full autonomy of information management and freedom of data organization. On the other hand, the framework should provide mechanisms for easy and instant knowledge sharing, e.g. by annotating all relevant pieces of the problem report with the

name of the responsible department, and associating group access with this annotation.

### 2.3 Focus of this Paper

It is clear that there is no simple solution to the introduced versatile complex of requirements. In this proposal, we restrict ourselves to the following major facets of the desired knowledge management framework:

- seamless *integration of knowledge* from different sources
- *flexible infrastructure* for knowledge representation for various application-driven contexts
- infrastructure for easy and instant *knowledge sharing* between experts, organizations, or platforms
- representation of expert *background knowledge* for better problem understanding
- *tracking of provenance* for decisions, changes, and data items
- ease of *knowledge acquisition* that should not require substantial additional efforts

On the one hand, our vision aims to overcome traditional limitations of data-driven information systems and addresses a wide range of applications with non-trivial requirements, such as product lifecycle management. On the other hand, the resulting technology forms the basis of the framework for knowledge sharing and re-use and provides direct support for its further components (such as information extraction or security mechanisms) .

## 3 Design Decisions

Following we present the design decisions made to cover the requirements of Section 2.3. Table 1 summarizes the stated requirements and indicates, which core technology addresses a specific requirement. Further, we explain the technologies employed, how they help us in achieving our goal and how they integrate with each other.

| Requirements \ Design Decisions | Tagging | Ontologies | Semantic Desktop | P2P | RDF |
|---|---|---|---|---|---|
| *integration of knowledge* | × | × | × | | × |
| *knowledge sharing* | × | | | × | |
| *flexible infrastructure* | | | | × | × |
| *knowledge acquisition* | × | × | × | | |
| *tracking of provenance* | | | × | | × |
| *background knowledge* | | × | | | |

**Table 1.** Requirements and technologies to cover them.

*Tagging.* Tagging denotes the association of keywords, so-called tags, with items of interest in a system. It proved beneficial in systems where a large community of users had to share content in a weakly controlled environment. Nonetheless it was shown that controlled vocabularies often emerge out of communities, which share interest in a topic [5] and we expect a similar behavior from users within our scenario working on joint problems. Further, tagging can dynamically adopt to new technologies or situations, while controlled vocabularies would need large effort in order to adopt them to a steadily changing environment. In our case, tagging specifically eases the *integration of knowledge*. Different sources of information (files, mails, ...) about the same subject can be tagged with same keyword, providing a semantically meaningful link between them. If tags are employed for *knowledge sharing*, a user can define access rights for remote users based on his own taggings and therefore distribute data implicitly. Finally, *knowledge acquisition* is done "on the fly", since the tags bear important semantics for the user and also within a community of users.

*Ontologies.* Ontologies are explicit conceptualizations of a shared understanding of domains of interest [8]. They usually model common terms and their relations agreed upon by a community, and can be viewed as controlled vocabularies with further support for reasoning. Ontologies offer important mechanisms for the *integration of knowledge*. By providing a common vocabulary and the relationships between terms, they assist expert users in finding suitable annotations. Moreover, ontologies also allow to infer some higher-level information from basic facts. Further, the *knowledge acquisition* is ameliorated by the use of ontologies, since it provides terminology that expert users are encouraged to share. This vocabulary itself can be considered as a product of collaboration within expert community and based on the taggings provided by particular experts or groups. Therefore, the problem-specific *background knowledge* might evolve, but might also be partly pre-defined, depending on specific requirements of the application domain.

*Semantic Desktop.* The Semantic Desktop [2] aims at providing a framework for the integration of applications and data based on metadata, so that links are kept between different files, mails and other sources of information. The *integration of knowledge* is one of the main goals of the semantic desktop. It is done in a way transparent to the user, i.e. the metadata is produced from within the usual applications, and not explicitly by additional third-party tools. Semantic desktop enabled applications support *knowledge acquisition* by providing metadata ether autonomously or with user interaction. An important aspect is *tracking of provenance* within such applications, i.e. storing information about the source of knowledge, such as the sender of a mail, the author of an document, or reasons for the particular decision.

*Peer-To-Peer.* A Peer-To-Peer (P2P) system is a decentralized network of computers (e.g. desktop PCs of expert users) without distinction between client and server functionality [7]. In other words, each machine acts both as a server and

as a client, depending on the action that it performs. Data intended for distribution within the P2P network is indexed by sophisticated algorithms that use decentralized data structures and is therefore efficiently available. A P2P infrastructure eases *knowledge sharing* by providing scalable and robust mechanisms for sharing of local knowledge sources with other users. The P2P architecture also offers the *flexible infrastructure* for our framework: no central storage and loose infrastructure of autonomous machines.

*RDF.* The *Resource Description Framework* RDF [6] is a graph-based model that allows for a flexible representation of metadata. Its schema-less model supports the integration of metadata from different, a priori unknown data sources. With RDF, integrating data sources is reduced to merging sets of triples, and applications only take those statements into account that are significant for them. RDF covers three important requirements as a data representation language. First, it eases the *integration of knowledge* due to its simple, graph based model. Second, it supports the *flexible infrastructure* on the data representation level, and we don't have to adhere to a static schema. Third, *tracking of provenance* can be also supported by RDF. If each item on the personal computer of a user is associated with an URI, provenance information can be easily represented by RDF statements. An important advantage is, that the resulting links become manifest in the data model.

## 4 Infrastructure

Based on the design decisions mentioned above we envision an architecture (Figure 1) for a networked semantic environment integrating semantic and legacy applications and allowing for easy, transparent exchange of metadata and content. A core element is a local RDF repository on each computer which is used to store metadata. Applications producing semantic metadata like the *semantics aware messenger SAM* directly use the repository. Legacy applications can access the repository via a filesystem interface called *SemFs*. A *semantic exchange architecture SEA* based on P2P technology allows for exchange of content and metadata. In this section we will describe each of these parts of the framework and give a usage example integrating the parts.

### 4.1 SemFs - a Semantic Filesystem

In order to integrate legacy applications into a semantic desktop we develop *SemFs*, a semantic filesystem, which provides a mapping from modifications of files and directory structures via the traditional filesystem interface to modifications of arbitrary information objects and corresponding annotations in an RDF metadata repository. The annotated information objects are not limited to local files, but can come from any information source, as long as they can be mapped into a file representation at access time.

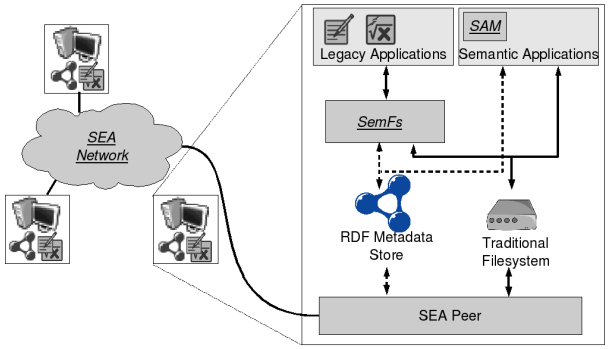


**Fig. 1.** A Networked Semantic Environment

SemFs translates filesystem paths into queries to functionally nested *views* on the underlying metadata repository. Views are predefined and can be parameterized. For example an 'author'-view could be defined, which would return all information objects having the author given as a parameter. The RDF graphs resulting from the filtering operations are mapped to a tabular structure containing names and URLs of the information objects. The names are then used for presenting the information objects as files in the filesystem. The URLs are used to dynamically retrieve the content of the information objects when the corresponding files are accessed. Possible query refinements are presented as subdirectories. The resulting filesystem is purely virtual: All files and directories are generated on-the-fly based on metadata and materialized only when accessed by a client application. When creating a file, placing it inside a directory adds all metadata to the repository, which is necessary to make the file appear within the view corresponding to the directory. Hence SemFs can be used for metadata based browsing as well as for annotation. As an example let assume that a fictional user Becky stores a report about engine number `E55-66-77` in a folder with the same name. Later, another user Chris looks for this report and remembers it was written by Becky. He opens the folder `author: Becky` and finds there all documents written by Becky. He refines his search by changing to the subfolder `E55-66-77` and finally obtains the desired report.

There are two benefits resulting from this flexible architecture: On the one hand we can transparently integrate local and remote information objects through a single, well known interface. On the other hand legacy applications can be provided with access to semantic metadata to a certain extent without having to modify the application. SemFs is described in more detail in [1].

### 4.2 SEA - a Semantic Exchange Architecture

To allow for autonomously exchange of annotated information objects without a centralized repository we develop the Semantic Exchange Architecture *SEA*. SEA provides easy exchange of information objects over a P2P network based on taggings. SEA works on the annotation produced by local applications, which

is stored in the RDF repository. Based on tags used for annotation the user can define access rights, for example "files tagged with `public` are freely accessible" or "files tagged with `E55-66-77` may be read by the users Adam, Becky and Chris". Corresponding to the mentioned examples we distinguish three kinds of data: *Private data* has no tags for which access rights are defined. *Publicly shared data* is freely accessible. *Protected data* is accessible by a restricted user group only.

For public data, tagging and location information are published in a structured P2P network using a distributed hash table, which guarantees that information can be found network-wide within a logarithmic number of hops. Protected data is searched for at a finite list of "known" peers, similar to a buddy list for an instant messaging application. This is sufficient, as the user granting access is expected to know the user she grants access to and vice versa. Shared data and annotation can be accessed via a REST API by local applications and remote peers. SEA is described in more detail in [3].

### 4.3 SAM - a Semantics Aware Messenger

SAM is an instant messaging tool - implemented as a plugin for the Spark[2] instant messenger - that is based on an ontology-based RDF representation for messaging data. Storing this data in a common repository allows for integration with desktop applications such as SEA or SemFs to improve message management. SAM features collective message tagging, i.e. messages can be annotated by multiple participants of a conversation. If one conversation partner tags a message, the tagging is automatically distributed to other participants of the meeting. The additional context information provided by taggings is used for message retrieval and to enable message collation, e.g. to summarize messages that address some particular problem or that are associated with some project or department. SAM has been implemented for the extensible messaging and presence protocol (XMPP) used by the Jabber network. The protocol has been extended to enable RDF data transfer and data request in form of SPARQL queries. SAM is our first step towards an integrated annotating communication infrastructure, which allows to retrieve knowledge produced during communication from archived conversations. We describe SAM in more detail in [4].

### 4.4 User Interaction

The infrastructure presented before integrates different data sources and applications to provide an environment for semantic metadata exchange and its collective management. In the following, the user interfaces that build upon this environment are introduced along the lines of the work setting described in Section 2. We consider here the purely fictional use-case scenario of problem analysis and resolution for the airline *ISWebFlights* that reports its trouble to the jet engine manufacturer.

[2] http://www.jivesoftware.org/spark/

*Vision of the Scenario Using ISWeb Tools.* The customer service associate Adam gets a phone call of a customer *ISWebFlights* that reports about potentially abnormal sound of one particular jet engine with serial number *E55-66-77*. Adam takes notes about this service request in a text file and uses the conventional *file save* dialog of his preferred text editor to save the file in the folder `ISWebFlights/serviceRequests/E55-66-77`. Adam then uses SAM to quickly write an instant message to the service engineer Becky to ask whether she can inspect the engine. Adam tags this message with `serviceRequests`, `E55-66-77`. Becky confirms that she can deal with the service request and utilizes the tags made by Adam to retrieve the request using SEA by browsing for the taggings made by Adam. When analyzing the engine, Becky takes pictures of the engine and records the engine's sound. She saves these documents on her harddisk under `E55-66-77` in the subfolders `audio` and `image`. She finds out that some loose filler was the problem of the abnormal engine by investigating the documents in the subfolders of `samples/loose filler`. She adds the existing taggings `loose filler` and `samples` to the instant message and other documents in the folder `E55-66-77` as another sample. Becky replaces the filler, writes a maintenance report that she saves in `E55-66-77/reports`, and writes an instant message to Adam that the engine problem is fixed. Since Adam might have forgotten which service request Becky dealt with, she also tags this message with `E55-66-77`. Upon retrieving the message, Adam forwards it to Chris who is responsible for the customer report and billing. Reading the forwarded message, Chris uses the tag `E55-66-77` that is attached to the original message to retrieve anything he can find for this tag with SEA: He retrieves Adam's service request, images, audio material, and Becky's maintenance report and collates this information to produce the problem resolution report and the bill for *ISWebFlights*.

*Browsing for Information.* SemFs transparently augments conventional file management interfaces, i.e. users continue to use conventional file management tools such as a file browser or *file open* and *file save* dialogs of legacy applications.

Besides functionality expected by file browsers, SemFs enables faceted file browsing, i.e. the same files can be retrieved through different access paths that resemble different user associations or perspectives.

Besides faceted browsing, the integration of multiple data sources allows to retrieve the same information from within different applications. For instance, a message tagged within SAM can be retrieved using SemFs by browsing for all information objects related to that tag. Furthermore, as messages are automatically associated to users (sender, recipients) one can also use SemFs to browse by user to find the message. On the other hand, from within SAM, file system content associated to a message is accessible.

*Organization of Knowledge.* Filing on the computer desktop means to deposit information, usually in a file for which a name needs to be created that describes the contained information. The file is then put into a particular folder that is created to summarize related information represented in further files. The purpose of filing is to improve later retrieval of filed information where more structure in

form of more sophisticated folder hierarchies can improve the retrieval process. Using the tools presented here, filing has the notion of annotating, i.e. adding metadata about some information object. This organizational paradigm is less complex than the creation of a folder hierarchy and provides more flexibility for browsing information as indicated before. In many cases, filing is not even required to support later retrieval as new information is already accompanied by useful metadata, e.g. the sender and recipient of a message. Users, however, can still be provided with user interfaces that allow them to add further metadata.

## 5 Conclusion

The aggregation and exploitation of lifecycle knowledge for complex industrial products is a hard and versatile problem. In this paper, we highlighted the semantic infrastructure technology that will help to achieve the challenging goals of knowledge management: acquisition, integration and sharing of knowledge across sources and platforms, its representation and provenance tracking for sophisticated application scenarios like analysis and resolution of non-trivial technical problems. The proposed networked semantic environment uses locally stored RDF repositories as a backbone for knowledge representation, the semantic exchange architecture SEA for knowledge sharing in a distributed P2P environment, the semantic filesystem SemFs for customizable knowledge representation, and the instant messaging toolkit SAM for expert exchange and background knowledge acquisition.

Our long-term objective is the integrated knowledge management framework for run-through support of knowledge extraction, capturing, representation, and re-use.

## References


1. S. Bloehdorn, O. Görlitz, S. Schenk, and M. Völkel. TagFS - Tag Semantics for Hierarchical File Systems. In *International Conference on Knowledge Management*, 2006.
2. S. Decker and M. R. Frank. The Networked Semantic Desktop. In *WWW Workshop on Application Design, Development and Implementation Issues in the Semantic Web*, 2004.
3. T. Franz, C. Saathoff, O. Gorlitz, C. Ringelstein, and S. Staab. SEA: A Lightweight and Extensible Semantic Exchange Architecture. In *Proceedings of the 2nd Workshop on Innovations in Web Infrastructure. WWW Conference*, 2006.
4. T. Franz and S. Staab. SAM: Semantics Aware Instant Messaging For the Networked Semantic Desktop. In *Proceedings of the 1st Workshop On The Semantic Desktop. ISWC*, 2005.
5. S. A. Golder and B. A. Huberman. The Structure of Collaborative Tagging Systems. *The Journal of Information Science*, 2006.
6. F. Manola and E. Miller. RDF Primer, February 2004. http://www.w3.org/TR/rdf-primer/.
7. S. Staab and H. Stuckenschmidt. *Semantic Web and Peer-to-Peer*. Springer, 2006.
8. S. Staab and R. Studer. *Handbook on Ontologies*. Springer, 2004.