Aletheia – An Architecture for Semantic Federation of Product Information from Structured and Unstructured Sources

Matthias Wauer
TU Dresden
Computer Networks Group
Dresden, Germany
matthias.wauer@tu-dresden.de

Daniel Schuster
TU Dresden
Computer Networks Group
Dresden, Germany
daniel.schuster@tu-dresden.de

Johannes Meinecke
SAP Research CEC Dresden
Dresden, Germany
johannes.meinecke@sap.com

ABSTRACT

Product-related information can be found in various data sources and formats across the product lifecycle. Effectively exploiting this information requires the federation of these sources, the extraction of implicit information, and the efficient access to this comprehensive knowledge base. Existing solutions for product information management (PIM) are usually restricted to structured information, but most of the business-critical information resides in unstructured documents. We present a generic architecture for federating heterogeneous information from various sources, and argue how this process benefits from using semantic representations. An reference implementation tailor-made to business users is explained and evaluated. We also discuss several issues we experienced that we believe to be valuable for researchers and implementers of semantic information systems, as well as the information retrieval community.

Categories and Subject Descriptors

H.3.4 [**Information Storage and Retrieval**]: Systems and Software; H.4.0 [**Information Systems Applications**]: General

General Terms

Design, Management, Experimentation

Keywords

Federated Information System, Product Information Management, Ontology, Semantic Web, Information Extraction, Architecture

1. INTRODUCTION

Product-related information is generated, accessed and manipulated along the product lifecycle in heterogeneous formats. Only part of this information can be accessed using state-of-the-art product information systems as large parts of this information are only available in unstructured sources or distributed along different databases and legacy systems. The challenge to create an all-embracing view on products is huge. Such a comprehensive product information system has to integrate and harmonize data from all phases of the product lifecycle, all different source formats like unstructured documents, sensor information or product databases as well as to even cross organization boundaries as different stakeholders may be responsible for the design, production, delivery, and service of a product.

The Aletheia project [1] is a unique attempt to bring together industry partners (ABB, BMW, Deutsche Post DHL, Otto, SAP) with five innovative application scenarios from different phases of the product lifecycle and five different landscapes of current state-of-the-art product information management. All these partners have a keen interest in improving the information flow internally as well as with their customers and partners and to open up new sources of product-related information like Web 2.0 pages.

In this paper we try to answer the research question if it is possible to federate structured as well as unstructured sources of product information along the product lifecycle. We use semantic technologies for this purpose and deploy and advance information extraction techniques. The scenario in Section 2 describes one of the use cases of the Aletheia project clarifying the opportunities of federated product information systems (FPIS). We further discuss requirements derived from this and other scenarios. A discussion of existing architectures for semantic information management and federation in Section 3 shows the need for a new architecture matching the requirements mentioned in Section 2.2. The contributions of this paper consist of

1. A high-level **component architecture for FPIS** in Section 4.1 including a concept for data sharing between organizations,
2. A detailed **concept of the Aletheia Service Hub** (Section 4.2), our central component for information federation within organizations,
3. An overview of our current **reference implementation** in Section 5.

Section 6 contains a discussion of the results achieved so far.

2. SCENARIO

In order to motivate our research, we discuss a scenario in the industrial sector. It is derived from a case study conducted in the Aletheia project, focusing on product lifecycle management (PLM) at ABB, a large company providing power and automation products, technology, and service. In addition to this company, the scenario includes a customer from the chemical sector that uses the company's products and services, and a logistics provider that stores and ships the company's spare parts. A similar use case is studied in more detail in [10].

2.1 Use Case

The customer has installed several products of the company at their local site. A service team of the customer notices that one of the devices is defective. Even though they have the knowledge which devices are applied at this installation, they lack the capability of identifying the actual cause and repairing the device. Hence, they contact the company's call center that records the service request. However, neither the customer's service team nor the call center associate have expert knowledge about the defective device. The service report therefore is frequently inaccurate, and preparing the service operation is laborious for an assigned service technician on the basis of this report.

On site, most of the suitable unstructured information is not consulted by the service technician because finding it based on the available information is cumbersome. If additional spare parts are required to repair the device, the service technician has to manually coordinate the order of a spare part and its delivery with the company's call center, the logistics provider, and the customer's service team. This causes several phone calls and requires much effort because the service technician's available information is not integrated with those of the other parties.

This use case can be optimized by three means. First, the customer should be able to solve well known problems with defined solutions without the need to consult a company's service technician. Aletheia can support this by providing such information related to the customer's actual installed base and corresponding historical information. On top of that, the different vocabulary of customers and service documents can be translated with semantic search. Second, this previously collected information is useful to more accurately define detailed problem descriptions for identifying the most appropriate service engineer, who can find related information from unstructured documents easier if they are extracted and semantically related to the respective device and problem symptoms. Third, the federation of RFID and sensor data correlated to the defective device can improve failure analysis by providing all relevant information, while the connected information helps assisting processes like ordering spare parts between the involved parties.

2.2 Requirements

Out of this and other scenarios from the use case partners we identified a large number of requirements. The requirements were acquired on-site in 2-day workshops at each industrial partner. The resulting large set of requirements then has been further analysed and condensed to the following main categories:

Req 1: Federated Information Retrieval

The central functionality users want to use an FPIS for is federated information retrieval, i.e. formulating an information need and retrieving relevant results from the FPIS. There are many features similar to classical search engines like natural language queries, auto-complete, clustering of results, personalization, and faceted search that users expect from an FPIS. Beyond that, we also identified FPIS specific requirements: search results that mix up document links and ontology facts relevant to the search query. A sort of federated ranking method is needed to bring order to this result list. Furthermore, one should be able to restrict the sources to search for by a query or some kind of intelligent source selection method should identify the best sources for each query.

Req 2: Information Exploration

Besides the retrieval part, users should also be able to navigate through the information space created by an FPIS and to explore connections between different information entities, documents, and related concepts. Information exploration and information retrieval can also be mixed up as exploration may be refined by a query as well as vice versa.

Req 3: Information Integration

Federation of product information means integrating existing sources of information that were formerly used separately to create an all-embracing view on all product-related information. Thus a large number of requirements targets on using existing databases and make them searchable within the FPIS. Other types of information sources include file shares with formatted documents such as Word, PowerPoint or PDF and information from the Internet of things, i.e. RFID and sensor data. Information integration includes appropriate mapping schemas, the management of access policies for the different sources as well as the actual access technologies like Web service interfaces or the like.

Req 4: Information Extraction

Full-text indexing of unstructured information (i.e., documents on file shares as well as websites) is not enough to reach the goal of semantic federation of product information along the product lifecycle. Information extraction techniques are needed to obtain information from unstructured documents. This mainly means (but is not restricted to) Named Entity Recognition (NER) to recognize entities with different keywords but belonging to the same semantic concept. Meta information should also be extracted from the documents to improve the relevance assessment.

Req 5: Information and Ontology Management

Once an FPIS gets deployed we also need means to directly manipulate the information presented to the user. It might be incorrect or important information may be missing.

This may optionally require update mechanisms to populate changes made in the FPIS back to the information sources. Another important point is the ability to easily manipulate the ontologies used to realize the information integration and information extraction.

Req 6: Information Sharing

Interestingly, the aspect of information sharing between organizations did not play an important role in the interviews. Only in the ABB case we actually found a use case where we need sharing of information as partner companies do part of the service for machines on behalf of ABB. But if FPIS will get used in organizations, the need for information sharing will soon arise and will be the next step in the evolution of FPIS. If we really want to create an all-embracing view covering all phases of a product's lifecycle, we need to share at least part of the information between a product's designer, producer, retailer, logistics provider, and service provider.

3. RELATED WORK

An early approach to federated search was presented as the Information Manifold [9]. The system uses source descriptions, describing contents and capabilities of different structured information sources, in order to determine appropriate execution plans for a query. In contrast to Aletheia, unstructured information is not discussed. Furthermore, the approach assumes a global schema, referred to as world view, to federate the information. Aletheia should instead provide the means to integrate information based on different models, as stated in requirement 3.

The complex nature of the presented requirements led to the investigation of Semantic Web technologies in order to handle the complex task of relating the federated information. In this context, the NeOn project provides a generic architecture [15] for ontology-driven applications. It separates the required services for ontology engineering and ontology usage with a clear focus on the engineering part. The aspects that are most important regarding Aletheia, such as the interaction of the core services and the extraction of information from the data sources, is not covered in detail. Instead, the creation and maintenance of semantic information is the key aspect of NeOn. With its focus on the usage of federated information, Aletheia instead needs to define components and interfaces that not only handle semantic information, but also integrate them with uncertain information that has been extracted from unstructured sources. This also requires the definition of appropriate services that enable access to this comprehensive knowledge base.

Regarding unstructured information, SMILA (Semantic Information Logistics Architecture) [14] presents a simple data extraction model for different unstructured sources and an architecture based on OSGi, SCA, and BPEL. This allows for dynamically switching extraction components and flexible management of the execution depending on specific use cases. Compared to the Aletheia requirements, it does not connect this data with structured information, and any semantic processing is designed to be executed on a higher level. Indeed, the ontology store proposed by the architecture is rather a wildcard for further extensions.

With regards to the extraction of information, the Unstructured Information Management Architecture (UIMA) [7] acts as a blueprint for the extraction components of Aletheia, separating the information access, analysis, and acquisition aspects. Some design decisions of UIMA, such as the annotation of metadata as simple sets of key-value pairs, may have to be revisited in order to gain major benefit of this data for the integration process, as explained in requirement 3.

Gaining precise knowledge from different sources is the objective of YAGO [8], which relies on few core sources that are assumed to provide correct information and semantically connects this information. Core extractors use rules to derive the knowledge base. The extracted facts are further restricted to those validated using the WordNet taxonomy. In a second two-step process, additional information is gathered from Web resources that is then judged with regards to the existing knowledge base. Although we generally follow a similar approach, the presented Aletheia use case would not benefit from publicly available but irrelevant core sources like Wikipedia, as intended by YAGO.

Considering distributed semantic information, projects like SemaPlorer [13] have shown the benefits of federating such data sources. It proposes the use of NetworkedGraphs, providing distributed views over RDF datasets that can be queried using SPARQL, and presents scalable reasoning by preprocessing a transitive closure of the SKOS hierarchies of configured datasets. Again, it only partly supports the requirements of Aletheia, as it neither connects arbitrary sources, nor is there any distinction between public and confidential information. Related to that, but based on a different motivation of the social semantic desktop, the NEPOMUK project [12] developed a distributed search system and different ontologies suitable for ordinary desktop entities. It proposes a P2P architecture for distributed storage and indexing of documents, but does not address the issue of actually extracting semantics from these documents. Thus, it does not address the issue of actually connecting heterogeneous information.

Focusing on product information, Brunner et al. [6] examine the use of semantic technologies in the context of Master Data Management (MDM). They argue that a subset of OWL DL is sufficient for most product information management scenarios. Furthermore, they present a generic metamodel for defining scenario-specific product information as well as a basic architecture for processing such product information. Although it shows how product information can be management semantically, it does not explain how to keep the complexity of the ontology from the user. The integration of existing data sets is not discussed either.

All of the presented research only solves part of the requirements. Hence, the architecture proposed subsequently aims to provide an integrated approach for a FPIS.

4. REFERENCE ARCHITECTURE

We identified four major entities that comprise a reference architecture fulfilling the made requirements, which are described in detail in the following. The central component connecting each of these entities is then explained in detail.

4.1 Components and Information Flow

Figure 1 illustrates the architecture of an Aletheia system. With regards to requirement 6 (information sharing), it includes the connection of instances across different organizations and departments. It is comprised of the following major components:

Application Servers enable clients to access the Aletheia system and maps domain-specific functionality to the generic interfaces. This includes a user-centered preparation of the available information. Both stand-alone and Web applications are supported as front ends.

Aletheia Service Hubs (ASH) are the key component for managing the semantic model, indexed documents, and stored facts and metadata that are stored inside the repository component. They also connect the clients to the information sources and can be connected to each other across organizational boundaries, with distributed query processing as in [16].

Information Providers act as wrappers to existing data and information sources that should be leveraged by the Aletheia system, and facilitate both push and pull access depending in the type of source. They are the components that actually access the data sources.

Registries store information about each ASH (external registry interface) and the connected information sources and services (internal registry interface), enabling discovery for more or less close cooperations between different departments and organisations.

With regard to distribution, each of the Aletheia Service Hub is the central node of a generally closed system that can be connected to external parties due to defined terms and conditions. This decision was an implication to the data sovereignty required by all the industry partners. Nevertheless, the platform may be configured to provide publicly available information via various channels, particularly with regards to linked data [4].

As the ASH performs several tasks, its specific composition is explained in detail subsequently.

4.2 Aletheia Service Hub

A more detailed view on the Aletheia Service Hub's components, corresponding to the architecture, can be seen in Figure 2, a fundamental modeling concepts (FMC) block diagram. Here, the client components are shown at the top, whereas the data sources appear at the bottom of the architecture. This detailed architecture can be separated into different layers. Please note that many connections have been left out in order to improve readability.

Front End Services. These Web services are supposed to abstract the user queries from the technical implementation of the repository, hence reducing the complexity of the system from the client's point of view. The major task of finding federated semantic information is supported by two services: the facts search service provides faceted search for extracted and stored knowledge, and a document search services enables full-text index search including semantic restrictions. Clients can add or extend the stored knowledge using the annotation service, which can occur for personal comments or incomplete automated extraction processes. In order to customize the service to specific user demands, preferences can be set via the personalization service. Finally, the current user context is handled by the session service.

Repository. Due to the different types of stored information, we conceived a combination of different repository components. The general and domain ontologies, stored facts, and the reasoner component are considered part of the semantic repository. This part of the repository is capable of managing different modules, which can be exploited for storing the individual ontologies of the participants for a certain use case. In addition to that, the uncertain information repository manages knowledge that has been automatically extracted using technologies such as natural language processing (NLP), i.e. it has a certain probability and cannot be safely trusted like facts. For handling full-text search, a syntactic repository is integrated as well. Finally, user context is designed to be stored separately.

Data Integration Layer. The actual federation of information is managed by the data integration broker, which includes a publish-subscribe service, enabling data provider components to send events in case of updated or newly available information. Unstructured information is also handled by this layer. Preprocessing and interpretation of provided documents can be configured by a processing pipeline, which is then fed into the uncertain information extractor applying appropriate algorithms for the extraction process. The result of this process might need to be transformed before it can be transferred to the repository by the dispatcher component, which also handles conflicts.

Information originating from the Internet of Things, such as wireless sensor networks (WSN), are attached using a gateway providing XML data. The actual sensor data is not embedded, but linked in the XML document because this can make up significant amounts of data that can rarely and does not need to be stored efficiently in the proposed repository. The platform still obtains all the necessary metadata available.

Vertical Services. Several component have not been discussed that are, however, crucial for the platform's functionality in real-life business environments. These services cannot be filed into the discussed layers because they are required throughout the platform. This includes the registry service for lookup of available information providers and platform functionality. Authenticating and authorizing access is also required for both client access and data federation, hence these services are provided as a separate component accessible by all layers. The adaptation service acts as a helper to allow for more personalized information provided by the individual front end services. In order to realise the intended cross-domain interactions, the Aletheia connector component enables synchronization between dif-

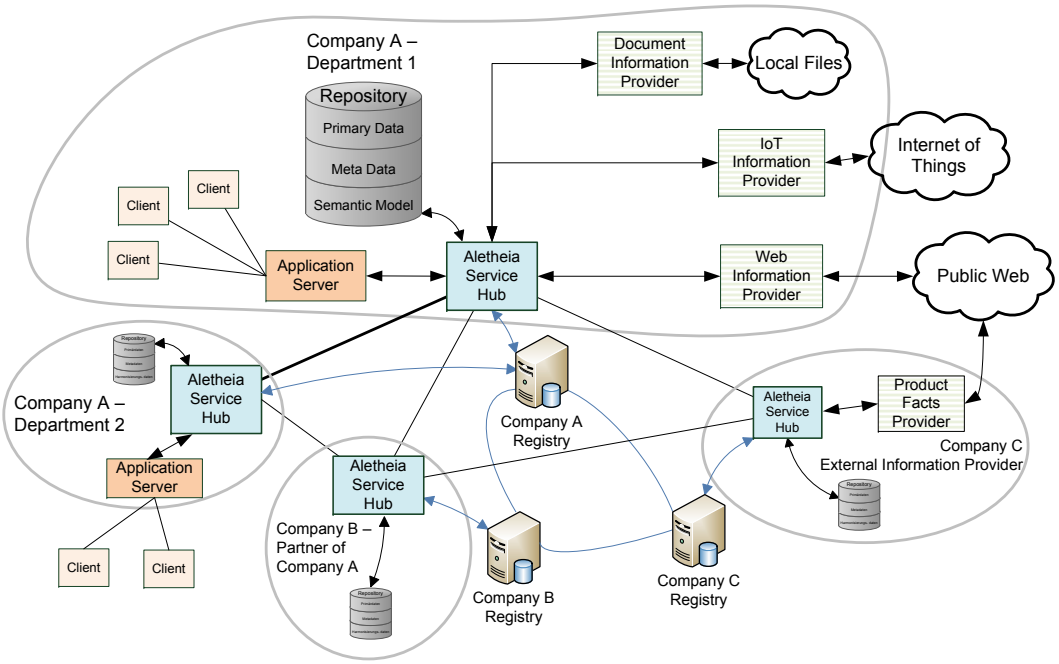

Figure 1: Aletheia reference architecture

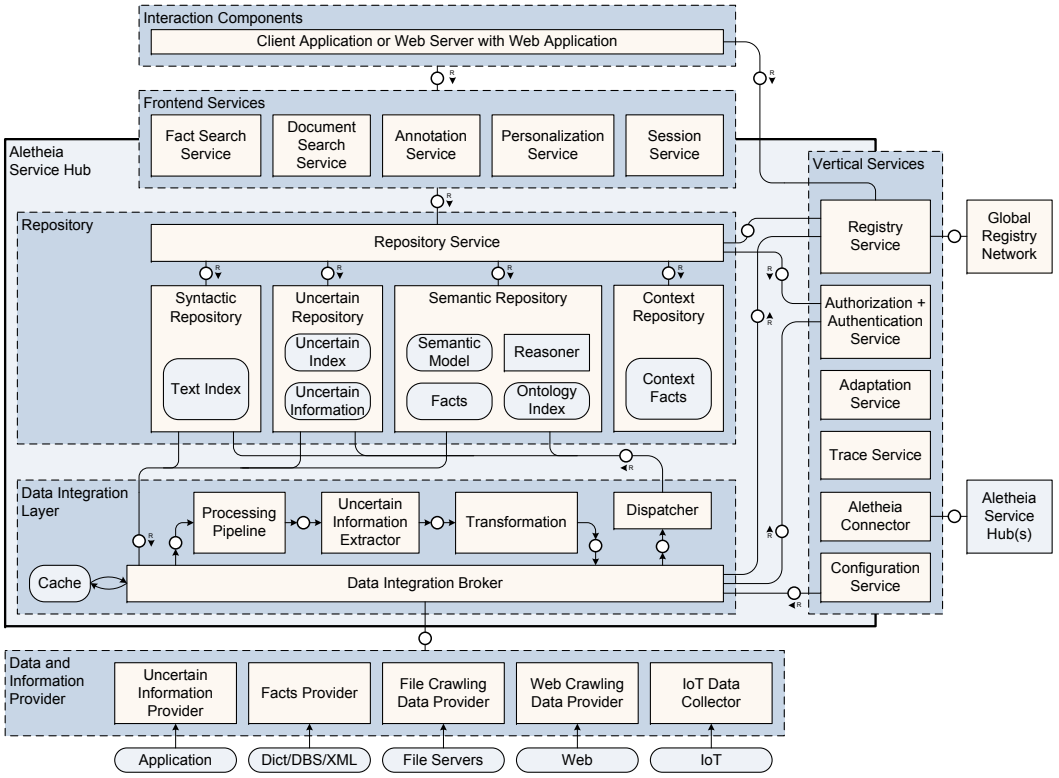

Figure 2: Architecture of the Aletheia Service Hub

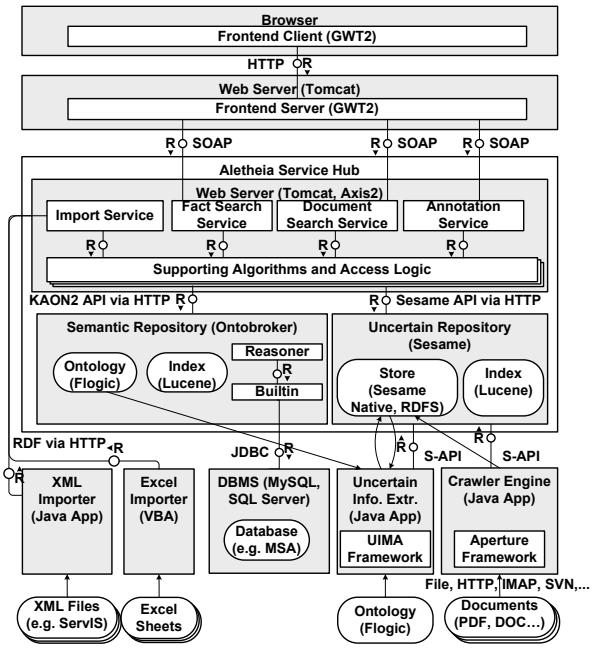

Figure 3: Overview of implemented components

ferent Aletheia Service Hubs. The configuration service actually is a front end service, but can access and modify settings on all the relevant system components, including the data federation. All the interactions between those services are logged by a trace service, which allows to debug the processes and improves monitoring.

5. IMPLEMENTATION

The presented architecture has been implemented with the most important components as a fully-working prototype within the Aletheia project and applied to real application partner data. Figure 3 gives an overview of the realized components and the technologies used for them. This implementation architecture can be seen as one possible instance of the reference architecture presented in section 4.

The front end of the Aletheia prototype has been realized as an AJAX Web application based on Google Web Toolkit (GWT2). This asynchronous user interface technology was chosen to account for the need for sending multiple complex semantic queries to the Service Hub. Hence, partial results can be presented to the user before all queries complete, greatly improving the user experience. The Aletheia Service Hub currently encompasses 3 major services for importing facts into the repository, searching for structured information and searching (semantically indexed) documents. As syntactic and uncertain repositories, we have integrated an Ontobroker [2] and a Sesame store [5]. With these choices, we take advantage of the Ontobroker support for querying large numbers of instances over live data sources (e.g. relational databases over JDBC) and of the interoperability of Sesame with the Aperture extraction framework [3] applied inside the Web crawling data provider (crawler engine). The semantic document annotation pipeline of the data integration layer is implemented in the uncertain information extraction component based on the UIMA framework. Other facts providers have been integrated via the import service's very lightweight interface that can easily be accessed from many different platforms, including from Microsoft Excel macros.

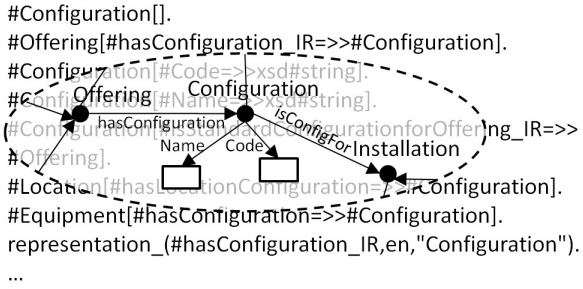

Figure 5: Clip of the ontology, as implemented with FLogic

In order to support the different domain models of individual use case participants, we utilize the Ontobroker capability of managing several ontologies in terms of multiple modules and respective namespaces. For example, the prototype can be switched from an ABB centric ontology to one that has been developed at BMW, simply by an appropriate selection in the front end.

The front end supports users in interacting (searching, navigating, annotating) product-related information as if the underlying data sources were one system. The underlying UI paradigm is to support complex structured queries (to take advantage of the semantic relationships), while not forcing the user to think in terms of complex queries (to account for the analyzed non-IT user needs in the PLM domain). The screenshot in Figure 4 shows an example where a user searches for configurations of machines used in the Chemical industry branch with 800 it their name.

The search terms are entered in a similarly simple way as users know from public Web search engines. However, to leverage the semantic model underneath, auto-complete suggestions with terms from the domain are offered to the user. This is supported by the fact search service that uses the ontology index of the semantic repository. When entered, the keyword query is than interpreted by the fact search service as a structured query, depending on disambiguated instance names, class names, role names and free-text terms. This is then executed by the semantic repository over the ontology designed collaboratively by domain and ontology experts. An excerpt of the ontology, modeled in FLogic, is shown in Figure 5.

The reasoner executes the query and, based on mapping rules, decides, for which sub queries to perform database SQL queries via the JDBC builtin (similar as in [2]). Other facts necessary for the query may be permanently stored in the semantic repository, e.g. imported from an XML dump of a legacy application via the XML importer before query time. After query execution, graphs are generated that ex-

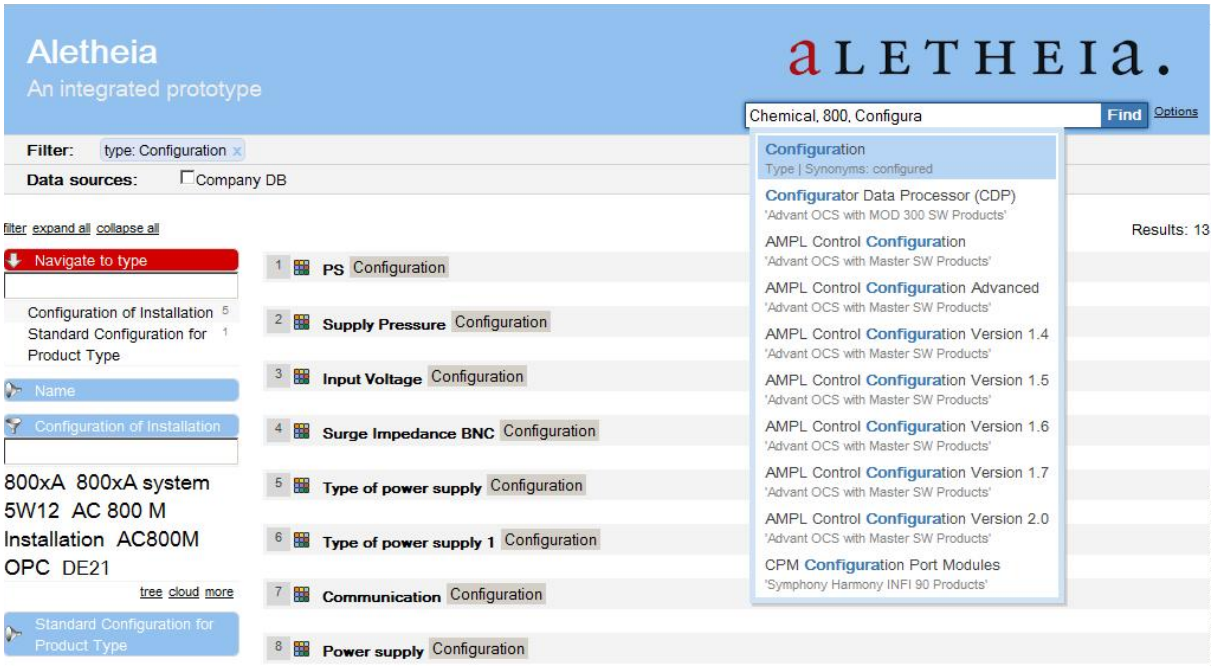

Figure 4: Screenshot of the ABB scenario prototype, showing a search example

plain to the user, why each particular found instance is considered to be a match for the entered keywords. This explanation is shown to the user as a mouse hover box. The same search box can also return documents from the uncertain repository. In this case, the query is interpreted by the document search service, not as a structured query, but as a vector of free text terms (via the document full-text index in Lucene) and disambiguated URIs (via the semantic annotations in the RDFS store).

6. EVALUATION AND DISCUSSION

Although the work on the reference implementation of Aletheia is still in progress, we already evaluated the initial results with regards to aspects such as usability, search functionality, data sources, interfaces, and added value due to the semantic technologies.

6.1 Evaluation

The evaluation involved all Aletheia project members and has been conducted using a survey on the prototype variants built for all of the five scenarios. With ten responses and a total of 59 rated criteria, this initial feedback was mixed:

- In summary, an earlier version of the implementation in Section 5 received 26 positive, 15 negative and 15 "partially fulfilled" ratings, with regards to the collected use cases and requirements. Here, partially fulfilled means that the evaluator did see some of the capabilities implemented w.r.t. a certain requirement, but not in its entirety that is expected for a final prototype.
- Several of the individual solutions that have been implemented for scenario-specific use cases, such as the recognition of product terms in full-text requests and graph visualizations, received very positive feedback, both for highlighting the benefits of semantics and usability of the search function. Hence, we target to include them as components to the reference implementation.
- As expected, some evaluators criticised that not all potential information sources are integrated already. They evaluated a comparatively early version of the prototype. Since then, the available data providers have been extended considerably.
- The benefit of semantics is not obvious for a few use cases related to the Internet of things. The implementation did not utilize this information in conjunction with the stored knowledge, and the higher-level integration is clearly necessary to show the advantage.

We will present the results of a more comprehensive evaluation to be executed later this year. Hence, we rather discuss the lessons learned to date with the development of this FPIS.

6.2 Discussion

As shown above, the current reference implementation exhibits many of the benefits of the intended Aletheia system, but is not completely finished with regards to the requirements and designed components.

Although the architecture proposes the federation of at least structured information at the time a client actually poses a query, we noticed that this is difficult to accomplish. For the auto-complete functionality presented in Figure 4, an index of the name and other properties a semantic entity can

be referred to *must* be available in the system in order to meet response time constraints (latency). This functionality is present in the employed semantic repository implementation, but is limited to the facts stored in the repository. Even though the repository federates information from other structured sources like relational databases, it does not make them available on this index unless this data is materialized, i.e. replicated to the repository.

Uncertain semantic information are separated from syntactic data, e.g. a full-text index, in the proposed architecture. While this is a sensible decision due to the different nature of those repositories, we employed the Sesame LuceneSail [11] component which combines both aspects. Hence, the management of extracted facts like `<Document1> <isAbout> <DriveComponentA>` is accomplished by the same component that stores the full-text index of that document.

We further noticed that the traceability of federated information is difficult. This is due to the overhead of storing provenance information for every fact in the repository, and gets even more complicated if the information should be annotated with confidence or trust ratings. While initially considering RDF reification, we are also studying whether named graphs can be used appropriately for such annotations. The requirement of federated ranking will benefit from such reliable confidence assertions.

The authentication and, to a greater degree, the authorization of users to access certain information remains an issue. Considering the federation during a client's request, existing single-sign-on (SSO) solutions can be applied. As the auto-complete issues have shown, the ad-hoc federation is an approach that causes several difficulties. However, federating the information prior to actual requests and, hence, not requesting the original information sources requires the repository to keep track of access rights to individual information. Additionally, this causes the data providers to determine and relay this authorization information in the first place. This is a critical aspect for all studied scenarios, and is subject to current research.

Nevertheless, the current implementation provides a complete "vertical cut" through the architecture, integrating different heterogeneous data sources. It also proved to be applicable to different domains by means of switching the semantic model, i.e. the ontology, and connected data sources.

7. CONCLUSIONS

In this paper, we have presented an architecture that supports the federation of heterogeneous information, originating from various data sources and arising throughout the product lifecycle. We propose this solution with regards to the limitations of current product information management products. These are less flexible than this semantic approach and usually only cover some aspects of the product lifecycle.ted an architecture that supports the federation of heterogeneous information, originating from various data sources and arising throughout the product lifecycle. We propose this solution with regards to the limitations of current product information management products. These are less flexible than this semantic approach and usually only cover some aspects of the product lifecycle.

Prior to that, we generalized a number of requirements derived from multiple real-life scenarios. The proposed architecture and associated reference implementation enables the exploration of information, both using semantic search and exploring related information. Many of the required data sources have already been integrated, based on generic solutions like XML to RDF transforms, in order to provide the desired all-embracing view. Existing frameworks for information extraction are attached that integrate unstructured information, using the respective domain ontology and existing knowledge. The presented information can also be modified by the users of the system, although the modifications are not yet pushed back to their origin. Finally, we presented our vision of enabling collaboration of such federated product information systems between different organisations, which is a requirement for exploiting the full potential of such a solution.

Compared to a previous publication [17], this paper presents the proposed architecture in much more detail and also discusses several research questions that we experienced developing the reference implementation.

8. ACKNOWLEDGMENTS

This work was partly funded by the German Ministry of Education and Research under the research grant number 01IA08001.

We would like to thank many Aletheia project members contributing to this architecture and reference implementation, including but not limited to Thomas Janke, Tobias Münch, Sandro Reichert, Robert Rieger, and Maximilian Walther, as well as ABB for fruitful discussions on use cases and requirements.

9. REFERENCES

[1] Aletheia project consortium. Aletheia - semantic federation of comprehensive product information. http://www.aletheia-projekt.de/, 2010.

[2] J. Angele and M. Gesmann. Data Integration using Semantic Technology: A use case. In *Rules and Rule Markup Languages for the Semantic Web, Second International Conference on*, pages 58–66, 2006.

[3] Aperture Framework. A Java framework for getting data and metadata. http://aperture.sourceforge.net/.

[4] C. Bizer, T. Heath, and T. Berners-Lee. Linked data–the story so far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22, 2009.

[5] J. Broekstra, A. Kampman, and F. Van Harmelen. A Generic Architecture for Storing and Querying RDF and RDF Schema. *The Semantic Web–ISWC 2002*, pages 54–68, 2002.

[6] J.-S. Brunner, L. Ma, C. Wang, L. Zhang, D. C. Wolfson, Y. Pan, and K. Srinivas. Explorations in the use of semantic web technologies for product information management. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, pages 747–756, New York, NY, USA, 2007. ACM.

[7] D. Ferrucci and A. Lally. UIMA: an architectural approach to unstructured information processing in

the corporate research environment. *Nat. Lang. Eng.*, 10(3-4):327–348, 2004.

[8] G. Kasneci, M. Ramanath, F. Suchanek, and G. Weikum. The YAGO-NAGA approach to knowledge discovery. *SIGMOD Rec.*, 37(4):41–47, 2008.

[9] T. Kirk, A. Y. Levy, Y. Sagiv, and D. Srivastava. The information manifold. *Proceedings of the AAAI 1995 Spring Symp. on Information Gathering from Heterogeneous, Distributed Environments*, pages 85–91, 1995.

[10] S. Kunz, F. Brecht, B. Fabian, M. Aleksy, and M. Wauer. Aletheia–improving industrial service lifecycle management by semantic data federations. *International Conference on Advanced Information Networking and Applications*, pages 1308–1314, 2010.

[11] E. Minack, L. Sauermann, G. Grimnes, C. Fluit, and J. Broekstra. The sesame lucenesail: Rdf queries with full-text search. *Technical Report 2008-1, NEPOMUK*, 2008.

[12] G. Reif, T. Groza, S. Scerri, and S. Handschuh. Final NEPOMUK Architecture – Deliverable D6.2.B. Public deliverable of the NEPOMUK project, Dec 2008.

[13] S. Schenk, C. Saathoff, S. Staab, and A. Scherp. Semaplorer-interactive semantic exploration of data and media based on a federated cloud infrastructure. *Web Semant.*, 7(4):298–304, 2009.

[14] T. Schütz. D11.1.1.b concept and design of the integration framework. Public deliverable of the Theseus-Ordo project, Sep 2008.

[15] T. Tran, P. Haase, H. Lewen, O. M. Garcia, A. Gomez-Perez, and R. Studer. Lifecycle-support in architectures for ontology-based information systems. In *Proceedings of the 6th International Semantic Web Conference (ISWC'07)*, November 2007.

[16] T. Tran, H. Wang, and P. Haase. SearchWebDB: Data web search on a pay-as-you-go integration infrastructure. Technical report, University of Karlsruhe, 2008.

[17] M. Wauer, D. Schuster, J. Meinecke, T. Janke, and A. Schill. Aletheia - towards a distributed architecture for semantic federation of comprehensive product information. In *Proceedings of IADIS International Conference WWW/Internet*, Rome, Italy, 2009.