UIMA and WebContent: Complementary Frameworks for Building Semantic Web Applications

Gaël de Chalendar

CEA LIST
F-92265 Fontenay aux Roses
Gael.de-Chalendar@cea.fr

1 Introduction

The main data sources available on the Internet today and even internally in companies are in very large part textual. Furthermore, the portions that are more controlled (structured) are often heterogeneous, relying on different formats and different ontologies. The development of Web applications is extremely complex and often requires unavailable programming skills or budget. It also requires the use of a wide variety of information processing modules. While building our original functional map, the WebContent consortium identified thirty such modules, including: data collection, acquisition, storage, indexing, transformation, normalization, description, ontology-based annotation, visualization, structuring, ordering, dissemination, sharing. Solutions come from various vendors and research laboratories, handle documents and metadata in various formats, and can take different forms: stand-alone applications, libraries, and Web applications. Integration is thus a serious problem. To overcome this issue, the WebContent project[1] has designed a document exchange format, a set of Web services specifications and guidelines to effectively build Semantic Web applications using distributed tools from various vendors.

UIMA offers an infrastructure in order to develop unstructured information management applications, allowing to chain together processing tools around data structures built upon a standardized API.

Both approaches are complementary because they handle well problems located at different levels of the software engineering stack. UIMA is good at the media (often text) processing chain and WebContent is well crafted for the middleware level. This paper presents the WebContent platform to the UIMA community and the way both frameworks will be used together in the Scribo project to build collaborative document annotation software.

2 The WebContent Platform

Figure 1 illustrates the overall architecture of a WebContent application. The platform presents a service integration infrastructure organized around a data exchange XML and

[1] The WebContent project was supported by the French National Research Agency (ANR) through the RNTL program. Its complementarity with UIMA is explored in the Scribo project funded by the Free Open Source Software Working Group in the System@tic-Paris-Region competitiveness cluster.

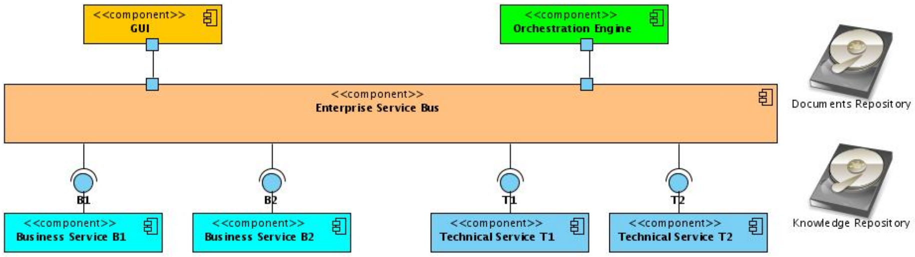

Fig. 1. Architecture of a WebContent application.

RDF-based format. WebContent applications are typically designed and implemented as Web portal applications, using portlets to build powerful graphical user interfaces.

Each service is independent from the others. It is specified by its interface and use conditions; it is normalized, the intention being to open the normalization process to a broader community. Similar components, providing the same kind of services, possibly based on different technologies, can be implemented by different suppliers. To accomplish a particular business objective through a complex process, several atomic services have to be combined. For that, we use a WS-BPEL (Business Process Execution Language) engine. BPEL enables the description of a complex business process using standard operations such as service invocations, conditional blocks, loops, etc. High flexibility is achieved with multi-level service composition and the ability to select endpoint at runtime.

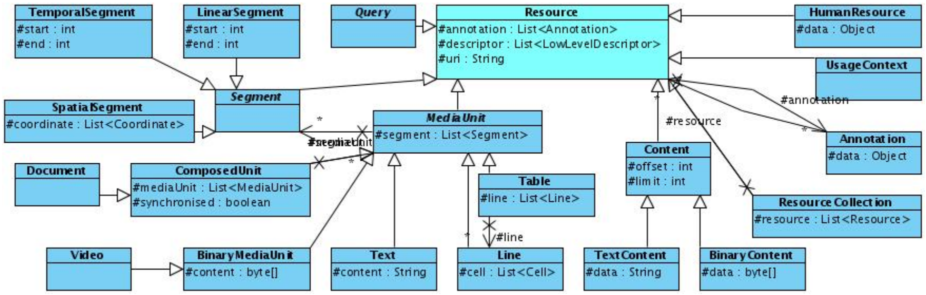

Fig. 2. Elements of the WebContent model.

To guarantee the full interoperability of services, it is not enough to normalize protocols and service interfaces. It is also necessary to define the structure and semantics of data shared or exchanged by services. The WebContent exchange data model must be able to describe all kinds of source documents that go through the textual information extraction process and be extensible to other kinds of media. Figure 2 shows a UML class diagram displaying the elements of the XML model. Each class inherits from the `Resource` class and is thus identified by a URI. RDF annotations attached to `Resource`s use this URI to refer to specific units in the documents.

The combination of data and annotations in the exchange data model allows for a better decoupling between services, each service receiving all the data it needs to do its processing in a single call. However, this model does not impose any restriction on the way the data is effectively stored and processed by each service. In fact, some WebContent applications use an RDF store and SPARQL to query the data, while others use the annotations as pure metadata attached to documents in a full-text index. For binary or large data, annotations can be received separately.

The WebContent platform can be deployed using an *ESB*[2]. In this context, the Web services are connected to the bus and a BPEL (Business Process Execution Language) orchestrator is used to manage their interaction in order to provide the application service.

The WebContent platform defines around twenty services (most of them being implemented by one or more furnishers). We describe here two of them to give an overview of the features. The platform defines an interface for a *storage* service and consequently a *query* service, to access the data that is stored. To illustrate the interfaces genericity, two implementations have already been released by WebContent partners. A first one around the MonetDB XML database and a second one around a P2P distributed index.

The semantic annotation service finds references to concepts and relations from a domain ontology in text segments. The corresponding text segments are then annotated with RDF statements referring to these instances. One implementation of this service is the CEA LIST one. The analyzer extracts entities and relations using manually-written or learned regular expressions. Next, the semantic annotator uses correspondence rules between entities and concepts of the domain ontology. Two other similar services by EADS and Thalès are available.

The user interface of WebContent applications is typically built as a Web application composed of portlet components assembled in the framework of Web portals such as Liferay[3]. The WebContent platform provides a set of generic, reusable portlets for visualizing data such as annotations contained in WebContent documents.

A development kit for WebContent is made available (BSD-like licence) by EADS under the name WebLab-core. It features the WSDL services APIs and XSD exchange model, tutorials to implement services and portlets, invoke them from the ESB, and orchestrate them with BPEL programs. A C++ framework by CEA LIST is also available, allowing the easy encapsulation of any C++ application in a WebContent-compatible service. The C++ framework is available under the LGPL license. All software and documentation is made available from the WebContent Web site[4]. While the platform is open and freely available, different service providers may have different release policies (free software, free use, commercial...).

3 UIMA and WebContent complementarity

In UIMA, documents are sub-specified letting the application developer free to handle any kind of document. CAS and Sofa allow to apply different analysis engines to dif-

[2] a second architecture is a P2P environment that will not be described here

[3] `http://www.liferay.com`

[4] http://www.webcontent.fr

ferent views of a document. This gives a great flexibility to the system but also open the door to difficult compatibility problems when time comes to use external tools and other COTS. Also, UIMA is primarily designed to run local software available in a same shared memory area. Even if UIMA-compatible RPC mechanisms are offered, this is still a high integration approach. On the contrary, WebContent puts the emphasis on API and data structures normalization in a Web services environment, emphasizing a loose coupling.

Concerning data, UIMA's goal is to handle unstructured information while WebContent's one is to handle structured documents from their acquisition and including their normalization. Leafs of WebContent documents are unstructured media that can be naturally handled by UIMA. In the multimedia processing stack, WebContent works at the document (and collection of documents) level while UIMA works at the individual media level. Finally, WebContent is an application design framework defining a broad range of tools, including data storage, crawling and search while UIMA processing chains aim at efficiently associating complementary processing tools working at a similar content level.

Marrying both approaches can help to solve a lot of problems when writing large multimedia applications depending on external vendors. There is two possible and compatible integration paths. On a one hand, an UIMA based application can call external WebContent services or service orchestrations. In this case, an analysis engine will take the application CAS and rewrite it into the WebContent format before calling the WebContent Web service. RDF annotations produced by the WebContent service will then be extracted and converted back into UIMA annotations.

On the other hand, a WebContent-based application can use an UIMA chain as the implementation of one of its services. In this case, upon receiving a request, this service will convert the received WebContent document into an UIMA CAS and then start the analysis. On termination, the UIMA annotations are then converted back to RDF annotations and inserted in the resulting WebContent document.

Naturally, hybrid applications using both approaches could be built, some UIMA service engines calling WebContent services and some WebContent services calling some UIMA service engines.

An example of an application that could profit from the use of both technologies is a watch application in which documents would be collected, normalized and stored by WebContent services. It would then be sent to an analysis service that would run an UIMA chain to extract entities, relations and facts. This would produce RDF annotations on the WebContent documents, used by a reasoner to produce the target knowledge.

Scribo is a project of the Free Open Source Software Working Group in the System@tic-Paris-Region competitiveness cluster. It aims at algorithms and collaborative free software for the automatic extraction of knowledge from texts and images, and for the semi-automatic annotation of digital documents. In Scribo we are currently designing an application using the first approach outlined above.

4 Conclusion

The WebContent platform provides means for opening existing tools to the Web services world and to develop applications based on the warehousing of Web resources. The project is both a platform for building real-world, industrial applications, and a testbed for research projects involving peer-to-peer technologies, ontologies or machine learning techniques to name a few. UIMA offers generic APIs and tools to build advanced information management applications.

UIMA and WebContent together will bridge the gap between very efficient applications and processing tools cluttered around the world. While the Scribo project is already on its way to demonstrate this complementarity, we would appreciate a lot if the UIMA community itself could have a look at this new Free Software tools.