DeveloperIntroduction

Linked Data made easy

Linked Data offers a big potential for enterprises in the media and knowledge management area. On the one hand, more and more datasets are published following Linked Data principles and thus give the opportunity to link enterprise content with background information and also allow disambiguation of concepts. No single enterprise would be capable of managing a standardised vocabulary the size of DBPedia or a location database the size of GeoNames . On the other hand, Linked Data also would offer a comparably simple solution for enterprise information integration *inside* companies.

However, enterprises are still hesitating to use Linked Data in their value chains. From our experience with working with industry partners, one of the main barriers in the adoption of Linked Data is that the technology is still not easy enough to use and does not integrate well with existing information systems. Particularly, accessing data from the Linked Data Cloud is still cumbersome, and Linked Data has so far mostly been seen as a read-only and metadata-only source, while enterprise data usually is highly dynamic and involves both content and metadata.

The Linked Media Framework (LMF) is a collection of software modules that make building applications based on Linked Data very easy. The LMF is published as Open Source Software under Apache license and available for download at [http://code.google.com/p/lmf/downloads/list Google Code] and through [CustomProjects Maven]. The key aspect distinguishing it from existing Linked Data servers is its complete Open Source approach, its ease-of-use, and its additional functionalities that are simplifying common development tasks. The LMF offers developers the following additional functionalities:

- easy publication and consumption of Linked Data using easy-to-use web services and providing transparent access to the Linked Data Cloud
- uniform treatment of content and metadata by extending the Linked Data principles for content management
- full integration with the Linked Data Cloud using Linked Data caching that is completely transparent to the developer: you link the resource, the LMF retrieves and caches it if needed
- integration with datasets that are not available as Linked Data, e.g. Social Media APIs from YouTube and Vimeo: you link the resource, the LMF retrieves and caches it as if it were an ordinary Linked Data resource

- developer friendly path-based querying over resources in the LMF and in the Linked Data Cloud based on our path language [http://code.google.com/p/ldpath/ LDPath]

- highly configurable Semantic Search component making use of linked concepts in the Linked Data Cloud
- stable and efficient rule-based reasoning component allowing to add implicit rule-based knowledge to the knowledge base, e.g. SKOS thesaurus transitivity, etc.

In the following sections we present three aspects of the Linked Media Framework: the LMF Server itself, how to build custom Linked Data servers on top of the LMF Server by building on top of the LMF Libraries, and how to use the LMF Client Library in conjunction with a LMF Standalone Server to integrate the Linked Media Framework in your own Java, PHP, or Javascript web applications.

The LMF Server

System Requirements

The LMF Server is an easy-to-deploy Java Web Application consisting of several optional modules that can be combined as needed in the individual application scenarios. A generic standalone configuration of the LMF Server can be downloaded as a .war file from the project website. As one of the design goals of the LMF was ease-of-use, the LMF Server has minimal system requirements. In its most minimalistic form, the following requirements need to be satisfied:

- a recent CPU and 1GB of memory
- Java 6 or higher
- Apache Tomcat 6.x or Jetty 6.x

In such a setting, the LMF Server will start its own embedded database to store RDF data and content. Depending on the application scenario, the LMF can be reconfigured to e.g. make use of PostgreSQL, MySQL, or other more performant databases. It is also possible to adapt the configuration to make better use of e.g. multiprocessor environments and caching infrastructures.

Resource Management: publishing and updating Linked Data and Content

The core module of the LMF is resource management. Resource management is based on the [http://linkeddatabook.com/editions/1.0/ Linked Data] principles but extends them in two important aspects. The first extension is concerned with Linked Data updates by making use of HTTP PUT, DELETE and POST for updating resource metadata and content, and complements the Linked Data GET with REST updating access to resources. The second is concerned with managing media content and metadata alike by providing an extended mapping of resource URIs to redirect URIs based on the mime type sent in the Accept (as in Linked Data) and Content-Type HTTP headers. In this way, it is possible to manage many different content types as well as distinguish between the metadata of a resource and the human-readable content of a resource. The details of the content negotiation are described in the [UsageResourceInteraction software documentation].

LDCache: transparent access to the Linked Data Cloud and beyond

The Linked Data Caching component offers applications building on top of the LMF transparent access to data from the Linked Data Cloud. Applications do not need to explicitly retrieve Linked Data resources; they can simply access them using the same API and libraries provided for local resources. Transparent Linked Data Caching has been perceived as a "killer feature" by many developers using the LMF, as it eliminates several common obstacles:

- it takes care of all necessary content negotiation and parsing of the returned data

- it provides proper caching mechanisms that take into account expiry information as sent by the server or defined by the user
- it integrates seamlessly with resources and triples that are stored locally

- it can be configured to take advantage of local or remote Linked Data cache endpoints like [http://sindice.com/developers/liveapi Sindice Live]

The LDCache component goes even beyond ordinary Linked Data by providing an API that allows implementing wrappers around other data sources on the Web. As an example, the LMF server ships with wrappers for the YouTube and Vimeo APIs.

LDPath: developer-friendly querying of Linked Data

Even though resources on Linked Data servers are typically interlinked and thus conceptually integrate data from potentially many different sources, querying such data is still very cumbersome. The main reason is that existing query languages for RDF like SPARQL are rather dataset-centric and do not easily query over distributed or even unknown sources. In addition to an ordinary SPARQL endpoint, the LMF server therefore offers path-based navigation through resources, called LDPath. LDPath follows similar ideas like XPath or SPARQL Property Path, but is specifically designed for Linked Data querying. In combination with the Linked Data Caching component, the path language becomes a powerful tool for transparently and efficiently querying data on the Linked Data Cloud. LDPath is described in more detail at its [http://code.google.com/p/ldpath/ project website].

Semantic Search based on Linked Data

One of the most common applications of Semantic Web technologies is Semantic Search, i.e. the use of semantic metadata to enhance search and retrieval of documents. The LMF Server therefore integrates a semantic search component based on [http://lucene.apache.org/solr/ Apache SOLR] that can make use of local and remote datasets. The search component is highly configurable to different application scenarios and allows mapping "LD Paths"' to index fields; by default we include rulesets for RDF, SKOS, Dublin Core and [GeoNames]. The LMF Search component can be accessed using a REST API that conforms to the [OpenSearch] standard and is compatible with the Apache SOLR search API. Existing SOLR clients can therefore be used for accessing the [ModuleSemanticSearch search functionality].

Rule-based Reasoning with sKWRL

The [ModuleReasoning LMF Reasoner] offers a subset of the rule-based reasoner sKWRL developed in the KiWi project for reasoning over triples contained in the LMF. Rules can be added and updated by the user during runtime and are evaluated using a forward chaining fixpoint algorithm. In addition to the inferred triples, the LMF Reasoner also computes so-called *justifications* that give explanations (base triples and rules used in the inference) why certain triples have been inferred. Justifications are used for efficient updating (so-called *reason maintenance* or *truth maintenance*) and for displaying to the user.

LMF Applications

The LMF has been designed as a generic platform for Linked Data and Linked Media applications. We therefore provide a Java-based build environment that allows building custom server applications based on the core services offered by the LMF. In this way, it is possible to develop "Linked Data-ready" web applications easily. The build environment is a compressed archive with a basic project structure and configuration, and a Gradle build file that will automatically download the required LMF libraries from a Maven repository and assemble a custom Java web application that bundles the LMF Server components. Detailed step-by-step instructions on how to setup a custom project are described in the [CustomProjects project documentation].

The typical scenarios where a developer would use a custom LMF Application would be:

- the generic LMF needs to be extended by custom functionalities, e.g. proprietary data importers, additional Linked Data wrappers, information extraction, custom user interfaces, etc.
- direct API access instead of web services is a requirement, e.g. for performance or security reasons

The main disadvantage of this approach is the restriction to the Java language and the LMF architecture patterns (CDI and Weld), as well as the necessary in-depth expertise regarding LMF development. Several custom LMF Applications have already been developed and are in production use, e.g. the [http://search.salzburg.com/ news search of Salzburger Nachrichten].

LMF Client Library

The second option to use the LMF for Linked Data development is to build applications based on the LMF Client Library and a standalone LMF Server (can be either the generic server or a custom server as described above). The LMF Client Library provides an API abstraction around the LMF webservices and is currently available in Java, PHP, and Javascript. While each language-specific implementation retains the same general API structure, the way of accessing the API functionality is customised to the respective environment, so that e.g. a PHP developer can access the LMF in a similar way to other PHP libraries. No in-depth knowledge of RDF, Linked Data, or the Linked Media Framework is required. The library provides the following functionalities:

- *Resource Management*: retrieving, updating and deleting resources and associated content/metadata
- *Configuration*: reading and updating the LMF system configuration
- *LDPath*: querying the LMF and the Linked Data Cloud using path expressions or path programs
- *SPARQL*: querying and updating the LMF using SPARQL 1.1 Queries/Updates
- *Semantic Search*: configuring semantic search cores and performing queries
- *Reasoner*: uploading and removing rule programs

The LMF Client Library is currently in use in several research projects, including the two Austrian industrial research projects [http://mytv.ipaustria.at/ MyTV] (personalised Web TV platform) and [http://connectme.sti2.org/ ConnectMe] (semantically interlinked videos).

Conclusion

This article gave only a brief overview over the core functionalities of the Linked Media Framework. Details can be found on the project website. A number of public demonstrations are also available at our [http://labs.newmedialab.at demo server]. The final release for the LMF Server is scheduled for 23rd March 2012.

The WWW developer presentation will first give an overview and show the capabilities of the Linked Media Framework on a demo scenario and then provide a short step-by-step guide how to use the LMF Client Library in conjunction with a standalone LMF Server to build a small PHP web application for connecting videos from YouTube and Vimeo.

Acknowledgments

This development has been funded by the Republic of Austria within the COMET project Salzburg NewMediaLab and by the the European Commission within the 7th Framework Programme project KiWi - Knowledge in a Wiki (No. 211932).