SOPPA - SERVICE ORIENTED P2P FRAMEWORK FOR DIGITAL LIBRARIES

Marco Fernandes, Pedro Almeida, Joaquim A. Martins and Joaquim S. Pinto
Universidade de Aveiro, Campus Universitário de Santiago, Aveiro, Portugal

Keywords: Peer-to-peer, SOA, Digital Libraries.

Abstract: P2P and SOA paradigms provide new opportunities for the development of new digital libraries and the redesign of existing ones. This paper describes the work being conducted to create a framework for the creation of digital libraries, which relies on a P2P network and service-enabled peers to achieve high modularity, reusability and performance in dynamic environments. While this framework supports data and metadata storage and management, this paper focuses on its service oriented approach.

1 INTRODUCTION

Current digital libraries face new challenges and demands. With the opportunities given by the Internet, these information systems must be able to deliver very high amounts of data to a growing number of users. Also, digital libraries must not act only as repositories – they should provide services for both humans and machines.

The centralized model, in which a server not only hosts the web site but is also responsible for all the underlying tasks required by the digital library, therefore lacks the necessary scalability and flexibility. A distributed approach, which promotes interoperability and cooperation, is a key element for success. The peer-to-peer (P2P) paradigm, in which a number of connected nodes (peers) act as both clients and servers in a decentralized and autonomous manner, offers an opportunity to create truly dynamic and scalable systems.

On the other hand, the Service Oriented Architecture model provides applications with well-defined interfaces to execute different (remote) functional units in a modular manner. The combination of SOA, Web Services and service composition has several advantages, such as increased automation, deeper process integration, higher reusability, and standardization of systems (Zimmerman, 2005).

There is a lack of frameworks to enable the development of new digital libraries which can take advantage of both service oriented architectures and a dynamic network infrastructure such as P2P. Also, most digital libraries are built as isolated systems, which respond only to the needs of a particular institution or community. Also, if new and more complex requirements are set, it is difficult to extend existing digital libraries. This has led us to develop SOPPA – a framework for a Service Oriented Peer-to-Peer Architecture.

2 OBJECTIVES

This work aims the creation of a digital library framework which can integrate a service oriented layer and P2P layer in order to achieve high decentralization, reusability and interoperability. Namely, the framework should allow the rapid development of new information systems and meet the requirements elaborated by the DELOS (Agosti, 2006) work group, from which we outline some of the most important:

- Availability of specialized services, such as search, indexing, annotation, and management of resources and metadata.
- Management of distributed, heterogeneous and autonomous services.
- Composition of complex processes based on existing services.
- High degree of availability: access needs to be guaranteed at all times.
- High degree of dependability/reliability.

Fernandes M., Almeida P., A. Martins J. and S. Pinto J. (2008).
SOPPA - SERVICE ORIENTED P2P FRAMEWORK FOR DIGITAL LIBRARIES.
In *Proceedings of the Tenth International Conference on Enterprise Information Systems - SAIC*, pages 215-220
DOI: 10.5220/0001681902150220

3 RELATED WORK

In this section we overview some of the tools and applications currently available which facilitate the creation of digital libraries and repositories.

The dLibra (Mazurek, 2005) digital library framework is a set of tools that allow the storage, management and access to collections of heterogeneous digital documents. It also supports users' rights management. Interoperability with other systems can be achieved by using the OAI-PMH and RSS mechanisms. DLibra is decentralized into six services, which together give dLibra functionality.

Edutella (Nedjl, 2002) is a P2P network infrastructure based on RDF aimed at the exchange of educational resources (metadata) between academic institutions. It is built on the JXTA framework and implements three different services: Query, which uses a query exchange language; Replication, to achieve metadata persistence and availability; and Mapping, Mediation, and Clustering, which perform mapping between schemas, mediate access between services and set up semantic routing and clustering. Edutella does not handle the data itself and is only responsible for the metadata.

P2P-4-DL (Walkerdine, 2004) aims to build a system for digital libraries which operates in a P2P network. It uses a brokered approach, by storing in a single node the global resource index. There is no replication or load balancing mechanism, as documents always remain only in the owner node.

Greenstone (Bainbridge, 2001) is a suite of open-source software for building and distributing digital library collections. It has an agent based architecture, in which agents have, or have access to, certain functionality. Although resources can be stored in a distributed manner by deploying more "sites", there are no built-in mechanisms for discovery, replication or orchestration. Communication between agents in different computers takes place using SOAP messages and latest versions support OAI-PMH.

Finally, some tools and frameworks currently allow the creation of digital repositories. DSpace (Tansley, 2003), ePrints (EPrints, 2007) and Fedora (Lagoze, 2005) are some examples of such systems. Besides empowering institutions with free applications which ease the creation of institutional repositories, they offer advantages such as facilitating digital preservation, employing standards conformance and supporting interoperability protocols such as OAI-PMH. Also, being open source and widely adopted in the academic world, they benefit from the collaboration of large communities. They are not, however, designed for a distributed scenario. Such platforms rely on a centralized architecture, in which data and services are made available in a server.

4 P2P ANALYSIS

The objective of the first part of this work is to define a P2P network layer which will support data storage and management in the framework. In this preliminary phase, it is required to analyze and select the best P2P topologies, data structures and tools. Also, a suited indexing engine must be selected and integrated in the layer.

4.1 Topology

Regarding the network topology P2P systems can be classified with one of four main categories: centralized, decentralized, hierarchical, or hybrid (Taylor, 2005).

In centralized or brokered P2P systems (such as Napster) nodes connect to the network by registering themselves in a central server. This server must maintain an index of all resources in the network and respond to search queries from all other peers. Although bandwidth efficient and easier to administer, centralized topologies are not scalable and, in case of a server failure, the system ceases to function properly.

Completely decentralized or pure P2P, such as Gnutella 0.4 (Ripeanu, 2001), do not rely on any centralized element. Peers connect to the network through any already connected node and searches are limited to a number of message hops (TTL). Pure P2P can grow to millions of connected nodes and is generally self adaptable, but it cannot provide deterministic query mechanisms and guarantee availability.

Hierarchical topologies usually follow underlying structures (social, geographical, etc.) and thus may make it easier to locate information based on locality. It may be difficult, however, to use it in very dynamic scenarios.

Most modern P2P applications use some sort of hybrid topology (Ma, 2005)(Mastroiani, 2005), which aims to achieve robustness and efficiency in dynamic scenarios by combining centralized and decentralized topologies. In a hybrid scenario, peers are clustered into groups which behave as small centralized P2P environments. The central nodes in

each group (the super-peers) communicate between them in a decentralized manner. This allows for a greater scalability and higher performance.

4.2 Data Structure

Regardless of the topology chosen, which defines how nodes connect themselves, one must decide how to actually populate peers with data. P2P systems usually take one of two basic approaches: structured or unstructured. Mischke and Stiller (Mischke, 2004) analyzed the problem of distributed searches in different structural data space designs.

In structured networks such as Chord (Ratnasamy, 2001) or CAN (Stoica, 2001) the data placement results of the execution of a predefined function or lookup in a hash table. By having a metric for the quick retrieval of data, structured networks are highly scalable. However, searching by metadata is a complex task which may require broadcasting of queries.

Unstructured networks, on the other hand, have no predefined rules for data storage. These are ideal in dynamic networks, where constantly updating a distributed hash table can be troublesome. While they may generate more traffic network in some situations, its flexibility makes them more attractive for digital libraries. Queries can be as complex as desired, and each peer will respond with its best possible answer.

4.3 Search Engines

Indexing and search services play a critical role in a digital library, as they allow to quickly and efficiently find resources based on previously indexed metadata. In a P2P based digital library, some aspects are relevant in the choice of search engines. Assuming a hybrid topology is used, each peer will have a local index which must be periodically sent to the super-peer where they are merged.

Ideally, the search/indexing engine should have the following features:

- Generated indexes should be transparent to the API/language used. If the P2P is to support different operative systems, indexes must be interchangeable.
- Decoupling of indexing and searching mechanisms. This allows applications to use super-peers without also replicating data.
- Indexes should be incremental and able to be merged.
- XML and XML namespaces support, either natively or by using plug-ins.

With these factors in consideration, we conducted a benchmark with six of the major free or open-source search engines: Indri, Lucene, MS Indexing Service, Swish-e, Terrier, and Zebra. The benchmark revealed that Swish-e and Lucene provide the best performance results and respond to most of the desired features for our digital library framework. Since Lucene is more actively supported by the community and is ported into a larger number of APIs, it is the chosen indexing engine. Some add-ons, such as proper support for XML indexing, were developed.

5 ARCHITECTURE

The framework is designed to take advantage of hybrid network topologies. In each peer group (small institutions, departments, I&D units, etc.) a super-peer maintains an updated index of the group resources (documents and services). Other peers periodically send their local indexes to be merged at the super-peer. When requesting a file or service provider, peers start by inspecting their local cache and, if no match is found, a query is made to the corresponding super-peer. Each super-peer may also forward queries to other super-peers.

The framework is built on top of the JXTA framework (Gong, 2001), which provides the basic P2P communication mechanisms. In JXTA, super-peers are named rendezvous peers and others are edge peers. In order to achieve communication with peer groups from other networks, at least one must have outside communication (relay peer).

As discussed in section 4.3, it is desirable to make use of index/search engines in order to achieve high performance queries. Lucene was therefore integrated with JXTA, so that it can index metadata from both files and services. Each super-peer maintains a merged index of the group's resources.

Figure 1 depicts the adopted architecture. Two interfaces are available at the peer - P2P

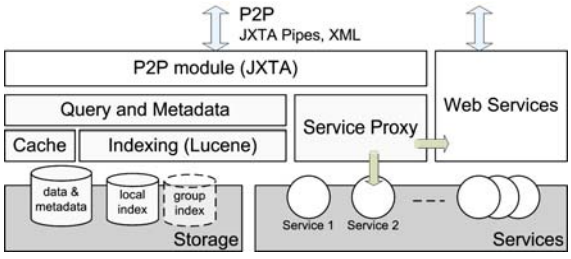

Figure 1: The adopted architecture for the framework.

communication or SOAP -, and both use XML as the message format. At the right end side, there are Web Services (developed using the framework or not) which can act as independent and standalone services. However, by registering the Web Services with the framework, applications can be developed to be more dynamic and autonomous.

5.1 Peer Services

Several services are required for the proper functioning of a digital library. For instance, a common submission of a thesis in the PDF format into a University's digital library may require invoking services such as: generation of a globally unique identifier for the thesis, converting the PDF into per-page image files, processing these images (resize, create thumbnails, etc.), extracting or recognizing (OCR) the text from the PDF, and indexing and storing the file. Some of these services are CPU-intensive and time consuming, and if deployed on the web server they will eventually degrade the overall responsiveness and decrease the system's throughput.

By using service oriented architectures, the workload of digital libraries can be distributed by many – eventually idle – machines. On the other hand, complex services can be designed as an orchestration of simpler, existing ones. With this framework, by using an overlay which seemingly integrates SOA and P2P layers, a service can be dynamically dispatched to any available peer who runs it.

5.1.1 Service Proxy

To automate the process of searching, choosing, and invoking remote (and local) services, a service proxy is included in each service-enabled peer. When a service call is made through the framework, the proxy is instantiated with the method URI:

```
ServiceProxy proxy = new
   ServiceProxy("srv://soppa/imaging/
   resize");
proxy.arguments = new
   object[]{image, width, format};
proxy.Invoke();
```

If the provided URI is an URL with the HTTP protocol, the proxy will simply execute the Web Service using SOAP. Otherwise, if it uses the special framework's srv protocol, when the proxy is invoked it will start by inspecting if the local peer has this service. If not, it continues by looking into the cache, and finally it sends a query to the group's super-peer. Each of these queries is delivered to the respective Lucene engine, which indexes the services signature and description. If successful, the query is replied with the peer/service address.

While on a common SOA scenario knowing the location of a given Web Service would suffice to invoke a service, P2P places a new challenge. Since it allows us to connect to otherwise unreachable nodes (firewalled, no public IP address), it is likely that two nodes on distinct local networks cannot invoke each other's Web Services.

The Service Proxy eliminates this barrier: if no "direct" connectivity is available between the calling peer and the service peer, the remote method is invoked through the P2P network between the two proxies. Locally, the proxy either calls a Web Service or the method's library or executable, if it is one of the core services.

5.1.2 Core Services

There are several basic built-in methods provided by the framework, such as the Get, Store, and Search operations, which are present in every peer. Apart from these, we have identified a number of core services required by most digital libraries developed at our laboratories, from document processing tasks to logging.

While it is not mandatory for peers to have all these services available, a common interface library is delivered as part of the peer logic. For the first release of the framework, this will include six main blocks: Imaging (which contains methods for image conversion, resizing, OCR, etc.), Text (along with some utilities there is a submodule for PDF handling), Security (users and groups management, cryptography), File (common I/O tasks, file compression), Logging (track operations, errors), and MetaInterop (methods for mapping between metadata formats, built-in OAI provider). When these services are called, the proxy is invoked under the hood to transparently find and execute the service in an available peer. The following code is therefore equivalent to the one on the previous section.

```
SOPPA.Imaging.Resize(image, width,
   format);
```

5.1.3 Replication and Load Balancing

As mentioned in the previous section, while every peer is "aware" of the interface for available core services, peers may only provide a subset of those services. In order to achieve a load balancing,

services (from the framework core or otherwise) may be replicated as long as they meet some requirements – for the libraries/executables to be portable and the destination peer to have the necessary operating system and platforms. These requirements are stored in peers as XML documents.

Core services already make use of open-source portable libraries, and can therefore be replicated on request. Also, the SOA/P2P decoupling allows for the same functionality to be provided by different libraries in various programming languages, as long as they have the same signature and URI. With every instance of a digital library, a common XML configuration file resides in every peer, where automatic replication may be set on or off. When set to on, peer services whose clients are being queued can broadcast a replication request to other nodes and copy the necessary libraries to the first peer to reply.

5.1.4 Other Services

Interoperability is a key factor in digital libraries, and OAI-PHM protocol is being widely adopted to promote a simple consumption (harvest) of metadata from institutional repositories. The SOPPA framework makes available a simple OAI provider service, which dynamically queries a peer group and delivers OAI-PHM responses with the data resources available. While it is more efficient to have this provider at the super-peer, it can be placed (and replicated) in edge peers.

To simplify the development of more complex services, one can also build composite services, which make use of existing, simpler ones. While this can be achieved with a simple sequence of proxy calls in the code, it is sometimes convenient to provide a declarative, XML based, composite service description. It is being considered the support for business process languages such as BPEL in the future.

6 CONCLUSIONS

Unlike existing digital library solutions, the proposed framework takes advantage of the benefits from both P2P networks and SOA architectures, by adopting an architecture which integrates the two paradigms. Although some features are still under development, it can already simplify the job of creating a new digital library from scratch.

As a future work, we expect to perform stress tests and performance measurements, and compare it to the existing digital library and archive from our institution, which holds thousands of heterogeneous documents and already takes advantage of some services distributed in other information systems. Future versions of the framework should include more services and solve issues such as declarative style business process execution. Also, support for REST-style web services should be added.

ACKNOWLEDGEMENTS

This work was funded in part by FCT – Portuguese Foundation for Science and Technology – grant number SFRH/BD/23976/2005.

REFERENCES

Agosti, M. et al., 2006. D1.1.1: Evaluation and comparison of the service architecture, P2P, and Grid approaches for DLs. Technical Report, DELOS – A Network of Excellence on Digital Libraries.

Bainbridge, D. et al., 2001. Greenstone: A platform for distributed digital library applications. In Constantopoulos, P., & Solvberg, I.T. (Ed.), *Research and Advanced Technology for Digital Libraries* (pp. 137-148). Springer.

EPrints, 2007. EPrints for Digital Repositories. http:// www.eprints.org

Gong, L., 2001. JXTA: a network programming environment. Internet Computing, 88-95. IEEE.

Lagoze, C. et al., 2005. Fedora: an architecture for complex objects and their relationships. In International Journal on Digital Libraries, 6 (4), 124-138.

Ma, Y., Aygun, R.S., 2005. Peer-to-peer based multimedia digital library. In Proceedings of the Thirty-Seventh Southeastern Symposium on System Theory, SSST '05, 130- 134.

Mastroianni, C., Talia, D., Verta, O., 2005. A super-peer model for resource discovery services in large-scale Grids. Future Generation Computer Systems, 1235-1248.

Mazurek, C., Werla, M., 2005. Distributed services architecture in dLibra Digital Library Framework. Future digital library management systems: System architecture and information access, 26-31. DELOS.

Mischke, J., Stiller, B., 2004. A methodology for the design of distributed search in P2P middleware. IEEE Network, 30-37.

Nedjl, W. et al., 2002. EDUTELLA: a P2P networking infrastructure based on RDF. In Proceedings of the 11th international conference on World Wide Web, 604-615.

Ratnasamy, S. et al., 2001. A scalable content-addressable network. ACM ISGCOMM, 161-172.

Ripeanu, M. (2001). Peer-to-Peer architecture case study: Gnutella network. In Proceedings of the First

International Conference on Peer-to-Peer Computing, 99-100.

Stoica, I. et al, 2001. Chord: A scalable peer-to-peer lookup service for internet applications. ACM SIGCOM, 149-160.

Tansley, R. et al, 2003. The DSpace institutional digital repository system: current functionality. In Proceedings of the 2003 Joint Conference on Digital Libraries (JCDL'03), 87-97. IEEE Computer Society.

Taylor, I.J., 2005. *From P2P to Web Services and Grids – Peers in a Client/Server World*. Springer-Verlag. London.

Walkerdine, J., Rayson, P., 2004. P2P-4-DL: digital library over peer-to-peer. In Proceedings of the Fourth International Conference on Peer-to-Peer Computing, 264-265.

Zimmerman, O. et al, 2005. Service-oriented architecture and business process in choreography in an order management scenario. OOPSLA'05, 301-312. ACM Press.