Web Services' Integration into a Peer-to-Peer BitTorrent Client

Francisco Barbosa, Maria Teresa Andrade, and Asdrúbal Costa

***Abstract*—Nowadays, the distribution of data is becoming more and more important. This lead to the massive adoption of the Peer-to-Peer (P2P) paradigm, in which every peer in a network is able to receive and to offer data to others. This type of architecture has significant advantages over the client-server architecture, increasing the speed of data's transfer and the availability of data. One of the major protocols used to share data between peers is BitTorrent, which has significant advantages, but also has some limitations, like the need to use specific software and the availability of contents, which becomes low with a torrent's lifetime.**

These limitation can, however, be overcome by using Web Services to access to a P2P network's contents and a special type of peer, equipped with plugins integrated within its BitTorrent client that can assure reasonable contents' availability.

***Index Terms*—BitTorrent, Peer-to-Peer, Web Services**

I. Introduction

MOSAICA is an European project that aims to distribute cultural contents to promote and increase tolerance between people. To accomplish this, MOSAICA uses the P2P paradigm to connect the peers, using the BitTorrent protocol to share contents over peers. As the main objective, MOSAICA intends to spread cultural contents as far as possible, which can be sometimes difficult, due to network's limitation, like traffic-shapping techniques, made by the Internet Service Providers, or even firewalls' configuration, restricting BitTorrent traffic. Together with this, there is also the common BitTorrent users' behaviour, who normally offer data to other peers only while they are downloading that same data. Once they finish the download, they stop sharing it, leading to a low number of resources available after some time.

To fight both exposed problems, and to get MOSAICA closer to its objective, two solutions were adopted: i) a set of Web Services, so data shared within a P2P network can be accessible to users through a Web browser; ii) a set of plugins to the Azureus BitTorrent client as a way to control and increase contents' availability.

II. State of the Art

Peer-to-peer traffic is at present a significant part of the Internet traffic, being still in expansion, according to [1]. Part of this traffic is from the BitTorrent protocol, the one chosen to be used within MOSAICA's peers. This protocol allows, beyond the file sharing ability wanted for MOSAICA and its inherent performance in terms of downloading times, the use of a structured network in which structured searches can be performed, taking advantage of the DHT-based search mechanisms implemented on most BitTorrent clients available.

The implementation of the MOSAICA P2P-based content management system was done using the JXTA framework to build an overlay on top of BitTorrent network. JXTA is a JAVA framework that offers a set of generalised and open protocols to build services on top of P2P networks. In MOSAICA this overlay was responsible for the distribution and access to metadata, implementing upload and search functionality. The BitTorrent network was responsible for the distribution and download of the actual content.

Keeping in mind the philosophy of the P2P paradigm, combined with that of Service-Oriented Architecture which is one of today's most important architectural style, mainly for enterprises [2], based in principles from distributed computing, nodes in a network can be seen as service consumers, service providers or even both. The services offered by the providers can be available to consumers through the Web, using HTTP, being called Web Services. These services are described according to the Web Service Definition Language (WSDL), informing client applications of each service's input and output. Together with WSDL, Web Services also enjoy of the SOAP protocol, turning Web Services language and platform independent, as SOAP is XML based, like WSDL. SOAP's main concerns are the encapsulation and encoding of XML data, and the definition of rules for sending and receiving that same data.

The creation of Web Services is at present a quite straight-forwarding task. One of the tools available for it is Apache Axis2, a JAVA-based SOAP implementation, that runs on Apache Tomacat server, combining SOAP's management with the hosting of services. Together with it, Axis2 can generate Web Services from Java classes and classes from WSDL documents.

III. Implementation

The performed work during the thesis can be divided in two parts, each of which aiming to reach both goals proposed in the introduction section:

- enable any Web user, through a normal Web browser, to access contents shared in a P2P network;
- control contents' availability, increasing resources of poorly distributed contents.

F. Barbosa and M. Teresa Andrade are with the Department of Electrical and Computers Engineering, Faculdade de Engenharia da Universidade do Porto, Rua Dr. Roberto Frias, s/n, 4200-465 Porto, Portugal
Emails: {ee01096, mandrade}@fe.up.pt

Asdrúbal Costa is with with the Telecommunications and Multimedia Unit, INESC Porto, Rua Dr. Roberto Frias, 378, 4200-465 Porto, Portugal
Email: acosta@inescporto.pt

In order to achieve the first objective, two Web Services were developed: the *Get Content* and the *List Azureus' Activities* Web Services. The first one accomplishes the desired objective by checking the content, generating and providing to the user the URL to the desired content. The second one was developed as an extra, with the sole goal of facilitating the user's task of consulting the activities being performed by Azureus.

The correct operation of the *Get Content* Web Service needs a support module so that, when the Web Service generates the URL to expose the content to the exterior, the Web server that enables the contents' HTTP transfer can know the exact location of the content in the peer's hard drive. This module is executed every time the MOSAICA system is launched, checking the location of the MOSAICA shared folder - the folder where the contents are downloaded to - and update the information of the *alias* defined in the Web server configuration file.

The *List Azureus' Activities* Web Service can inform users, in real time, of what is being transferred by Azureus, showing, for each each content, its name, the percentage of data downloaded, the estimated remaining time, whenever the download is not concluded, the actual transfer speed, according to the content is being downloaded or uploaded and also the share ratio. This Web Service retrieves information directly from Azureus, using for it Java Remote Method Invocation.

To achieve the second objective, it was proposed a group of plugins for Azureus that control the contents distributed within the P2P network. The first plugin is RSS Import, a plugin developed by Markus Baeker [3], whose function is to contact a RSS server and retrieve a feed with a torrent file of a content. Modifications made to this plugin allow it to check, for every feed retrieved, the content's size, together with the size occupied by the shared folder. In turn, this enables users to define a maximum size for that folder, corresponding to the maximum disk space that they may want to share. This way, Azureus will only download contents that do not make the occupied space overcome the one defined by the user. The definition of the space that users may desire to share can be configurated in the *configuration panel* of the plugin itself and from an applet, developed to allow users to configure the plugin in an easy way and from a Web interface.

Together with this plugin, another one was developed - the *Seed Limiter*. For each content obtained by the *RSS Import*, this plugin checks the number of available complete copies - the seeds - and compares it against a number defined by the user. If the number of seeds is greater than the latter, the content is transferred and seeded. If not, the content is discarded, and the plugin waits for another content from the RSS Import, repeating this mechanism.

The developed components interact with each others and with the users as represented in figure 1.

The use of Web Services was also tested, so that the additional latency imposed by the use of Web Services could be measured. These latency times were obtained with JAMon [4], in a test with 10,000 requests, which returned the latencies showed in table I.

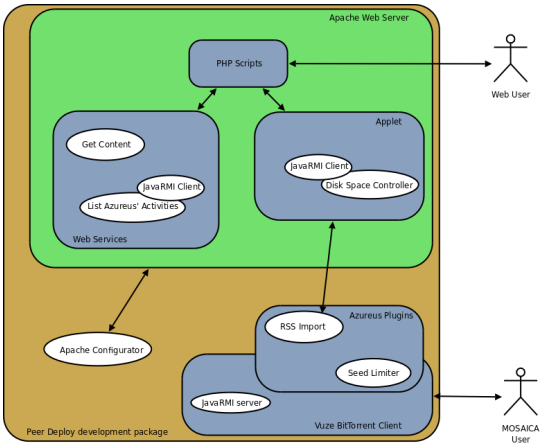

Figure 1. MOSAICA's developed components

Table I
LATENCY FOR EACH REMOTE REQUEST MADE TO A WEB SERVICE

Developed Web Service	*Average*	*Minimum*	*Maximum*
Get Content	2.12 (ms)	216.10 (μs)	24.97 (ms)
List Azureus' Activities	101.58 (μs)	61.00 (μs)	891.00 (μs)

IV. CONCLUSION

The developed Web Services assure that users can enjoy contents shared in P2P networks without the need to install any specific P2P software. Instead, all users need is a Web browser, which is a component usually installed by default in any operating system. The use of these Web Services do not imply additional signifcant latencies, which can be observed from the values in table I.

In what comes to the second objective, all the contents can be controlled with both developed plugins, and the *SeedLimiter* plugin assure that peers will only seed poorly distributed contents, while the *RSS Import* plugin assures that all the contents are controlled and obtained from all the feeds published in the RSS server, all this without any user's intervention.

ACKNOWLEDGES

I would like to acknowledge my supervisors, Dr. Ing. Maria Teresa Andrade and Ing. Asdrúbal Costa, for the enormous assistance and support they both gave me, which was fundamental to the accomplishment of the work performed during the thesis. I would also like to express my special gratitude to my family - my parents Francisco and Odete, my sister Helga, Hélder and Tomás -, for their huge encouragement and the constant optimism about my future, and to Vera, for all her love and support.

REFERENCES

[1] P. Networks, "The p4p working group," http://www.pandonetworks.com/p4p.
[2] M. Ibrahim, B. Michelson, K. Holley, D. Thomas, N. M. Josuttis, and J. deVadoss, "The future of SOA: What worked, what didn't, and where is it going from here?" IBM, Beddara Research Labs, Microsoft, Elemental Links and SOA Consortium, Tech. Rep., October 2007.
[3] M. Baeker, "RSS Import 2.2.2," http://azureus.sourceforge.net/plugin_details.php?plugin=RSSImport.
[4] S. Souza, "JAMon - Java Application Monitor," http://jamonapi.sourceforge.net/.