Web2Peer: A Peer-to-Peer Infrastructure for Publishing/Locating/Replicating Web Pages on Internet

Heverson Borba Ribeiro, Lau Cheuk Lung, Altair Olivo Santin, Neander Larsen Brisola.

Graduate Program in Applied Computer Science- PPGIA
Pontifical Catholic University of Paraná – PUCPR - Curitiba - Brazil
e-mail: {heverson, lau, santin, neander}@ppgia.pucpr.br

ABSTRACT

This paper presents a decentralized infrastructure, called Web2Peer[1], which makes Web pages available on Internet through P2P networks. Different from the conventional Web, the proposed approach does not need, for instance, a HTTP address or a central Web Server. Web2Peer provides a set of functions for publishing, locating, and replicating Web pages on Internet ensuring high availability and fault tolerance. Our infrastructure allows anyone who has a computer connected to the internet, even through a domestic ADSL connection, to publish Web pages through their own machines without any additional cost.

Keywords: P2P, DHT, jxta, web publishing, web content.

I. INTRODUCTION

The World Wide Web became, in the last decade, the main source for searching and publishing information in the world. Being worldwide spread, the network became the place where any answer can be found, anywhere, anytime. Although technology has come a long way, for instance with the evolution of personal computers and the significant increase of domestic broadband connections, the mechanisms used to publish Web page on Internet are still the same. The network preserves its centralized structure, based on client/server model [1]. When a user wants to access any page he/she needs to request it to a particular web server that provides the content, using URLs [2] and URIs [3].

When an organization or user needs to publish any content (e.g. an enterprise website or a personal web page) on the Internet, they must necessarily request it either to a service provider (such as a blog server for ordinary users), being this a rewarded provider or not, or ask to a regulatory entity for receiving an own IP address and internet names [4]. It is important to note that a free webpage provider does not grant any administration mechanism to the user. This scenario brings two interesting considerations related to the administration and cost of the infrastructure. First, deciding to contract a rewarded provider to publish the pages will cause dependency on content administration. Deciding to use free providers will result in resource limitation. However, if decided to use an own server to provide the content will result in a high-cost with IP addresses, internet domain (HTTP address), internet links, therefore resulting in subordination of regulatory entities that provide these requirements [4].

Regardless of the choice, an important consideration is that none of them can guarantee that the web pages will be always available. In centralized structures, the content provider (a website) acts as a single point of failure, which may be unavailable anytime. In some applications, it can be a reasonable condition for a while, but it may not be acceptable in others. Therefore, today's model is characterized for its dependency and subordination to regulatory entities and its vulnerability to failures (server or communication link's fail) resulting in the web page's unavailability.

The peer-to-peer architecture creates a network layer over the topologies already used, creating a new virtual network over the other, eliminating the physical limits on these topologies. In the peer-to-peer systems, all nodes are consumers and providers, resulting in the increase of the resources availability on the network. This is achieved by the passive replication which happens when a resource is accessed by a consumer node, so this node also becomes a new content provider [5].

In this paper we propose a new distributed infrastructure for publishing, searching and replicating web pages on the Internet, based on peer-to-peer networks. The suggested solution eliminates the problems mentioned above, involving entities dependency and content availability, by the user side. The present work offers a new point of view of the current technology, changing the way we publish web pages. The proposed solution allows anyone who has a computer connected to the internet, even through domestic ADSL connection, to publish web pages in their own machines. Furthermore, the proposed infrastructure offers a searching mechanism based on DHT (*Distributed Hash Table*), which acts as a search server, just like Google does, providing a way for users to find pages from keywords. The user interface for this infrastructure is a web browser called Web2Peer, which carries all functions for finding P2P address from subject search, allowing the user to download the web pages from remote peers and viewing these pages on the browser screen. Web2Peer does not use HTTP address.

This paper is divided in the following sections: Section 2, an explanation of peer-to-peer technology, its main features and the JXTA protocols. Section 3, a brief explanation of

[1] This work is supported by CNPq (*Brazilian National Research Council*) through processes 481523/2004-9 and 506639/2004-5.

distributed hash tables (DHT). In Section 4, it is presented the proposed infrastructure. Section 5, some considerations about the implementation of the prototype, tests and results obtained. Section 6, some related work. In section 7, conclusions and future works are presented.

II. PEER-TO-PEER NETWORKS

A. Concepts

Distributed architectures called peer-to-peer are projected for sharing resources through direct interaction, instead of using the intermediation or support of a centralized entity. The peer-to-peer networks are content orientated and are distinguished by scalability and adaptation of variations on number of present nodes. [6] Other peer-to-peer systems features can be mentioned:

- Resistance to the centralized control;
- Encouraging the resources access;
- Administration, maintenance and operation distributed between the users.

The peer-to-peer networks are classified according to two main parameters: the structure and the centralization of the network. By network structure we must understand the way the network is created, if deterministically or not. The structured networks follow specific rules for adding nodes and resources. The unstructured networks do not follow any rule. By network centralization we must understand how centralized the systems are. It defines the topology of the overlay network. According to this parameter, peer-to-peer networks can be: purely decentralized architectures, partially centralized architectures and hybrid centralized architectures [6, 7].

The motivation for using peer-to-peer technologies is its capability of working in a scalable and self organized mode in environments with a dynamic number of nodes and computational faults, without a centralized control.

B. JXTA Protocols

The JXTA project was initially conceived by Sun Microsystems at 2001 and was later developed by a group of academic institutions. The JXTA protocols establish an overlay network over the internet, allowing peers to interact directly with each other and auto-organizing themselves independently from their original network topologies[8]. When peer-to-peer services and architectures started appearing, they did not usually interact with each other. One of the main JXTA proposals is to offer a set of protocols to build peer-to-peer networks that allow interoperability between distinct applications. JXTA defines the minimum requirements to develop peer-to-peer networks allowing the developers the possibility of choosing the precise way to build their networks. JXTA specification was created to be independent of the programming language, operating systems, services and network protocols [9].

C. JXTA Network

As mentioned before, the JXTA protocols create an overlay network over the physical infrastructure, where peers can exchange messages wherever it is in the network. Another objective of the protocols is to hide the complexity of the network topologies giving the messages the possibility of being routed transparently between networks, even by peers behind the firewalls and NATs[8].

JXTA organizes their nodes in peer groups, whose function is to isolate the applications inside the peer-to-peer network. It also has special peers called relay peers and rendezvous peers. These peers are super-peers because they have special capabilities. Another import part of the JXTA specification is the pipe. Pipes are virtual communication channels used for sending and receiving messages between peers. There are two types of pipes: input pipes and output pipes. The pipe type depends on who is sending (output) and who is receiving (input) the message [10]. The JXTA has a set of six different protocols, as follows [11]:

- Peer EndPoint Routing Protocol (PEP): route discovering.
- Rendezvous Protocol (RVP): group propagation.
- Peer Resolver Protocol (PRP): message exchanging.
- Peer Discovery Protocol (PDP): publishing/discovering resources.
- Peer Information Protocol (PIP): peer status.
- Pipe Binding Protocol (PBP): virtual connection creation.

III. DHT - DISTRIBUTED HASH TABLE

Distributed hash tables are tables that provide a mapping between keys and values such as traditional hash tables, but in this case the table is partitioned across the participating nodes. Such as traditional hash tables, the keys are obtained from a hash function applied on the data, called value, which will be stored in the table [12, 13] creating the pair *<key, value>*. In the DHT every node is responsible for mapping a group of keys. This is achieved through internal routing tables present in each node, which are used for searching and locating nodes responsible for certain keys. This search is done asking to the neighbor nodes and their neighbor nodes until the target node is found. This target node provides the value indexed by the key. The first DHT systems were introduced in 2001 by CAN [14], Chord [15], Pastry [16] e Tapestry [17].

IV. PROPOSED INFRASTRUCTURE

The presented architecture uses the advantages of peer-to-peer network through the JXTA protocols for publishing web pages on internet. All users on this network are peers who can publish pages on the internet at same time that are accessing other contents. Each node has a Web Server internally, which is responsible for providing the pages, and a HTTP Client that access the contents, for this communication both use the JXTA API, performing the communication through the JXTA network. The JXTA network has its own protocol for

searching resources, the PDP (*Peer Discovery Protocol*). This search is done by flooding the network looking for specific resources. This technique results in a poor response time, in order to improve this time the infrastructure presented has adopted an indexing service based on distributed hash tables (DHT) which provides faster and deterministic answers, if compared to the JXTA mechanism. The indexing service creates a relationship between the content location, inside the peer-to-peer network, and one or more keywords. The infrastructure has six main components shown in figure 1:

- Browser: User interface, with an internal html editor;
- Publisher: publishes web pages and its keywords;
- DHT: searching and locating web pages by keywords;
- HTTP Client: mechanism for downloading web pages;
- Web Server: provides the web pages in the p2p network;
- JXTA API: provides all peer-to-peer capabilities.

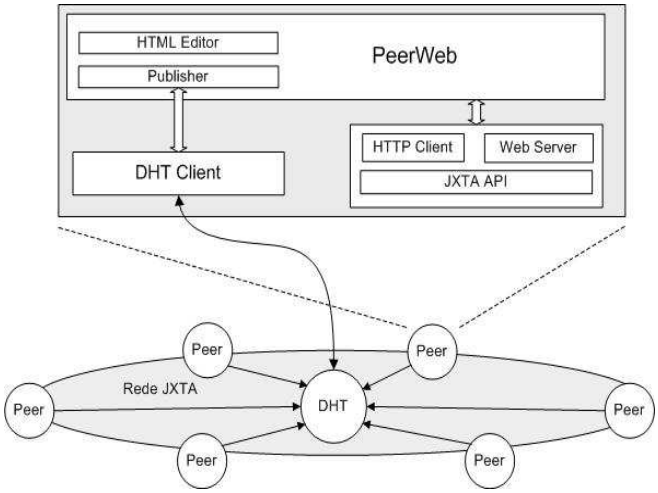

Figure 1. The Web2Peer infrastructure.

The publisher is one of the most important Web2Peer mechanisms. It is responsible for publishing web pages, making them available to other peers.

The presented infrastructure, by conceptual view [6] , is a partially centralized peer-to-peer network, based on JXTA protocols, using a distributed index service, for searching pages, based on DHT. JXTA protocols define this network because some of the nodes have special roles.

At this point, there is a clear difference between the conventional Web and our approach presented in this paper. Nowadays when somebody needs to get a document on internet he needs to use URL and URI address representing the server whose document belongs to. It is a very practical way to find a web page when you know the source server.

On the other hand, when the user does not know the URI/URL for a specific content he will depend on some external mechanism for searching these addresses. Nowadays, Google is certainly the most used searching mechanism [18] to locate web pages by context. The main function of the DHT in this infrastructure is to act just like Google does, providing the location (using P2P address) of the web pages searched/requested.

The infrastructure proposed has three main functionalities, as follows:

A. Web Page Publishing

To create and publish a web page in Web2Peer, the user must use the HTML editor embedded into the browser. After creating the page, the publisher can be activated, by clicking the publish button. For each new web page, a number *n* (where $n \geq 1$) of keywords must be chosen by the author. Keywords are defined according to the subject of web page, and are inserted into the web page through the HTML meta-tags. Each of these keywords will be used by publisher mechanism for indexing the web page into DHT. The figure 2 shows step-by-step how the publishing procedure takes place.

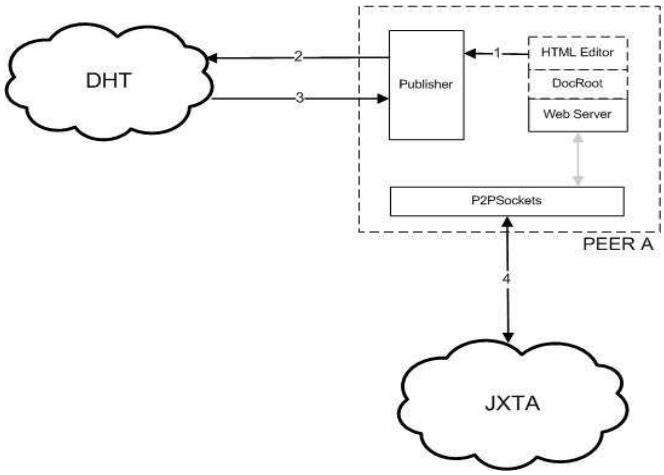

Figure 2. Publishing Web Pages using Web2Peer.

When Publisher is activated by the HTML editor, it captures all the keywords written in the HTML document, during the page creation, as HTML meta-tags. The following example shows a meta-tag which has two keywords associated with it: *p2p* and *web*.

<META NAME="Keywords" CONTENT="p2p, web">

After extracting these keywords, the Publisher access the DHT and publishes the references (steps 1, 2, 3 of the Figure 2), registering the pairs <*key*, *value*> into DHT associating keywords to document location (P2P address) as shown in figure 3.

keyword	p2p address
p2p	p2p://MyPeer/p2p.html
web	p2p://MyPeer/p2p.html

Figure 3. DHT Publishing Example.

B. WebPage Searching

To locate a web page in the peer-to-peer network, the Web2Peer browser access the DHT searching for documents indexed according to the desired subject (keywords). The browser will receive a list, with all peers that have documents published with this same subject, from the DHT. In this list, the user can select the peer from which the web page will be downloaded, usinng JXTA network. This operation is similar to what is done using Google [19] in the conventional web. After that, the downloading procedure will be performed in the JXTA peer-to-peer network. At this point, the browser runs the HTTP Client, using p2psockets library [20], to download the file from the selected peer. When the content of the page arrives in the requesting peer it is shown in the browser screen.

Figure 4 shows the steps for locating and viewing a web page on this system. In this figure, the target page is *Page B* provided by *Peer B*. Peer A starts searching into DHT for peers that provide *Page B* and receives a list of these peers (steps 1, 2). In this example, there is only one peer that provides *Page B*, which is *Peer B*. By clicking in this option, the browser requests to its HTTP Client to download that page (steps 3, 4) from the provider peer, *Peer B*. Finally, the page is downloaded through JXTA (steps 5, 6) and is loaded in the browser screen.

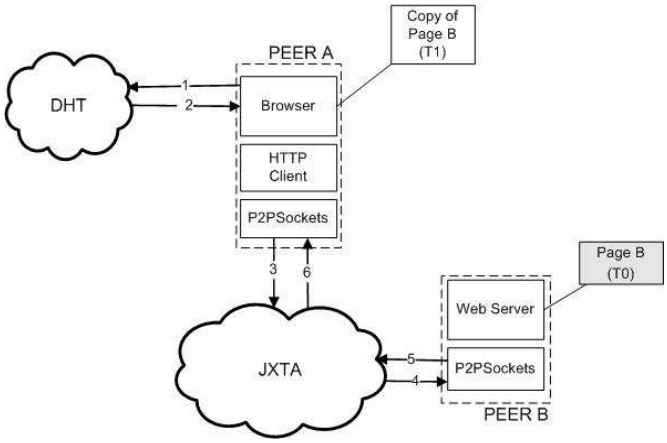

Figure 4. Searching Pages on Web2Peer.

C. Replicating Web Page

JXTA performs passive replication of the published resources across the network when peers access resources in remote peers. But the DHT indexes are not updated automatically by this JXTA replication. In order to show accurate information about the location of documents, the publisher must be aware of this replication and update the DHT indexes. When a web page is accessed, the replication procedure calls the publisher to create a new DHT entry. It shows to the entire peer-to-peer network an optional location for that web page.

Figure 5 shows the replication procedure. After searching into DHT (steps 1, 2) the Peer C receives a list with two possible locations for the target page B. This page was originally created by Peer B (Page B Copy 0) and it has a copy created by a replication on Peer A (Page B Copy 1). If the user decides to download the page from Peer A (steps 3, 4, 5) the Peer C will receive the page through the JXTA network (steps 6, 7, 8) and it will also have a copy (Page B copy2). After receiving the page, the publisher of peer C will update the DHT (step 9) and the replication is done. It makes Peer C a new provider of the page B.

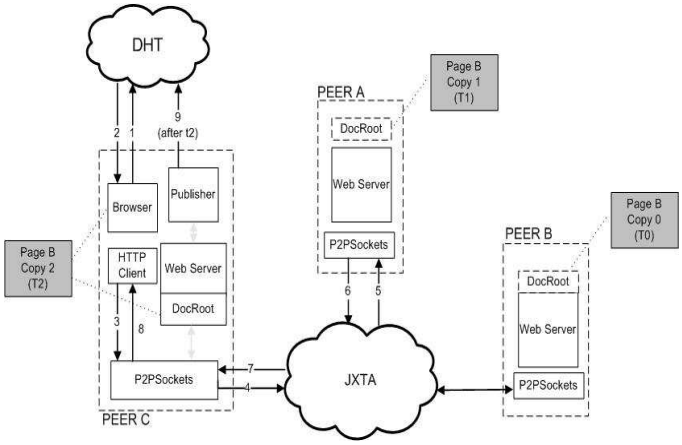

Figure 5. Web2Peer Replication Process.

V. PROTOTYPE IMPLEMENTATION AND RESULTS

The devised prototype has preserved the functionalities of a conventional browser, with the ability to navigate in the current WWW environment. Java was used as programming language, due to its platform independence. Figure 6 shows the browser's interface, and its main components:

1. Control buttons (*backward*, *forward*, *stop*, *refresh*, *editor*, *homepage*, *tools*, *preferences*);
2. Address bar;
3. Search bar (DHT x JXTA);
4. Display area.

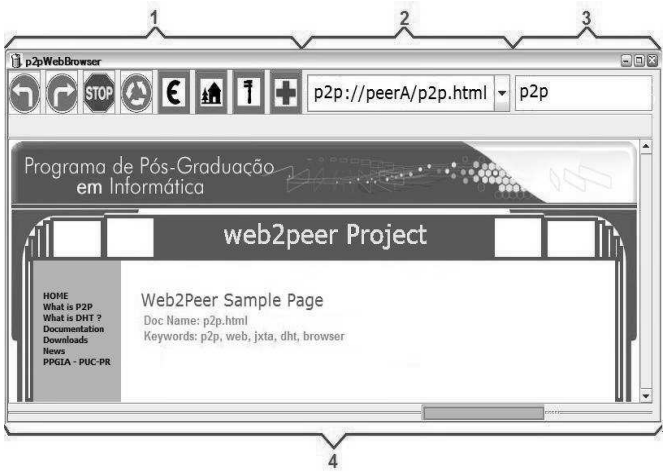

Figure 6. Web2Peer Interface.

While loading the browser, a connection to the JXTA peer-to-peer network is created. During this connection, the peer will receive an ID from JXTA network. This connection is performed through the peer rendezvous or peer relays, depending on the network environment where the peer is located. Moreover, at this moment the browser will create an input pipe that will be listening for web page requests. The web server Jetty, which was ported to answer the HTTP requests through the JXTA network, provides the possibility to transfer all desired pages through the peer-to-peer network. Thus, the loading operation of the new browser is not so different from the current ones. When the browser finishes loading, the user can do any query in the network, from search bar, as shown on item 3 of figure 6. When the keywords are inserted on this field, the browser searches the DHT to find where the page, associated to this keyword, can be requested. The browser itself can save the P2P addresses in its bookmark, if the provider peers are considered reliable.

A. Distributed Hash Table

The three main functions of the infrastructure use the DHT. Among some possibilities, we have chosen to use a DHT implementation previously tested. The system chosen was the *openDHT project* , which is a public service that is currently performed in 700 nodes spread in about 300 sites across the world, controlled by *PlanetLab* [21]. Each one of these nodes uses Linux as operating system that runs a *Bamboo* implementation. *Bamboo* is an implementation of the Pastry protocols. *Pastry was* presented in 2001, when the first DHT architectures started to appear. Each DHT node stores in its local disks a portion of the entire DHT storage. The DHT nodes can execute inserts, query, and removal tasks and other message routing operations. Each openDHT node is also known as DHT-*gateway*, it allows the clients that is not part of DHT to store, query and remove pairs such as any other node present in DHT. It allowed the *browser* to create its content indexing service.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodCall>
   <methodName>put_application_name</methodName>
   <params>
      <param><value> key </value></param>
      <param><value> value </value></param>
      <param><value> secret_hash </value></param>
      <param><value><int> ttl </int></value></param>
   </params>
</methodCall>
```

Figure 7. DHT Insert Message.

To perform the DHT functions, it was created a DHTClient which uses XMLRPC to communicate to DHT-gateway. The *DHTClient* performs the setting up of the XML messages to execute each desired function (insert, get, remove). To insert a value into DHT, a message must be created as shown in figure 7. Where the field *key* represents the key to be inserted on the DHT and *value* is the value referred to this key. There are other control fields as *put_application_name*, the field that identifies the method to be executed, *secret* is the password used for removing an entry, and *ttl* is the time, in seconds, that this entry should remain in the DHT. The return message to the insert request is a message with three possible answers: success, out capacity, or failure.

To search a value on the DHT, the *DHTClient* creates another message, as shown in figure 8. This message is more simple than that of performs an insert. Further the key, only three other control parameters are needed and then the DHT can answer with the value associated to this key. These control parameters are *get_application_name*, *maxvals* and *placemark*, which represent the name of the executed method, and the maximum number of elements that should be returned by query.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<methodCall>
   <methodName>get_application_name</methodName>
   <params>
      <param><value> key </value></param>
      <param><value> maxvals </value></param>
      <param><value> placemark </value></param>
   </params>
</methodCall>
```

Figure 8. DHT Search Message.

The return of this query is an *array* with all found values to the informed keys on the query.

Because there are a lot of current active DHT-*gateways* the Web2Peer browser has a list of them. If needed, this list can be updated via HTTP using the browser tools. These tools also allow checking the availability of each DHT-gateway.

B. JXTA Peer-to-Peer Network

As mentioned before, the JXTA network was created by using the p2psockets API [20]. P2psockets is a reimplementation of the java socket, and has some services already ported to work on the JXTA network. Two of these services are essential to the proposed infrastructure: the JXTA Web Server and the JXTA HTTP Client. With these services, it is possible to transfer HTML files through the peer-to-peer network, so both need to be connected to the JXTA network. This connection is performed through a peer relay and/or a public peer rendezvous, available over a set of public nodes on the internet. The connection process takes place as it follows: after connecting to the public peers, peer connects to a JXTA network peergroup, receives a PeerID from network, and then launches an input pipe, then the other peers can connect and access the pages published by it. However, all the complexity of these JXTA activities is minimized by using p2psockets. For instance, Web Server and HTTP Client connections to the JXTA network are achieved by using any of the following methods: *signin* and *autosignin.*

Table I shows a list of parameters used by the HTTP Client during a request for pages, and also the parameters used by

the Web Server (jetty) at the moment the browser is initiated. Each of these parameters is sent to its respective method, according to p2psockets API specification.

JXTA HTTP CLIENT	JXTA WEB SERVER
Peer Name	Pipe Name
Password Local	PeerGroup
PeerGroup	Peer Name
Pipe Name + HTML file	Password Local
	Virtual Port (default=80)

Table I. Parameters Used By Internal Web Server and Http Client

There are also parsers mechanisms built based on the identification of pre-defined strings and its breaking tokens among these strings. Usually, the results were stored in arrays of strings for later use.

C. Results

The working environment used for testing the prototype is composed of two edge peers running on different computers with Windows XP operating system. The prototype was created using Eclipse 3.1 IDE and the Java Virtual Machine 1.5 version. In addition, a peer relay was created running Linux UBUNTU 6.06/kernel2.6 as operating system. This relay peer, connected to the JXTA network, works as a gateway between peers inside of firewalled networks and the JXTA network. During the tests two edge peers were connected from different physical networks through the internet. One of them connected to a domestic ADSL connection and the second one connected to a 100 Mbps Ethernet in the university laboratory. There was also a relay peer connected straight on the internet, listening on the ports 80, 22, and 9701, as shown in the figure 9. To check the proper functioning of the system it was created a simple HTML document in one of the peers, as shown in figure 6, some keywords were associated to this document, and published into DHT. Next, the other peer searched one of the used keywords and it received a list with the peer that was providing that page. After clicking on the peer, the page was downloaded through the JXTA network, as expected.

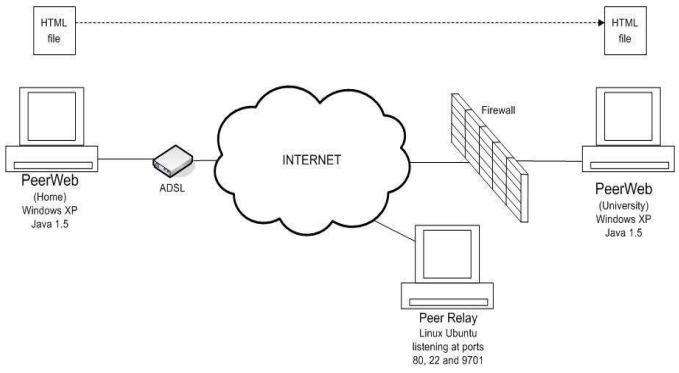

Figure 9. Working Environment.

After showing the page on the Web2Peer screen, it was done a new search with the same keywords. At this point, it was possible to check that the same file was available in two different peers, now proving the replication was working as expected.

For queries done from keywords in the DHT, using samples that bring back 10 results each attempt, such as Google shows its answers, the spent time was a little bit higher than Google. However, as mentioned before, the environment in which Google is running is very different in terms of hardware and software therefore providing such result. Even so, the obtained results were pretty good justifying the usability of the system. In figure 10 is shown the times obtained in 10 queries with different number of keywords.

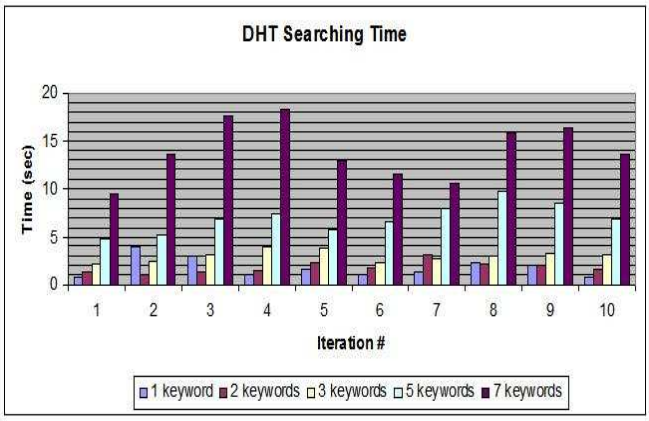

Figure 10. Searching Time.

After searching the pages and choosing the source peer for downloading it, there is a time spent by a *peer* to transfer such file over the JXTA network. For this transfer it was captured the time elapsed between the *click* of the mouse, that selects the source peer, and the local saving of the specific file. This test was repeated 10 times and the results are presented on figure 11.

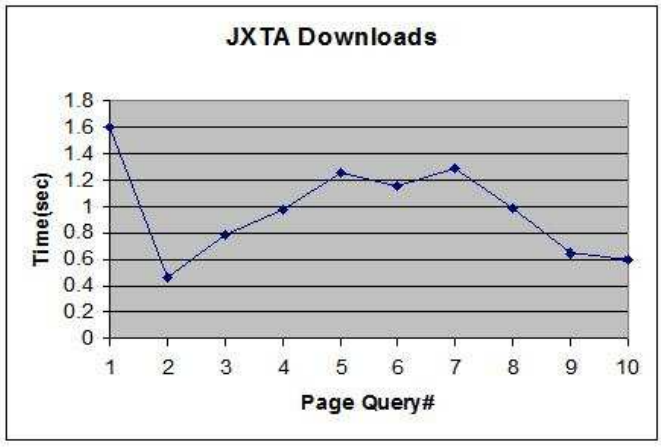

Figure 11. JXTA Downloading Time.

VI. Related work

Although other work have presented similar ideas, none of them has presented the set of facilities that our project does. Everything the user needs to do in this case is to download the browser, and create a web page. Some of these projects are mentioned below.

In 2001, the Freeweb project [22] has introduced itself as an alternative for publishing web pages on internet preserving the identification of the publishers, emphasizing the anonymity and censorship. This project provides in fact a simple interface for publishing HTML files in the Freenet network [23]. Freenet is a peer-to-peer network that associates contents to keywords and these keywords to peer-to-peer network locations. The peer location on Freenet is based on IP address of the node. This feature makes the Freeweb dependent on the network architecture and works only with TPC/IP protocols. Our JXTA approach is platform independent. The Freeweb offers independency of regulatory entities like our project does but its search mechanism is based on flooding, resulting in a slower mechanism if compared to deterministic mechanisms offered by DHT, as our approach suggests.

Even being a considerable job, this project did not have improvement. The project web site [22] shows that the project started in March 2001 and after November, in the same year, nothing else was done.

In 2003, another project [24] created a web browser over a peer-to-peer network using JXTA protocols. But this web browser had a specific use. It was a collaborative web browser (groupware) based on peer-to-peer networks. This browser allowed the users to share their view with other users in a synchronous mode. The document visualized by a single user is shared with every member in the group and the actions on this web page are synchronized with the others. This browser was developed using Java language, using a small protocol which exchanges few messages to synchronize the peers. These messages are exchanged through the JXTA network. It is an interesting work in terms of collaboration, but it does not provide any mechanism for publishing local contents on the peers. All the shared pages by peers are in centralized servers on the internet.

There are other ideas such as content distribution network that suggests the solutions to improve the content availability by the server side, on client/server architectures. In this case a peer-to-peer network is used in the server side. Our approach provides the availability by the servant side, comparing with client/server model should be called client side.

VII. Conclusions and future work

The main goals of this work are to improve the availability of the web pages published on the internet and make the publishing process easier by breaking the dependency of the regulatory entities resulting on a more democratic Internet. These objectives were achieved by using JXTA protocols and a set of other existing technologies and mechanisms, providing a user-friendly web browser that works in a conventional Web and in a peer-to-peer network.

Some aspects were not considered in the first scope of the project. These points must be, or are already being, improved in a short term. Examples of such improvement are: a mechanism for controlling the version of the updated pages, development of a security mechanism that allows a user to digitally sign web pages before publishing them, providing authenticity and integrity of the pages, development of some cryptographic mechanism for providing confidentiality of the web page, development of reputation mechanism for scoring the peers and their behavior on the network, and access control mechanism based on public key infra-structure (PKI).

The Web2Peer browser prototype can be downloaded from the project site http://www.ppgia.pucpr.br/~lau/Web2Peer for testing.

ACKNOWLEDGEMENT

This work was sponsored by CAPES – (Brazilian Ministry of Education Agency) with a Master' Scholarship Program.

REFERENCES

[1] R. Orfali, D. Harkey, and J. Edwards, *Client/Server Survival Guide*: John Wiley & Sons, Inc. New York, NY, USA, 1999.

[2] T. Berners-Lee, L. Masinter, and M. McCahill, "RFC1738: Uniform Resource Locators (URL)," *Internet RFCs*, 1994.

[3] T. Berners-Lee, R. Fielding, and L. Masinter, "RFC2396: Uniform Resource Identifiers (URI): Generic Syntax," *Internet RFCs*, 1998.

[4] ICANN, "ICANN WebSite, http://www.icann.org/, Accessed: October 2006," 2006.

[5] Rüdiger Schollmeier, "A Definition of Peer-to-Peer Networking for the Classification of Peer-to-Peer Architectures and Application," presented at First International Conference on Peer-to-Peer Computing (P2P`01), Linköpings universitet, Sweden, 2001.

[6] S. Androutsellis-Theotokis and D. Spinellis, "A Survey of Peer-to-Peer Content Distribution Technologies," presented at ACM Computing Surveys (CSUR), 2004.

[7] E. K. Lua, J. Crowcroft, M. Pias, R. Sharma, and S. Lim, "A Survey and Comparison of Peer-to-Peer Overlay Network Schemes," presented at IEEE Communications Survey and Tutorial, 2004.

[8] L. Gong, "Project JXTA: A Technology Overview," Sun Microsystems, Inc., Palo Alto, CA, USA October 29 2002.

[9] B. Traversat, A. Arora, Mohamed Abdelaziz, M. Duigou, Carl Haywood, J.-C. Hugly, and B. Y. Eric Pouyoul, "Project JXTA 2.0 Super-Peer Virtual Network," Palo Alto, CA, USA May 25 2003.

[10] I. Sun Microsystems, "JXTA v2.3: Java Programmer's Guide," I. A. r. r. Sun Microsystems, Ed., 2005.

[11] B. J. Wilson, *JXTA*. Indianapolis: New Riders Publishing, 2002.

[12] S. Rhea, B. Godfrey, B. Karp, J. Kubiatowicz, S. Ratnasamy, S. Shenker, I. Stoica, and H. Yu, "OpenDHT: a public DHT service and its uses," *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pp. 73-84, 2005.
[13] J. A. Mussini, "Distributed Hash Table: The State of the Art," 2006.
[14] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content Addressable Network," presented at ACM SIGCOMM Special Interest Group on Data Communications (SIGCOMM`01), San Diego, California, USA, 2001.
[15] I. Stoica, R. Morri, D. Karger, M. F. Kaashoek, and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for internet applications," presented at Conference on Applications, technologies, architectures, and protocols for computer communications San Diego, California, United States, 2001.
[16] A. Rowstron and P. Druschel, "Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems," *Lecture Notes in Computer Science*, vol. 2218, pp. 329–350, 2001.
[17] B. Y. Zhao, L. Huang, J. Stribling, S. C. Rhea, A. D. Joseph, and J. D. Kubiatowicz, "Tapestry: a resilient global-scale overlay for service deployment," *Selected Areas in Communications, IEEE Journal on*, vol. 22, pp. 41-53, 2004.
[18] J. Li, B. T. Loo, J. Hellerstein, F. Kaashoek, D. Karger, and R. Morris, "On the Feasibility of Peer-to-Peer Web Indexing and Search," *IPTPS'03*, pp. 207-215, 2003.
[19] Google, "Google WebSite, http://www.google.com , Accessed: October 2006," in *3*, 2, Ed.: 4, 2006.
[20] j. bradneuberg, "P2psockets Project Home, http://p2psockets.jxta.org, Acessed: Jul, 2006," 2003.
[21] P. L. Consortium, "PlanetLab Consortium , http://www.planet-lab.org/, Acessed: Jun, 2006," 2006.
[22] F. Project, "Freeweb Web Site, http://freeweb.sourceforge.net, Acessed: August 1st, 2006," 2001.
[23] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong, "Freenet: A Distributed Anonymous Information Storage and Retrieval System," presented at ICSI Workshop on Design Issues in Anonymity and Unobservability, International Computer Science Institute (ICSA), Berkeley, California, USA, 2000.
[24] M. Nakamura, J. Ma, K. Chiba, M. Shizuka, and Y. Miyoshi, "Design and implementation of a P2P shared Web browser using JXTA," presented at 17th International Conference on Advanced Information Networking and Applications (AINA 2003) 2003.