Cigarra - A Peer-to-Peer Cultural Grid

Alexandre Freire , Francisco Gatto , Fabio Kon

[1]Department of Computer Science
Institute of Mathematics and Statistics
University of São Paulo

{alex,gatto}@arca.ime.usp.br kon@ime.usp.br
http://cigarra.arca.ime.usp.br

Abstract. *The Ministry of Culture in Brazil, started in 2004 a digital inclusion project in which distribution of multimedia content is paramount. In light of this incentive, and by request of the Ministry, the XP lab at IME-USP developed Cigarra, an open-source, peer to peer file distribution software that allows the cultural content produced by local communities to be distributed over the Internet. This paper describes the development of Cigarra, how it works and the benefits it provides to users. We will also describe the free-software components that were used and refactored during the development, and our interactions with the free-software community. We will then describe the evolution of Cigarra during the XP lab, and how XP and free-software fit nicely together.*

1. Introduction

The production and distribution of cultural content has been ground for discussions worldwide; a recent article in Wired Magazine points Brazil as the country leading the way in this debate [14]. Copyright issues impose restrictions that technology could counter; thus many give up on pursuing ways to ease distribution of content in the Internet. Meanwhile, Brazil is a developing country, with a "world famous" culture, exporting from songs and artists to regional dishes worldwide. Although this is visible worldwide, a whole new ecology of culture flourishes in Brazil, regional culture, rich and meaningfull, which is not distributed even inside Brazilian borders, much less internationally. In order to sponsor this regional cultural content production, the Ministry of Culture has started a digital inclusion project called "Pontos de Cultura" (Culture Hotspots). The basic idea is to install cultural hotspots all over the country, where cultural projects with proven actions in poor and remote regions will receive annual funding and will be equipped with a multimedia production kit, composed of audio and video production capable hardware, microphones and cameras, a broadband connection to the Internet and a pre-configured server that will join them in a network. People from the communities around such hotspots will be stimulated to use those facilities and record songs, movies, video clips, etc., always being encouraged to license their creative products with CreativeCommons [11] like licenses and to share the result over the Internet.

All this new cultural content, video and music artifacts as well as pictures and texts, must be published and made available to anyone interested, anywhere in the world. And so arises a new problem: How would the Ministry distribute this new (and potentially massive) content to the population? How would the population filter the content, to find just what they are interested in? Due to limited funding, the possibility of centralizing the content was not a choice, and so consultants for the project suggested that a peer-to-peer architecture be adopted and a new software be developed rising to such challenges as

extreme ease of use and scalability. This software was to be developed as an assignment on a popular elective course that teaches students eXtreme programming [13], IME-USP's XP lab, which have already produced free-software previously, but usually for demands generated internally at the University of São Paulo [15].

In Section 2, we will describe the requirements and approaches to the distribution strategy. In Section 3, the solution is detailed, as well as the free-software technologies, frameworks, tools, and software that were used, refactorated and incorporated in Cigarra. Section 4 analyzes the advantages and disadvantages of undertaking a free-software project and the benefits of doing so using the eXtreme programming method. Finally, Section 5 wraps up drawing some conclusions about the problem, the solution, and about free-culture in general.

2. The Need to Distribute

Making new content produced on the cultural hotspots available to the population on a country as big as Brazil can be quite a challenge. The large area itself is a problem, but not the only one. The cultural differences between regions, and the (potentially) massive amount of information generated by the hotspots, should also be taken into account. These demands, teamed up with the possibility that some of the content produced will "tip", and have very high demand, while others will not, call for a very scalable, simple to use, distributed approach to the publishing problem. And yet, those are minor issues, when compared to problems such as energy availability to power the multimedia kits in indigenous tribes in the Amazon rain forest, for example.

On the other hand, having a lot of information available can be quite stressful when one is looking for something specific. Many different people, from many different backgrounds will produce media content and the consumers will need means to find the needle on the hay stack. So artists must have a way to describe their work in detail, in such a way that it can be found later by their fans, relatives, or artistic soul-mates.

The solution proposed and implemented by the XP lab was the distribution of the multimedia files by a peer-to-peer network, where, when publishing, the content producer would provide meta-information about what was being published, such as author, license, creation date, type of media(e.g., music, video, picture, text) and even lyrics for a song; so that users could use that information to search for specific content.

The meta-information is an important part of the solution, because without it publishing would not be useful. It indicates the type of content and in addition to general fields, such as author and creation date, proposes different fields for different types of media (e.g., a director field for a movie or a lyrics field for a song). In particular the application prompts the user to specify the licensing of the content being published, so other users know the restrictions, or lack thereof, of using published content both as spectators and as artists creating new works of art by recombining and modifying content produced by others.

3. The System

The XP lab at IME-USP is a semester long course [16] targeted at undergraduate students on the 3rd and 4th year of the Bachelors program in Computer Science. In the course, students learn the eXtreme programming method [13] while using it in a real world context. In this case to answer a request from the Ministry of Culture: to allow the cultural hotspots to freely publish content produced locally. Another point made by the government was that everything had to be free-software.

So, the 2004 class of the XP lab developed Cigarra, a peer-to-peer cultural grid, developed on top of the BitTorrent technology [1]. BitTorrent is a protocol for distributing files that integrates seamlessly with the Web, i.e., users start their downloads from a URL, just like they would to get a file from a Web site. Then BitTorrent kicks in and the download happens from other users instead of from the server. The important thing to notice about the peer-to-peer network created by BitTorrent is that it forces users to cooperate: Anyone downloading a file is also using their upload capacity to share the file with others. This results in a virtually limitless distribution network, since each new participant brings not only demand, but also supply. That way, if a specific file becomes a popular "hit", the system will not fail, for it is scalable and fault tollerant.

A BitTorrent network has four separated components: *metainfo* files containing all the information needed to allow others to download the file; a regular Web server that servers the *metainfo* files; a *tracker* application that keeps track of the content files; and the *clients*, that download and upload from each other as well as updates the tracker.

Figure 1 shows how it works.

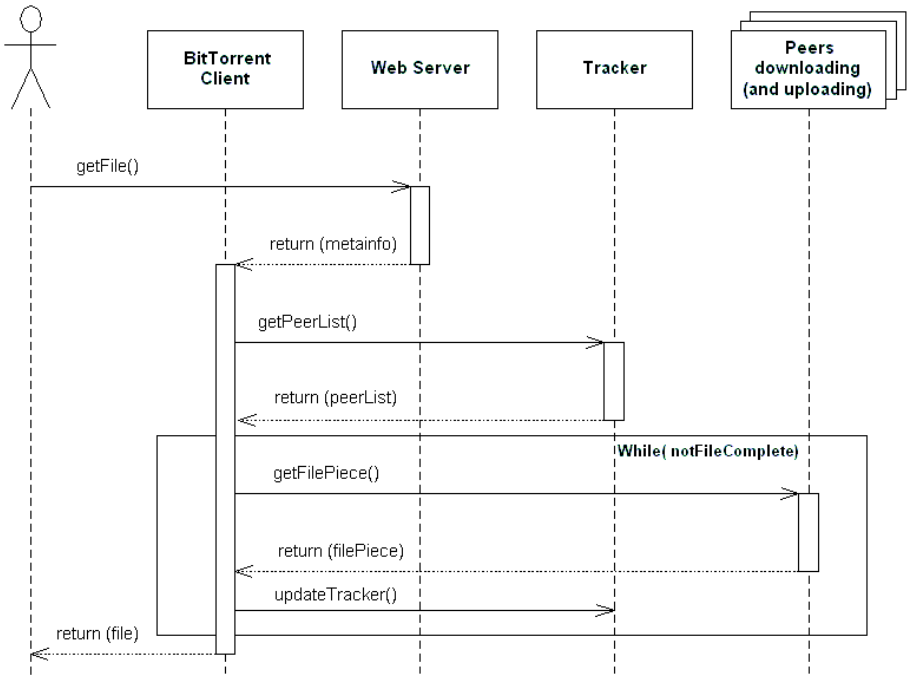

Figure 1: The BittTorrent file distribution protocol.

Each *metainfo* file has the URL of the *tracker* managing the published file; the length of the file; the size and number of pieces the file will be splited and a calculated hash for each piece of the file. The Web server responds to request for static *metainfo* files as usual, but the BitTorrent client starts on the user's computers and requests a peer list to download from, to the the *tracker* specified in the *metainfo*. This is done over HTTP or HTTPS. The downloaders then connect over TCP directly to other peers on their list to download pieces of the file. When the download of a piece completes, the BitTorrent client uses the hash provided in the *metainfo* to verify its integrity. Once each

piece completes, the downloader is ready to upload that piece to others. Periodically, the downloaders send update to the *tracker* to keep it informed of their progress.

Cigarra extends BitTorrent by supplying a *rich client* and a Web application to go with it, and extending the *metainfo* files, so now they also carry information about its content, such as title, author, creation date, licensing, etc. As an extension to the BitTorrent protocol, the new *metainfo* file is completely backward compatible with other BitTorrent tools.

The *rich client* is a tool that runs on each peer and retrieves a list containing the result of a search for some content and displays it to the user, who can ask for more detailed information about a specific file and download it. The peer tool is also used for publishing new multimedia content. This is done by selecting a file from the local computer and providing general meta-information about its content. After that, the user is queried about the type of content being published e.g., music, picture or video, and then asked to provide specific meta-information about that type. The tool then generates a *metainfo* file, uploads it to the Web application, alerts the *tracker* that new content has been published and starts to download it as the origin.

The Web application was designed to interact with the *rich client*, providing the content list and receiving the *metainfo* file for published content. It is thought of as one possible aggregator of the information being published, a central place where one can search trough everything that is published by the cultural hotspots.

Figure 2 shows the Cigarra extensions to BitTorrent.

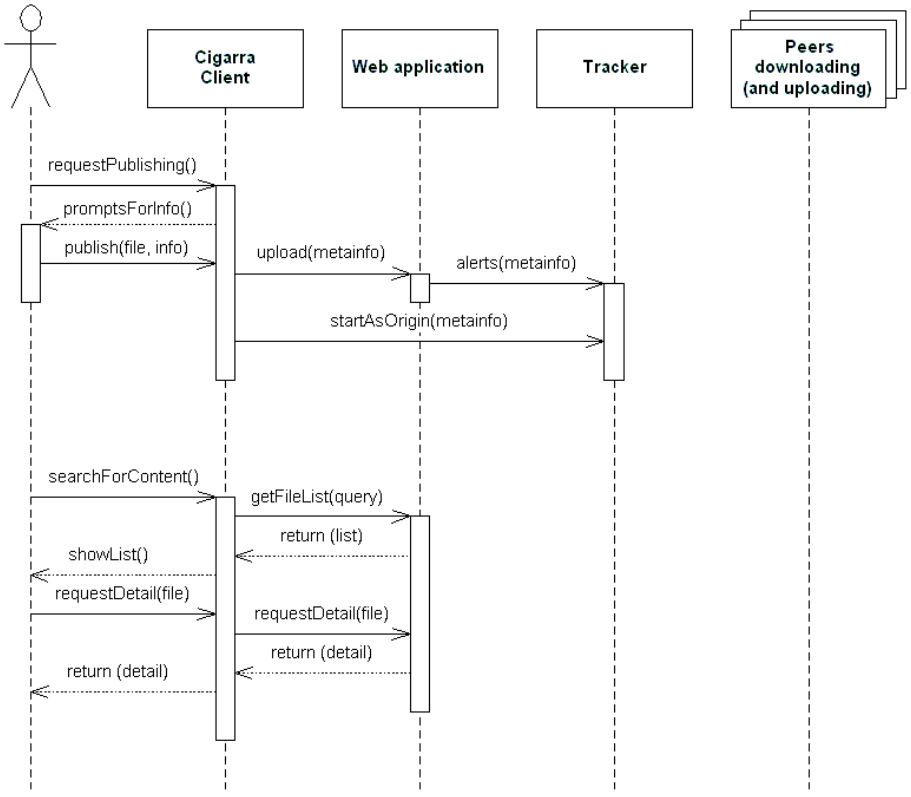

Figure 2: Cigarra extensions to the BittTorrent protocol.

3.1. Free Software Tools

Since its conception, Cigarra was based on a free-software infrastructure. It was designed on top of the already popular bittorrent protocol and developed in Java using SWT technology from the Eclipse Framework [3]. Cigarra was developed on GNU/Linux workstations and tailored to work on a GNU/Linux environment (such as will be available to users of the Cultural Hotspots). The project was hosted on Arca's [4] CVS server and was accessed by the developers both from the lab and from home.

To bootstrap the development and accelerate the project, we used code extracted from the Azureus [5] project, a bittorrent client developed in Java. We also used Azureus as a benchmark to compare with Cigarra.

The Eclipse framework was used both as a developing tool and as a base for our interface, totally based on SWT. XML technology was used on the server, to create lists of all files published and also to convert these lists to RSS format. Our Web application followed the J2EE specification and was deployed in a Tomcat container.

BitTorrent technology was incorporated not only from Azureus but also from the original BitTorrent implementation. Cigarra is currently using the original bittorrent tracker, developed in Python [2] by Bram Cohen, for example.

3.2. Current Status

In its current status, Cigarra is ready for deployment, offering a fully functional interface that allows one to publish multimedia files, add meta data, and search and download multimedia artifacts. It has some user interface flaws and sometimes depends too heavily on external tools (such as the BitTorrent tracker). But it has a complete test suite and it passes all functional tests proposed by the Ministry's consultants as necessities for the 1.0 release. It is currently being maintained by the Arca community and can be freely downloaded from `http://cigarra.arca.ime.usp.br`.

3.3. Future Developments

We will have to work on making the interface simpler and easier to use. At the moment, there are no progress bars to indicate evolution of a download, for example. We will also tackle the challenge of creating a Web interface for those that do not want to use the rich client to publish content, and also to allow others to edit meta-information for files that have already been published.

We also plan on using Kenosis [6] to enable the BitTorrent trackers to work in a distributed fashion. Our architecture currently has one central point of failure, the tracker. It keeps track of peers and manages a list of who has which files, and where. Without it, peers don't have access to bittorrent's greatest strengths. Kenosis is a fully-distributed peer-to-peer RPC system built on top of XMLRPC. Nodes are automatically connected to each other via a Kademlia-style network [17] and can route RPC requests efficiently to any online node. Kenosis does not rely on a central server - any Kenosis node can effectively join the network from any connected node. Using Kenosis-enabled BitTorrent we plan to eliminate this central point of failure for BitTorrent downloads.

3.4. Related Projects

Our work is related to a couple of other free (or not) software development efforts. We feel we have learned a lot from these projects and would like to mention a couple of them from where we got some of our ideas.

The first one is ApnaOpus [7], a software being developed to be used as a semi-decentralized content management system in the labs of the cybermohalla project in India. ApnaOpus helped us better understand how our meta-data model should work. This project and our own were developed at about the same time, by having access to ApnaOpus documentation we found insight into how to expand the BitTorrent protocol to allow the use of meta-data.

We would also like to point that we were inspired by other fresh projects such as BlogTorrent [8], TorrentFlux [9], and DropTorrent [10]. The mere fact that these projects exist serve as inspiration for a software like Cigarra. All of these projects pretend to do something new with BitTorrent. BlogTorrent wants to make it easier to use, so that bloggers whom may not have technical backgrounds can quickly publish their photos or videos using the BitTorrent protocol. DropTorrent is developing a very easy to use client shortcut (currently only available for windows platforms) that will enable users to drag-n-drop files to publish them in a BitTorrent network. TorrentFlux is a PHP client that allows one to publish and download torrents in a Web Sever controlled by a simple web interface. These projects show that Cigarra is clearly something that many will be able to benefit from.

4. Free-Software and Extreme Programming

We have had a great experience using eXtreme Programming (XP) techniques during the development of Cigarra.

A common problem when introducing XP is the bootstrap story [12]. It shows that pair-programming [13], continuous integration [13] and collective code ownership [13] (important practices in the method) are hard to apply when the project is starting and there is not much code to build upon. We found that using an existing free software code base is an excellent idea to bootstrap XP projects.

During our experience, we were teaching students how to write automated tests. We suggested that students ignore the external free software tools and simply assume that these tools worked, for they were tested by the free software community. In retrospect this was not a very good idea. Although students applied themselves to writing tests for the code they wrote, some where skeptical about the free software code we were reusing and refactoring. When bugs in the software stopped our production, many students where quick to point the problem as coming from this external code base. As we found out, this was never the case. We suggest that a solution for this problem is to write automated tests for every open or free software library or tool being used in the project. These tests can serve two purposes, the first is to teach automated testing to students, the second is to give them more confidence that the external code is working. These tests could also serve as a nice contribution to the free software community, which rarely has test suites for their projects.

It was hard to have follow the “customer always present” XP practice [13] since the consultants from the Ministry of Culture were not available all the time and because they did not know where the cultural hotspots would be installed. We tried to solve this problem by electing the coach to play a customer role. For this he had a special hat only to be used when he was speaking in behalf of the customers.

We also believe that the software produced has none, or very few bugs, and has delivered value to our customer. We have a full test suite and have passed customer’s acceptation tests defined for the 1.0 release.

5. Conclusions

We hope that in its current version Cigarra will satisfy the needs of the Cultural Hotspots. We believe BitTorrent was an excellent choice of protocol and that our addition of metadata is an important contribution. Even tough some improvements are necessary, we are eager to see Cigarra in action. We had a nice experience using XP to build this software and hope to continue building upon our codebase, which we believe is of high quality.

References

[1] Bittorrent project. http://www.bittorrent.com.

[2] Python. http://www.python.org.

[3] Eclipse. http://www.eclipse.org.

[4] Arca - free software development collective in brazil. http://arca.incubadora.fapesp.br.

[5] Azureus project. http://azureus.sourceforge.net.

[6] Kenosis. http://freshmeat.net/articles/view/1440/.

[7] Apnaopus. http://apnaopus.var.cc/wiki/index.php/Main_Page.

[8] Blogtorrent. http://battletorrent.sourceforge.net.

[9] Torrentflux. http://www.torrentflux.com.

[10] Droptorrent. http://writtorrent.sourceforge.net.

[11] Creativecommons. http://www.creativecommons.org.

[12] Jennitta Andrea. Managing the Bootstrap Story in an XP Project. *XP 2001*, 2001. http://www.agilealliance.org/articles/reviews/Andrea1/articles/ManagingTheBootstrapStory.pdf.

[13] Kent Beck. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 2000.

[14] Julian Dibbell. We Pledge Allegiance to the Penguim. *Wired Magazine*, pages 190–197, November 2004.

[15] Alexandre Freire, Alfredo Goldman, Carlos Eduardo Ferreira, Christian Asmussen, and Fabio Kon. Mico - University Schedule Planner . *Anais do 5o Workshop sobre Software Livre*, pages 147–150, 2004.

[16] Alfredo Goldman, Fabio Kon, Paulo J. S. Silva, and Joseph W. Yoder. Being Extreme in the Classroom: Experiences Teaching XP. *Journal of the Brazilian Computer Society*, 10(2):1–17, November 2004.

[17] P. Maymounkov and D. Mazieres. Kademlia: A peer-to-peer information system based on the XOR metric. *In Proceedings of IPTPS02, Cambridge, USA.*, March 2002.