Towards a Hybrid Client/Server and P2P Architecture for Content Delivery over the Internet

Soufiane Rouibia[1], Majd Ghareeb[2], Benoît Parrein[2], Marco Biazzini[3], Raziel Carvajal-Gomez[3], Adriana Perez-Espinosa[3], Patricia Serrano-Alvarado[3]

[1]TMG
St Sébastien sur Loire – France
rouibia@tmg.eu

[2] LUNAM Université
Université de Nantes,
IRCCyN UMR CNRS 6597
Polytech Nantes, France
name.lastname@univ-nantes.fr

[3]LUNAM Université,
Université de Nantes,
LINA UMR CNRS 6241,
Nantes, France
name.lastname@univ-nantes.fr

Abstract— **Regarding the limitations of the traditional Client/Server communication mode from the one hand, and of the P2P mode from the other hand, hybrid network architectures have recently gained large popularity. In this paper we propose a new hybrid architecture that is called P2PWeb, between the centralized client/server and the non-centralized P2P architectures for content delivery over the Internet. The main objective of this proposal is to reduce the load over the server in order to provide a better Quality of Service (QoS) for the end-users. A BitTorrent-like protocol has been implemented and deployed to reach the objective. Moreover, we support our system with a user-satisfaction control technique that helps to improve the provided service of the hybrid P2PWeb architecture. The experimentation results and the performance evaluations that we have made show the efficiency of the proposed system in terms of QoS evaluations.**

Keywords- Content delivery; Client/Server; P2P; Hybrid architectures; P2PWeb; User-satisfaction; WUW.

I. Introduction

The rapid development of networking technologies has tremendously facilitated the content delivery over Internet. In this scope, the traditional client/server (C/S) communication mode has suffered from several drawbacks that necessitated the deployment of alternative solutions such as the Peer to Peer (P2P) communication mode. However, in order to provide an efficient reliable and scalable content delivery service, neither a C/S-based, nor a P2P-based architecture can reach the target alone. The centralized server of C/S architecture may not support the huge loads over the Internet, which undermines the system scalability and leads to a bad content delivery service. It requires high accessibility features that may not be easily and cheaply provided [1]. On the other hand, P2P architecture needs a sufficient number of 'seeders' (content sources) to launch the content delivery service. Besides, it is not easy to fairly determine the contribution of each peer, which may also affect the reliability of the system. To cope with these issues, different researches have been already proposed, to provide hybrid compromise network architectures between C/S and P2P architectures [1,2,3,4]. In some of these works, authors propose to release the server after a given instant and to switch to a pure P2P communication mode [1,2]. Other approaches propose to start the content delivery with a P2P communication mode, and then to redirect the clients' requests to the content server in case of need [3]. Nevertheless, neither of the above listed researches has considered the user satisfaction about the provided service. In this article we propose a new hybrid architecture (P2PWeb) that helps to integrate the P2P technologies in a web environment in order to provide an efficient content delivery service over Internet. In our proposal, the server continues to response to clients' requests as long as it is not overloaded. Then, when it arrives to a saturation threshold at which it cannot deliver the content to any new client, it stays in the system and it starts to redirect the new clients (peers) to retrieve the content from the other clients/peers that are already present in the system and that are receiving that content. P2PWeb aims at reducing the load over the server and providing better QoS (Quality of Service) and QoE (Quality of Experience). On the other hand, and in order to improve the users' satisfaction in our hybrid architecture, we propose a technique for user-satisfaction control that is called WUW (What User Wants). WUW main objective is to personalize the end-users preferences in order to ensure the best user satisfaction from both points of view, the delivered service (server) and the perceived service (clients/peers).

Section II of this paper presents the hybrid P2PWeb architecture and its networking protocol in more details. In its turn, Section III shows the advantages of using this architecture by demonstrating some of the experimentation results and the performance evaluations that we have obtained. Furthermore, Section IV describes WUW technique and its efficient utility.

II. The Hybrid P2PWeb Architecture and Design

In this section, we introduce the hybrid architecture P2PWeb in more details. Then, we explain its network protocol and the P2PWeb compatible browser-plugin that we have developed. Moreover, we illustrate how the network coding can be efficiently used to improve the provided service and to ameliorate the availability of distributed data chunks.

A. *System Architecture*

In P2PWeb architecture, the server plays the role of a "*content provider*" and of an "*index server*" at the same time. For a given content, the delivery start in a C/S mode. Then, at a given clients' number threshold at which the server becomes saturated, the system changes the delivery to a P2P mode among the already connected P2PWeb clients and that potentially accept to share their downloaded chunks.

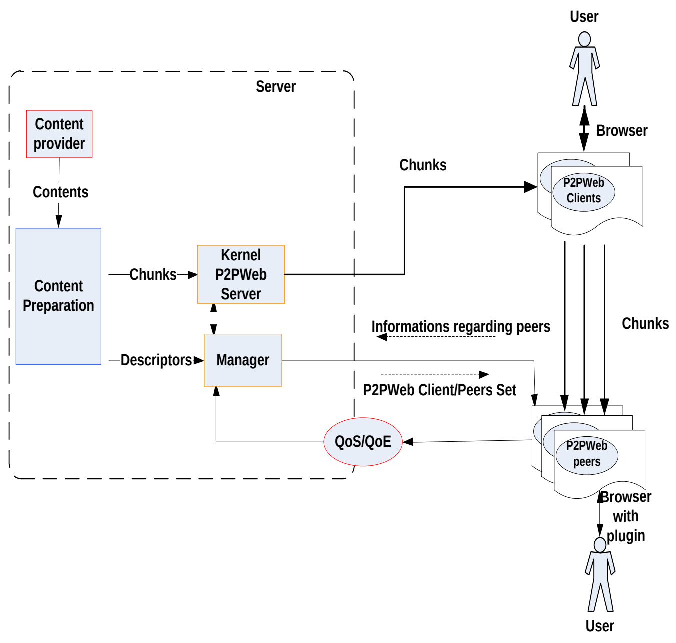

Figure 1. P2PWeb hybrid architecture.

Figure 1 represents the system architecture blocks. The first server's task is to provide the different types of contents (web pages, text, images, audio, video, etc) that will be distributed in the P2PWeb system. The content passes through a *Content Preparation* phase, in which it will be divided into variant number of chunks to be injected in the system. Chunks generating process depends on the type of content. For example, for the real-time media streaming, each chunk will represent the data to be delivered for a given portion of time. Hence, some important factors such as time stamp and chunk order should be taken into consideration. On the other hand, non-real-time content downloading has typically less restrictions in term of delay. Each content file will have its own related list of chunks that is signed by the content preparation service and that will be transmitted to the *Manager* by the *kernel P2PWeb Server*. This list contains the associated content ID, the chunk IDs, the chunk time stamp and the chunk hashes. To ensure the safety of the contents, these hashes are managed in a centralized manner via the Manager. P2PWeb Manager is the general coordinator that has a global knowledge about the system state. It knows:

- what content is available, where it is located and in which quality;
- which P2PWeb clients/peers are on-line and the resources/content they have;
- which and how many P2PWeb clients/peers request a given content.

As long as the server is not saturated, the kernel P2PWeb server responses to the new arriving request in a C/S content delivery mode, by sending the chunks that are formatted to be easily used in the P2P exchange between peers. Moreover, it declares the new client to the Manager as a new content owner. Although P2PWeb clients are receiving the content directly from the server, but we have to pay attention that they are not typical clients, since they are receiving the content in terms of chunks and at a given instant, they become sources for this content.

When the server arrives to a saturation threshold, with which it will not be able to deliver the content to any new client, the Manager takes the responsibility of responding to the new requests. This will be done by sending meta-data information about the content with a list of P2PWeb clients and peers that are already downloading this content to the requested user. A special P2PWeb plugin that is installed on the user browser will analyze the received information and will consequently start to download the content. Here, we can notice that as long as the server is at its saturation threshold, all the new arriving nodes will not be treated as clients, but as peers. Hence, instead of downloading the content chunks directly from the server, these peers will retrieve their requested chunks from the P2PWeb clients/peers that are already connected to the system and that are downloading the content in question.

P2PWeb hybrid architecture can be supported with an adaptive QoS/QoE management technique, like the one presented in [5]. With such a technique, P2PWeb clients and peers will feedback the Manager with activity reports about the perceived quality and the user satisfaction, so it will be able to improve the delivery service accordingly and in real-time.

Instead of starting from crash, we adapted the BitTorrent Open Source code, developed in Python Language that has already proved a good and reliable performance. The modifications that we have applied consist of adapting the queries for a Web exchanges. The first chunks to be downloaded are organized in order to have sequential and not random chunk downloading. We also added some parameters to measure the QoS and manage the overlay construction using these measurements.

B. *P2PWeb use case design*

Figure 2 depicts the sequence diagram of a use case for a content delivery process in P2PWeb system. In the first case, when the server did not reach its clients' number threshold yet, a simple content delivery process between the server and the user will be done in a C/S mode. On the other hand, the second case represents the steps of content delivery in P2P mode when the server threshold is reached. In this case, P2PWeb plugin will obtain the meta-data information and a set of P2PWeb clients/peers that are already downloading the content from the Manager. Then it will start to retrieve the content from this set.

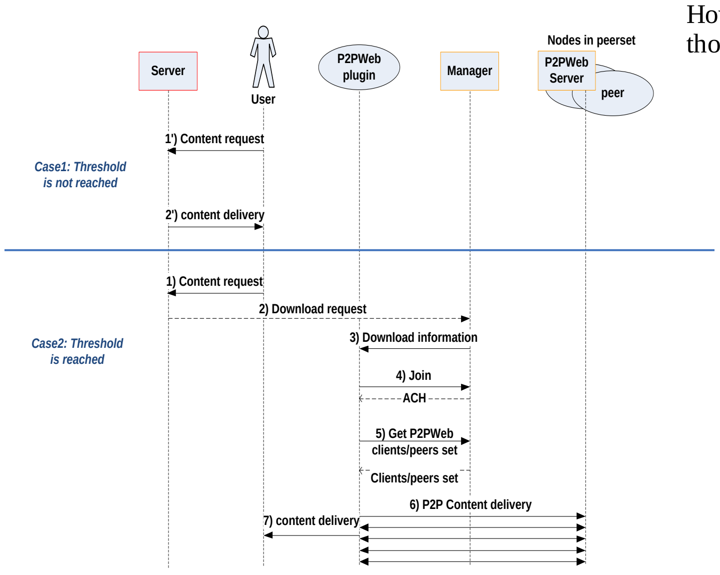

Figure 2. A use case for content delivery process in P2PWeb system

C. *Network coding*

A key factor that can help to improve the QoS in P2PWeb architecture is the introduction of redundant content chunks into the network. These chunks can be basically produced by linear combinations of original chunks. Many Forward Error Correction (FEC) algorithms can be used for this purpose, from the simplest ones e.g. XORing original chunks to the more complex encoding ones that includes optimal MDS property *i.e.* Maximum Distance Separable codes. By definition in error control theory, redundant codes produce n from k original message packets with $n \geq k$. With optimal MDS codes, it is possible to retrieve the original k message packets (chunks in our case) from any k packets out of n. Codes construction could be systematic, in which the k first packet are the original packet, or non systematic, in which e the n output packets are totally encoded. To reduce the complexity at the decoding side (for the peers), we adopt the systematic for of the codes in P2PWeb protocol. Hence, the complexity will be reduced to zero when all the original chunks are available. Figure 3 summarizes the erasure coding into the P2PWeb architecture and the MDS property.

Reed-Solomon (RS) codes are the most common MDS codes. These codes are used in the famous distributed file system HDFS (Hadoop Distributed File System) within his RAID module. HDFS-RAID has been experimented today within a very popular social network [6], and it gave substantial benefits compared to the simple replication mechanism [7]. Recently, new MDS erasure coding algorithms are proposed based on discrete geometry and tomographic operators [8]. These algorithms help to get flexible redundant rate with linear complexity in coding and decoding, by making adequation between chunks and geometrical projection of data. An implementation of erasure codes in P2PWeb is done in [9]. However, the comparison between Reed-Solomon codes and those new codes are out of the scope of this paper.

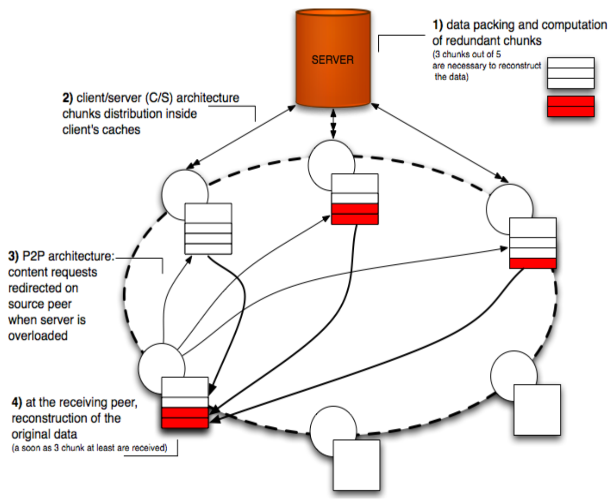
SERVER
1) data packing and computation of redundant chunks
(3 chunks out of 5 are necessary to reconstruct the data)
2) client/server (C/S) architecture chunks distribution inside client's caches
3) P2P architecture: content requests redirected on source peer when server is overloaded
4) at the receiving peer, reconstruction of the original data
(a soon as 3 chunk at least are received)

Figure 3. Redundant chunks production by MDS erasure coding within the P2PWeb hybrid architecture

III. Experimentation Results and Performance Evaluations

In this section we present some of the experimentation results that we have realized on P2PWeb architecture. We first present comparison results to show the advantage of using P2PWeb hybrid architecture against a traditional Client/Server transmission mode. Then, we study the performance of our hybrid proposal in different scenarios and in larger scales.

A. *Experimentation setup*

Our experimentations have been done in a virtualisation environment that contains a server and two clusters of 23 and 50 virtual machines respectively, with public IP-addresses for the different machines. Several content files with different sizes have been tested to validate our experimentations. We present the results obtained by using a JPG content file of 39 MB size. At this step of experimentation, the other file sizes did not present a significant difference.

B. *Experimental Results*

1) P2PWeb vs pure Client/Server architecture

The objective of the first set of experiments was to show the advantage of the proposed P2PWeb hybrid architecture against a traditional C/S communication architecture in terms of QoS parameters represented by the delay. For the two scenarios (C/S and P2PWeb), we ran a server with the same downloading rate of 2MB/s using the first cluster of virtual machines. In the former scenario, the 23 nodes will act as pure clients and will download the content from the server directly. Furthermore, in P2PWeb scenario the server will start to upload the content to the first 8 arrived nodes as pure clients. Starting from the 9th arrived node to the network, the server responses to user requests by sending a list of P2PWeb clients/peers that are already connected to the network and that accept to share the content with a maximum uploading rate of 300KB/s for each client/peer.

We can notice in Figure 4 how in C/S scenario, the more the number of connected clients is, the more the demanded time for downloading the content will be. On the hand, by using P2PWeb architecture, the first 8 nodes (clients) will take the same time to download the content as in C/S scenario. While, starting from the 9th node, the new arriving nodes (peers) will take considerably less time to download the content, since they will not be limited only to the server, but they will use the other clients and peers on the network as content providers.

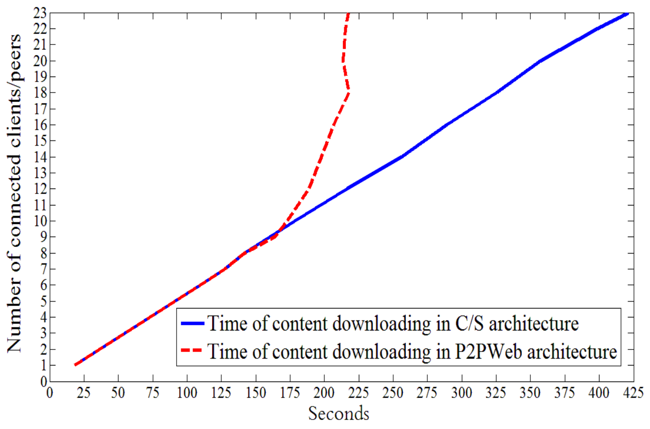

Figure 4. P2PWeb vs pure Client/Server architecutre

2) QoS P2PWeb performances

After illustrating the benefits of P2PWeb hybrid architecture, our goal in the second set of the experiments is to study the performance of the proposed architecture in a way that gives better results in terms of some QoS metrics which are the delay and the network throughput. Besides, we aimed at demonstrating the ability of applying our hybrid architecture in large scales. Hence, we extended the platform of 23 nodes that we have used in the first set of experimentations with another platform of 50 nodes as Figure 5 shows.

To reach the objective, and in order to give more realistic results, we made our measurements on a machine that is put behind a firewall like the most part of the connected terminals over the Internet [10]. Moreover, we limited the measurement time to the first 60 seconds. This minute corresponds in many P2P applications to the TTL (time to live) of chunk delivery [11,12]. Then, we tried to find the most suitable P2PWeb topology (clients vs. peers) for this precise period. We studied the impact of three important parameters variation on the hybrid P2PWeb architecture, which are:

- the number of P2PWeb clients that retrieve the content from the server directly;
- the number of peers that retrieve the content from P2PWeb clients/peers which are already present in the system and receiving that content;
- the uploading rate for the peers in the network.

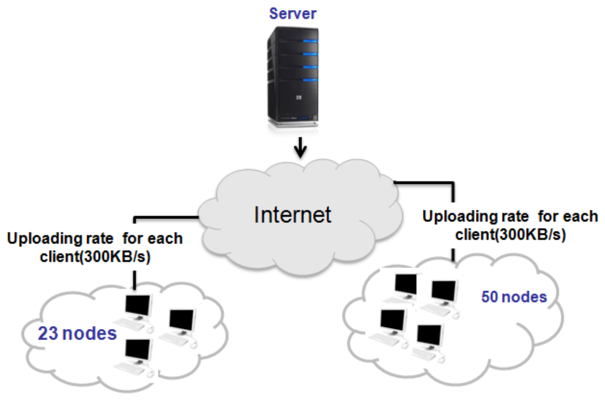
Server
Internet
Uploading rate for each client(300KB/s)
Uploading rate for each client(300KB/s)
23 nodes
50 nodes

Figure 5. P2PWeb platform based on virtual machines (two clusters of 23 and 50 machines)

Different scenarios have been tested. In the following, we detail some scenarios that illustrate well the benefit of deploying clients and peers together in the system in order to provide better QoS and potentially QoE.

a) P2PWeb: 10 clients & 63 peers with 300KB/s upload rate:

In this scenario, we assume that the server can serve 10 clients directly in a C/S mode. The 63 arriving nodes to the network -besides the measurement peer- will be then served as peers. In this case, the Manager will pass to each arriving node a set of 50 P2PWeb clients/peers that are connected to the network and that are concerned about the content in question. Figure 6 represents the number of P2PWeb clients and peers that are connecting to the measurement peer during the first 60 seconds of the content delivery process.

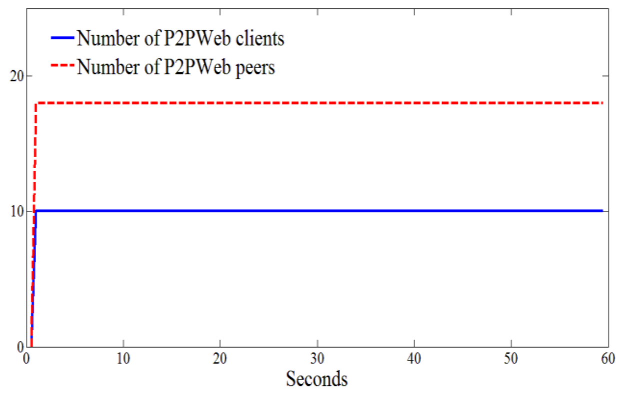

Figure 6. Number of connected P2PWeb Clients and Peers per second

On the other hand, Figure 7 shows the interaction between the measurement peer and these P2PWeb clients/peers. From this figure, we can see that the largest part of the downloaded content has been obtained by the clients, while a modest contribution has been considered from peers' side. This result was expected, since all the peers are connecting to the network at almost the same time. Thus, the peers will not have enough number of chunks to exchange with the new arriving peer as the P2PWeb clients do. On the other hand, by mapping Figure 7 to Figure 6, we can notice that although the measurement peer is connected to 10 P2PWeb clients and to 18 peers as soon as it arrives to the network, the interaction with these clients/peers will not start directly. Thus, a slight "startup" delay takes place before starting the content downloading process. The reason is that the new arrived peer has no content to share yet. Hence, at its arrival, nobody on the network will be interested to exchange content chunks with him.

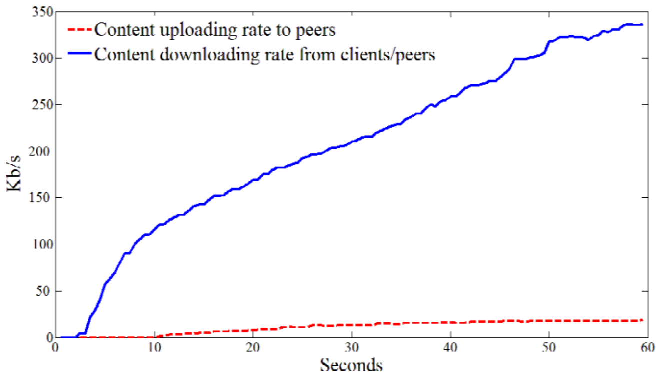

Figure 7. Uploading and Downloading rates over a peer

In its turn, Figure 8 demonstrates the percentage of the downloaded data by the measurement machine in 60 seconds. We can notice how with 10 P2PWeb clients, 63 peers and an uploading rate of 300KB/s for each client/peer; the measurement peer could only download 45% of the entire content in 60 second, which necessitated the re-tuning of the experimentation parameters in order to obtain better results.

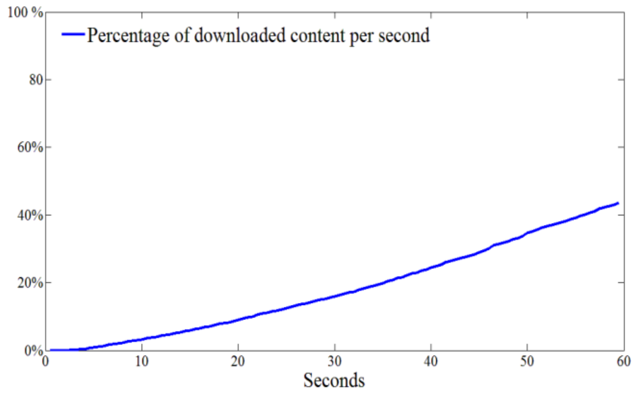

Figure 8. Percentage of downloaded data over the meauserment peer

We tested different scenarios with variant uploading rates and different number of clients and peers. Nonetheless, in order to be compatible with the main objective of the work, which is to improve the system performance in terms of QoS parameters, we made our choice about the second scenario that will be presented in this paper. In this scenario, we chose to keep an uploading data rate of 300KB/s for each P2PWeb client/peer that corresponds to a reasonable and realistic data rate over the internet nodes. Hence, we needed to tune the number of P2PWeb clients for obtaining better results.

b) P2PWeb: 25 clients and 48 peers with 300KB/s upload rate:

In this scenario, the server can serve 25 clients directly in a C/S mode before becoming saturated. The rest of the nodes (48 in our virtual platform besides the measurement peer) are then served as peers. Figure 9 represents the connecting P2PWeb clients and peers to the measurement peer at the first 60 second of the content delivery. In this figure we can notice how network condition fluctuation could affect the communication speeds between the network elements from time to time.

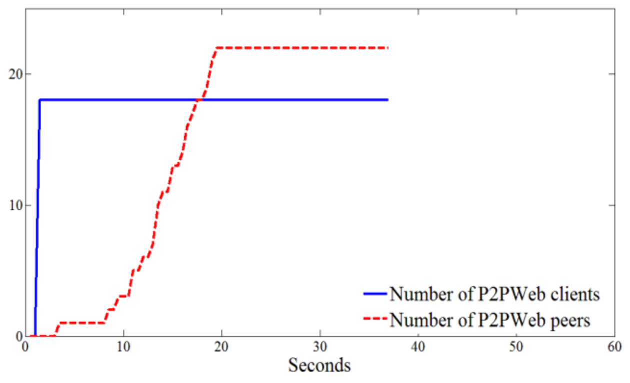

Figure 9. Number of connected P2PWeb Clients and Peers per second

In its turn, Figure 10 shows the downloading and the uploading rates of the measurement peer, regarding the already connected clients and peers to the network. In this scenario, the contribution of the peers in the downloading process will be less than it was in the previous scenario, since the number of P2PWeb clients that are used as content sources is bigger.

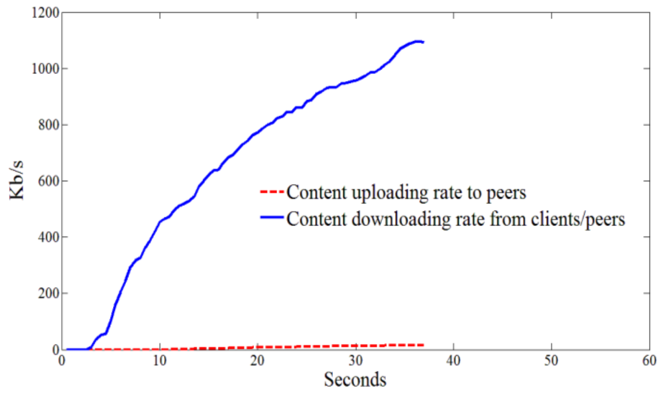

Figure 10. Uploading and Downloading rates over a peer

However, as we can see in Figure 11, with 25 P2PWeb clients, 48 peers (about the double number of clients) and an uploading rate of 300KB/s for each client/peer; the measurement peer could download the entire content in less than 40 seconds.

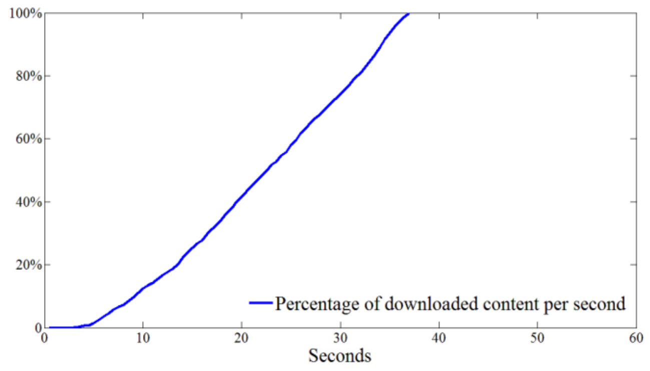

Figure 11. Percentage of downloaded data over the measurement peer

As we have noticed in the presented results, two issues need to be investigated, the startup delay of content downloading on the peers and the ratio of clients and peers participation for delivering the content to the new arriving peers. In-progress tests are currently done to cope with these issues, and to improve the performance of P2PWeb architecture. In these tests, even for the P2P part of the architecture, the server will not only play the role of an index server. Hence, besides passing the list of clients/peers that already have the content to the new arrived peer; the server will also pass a chunk of the requested content to the peer in question. This process will help to accelerate the contribution of this new peer in the system, which means less startup delay and more fairness in the participation ratio of both P2PWeb communities, the clients and the peers.

IV. User-SATISFACTION control

Since users' resources play an important role in the P2PWeb architecture, it will be necessary to satisfy users' preferences concerning the usage of their resources. In general, P2P-based applications do not take into account user preferences, other than QoS-related parameters, like the available bandwidth or the maximum number of connections. Motivated by these reasons, we thought of supporting our P2PWeb hybrid architecture with a user-satisfaction control technique that we call WUW (What Users Want). WUW main goal is to allow users to strategically impact the composition of their local neighborhoods, according to their own personal preferences. This service has been inspired from the centralized satisfaction-based load balancing approach that is presented in [13]. Authors of [13] proposed a centralized C/S mechanism that uses some generic notions (preferences, intentions, strategy) to help balancing clients and servers' whishes. Besides, they defined satisfaction and adequation measures to have a feedback about the mechanism itself. In our contribution, we keep the definitions of preferences, intentions and strategy. On the other hand, we provide a new definition of the feedback measures, in which we completely change the way of obtaining and computing them. Moreover, the WUW service has an entirely new design that considers the decentralized P2P part of P2PWeb hybrid architecture and that deals with peers and not only with producers-consumers.

A. *WUW overview*

WUW takes into account users' preferences during their participation in the system. Preferences are users' personal choices. More formally, a preference $p \in P$ is represented by a couple p = < label, value >, e.g. <type of content, movies>, <interest, high>, <location, Europe>, etc. A strategy $s \in S$ is a function $s: P \rightarrow \mathbb{R}$ that maps a set of preferences into real numbers, called intentions. The concept of strategy defines the basic way for a user to determine which neighbors he considers interesting to trade with. WUW computes the intentions of the local user at each P2PWeb client/peer and uses the values of these intentions to score the user neighbors and to build a local ranking. Only the best ranked neighbors will then be kept in the peer's local neighborhood. In order to assess the way with which the applied *strategies* satisfy users' expectation and limit the extent to which bad strategies may affect performance, WUW gives a feedback to the users. Users are thus able to evaluate the ongoing behavior of the system with respect to their preferences and modify their strategies to meet their goals.

B. *WUW iterative algorithm*

WUW runs on each P2PWeb client/peer between the browser plugin and the Manager. The main functions of WUW are: (i) to evaluate the job of the content delivery application in use with respect to the preferences expressed by the local user; (ii) to rank the set of P2PWeb clients/peers that is associated to the content and that has been passed by the Manager and then, to select a subset of this set consisting in the peers whose users most satisfy the local user's expressed preferences.

These actions are repeatedly performed at regular intervals, considering the most up-to-date locally available information. At each interval, each user builds his own ranking locally. To make meaningful choices, users must have some information about the other P2PWeb clients/peers "aims" and how they are behaving (recent uploads/downloads). Obviously, sharing preferences and strategies may reveal sensitive information about the users. The intentions are real numbers that quantify how much a user is willing to share a given content with other users at a given time. They reveal nothing about the reasons behind the number, which are most probably the sensitive information that may create privacy concerns. Intentions are thus the right piece of information to be shared without affecting users' privacy.

WUW disseminates intentions and updates about recent upload/downloads of the P2PWeb clients/peers via an epidemic protocol. From a long time, epidemic protocols are known to be efficient and robust ways to spread information in decentralized networks [14]. Each instance of WUW, running on each peer, participates in the dissemination of information coming from all the other peers, according to the epidemic paradigm. Each user is thus able to know the intentions of his neighbors towards him. By combining his own intentions and the neighbors' intentions in the ranking phase, the resulting ranks turn out to be an informed choice.

C. *WUW feedback measures*

As we have mentioned above, WUW provides a way for the user to understand how his preferences and the applied strategy affect the local performances and fit in the rest of the P2PWeb network. WUW uses what we call "*items*" as a measurement unit to compute and update its feedback information. In P2PWeb, the chunks that have been already generated at the content preparation phase will be mapped into items in order to be used by WUW. WUW provides two measures: satisfaction and adequation. Both are related to each user and locally calculated on each P2PWeb client/peer, considering the dual nature of these elements as downloaders and uploaders.

The satisfaction "as a downloader" (resp. "as an uploader") measures to which extent the application prefers highly ranked users over all those who are in the current neighborhood, to download (resp. upload) a given content. Intuitively, a user is more satisfied if he downloads from or uploads to users who have the best scores, according to the local ranking provided by the "strategy and preferences setting".

The adequation "as a downloader" (resp. "as an uploader") measures to which extent the content being downloaded (resp. uploaded) is provided (resp. requested) by highly ranked users, over all those who are in the current neighborhood. Intuitively, a user perceives the system as more adequate if the content he is currently sharing is making his having more exchanges with the best scored users.

Both measures (for both the "downloader" and the "uploader" perspectives) are expressed by real numbers, whose value can vary between 0 and 1, with 1 denoting the best possible choices are made. Consistently low values during the execution may denote that the user has chosen a very constraining set of preferences or an ineffective strategy to evaluate his neighbors.

D. *WUW satisfaction experimentations*

WUW tries to optimize the local neighborhood at each peer, in order to improve its compliance with the local user preferences. By selecting only subsets of the available remote peers, according to criteria (user's preferences) which the P2P application in use knows nothing about, some QoS-related performance problem may arise as a side-effect. A proper service parameterization must therefore be found, in order to maximize user's satisfaction and adequation while minimizing the impact on the provided QoS.

We have tested the WUW service in order to verify the amount of overhead imposed to the rest of the application and the effectiveness of the local ranking in improving user satisfaction. Preliminary results show a negligible impact on the global content sharing performance; we can say that the global content sharing performance gets worse by less than 1%.

Ongoing experimentations are also dedicated to test the effectiveness of the local ranking in improving user satisfaction. For that, in a first test we will observe the measures of satisfaction and adequation without WUW influencing the P2PWeb BitTorrent-like protocol. In a second test, the list of peers sent to this protocol will be modified by WUW. WUW will send only the best set of peers from its satisfaction and adequation point of view.

The full integration of WUW in the rest of the P2PWeb architecture will immediately follow the successful completion of the ongoing tests.

V. CONCLUSION

Our main objective in this paper was to provide an efficient reliable and scalable content delivery service. We clarified how neither a C/S-based, nor a P2P-based architecture can reach this target alone, since each of the two architectures has its own advantages and limitations. To cope with this issue, we proposed a new hybrid architecture that is called P2PWeb, between the centralized client/server and the non-centralized P2P architectures for content delivery over the Internet. This architecture helps to reduce the load over the server in order to provide a better service for the end-users. To reach the goal, we based on an open source of BitTorrent protocol to implement a P2PWeb network protocol. By our experimentation results, we prove the advantage of our proposed architecture against the traditional C/S communication mode. Besides, we studied the different network configurations in term of client's number threshold and maximum uploading rate in order to select the best one for our contribution. Finally, we supported our system with a user-satisfaction control technique that helps to improve the provided service of the hybrid P2PWeb architecture. As mentioned in Section III, we are working on enhancing P2PWeb performance by passing a content chunk to the new arriving nodes. Besides, we are currently studying the feasibility of the proposed architecture for different applications, such as the real-time streaming of scalable video content by using the Scalable Video Coding (SVC). Moreover, more development steps are being investigated on the WUW technique to improve its performance and its compatibility and utility in P2PWeb architecture.

ACKNOWLEDGMENT

The work presented in this paper has been supported by P2PWeb project with support from OSEO and the Bretagne and Pays de la Loire regions. The P2PWeb project is labeled by the "Images et Réseaux" cluster. Adriana Perez-Espinosa acknowledges the CONACyT Scholarship Program of the Mexican Government.

REFERENCES

[1] D. Xu, S. Kulkarni, C. Rosenberg, and H. Chai, "A CDN-P2P hybrid architecture for cost-effective streaming media distribution," Computer Networks, vol. 44, pp. 383-399, 2004.

[2] Y-C Tu, J. Sun, M. Hefeeda and P. Sunil, "An analytical study of peer-to-peer media streaming systems", ACM Trans. Multimedia Comput. Commun. Appl, vol. 1, no. 4, pp. 354-376, 2005.

[3] A. Bakker, R. Petrocco, J. Gerber, V.r Grishchenko, D. Rabaioli and J. Pouwelse, “Online Video Using BitTorrent and HTML5 Applied to Wikipedia”, Int. Conf. on Peer-to-Peer Computing, pp.1-2, 2010.

[4] R. Mattson, “Enhancing HTTP to improve page and object retrieval time with congested networks”, Ph.D. thesis, La Trobe University, Melbourne, Australia, 2008.

[5] M. Ghareeb, A. Ksentini and C. Viho, "A multipath Video Streaming Approach for SNR Scalable Video Coding (SVC) in Overlay Networks," IEEE CCNC: Consumer Communications and Networking Conference, pp. 605-610, Las Vegas, NV, USA, 2011.

[6] M. Sathiamoorthy, M. Asteris, D. Papailiopoulous, A. Dimakis, R. Vadali, S. Chen, D. Borthakur, “XORing Elephants: Novel Codes for Cloud Storage”, to appear.

[7] H. Weatherspoon and J. Kubiatowicz, “Erasure coding vs. replication: A quantitative comparison,” in Revised Papers from the First International Workshop on Peer-to-Peer Systems, IPTPS’01, pp. 328-338, London, UK, Springer Verlag, 2002.

[8] N. Normand, I. Svalbe, B. Parrein, and A. Kingston, “Erasure Coding with the Finite Radon Transform,” in Wireless Communications & Networking Conference, pp 1-6, Sydney, Australia, April, 2010.

[9] Benoît Parrein, Nicolas Normand, Majd Ghareeb, Giulio d’Ippolito, Federica Battisti, Finite Radon Coding for Content Delivery Over Hybrid Client/Server and P2P architecture, 5th International Symposium on Communications, Control and Signal Processin, pp 1-4, Roma, Italy, May 2012.

[10] L. D’Acunto, J. Pouwelse, and H. Sips, “A measurement of NAT & firewall characteristics in peer to peer systems,” in Proceedings of 15th ASCI Conference, pp. 1-5, June 2009.

[11] L. Golab, K. G. Bijay, and M. T. Ozsu. • On concurrency control in sliding window queries over data streams. In Proc. Int. Conf. on Extending Database Technology (EDBT), pp. 608-626, 2006.

[12] B. Cheng, H. Jin, and X. Liao. “RINDY: A Ring Based Overlay Network for Peer-to-Peer On-demand Streaming”. In 3th IEEE Conference on Ubiquitous Intelligence and Computing ,pp. 1048-1058, Wuhan, China, September 2006.

[13] J.A. Quiane-Ruiz, P. Lamarre and P. Valduriez, “A self-adaptable query allocation framework for distributed information systems,” The VLDB Journal, vol. 18, no. 3, pp. 649–674, June 2009.

[14] B. Pittel, “On spreading a rumor,” SIAM Journal on Applied Mathematics, vol. 47, no. 1, pp. 213-223, February 1987.