Towards improving user satisfaction in decentralized P2P networks

Marco Biazzini, Patricia Serrano-Alvarado, Raziel Carvajal-Gomez

Towards improving user satisfaction in decentralized P2P networks

Marco Biazzini
INRIA - Bretagne Atlantique
Marco.Biazzini@inria.fr

Patricia Serrano-Alvarado
Université de Nantes - LINA Lab
Patricia.Serrano-Alvarado@univ-nantes.fr

Raziel Carvajal-Gomez
Université de Nantes - LINA Lab
Raziel.Carvajal-Gomez@univ-nantes.fr

***Abstract*—Peer-to-Peer (P2P) architectures are more and more used in recent content distribution platforms because of their valuable characteristics as scalability, performance, and negligible maintenance and distribution costs. In general, P2P applications allow users to provide preferences that are mainly related to performance, like number of connections and bandwidth limits. As user resources are the wealth of P2P systems, we think it is important to satisfy user preferences in a more meaningful and personalized way. Users should be able to define the kind and quality of peers they prefer to exchange with. In this work, we present What Users Want (WUW), a framework to measure and improve the satisfaction of the users based on personal preferences that reflect their expectations from the P2P system. We then present the design of a distributed P2P service that implements our framework. Experimental results, obtained with a prototype running on top of BitTorrent, show improvement of user satisfaction and the possibility to minimize the impact on the overall performance of the content distribution.**

***Keywords*- User satisfaction, user preferences, P2P unstructured overlays, P2P content distribution, P2P distributed computation.**

I. Introduction and motivation

Peer-to-Peer (P2P) architectures have recently been coupled with different architectural paradigms because of valuable characteristics as scalability and performance, that come with almost negligible maintenance costs. They are increasingly used in Content Delivery Networks (CDN) [1] [2], because traditional client-server architectures generate high maintenance costs, whereas P2P systems are easily set up and can offload a relevant amount of traffic. Besides, P2P architectures are more performing than client-server ones to distribute highly requested content in a short period of time.

Large content aggregators as Wikipedia are experimenting some P2P-assisted form of content distribution, in order to reduce the load of servers when large contents are requested in a bursty fashion [3]. Recently, some research on coupling P2P and cloud computing paradigms has shown that a profitable cooperation is possible and convenient for both service providers and end-users [4]–[6]. Telecommunication companies provide their customers with set-top boxes which include "seedboxes" using P2P protocols[1] [2] [3].

On the one hand, we think that this trend will grow and lead to interesting research challenges, as earlier works have already shown [7]. On the other hand, while in P2P systems users' resources are the wealth of the system, little attention has been given to improve user satisfaction beyond the standpoint of system performance and reliability. In general, P2P applications allow users to configure parameters related to quality of service (QoS), *e.g.*, the maximum bandwidth allowed. The recent trend of personalization of online services according to users' tastes and decisions, also driven by the huge momentum of the social networking phenomenon, clearly demands something more.

A growing number of users regularly participates in online P2P-based services/communities using a single personal digital identity [8]. Such P2P systems are characterized by relatively long uptime and reasonably large resource availability. While users may share information, opinions and collaborate at will in these digital communities, they do not have the possibility to tune the behavior of their P2P application in a way that reflects their personal attitudes. This fact may raise concerns about their convenience as participants. In these P2P architectures, improving users' *satisfaction* is essential, because if users leave, the amount of system resources decreases.

We envision a scenario in which digital communities or distribution platforms, that exploit unstructured P2P overlay networks, aim at giving users the possibility to truly personalize the way their resources are shared in the community. With users consent, such platforms may profile each authenticated user, by collecting information directly from users, or by retrieving it from social networks. Each user may choose the part of information in her *profile* that can be publicly disclosed. The ensemble of these public information may then be organized by the platform to obtain configurable *settings*, which users can exploit to personalize the way they share resources.

As an example, let us consider a content distribution platform which builds user profiles containing the following information: geographic location, favorite music genre and a reputation score provided by an online community. Additionally, the platform is able to know with whom a given user has already exchanged content in previous sessions. Corresponding settings that users can configure could be: choice of the location of remote users, priority for unknown users, choice of affinity with remote users with respect to music tastes, choice of minimal reputation scores of remote users. A user could then choose to avoid some users that have participated in a previous sharing, to lower the risk to be tracked; she may prefer users who like music that she likes, people with

[1] http://www.p2pon.com/2012/12/17/worlds-first-bittorrent-certified-set-top-box-sees-the-light/
[2] http://boingboing.net/2013/01/14/bittorrent-set-top-box.html
[3] http://www.free.fr/adsl/pages/multimedia/nas.html

high reputation, living in a country where human rights are respected or where digital data protection is well regulated, *etc.*

A platform that gives the possibility to configure such settings allows users to personalize the way their resources are used and whom their contents get shared with.

We do not address the way a platform collects and organizes user information. Even assuming that user profiles are available and settings based on public information can be provided to users, several challenges must still be tackled.

- *Decentralized computation of neighbor matching*. A centralized computation of the best matching neighbors for each user may be impractical for several reasons: it may present an excessive computational load; it is a typical bottleneck from the standpoint of both networking performance and system robustness; it may arise concerns of the user about their privacy: even if a user agrees on disclosing some information about herself, this does not imply that she is happy to let other users (or even the hosting platform) know which kind of users she prefers.
- *Processing of heterogeneous public profiles*. Different users will agree on disclosing different subsets of the overall set of data collected in their profiles. *E.g.*, a user may agree on disclosing her location, but not her reputation score. Heterogeneous profiles must be evaluated at each peer in a globally coherent and comparable way.
- *Computation of user satisfaction*. QoS-related metrics cannot quantify the satisfaction of a user with respect to her personal choices, expressed in her local configuration of settings. For instance, the fact that a content is shared with a satisfactory (or unsatisfactory) bandwidth usage, may not be related to the fact that the content is shared mostly among users that highly (or poorly) match the settings chosen by the local user. Thus, metrics for user satisfaction are needed, that take into account personal choices expressed in the system.

In this work, we propose **WUW** (What Users Want), a general framework to address the aforementioned challenges. Thanks to **WUW**, at each peer a user decides how to evaluate heterogeneous user profiles via a fully decentralized computation. Then **WUW** modifies the composition of local neighborhoods in a way that accounts for the configuration of settings chosen by each user, without disclosing these configuration to others. Moreover, **WUW** provides a way to measure, at each peer, user satisfaction and impact on the system, as derived by user choices.

We then present the architectural design of a P2P service that implements our framework. We detail its modularity and describe the functions provided by all its components.

Additionally, we show results of an initial evaluation of **WUW**, obtained by deploying a prototype implementation on a distributed grid platform.

The rest of the paper is organized as follows. Section II discusses related work. Section III describes the concepts of **WUW** and their peculiarities. Section IV details the design of the decentralized service implementing **WUW**. Section V shows our preliminary experimental results. Conclusion and future work are discussed in Section VI.

II. Related Works

P2P applications for content sharing are many and present their own particularities, but all of them show the same lack of attention towards user personal preferences. As a representative example, P2P applications as GoalBit [9], uTorrent[4] and vidTorrent[5] only allow users to decide: the amount of upload/download bandwidth (KBs), the maximum number of concurrent connections, the maximum number of connections per content, and the port number used for incoming connections. In general, these applications only choose peers based on their QoS-related settings. None of them allows a user to influence her local neighborhood according to settings other than QoS-related ones.

Friend-to-friend networks, as proposed by OneSwarm [10] are recently emerging as a way to build privacy preserving P2P network of trusted users. These works target scenarios that differ from ours. Nonetheless, an interesting research trend, somehow closer in spirit, is exploring the possibility to exploit social network overlay topologies for content distribution [11] [12], achieving promising results.

Other works focus on obtaining social-based P2P systems. They compare users preferences and cluster peers that have similar preferences. In Tribler [13], the similarity is based on the most recently downloaded content. Another work [14] proposes a graph-based model that computes users' descriptor vectors to calculate similarities. Lin *et al.* [15] measure the cosine similarity of users' preferences, in order to enhance the distributed querying for multimedia content. Following the assumption that people with similar preferences are likely to have content matching the shared preferences, a semantic social-based overlay is constructed by exploiting information about other peers' tastes. By querying the content in this overlay, the authors show that it is possible to improve the number of successfully located content per query. All these works consider a predefined set of preferences for all peers and aim at improving the overall content sharing performance. In **WUW**, each user is allowed to choose her personal set of preferences.

A recent work by Lin and Chou [16] proposes a preference-aware content dissemination protocol for mobile social networks. The main contribution of this paper is the definition and the evaluation of a decentralized algorithm that estimates the global utility for the peer (as defined by local preferences) in choosing neighbors to exchange with. With this approach, authors successfully tackle the issue of the opportunistic and discontinuous connectivity that characterizes these networks. This work undertakes a problem different than ours and in a different use case scenario, but as we do, it considers individual preferences as decision drivers for content sharing and does not assume their uniform definition across the system, therefore relying on a decentralized local computation at each peer. Nevertheless, the satisfaction of users is not addressed.

Works that are interested in measuring users' satisfaction generally focus on QoS-related measures. In [17], a model to quantify VoIP user satisfaction based on an analysis of the call

[4]http://www.utorrent.com

[5]http://web.media.mit.edu/~vyzo/vidtorrent

duration from Skype traces is proposed. Such model quantifies the influence of QoS parameters like bit rate, jitter and packet loss on call duration. Qiu and Cui [18] propose a quantitative model to study user satisfaction in a client-server online video streaming. Their approach is based on the ratio of the duration of a session to the length of the video. But they also take into account subjective factors like user inclination and content popularity. These factors are included as a variable in their cosine-based formalism, but not measured in the evaluation tests. These works are oriented to particular applications and the user has no flexibility on defining personal strategies as it is done in **WUW**.

The Satisfaction-based Query Load Balancing framework (SQLB) has been introduced as a generic framework to measure satisfaction of participants in the context of "query allocation" [19]. In this framework, each provider (respectively consumer) computes her intention to treat a query (respectively, to obtain her query be treated by some specific provider). It is up to each participant to take into account any information she considers relevant as general objectives, local context (capacity, actual load, task load, ...), and history (past jobs, satisfaction, ...). This information is kept private and only the intention is disclosed. A centralized component (mediator) collects participants' intentions and takes an informed decision about who will treat which query. Such decision may please some users and displease others. Participants are assumed to be able to understand they cannot be satisfied every time, thus a single episode of dissatisfaction can be tolerated, while it is important to avoid dissatisfaction to repeatedly occur on the long term.

The framework we propose is profoundly inspired by SQLB. Consistently with our purposes, the main goal of this framework is to measure but also to increase user satisfaction, and motivate users to stay longer in the system. SQLB proposes a model to define several fundamental notions, that in **WUW** have been deeply rethought and redefined, in order to fit a radically different context.

While SQLB was designed to be implemented as a centralized mechanism, **WUW** focus on fully decentralized P2P networks where no central component is given, or, even if it is, it cannot reasonably bear the burden of a combinatorial computation that should be repeatedly performed considering each peer with respect to all the others for every shared content. Thus, in our framework, all the necessary information for the users to impact the current distribution overlay of their peers is computed locally at each peer, as it is its evaluation in terms of user satisfaction. This implies that our framework operates to locally improve the situation at each peer, while forcefully lacking a global knowledge of the P2P overlay. How far **WUW** actions are from a globally optimal situation will be object of our future research.

A second important difference between SQLB and **WUW** is related to the very nature of their actors. While SQLB's main actors are producers and consumers, typical connotations of a client-server system design, **WUW** operates with peers. In its very nature, every peer is simultaneously a client and a server, with respect to the other peers sharing a given content. Thus, when considering users interactions, **WUW** takes into account both faces of the peer nature.

III. WUW FRAMEWORK

Our framework has two main objectives: (i) it takes into account user profiles and settings in order to shape the distribution overlays accordingly and (ii) it provides a quantified feedback to users, who are thus able to understand how their choices are taken into account and how they affect the performance. Section III-A gives details about the first point, while Section III-B formally describes the notions used in the second point.

A. Strategy Definition

Let us recall the scenario described in Section I. Each user of a P2P content sharing platform has a public profile. She has also the possibility to configure personal settings, provided by the platform and based on public user information.

We call *preferences* of a user a chosen configuration of their personal settings. Our framework does not define what preferences should be and thus impose no constraint on them. The actual definition of sets of preferences is something directly related to the community the users are part of and must thereof come from the platform that uses **WUW**.

WUW provides the way to tune local neighborhoods towards those users whose profile better matches local user preferences. A *strategy* evaluates, at each peer, the neighbors of the local user, using available information as public profiles, private preferences (of the local user), recent history of content exchanges, *etc*. In order to precisely define the strategy, we first present the key notion of *intention*.

Intentions are single numbers that quantify in a compact way the attitude of users towards each others for any shared content. Considering the preferences of the local user and public profiles of other users, **WUW** allows to compute at each peer the intention of the local user to share a given content with every remote user. An intention is defined as a real number whose value ranges from -1 to 1. The value 1 denotes that the remote user perfectly matches local user's expectations. The less the local user wish to exchange with the remote user, the smaller the value of her intention. By "translating" private user preferences into intentions, it is possible to compare the attitudes of the users towards each other by only considering their respective intentions, notwithstanding the fact that these intentions may result from possibly quite different sets of preferences. User preferences are private and never disclosed to other users. The platform that provides settings for the users to configure does not need to know how each user chooses to configure them. Intentions are instead public information shared among users. In the following we explain how they are used at each peer to evaluate the attitude of the local user towards remote users for any shared content.

As Fig. 1 shows, a strategy is composed of three distinct procedures:

- *Intention Computation* — The way personal preferences are translated into intentions strongly depends on the kind of preferences and thus different algorithms can be devised. **WUW** framework requires that for each remote

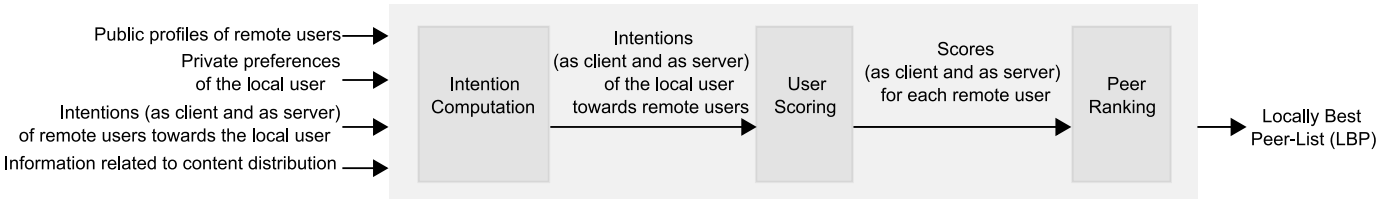

Figure 1: WUW STRATEGY.

user, with respect to each shared content, two intentions are computed at each peer: one considering the local user as client (downloader), the other considering the local user as server (uploader).
Heterogeneous user profiles are handled, in the intention computation procedure, by defining appropriate actions whenever a profile does not contain information considered by a local user preference.

- *User Scoring* — Intentions are locally used at each peer to assign scores to every remote user. Remote users are separately scored as clients and as servers. Each score takes into account intentions of both local and remote user. Thus, the score a local user assigns to a remote user as client, depends (i) on the intention of the remote user as client towards the local user as server and (ii) on the intention of the local user as server towards the remote user as client. Whereas the score a local user assigns to a remote user as server, depends (i) on the intention of the remote user as server towards the local user as client and (ii) on the intention of the local user as client towards the remote user as server.
- *Peer Ranking* — Once these two scores have been assigned to each remote user for every content, it is possible to assign to remote peers a position in a ranking. The K best ranked peers sharing a given content will compose the *Locally Best Peer-list* (LBP) for that content. The $LBPs$ for all contents to be shared are given to the P2P application to "tune" local neighborhoods.

We formally define these key concepts as follows. The subindex c stands for "of the peer as client", whereas the subindex s stands for "of the peer as server". To simplify we consider only one content.

- Let $P_{c|s}$ denote a set of personal preferences defined by a user. Let U be the set of public profiles of users.
- An intention computation function $f_{c|s} \in F$ is a function that maps a set of preferences and a user public profile into intentions :
  $f_{c|s} : \wp(P_{c|s}) \times \mathrm{U} \rightarrow \mathbb{R}_{[-1..1]}$.
- A scoring function $k_{c|s} \in K$ is a function that takes couples of intentions associated with a user and returns a score:
  $k_{c|s} : \mathbb{R}_{[-1..1]} \times \mathbb{R}_{[-1..1]} \rightarrow \mathbb{R}$.
- A ranking function $r \in R$ is a function that takes couples of scores and returns a position in a ranking:
  $r : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{N}+$.
- A strategy $t \in T$ is a tuple $\langle f_c, f_s, k_c, k_s, r \rangle$.

A strategy is thus defined by a set of procedures that (i) computes intentions of the local user, as client and as server, towards any remote user, (ii) assigns scores to remote users, as clients and as servers, according to the intentions of both sides, and (iii) ranks the peers of the remote users according to these scores, to decide which peers are to be preferred to exchange a given content. An example of a simple strategy we implemented in our experiments is detailed in Section V-A.

Different strategies may be used at each peer, according to local user decisions, to better reflect personal goals and expectations. Through an appropriate user interface, it is possible to conceive procedures to let users pick, in a set of strategies, the one they prefer and even change it at runtime. Users are not informed about which strategy is used by others, nor they are informed about the preferences of other users. Only public profiles and intentions are shared.

Once the peer ranking has been computed on each peer, the P2P application is feed with a LBP for each shared content, rather than with original peer lists provided by the P2P overlay manager (*e.g.*, a tracker, a DHT, *etc.*). The number of neighbors in the partial views of each peer is then probably reduced and the randomization of the local neighborhoods affected. More peers—that are better according to the preferences expressed by the user and to her strategy—will be part of the local neighborhoods. One of our research hypothesis is that, in the context of stable online communities, this operation may be not detrimental for the system QoS. No change of the internal algorithms of the P2P application is attempted in order to enforce the use of the best ranked peers by the P2P exchange protocol.

It is worth noticing that the WUW framework first ranks all peers sharing any content in a single ranking and then build the various $LBPs$ associated to each content. This makes it possible to devise strategies that globally shape the attitudes of users across different sharing tasks. In a context where users are identifiable and keep a stable identity over multiple sessions, possibly maintaining historical records, this feature opens interesting ways to evaluate users across sessions.

B. Feedback computation

It is important to understand to which extent user preferences and her strategy are effective, *i.e.*, they are making the local user share content with the users she likes the most. WUW provides a feedback quantified in the *satisfaction*, *adequation*, and *system evaluation* measures. They are inspired by the SQLB framework, but redefined to fit our radically different environment and computed in a fully decentralized fashion.

Locally computed measures are needed to evaluate the actions of WUW on the local neighborhood, because our

framework does not force the P2P application to use only the best ranked peers. **WUW** modifies the neighborhoods by proposing good peers and discarding others, but then the P2P distribution protocol remains in charge of deciding the frequency of exchanges among peers. **WUW** cannot predict which users will be able to provide most of a content. An aggressive intervention on the local neighborhoods (like limiting too much the size of the $LBPs$, or failing to account for the current performance in the strategy) would almost surely be detrimental to the QoS. Thus, users need to know about the actual results of their actions. They may then decide to change some of their preferences or part of their strategy.

The feedback measures are computed locally at each peer and are related to the local user. Their formal definition is based upon some assumptions. It is assumed that any content can be logically split in a set of non overlapping *items*. Items are the units of measure used to compute the feedback measures. The precise definition of item must be devised in compliance with the P2P application used for content distribution. Items are anyway not meant to be atomic: they can be exchanged partially or completely. The computation of feedback measures needs information about the status of the current content distribution. It is assumed that some information can be obtained locally from the P2P application being used. It should be possible to know at least the items completely or partially downloaded from (or uploaded to) a given peer and the items requested to given peers but not retrieved.

Considering a content C as a set of items $i_1, \ldots, i_n$, we define the feedback measures related to this content with the help of the notation in Table I. Since feedback measures are periodically recomputed by each peer, in this table we use the symbol ‡ to imply the constraint: "since the last time the measure was computed". In the following, the expression "download (resp. upload) events" means the download (resp. upload) of an item or part of it.

The *Satisfaction* S_c of a (local) user as client (*i.e.*, as a "downloader") is computed as follows. For each item $i \in C$ whose download the local user has completed, let $S_c[i]$ be the sum of the local user's intentions towards the users who have provided her item i, multiplied by the number of successful download events related to i, divided by the number of times i, or part of it, has been requested by the local user. That is :

$$S_c[i] = \frac{\sum_{u \in P_i}(((Ic_C^u + 1)/2) \cdot \mid D_i^u \mid)}{\mid LQ_i \mid} \tag{1}$$

Then S_c is the average computed by aggregating the values $S_c[i]$ of the latest items downloaded by the local user:

$$S_c = \frac{\sum_i S_c[i]}{\mid D \mid} \tag{2}$$

Intuitively, S_c measures to which extent the P2P application prefers "good users" over the others, to get a given content. Its value may vary between 0 and 1, with 1 denoting the best possible choices are always made.

The Satisfaction S_s of a (local) user as server (*i.e.*, as an "uploader"), is instead computed as follows. For each item $i \in C$ whose upload the local user has completed, let $S_s[i]$ be the sum of the local user's intentions towards the users who have downloaded item i from her, multiplied by the number of successful remote download events (upload events from the local user point of view) related to i. That is :

$$S_s[i] = \sum_{u \in RD_i}(((Is_C^u + 1)/2) \cdot \mid RD_i^u \mid) \tag{3}$$

Then S_s is the average computed by aggregating the values $S_s[i]$ of the latest items uploaded by the local user:

$$S_s = \frac{\sum_i S_s[i]}{\mid RD \mid} \tag{4}$$

Intuitively, S_s measures to which extent the P2P application prefers "good users" over the others, to distribute content. Its value may vary between 0 and 1, with 1 denoting the best possible choices are always made.

The *Adequation* A_c of a (local) user as client (*i.e.* as a "downloader") is computed as follows. For each item $i \in C$ for which a request has been issued by the local user, let $A_c[i]$ be the average of the local user's intentions towards all the users who currently have item i. That is :

$$A_c[i] = \frac{\sum_{u \in H_i}((Ic_C^u + 1)/2)}{\mid H_i \mid} \tag{5}$$

Then A_c is the average computed by aggregating the values $A_c[i]$ of the latest requests issued by the local user:

$$A_c = \frac{\sum_i A_c[i]}{\mid LQ \mid} \tag{6}$$

Intuitively, A_c measures to which extent the preferences of the local user and her strategy are assigning higher scores to useful peers over the others, to get a given content. Its value may vary between 0 and 1, with 1 denoting the best possible matchings are always made.

The Adequation A_s of a (local) user as server (*i.e.* as an "uploader"), is instead computed as follows. For each item

Table I: NOTATION USED TO DESCRIBE FEEDBACK MEASURES COMPUTATION ON EACH PEER.

Symbol	Meaning
H_i	Set of all remote users who currently have item i
P_i	The set of users who provided item i ‡
Ic_C^u	The local user's Intention "as client" toward the remote user u for content C ‡
Is_C^u	The local user's Intention "as server" toward the remote user u for content C ‡
D_i^u	Set of all download events from a remote user u related to item i ‡
D	Set of all download events ‡
LQ_i	Set of all the request events issued by the local user related to item i ‡
LQ	Set of the request event issued by the local user ‡
RD_i^u	Set of all the complete download events to a remote user u related to item i ‡
RD_i	Set of all the complete download events to remote users related to item i ‡
RD	Set of all the complete download events to remote users ‡
RQ_i^u	Set of all the request events issued by a remote user u related to item i ‡
RQ_i	Set of all the request events issued by remote users related to item i ‡
RQ	Set of all the request events issued by remote users ‡

$i \in C$ whose upload has been requested to the local user, let $A_s[i]$ be the sum of the local user's intentions towards the users who have requested item i, multiplied for the number of request events related to i. That is:

$$A_s[i] = \sum_{u \in RQ_i} (((Is_C^u + 1)/2) \cdot \mid RQ_i^u \mid) \quad (7)$$

Then A_s is the average computed by aggregating the values $A_s[i]$ of the latest requested items:

$$A_s = \frac{\sum_i A_s[i]}{\mid RQ \mid} \quad (8)$$

Intuitively, A_s measures to which extent the P2P application strives to distribute those contents that are more requested by the "good users". Its value may vary between 0 and 1, with 1 denoting the best possible matching are always made.

At any given time, the *System Evaluation* of a (local) user as client (respectively: server) is the ratio S_c/A_c (respectively: S_s/A_s). Intuitively, the System Evaluation measures how much the local user can be happy about the impact of her preferences and strategy on the ongoing content distribution, both as client and as server. Its value may vary in the interval $[0...\infty]$ (the higher, the better), with 1 denoting a neutral impact.

IV. WUW SERVICE DESIGN AND IMPLEMENTATION

WUW is devised as a generic, autonomous, and fully distributed service that runs on top of unstructured P2P systems. It is a modular application. Each of its modules implements different sets of functionalities and handles inputs and outputs through well defined interfaces.

We present an overview of the architecture of the service in Section IV-A and its modular components in Sections IV-B to IV-E.

A. WUW *design overview*

The architecture of **WUW** is shown in Fig. 2. **WUW** has been conceived to be independent from the P2P application used to distribute content. It is located on top of the P2P layer, so it is generic enough to be adapted to different unstructured P2P protocols.

The **WUW** service acts as a middleware between the local instance of the P2P application and the overlay management system of the P2P network. This means that the P2P application communicates with **WUW** to know about other peers in the overlay and **WUW** communicates with the overlay coordinator (*e.g.*, a tracker, a DHT protocol, or other, depending on the P2P system being used) to get information about the state of the overlay. To be able to affect the list of peers used by the local P2P application, **WUW** intercepts the communications between the local peer and the overlay coordinator and modifies the peer list by only including the peers in its current LBP, thus excluding the less interesting ones from the standpoint of the local user.

The typical sequence of actions of **WUW** can be summed up as follows:

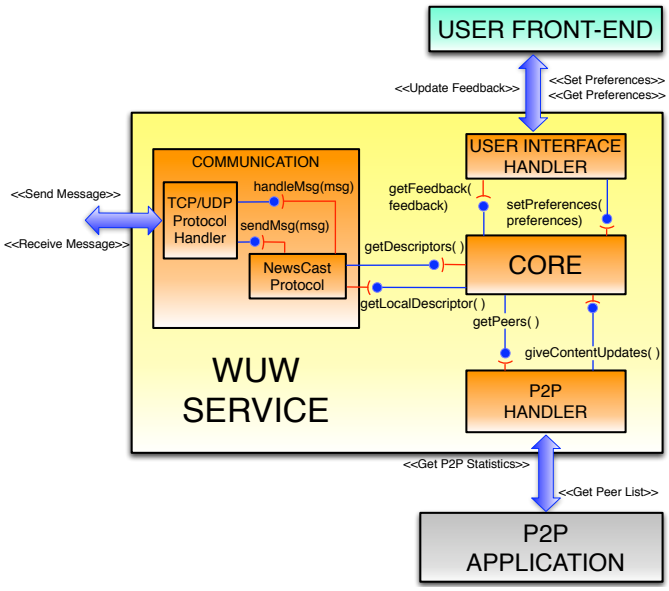

Figure 2: **WUW** architecture design.

- retrieve information about the content to be exchanged and the peers to exchange it with;
- get the user preferences about the given content;
- periodically exchange messages with the remote users to share user profiles, intentions and information about the state of the task;
- periodically retrieve data from the local P2P application instance about the state of the task;
- periodically compute/update the feedback measures by consuming, at each iteration, information coming from the local P2P application and the remote **WUW** instances;
- for each content, apply the local strategy to the current user preferences and update local user intentions towards its neighbors;
- score the neighbors according to the intentions associated to each of them;
- build a global ranking of all scored neighbors;
- for each content, build a LBP composed at most by the K best ranked peers that are sharing the content;
- send the new peer lists to the P2P application;
- update the feedback measures and make them available to the user.

B. *The User Interface Handler module*

The primary source of information for **WUW** is the local user. The User Interface Handler (UI Handler) module takes care of getting input from the user (her profile and preferences) and outputs the computed feedback measures. This module is independent from other's **WUW** modules, so it is possible to implement different user interfaces at will.

C. *The Communication module*

The Communication module implements basic functions to facilitate communications, over TCP or UDP, among **WUW** instances at different peers.

Moreover, an epidemic protocol is provided to disseminate among the peers: user intentions, user public profiles and additional data related to content dissemination, *e.g.* the number of content items at each peer. The epidemic dissemination of up-to-date information makes it possible for any local user to know what are the recently computed intentions of remote users towards her and what is the state of the diffusion of a given content.

The overlay used by the epidemic protocol is different from the content distribution overlay and it is locally maintained at each peer with negligible overhead. The delay between the local generation of an updated descriptor and its fruition by remote peers, modeled by the known properties of the epidemic protocols [20], can be modified by tuning the parameters of this module.

D. The P2P Handler module

This module is the only part of **WUW** that is aware of the specific P2P application being used by the local user. The interfaces provided by this module allow other modules to transparently exchange information with a specific distribution protocol.

The P2P Handler module contains the logic that defines content items and their parts in a way that is consistent with the P2P application.

It comprises all that concerns the specific way **WUW** get information from and gives information to the rest of the P2P system. This includes the communication with the P2P overlay manager, concerning the global list of peers sharing a given content. For each content being shared and for each neighbor in the overlay, the P2P Handler module retrieves data about any distinct download or upload event from/to a given neighbor, related to a given item.

Finally, through this module **WUW** sends to the P2P application the periodically renewed LBP for a given content.

Thus the main tasks of the P2P Handler module are:

- to retrieve from the P2P overlay management system the list of peers associated to a given content;
- to give to the P2P application the LBP for that content;
- to retrieve from the P2P application the information related to the content sharing activity
- to produce, from the retrieved information, data structures that are usable by the rest of the service to perform the proper computations.

E. The Core module

The main functionalities of **WUW** are placed in this module, which is the orchestrator of the service. User intentions and feedback measures are computed in a timely way. Every n seconds, a routine is started which performs the following steps:

1) Get updates about the remote users from the Communication module.
2) Get the latest information about the activity of the local P2P application from the P2P Handler module.
3) Get the latest changes in the users preferences from the UI Handler module.
4) Update the local state with the collected information.
5) Compute the feedback measures related to the latest local activity, considering the current intentions for all the neighbors and all the contents.
6) Make the feedback measures available to the local user via the UI Handler module.
7) Compute the intention values to be associated to every neighbor for every content, according to the local strategy.
8) Build a global ranking of all neighbors, considering the associated intentions for all the contents.
9) Create a LBP for each content currently shared by the local user.
10) Send the $LBPs$ to the P2P application, via the P2P Handler module.

By repeating this routine periodically, the local user is updated about the impact that her choices have on the local computation. While any QoS-related performance issue is usually reported by the P2P application being used, **WUW**'s feedback measures enable users to evaluate the overall "quality" of their neighbors as determined by user profiles and strategies and the average quality of the **WUW** service itself.

V. Preliminary experimental results

Our **WUW** prototype is implemented in Java and requires JVM 1.7 or higher. The source code is available under GPL license on Github[6].

The current implementation features a P2P Handler module that is able to interact with BitTorrent MainLine 3.1.9 (in the following: BT). We provide an instrumentation class in MainLine that periodically collects the required information from the working memory of the BT process and sends it to **WUW** via a local socket.

The Communication module hosts the NewsCast epidemic protocol [21] to maintain the overlay used to disseminate user profiles, intentions and content related information.

The UI Handler module is currently a facility to handle the service configuration via XML files.

In the following we present experimental results obtained by deploying our **WUW** prototype on the Grid5000 distributed platform [22]. The goal of these experiments is to start investigating the feasibility (impact of **WUW** on the time to complete the download of a content) and the effectiveness of our framework in improving user satisfaction. Being them the first exploratory experiments, the use case scenarios are quite simple, preference and strategy definitions are very basic and no runtime change of user preferences is considered. We do not consider churn and each instance of **WUW** eventually knows the descriptors of all the peers in the system via gossip communication.

A. Strategy implementation

We implement a simple strategy that considers public profiles represented by a single integer value in $[0..4]$. Intuitively,

[6]http://github.com/marbiaz/WhatUsersWant

our strategy (i) balances user preferences and overall performance in intention computation, (ii) balances local and remote user intentions while scoring users and (iii) balances scores of users as clients and as servers while composing the LBP ranking. We briefly give here some more details.

In the implemented strategy, for each remote user u and for each content traded with her by the local user o, let us call $I_c(u)$ the intention of u as client towards o as server and $I_s(u)$ the intention of u as server towards o as client. Conversely, let $I_c(o)$ be the intention of o as client towards u as server and $I_s(o)$ the intention of o as server towards u as client. We recall that intentions are in the range $[-1..1]$.

a) Intention computation: First, a real number $p \in [-0.5..0.5]$ is computed, which is inversely proportional to the difference between the preference of o and u.

Then, two real numbers $r_c, r_s \in [-0.5..0.5]$ are computed, which are inversely proportional to the number of failed exchanges between o and u in the recent past (*i.e.*, since the last time the intentions were computed). The first (r_c) accounts for the failure in downloading a given content; the second (r_s) accounts for the failures in uploading items of the same content to u.

Finally, the intentions of o towards u are computed as $I_c(u) = p + r_c$ and $I_s(u) = p + r_s$. In this way, higher intentions are assigned by u to remote users whose preference value is closest. Failed exchanges (if any) penalize remote peers by lowering the values of the associated intentions.

b) User scoring: The scores assigned to remote users are computed at each peer as the average of the intentions related to the users. The scores assigned by o to u for any given content are then:
$k_c = (I_c(o) + I_s(u))/2$ and $k_s = (I_s(o) + I_c(u))/2$.

c) Peer ranking: The final ranking of remote peers is computed by averaging the scores assigned to every user and sorting the list of users according to these averages.

B. Experimental settings

In all the experiments we consider a P2P network of 300 peers. Each peer runs an instance of the **WUW** service and an instrumented instance of BT. Only one target content is considered, a file of 1.1 GB. A standard BT tracker is also deployed as overlay manager. 30 peers are seeders (they have the full content since the beginning) and 270 peers are leechers. The download/upload bandwidth of each peer is set to $1024/512$ kbps. At each peer, **WUW**'s epidemic protocol sends information about the known peers to one randomly chosen peer every 3 seconds, while the feedback measures and the peer lists are updated every 8 seconds.

Our experiments consider two different scenarios:

- *Scenario 1* : standard BT execution. Each peer receives 40 random peers from the tracker every 100 seconds. **WUW** computes intentions and measures, but does not change the peerlist of BT. In this scenario, **WUW** is basically monitoring user satisfaction and adequation in a standard BT-based content sharing use case.
- *Scenario 2* : **WUW** in action. Every 100 seconds **WUW** receives 40 random peers from the tracker. The first time **WUW** gives to BT the 15 peers, among which 5 are the best according to its locally computed ranking, and the remaining 10 are chosen at random. All the other times, **WUW** gives to BT the best 15 peers in its updated ranking. Thus in this scenario each peer starts with 15 neighbors and receives 15 best ranked neighbors every 100 seconds.

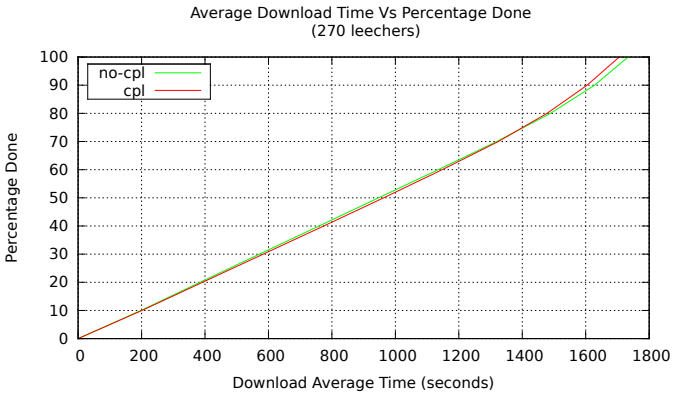

Figure 3: Average download time.

For both scenarios we measure: time to complete the download, average satisfaction and average adequation in the overlay. For each scenario 20 runs are executed. We then take the averages of the measured values.

C. Discussion of the results

Fig. 3 shows the average time taken by the peers in the overlay to complete the download of the target file. The parametrization of our strategy, described in Section V-A, makes it possible to obtain almost identical performance in the two scenarios. In general, during our heuristic quest for performance, we obtained for Scenario 2 performance from 1.2 to 2 times worse than for Scenario 1.

It is known that tinkering with local overlays usually worsen overall performance. Nonetheless, this result shows that it is actually possible to obtain good performance if there is the possibility to tune the overlay in a way that accounts for the relations among the number of peers, their average amount of bandwidth and the size of their local neighborhoods. For highly stable P2P overlay networks this does not sound completely unreasonable. We are currently investigating to get further insights towards strategy tunings which do not excessively penalize performance. As of now, the setting of our experimental scenario are too simplistic to support any claim.

A more interesting finding of our quest concerns a current limit of BT, for which dropping connection and creating new ones is extremely expensive. The strategy parametrization we finally devised minimizes the occurrence of these events, while ensuring that a local peerset is composed of highly satisfactory peers. In fact, due to BT default behavior, no new peer is allowed in the peerset once 40 neighbors are in, unless a neighbor becomes unreachable. Thus in our faultless and error-free settings, after roughly 400 seconds no new peer is added to BT neighborhoods anymore. By analyzing the logs, we found

that the deteriorations of performance we experimented while trying different parameterizations seemed to be mainly related to the overhead caused by the higher number of connections that were cut and opened each time the neighborhood in BT was modified. An improved connection handling or the switch to a connectionless protocol could open easier possibilities to our service.

Fig. 4 to 6 show aggregate values (minimum, 0.25 percentile, median, 0.75 percentile and maximum) of **WUW** feedback measures in the overlay at different time snapshots. The two experimental scenarios are thus compared with respect to the same measure at the corresponding time.

The boxplots in Fig. 4 clearly shows the positive impact of **WUW** on user satisfaction as client. All relevant aggregate values are better in Scenario 2 than in Scenario 1. The flooring effect that is visible at $y = 0.5$ is an artifact due to the definition of our strategy. In faultless and basically contention-free environments, the minimum satisfaction level has a lower bound in 0.5, as a consequence of how the intentions are defined. Thus, considering the narrow range of possible values, we see that **WUW** is quite effective in globally improving the satisfaction of the users as downloaders.

Fig. 5 demonstrates a similar effectiveness with respect to the satisfaction of users as servers. In this case, though, the values measured in Scenario 2 are lower at the beginning. From the standpoint of uploaders, a very small overlay, not (yet) optimized by **WUW**, is of limited satisfaction. The values rapidly improve when the BT instances receive more and better peers (after 100 and 200 seconds). The relation between the initial size of the local neighborhood provided by **WUW** and its impact on BT performance at each peer are objects of ongoing investigation.

While the adequation as server (not shown) globally follows a pattern that is very similar to the one illustrated so far, a different situation is depicted by Fig. 6. Adequation as client does not improve in Scenario 2 and actually presents slightly worse aggregate values. We see here that the adequation reflects the extreme ease of the downloading task, determined by our settings and resulting in a faultless and over-provisioned network of peers. If resources are not an issue, peers are equally adequate in the two scenarios, with respect to the content they are downloading from one another. To put it differently: if a content (or at least the part of it that is currently requested) is readily available from a large number of peers in the overlay, the profiles of the users that have it are, on average, not significantly better for any particular user. In our setting, at a given time the same items are available from users whose position in the local ranking at each peer is quite different. Thus the adequation as client of each user, measured on the intentions towards those who can provide these items, cannot improve that much by tuning local neighborhoods.

To conclude the discussion of our experimental results, we underline that, even in a simplified and optimistic setting as the one we examined, we can show how **WUW** allows interesting insights on the dynamics of the P2P content sharing network, from the novel standpoint of personal user satisfaction.

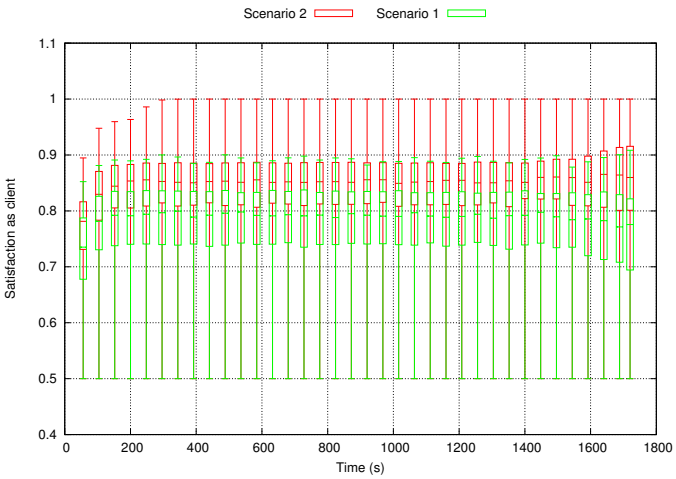

Figure 4: Satisfaction as client.

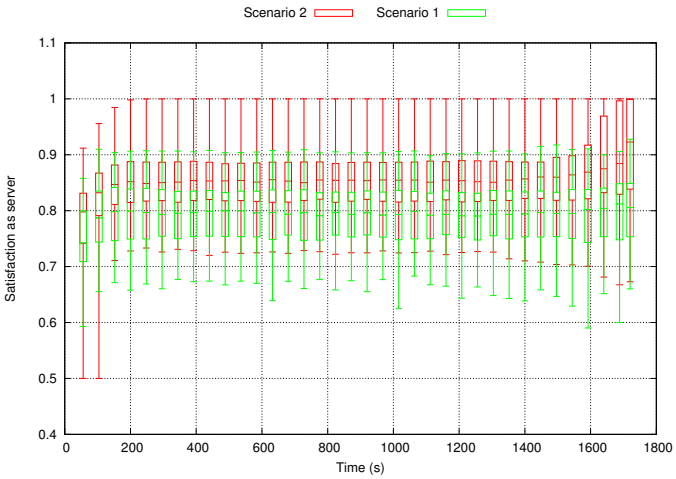

Figure 5: Satisfaction as server.

VI. Conclusion and Future Work

This paper focuses on the feasibility and the challenges of hybrid P2P systems that let personal preferences of the users influence the way their resources are used.

We presented **WUW** (What Users Want), a framework defining concepts to achieve, in a fully decentralized way, the following goals:

- to take into account user preferences at each peer in order to shape distribution overlays accordingly;
- to provide a quantified feedback to users, which are then able to evaluate the impact of their choices on their satisfaction and adequation to the system.

We detailed the procedures through which **WUW** makes it possible to evaluate users, characterized by heterogeneous profiles, with respect to personal, undisclosed preferences.

We described the design of a P2P service, which implements our framework and is oriented to unstructured P2P overlays. Finally, we presented and discussed experimental results, obtained with a prototype implementation of the **WUW** service, deployed on a distributed grid platform. They picture our first steps in the exploration of the various issues raised by the relations between performance and overlay tuning.

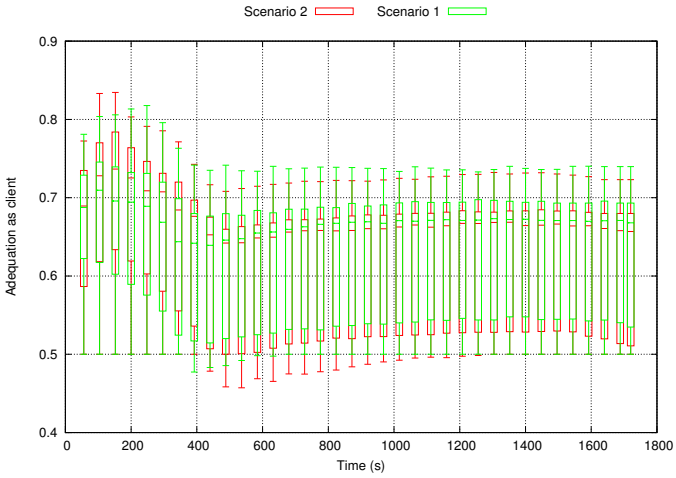

Figure 6: Adequation as client.

Our future work will evolve along three main directions. The first is related to extending the support to different P2P applications. The second concerns the experimental study of ways to tune local neighborhoods towards greater user satisfaction, while affecting performance in a way that can be estimated or controlled. The third is the characterization of different interesting use case scenarios, *e.g.* scenarios (a) in which satisfaction and adequation enlighten differently the state of user contribution (low satisfaction, high adequation), (b) where users dynamically change their preferences in order to ameliorate their satisfaction and adequation during the distribution of a given content, (c) with multiple contents and where the intersection of the $LBPs$ is not empty, (d) with different strategies per user, (e) with large number of peers and occurrence of churn.

ACKNOWLEDGEMENTS

This work was partially funded by the Region Pays de la Loire. We thank our colleagues of the P2PWeb project for our useful discussions all along this project. We sincerely thanks Philippe Lamarre for his support and very interesting discussions at the beginning of this work.
Experiments presented in this paper were carried out using the Grid'5000 experimental testbed, being developed under the INRIA ALADDIN development action with support from CNRS, RENATER and several Universities as well as other funding bodies (see https://www.grid5000.fr).

REFERENCES

[1] D. Xu, S. S. Kulkarni, C. Rosenberg, and H.-K. Chai, "Analysis of a CDN-P2P Hybrid Architecture for Cost-Effective Streaming Media Distribution," *Multimedia Systems*, vol. 11, no. 4, pp. 383–399, 2006. [Online]. Available: http://dx.doi.org/10.1007/s00530-006-0015-3

[2] M. Eberhard, A. Kumar, S. Mignanti, R. Petrocco, and M. Uitto, "A Framework for Distributing Scalable Content over Peer-to-Peer Networks," *Journal On Advances in Internet Technology*, vol. 4, no. 1,2, 2011.

[3] A. Bakker, R. Petrocco, M. Dale, J. Gerber, V. Grishchenko, D. Rabaioli, and J. A. Pouwelse, "Online Video Using BitTorrent and HTML5 Applied to Wikipedia," in *Peer-to-Peer Computing (P2P)*, 2010, pp. 1–2. [Online]. Available: http://dx.doi.org/10.1109/P2P.2010.5569984

[4] F. Marozzo, D. Talia, and P. Trunfio, "P2P-MapReduce: Parallel Data Processing in Dynamic Cloud Environments," *Journal of Computer and System Sciences*, vol. 78, no. 5, pp. 1382–1402, 2012. [Online]. Available: http://dx.doi.org/10.1016/j.jcss.2011.12.021

[5] A. Montresor and L. Abeni, "Cloudy Weather for P2P, With a Chance of Gossip," in *Peer-to-Peer Computing (P2P)*, 2011, pp. 250–259. [Online]. Available: http://dx.doi.org/10.1109/P2P.2011.6038743

[6] I. Trajkovska, J. S. Rodriguez, and A. M. Velasco, "A Novel P2P and Cloud Computing Hybrid Architecture for Multimedia Streaming with QoS Cost Functions," in *ACM Multimedia*, 2010, pp. 1227–1230. [Online]. Available: http://doi.acm.org/10.1145/1873951.1874193

[7] V. Janardhan and H. Schulzrinne, "Peer Assisted VoD for Set-top Box Based IP Network," in *Workshop on Peer-to-peer streaming and IP-TV*, ser. P2P-TV '07, 2007, pp. 335–339. [Online]. Available: http://dx.doi.org/10.1145/1326320.1326327

[8] M. Meulpolder, L. DAcunto, M. Capota, M. Wojciechowski, J. A. Pouwelse, D. H. Epema, and H. J. Sips, "Public and Private Bittorrent Communities: A Measurement Study," in *Conference on Peer-to-Peer Systems (IPTPS)*, 2010, pp. 10–10.

[9] M. E. Bertinat, D. De Vera, D. Padula, F. R. Amoza, P. Rodríguez-Bocca, P. Romero, and G. Rubino, "GoalBit: The First Free and Open Source Peer-to-Peer Streaming Network," in *Latin American Networking Conference (LANC)*, September 2009, pp. 49–59. [Online]. Available: http://doi.acm.org/10.1145/1636682.1636691

[10] T. Isdal, M. Piatek, A. Krishnamurthy, and T. Anderson, "Privacy-preserving P2P Data Sharing with OneSwarm," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 111–122, August 2010. [Online]. Available: http://dl.acm.org/citation.cfm?id=2043164.1851198

[11] M. Fonville, "Confidential Peer-to-Peer File-Sharing Using Social-Network Sites," 2010. [Online]. Available: http://referaat.cs.utwente.nl/conference/13/paper

[12] W. Galuba, K. Aberer, Z. Despotovic, and W. Kellerer, "Leveraging Social Networks for Increased BitTorrent Robustness," in *Consumer Communications and Networking Conference (CCNC)*. IEEE, 2010, pp. 1–5.

[13] J. A. Pouwelse, P. Garbacki, J. Wang, A. Bakker, J. Yang, A. Iosup, D. H. J. Epema, M. J. T. Reinders, M. van Steen, and H. J. Sips, "TRIBLER: a Social-based Peer-to-Peer System," *Concurrency and Computation: Practice and Experience*, vol. 20, no. 2, pp. 127–138, 2008. [Online]. Available: http://dx.doi.org/10.1002/cpe.1189

[14] P. Androutsos, D. Androutsos, and A. Venetsanopoulos, "Small World Distributed Access of Multimedia Data: An Indexing System That Mimics Social Acquaintance Networks," *IEEE Signal Processing Magazine*, vol. 23, no. 2, pp. 142 –153, march 2006.

[15] C.-J. Lin, Y.-T. Chang, S.-C. Tsai, and C.-F. Chou, "Distributed Social-based Overlay Adaptation for Unstructured P2P Networks," in *IEEE Global Internet Symposium*, 2007, pp. 1–6. [Online]. Available: http://dx.doi.org/10.1109/GI.2007.4301422

[16] K.-J. Lin, C.-W. Chen, and C.-F. Chou, "Preference-Aware Content Dissemination in Opportunistic Mobile Social Networks," in *INFOCOM*, 2012, pp. 1960–1968.

[17] K.-T. Chen, C.-Y. Huang, P. Huang, and C.-L. Lei, "Quantifying Skype User Satisfaction," in *SIGCOMM*, 2006, pp. 399–410. [Online]. Available: http://doi.acm.org/10.1145/1159913.1159959

[18] F. Qiu and Y. Cui, "A Quantitative Study of User Satisfaction in Online Video Streaming," in *IEEE Consumer Communications and Networking Conference (CCNC)*, January 2011, pp. 410 –414. [Online]. Available: http://dx.doi.org/10.1109/CCNC.2011.5766502

[19] J.-A. Quiané-Ruiz, P. Lamarre, and P. Valduriez, "A Self-Adaptable Query Allocation Framework for Distributed Information Systems," *Very Large Databases Journal (VLDB)*, vol. 18, no. 3, pp. 649–674, June 2009. [Online]. Available: http://dx.doi.org/10.1007/s00778-008-0114-1

[20] B. Pittel, "On Spreading a Rumor," *SIAM Journal on Applied Mathematics*, vol. 47, no. 1, pp. 213–223, February 1987. [Online]. Available: http://dx.doi.org/10.1137/0147013

[21] M. Jelasity, S. Voulgaris, R. Guerraoui, A.-M. Kermarrec, and M. van Steen, "Gossip-based Peer Sampling," *ACM Transactions on Computer Systems (TOCS)*, vol. 25, no. 3, August 2007. [Online]. Available: http://doi.acm.org/10.1145/1275517.1275520

[22] R. Bolze, F. Cappello, E. Caron, M. Daydé, F. Desprez, E. Jeannot, Y. Jégou, S. Lanteri, J. Leduc, N. Melab, G. Mornet, R. Namyst, P. Primet, B. Quetier, O. Richard, E.-G. Talbi, and I. Touche, "Grid'5000: A Large Scale And Highly Reconfigurable Experimental Grid Testbed," *Journal on High Performance Computing Applications*, vol. 20, no. 4, pp. 481–494, November 2006. [Online]. Available: http://dx.doi.org/10.1177/1094342006070078