Software Agents for Distributed Social Networking

Enrico Franchi, Michele Tomaiuolo
Dipartimento di Ingegneria dell'Informazione
Università di Parma
Parma, Italy
{efranchi, tomamic}@ce.unipr.it

Abstract—**Especially in the case of completely distributed or federated social networking platforms, multi-agent systems can play an important role. In particular, multi-agent systems have been used as (i) an underlying layer or a middleware for developing social networking platforms, (ii) a technology to increase the autonomous and intelligent behaviour of existing systems and (iii) a tool to develop simulation environments for studying both online and offline human social networks. In this paper we propose the integration of multi-agent technology into Blogracy, a novel peer-to-peer, anonymous and uncensurable social networking platform. The resulting system augments the platform with locality and proximity groups, making it fit for pervasive computing scenarios, exploiting the adaptivity, proactivity and negotiation ability of multi-agent systems.**

Distributed Micro-blogging, Peer-to-peer Computing, Social Networking, Multi-Agent Systems

I. Introduction

Users of popular social networking sites are becoming increasingly wary of the privacy issues they face. Often, the perceived problems are related to the possible leakage of personal information to extraneous persons, or to workmates. However, privacy threats can also come from the service providers which, being mostly centralized systems, maintain full access and control over published data.

From the technical point of view, scaling centralized systems to tens or hundreds of million of users is a hard challenge, which can be faced if enough resources are deployed. Most companies rely on mining users' data for supporting targeted advertisement. This behavior poses serious threats to privacy and data protection issues. Quite consequently, social networking sites guide their users into "walled gardens", without giving users full control over their own information because such information constitutes much of their company value [1]. Moreover, service providers are in the position to effectively perform a-priori or a-posteriori censorship, or to disclose all the information they have, no matter how private, to other entities. They can perform such actions either motivated by selfish interests or forced under legal terms and other forms of pressure.

On the other hand, P2P systems essentially achieve automatic resource scalability, in the sense that the availability of resources is proportional to the number of users. This property is especially desirable for media sharing social networking systems, considering the exceptionally high amount of resources needed. Moreover, regarding censorship issues, a P2P system essentially solves them by design. Without a central entity, nobody is in the position of censoring any data nor may be held legally responsible for the diffusion of censurable data: the sole owners and responsible of the data are the users themselves.

Especially in the case of completely distributed or federated social networking platforms, multi-agent systems can play an important role. In fact, one of the very specific features of multi-agent systems is the sociality of agents, i.e. their ability to communicate in a semantic way and develop trust relationships among them. Moreover, agents can express their communication acts by means of acknowledged standards, like FIPA, for interoperability among diverse systems, and exchange messages directly, in a peer-to-peer way. So, it is not surprising that these two technologies are often applied together for developing advanced social platforms. In particular, multi-agent systems have been used as (*i*) an underlying layer or a middleware for developing social networking platforms, (*ii*) a technology to increase the autonomous and intelligent behaviour of existing systems and (*iii*) a tool to develop simulation environments for studying both online and offline human social networks.

For the first type of solution, many of the distinguishing features of multi-agent systems can be fully exploited. In fact, multi-agent systems provide semantic communication among agents, which is handy for expressing all the different actions that users can perform on a social platform. The different performatives of messages can be understood according to their pragmatics meaning, and applied according to existing trust relations among the users and their respective agents. Also, complex negotiation protocols can help creating acknowledgements and trust among users, in an automatic or assisted way, without exposing sensitive data. Mobility can also be useful for moving the computation closer to data, if massive analysis has to be performed, but can also be handy for adding functionality to a node of a distributed social platform or to a user's client application.

In the second case, agents are mainly exploited because of their proactive and reactive behaviours, for providing recommendations of both users and content and for providing personalization of results. Reactive abilities fit particularly well into a social networking environment, where events happen continuously and users can be easily distracted by the huge information overflow which is associate with richly interconnected social networks. Sensing the environment and executing automatic tasks can reduce this overload significantly. Goal-oriented behaviours, on the other hand, can support users in persecuting their long term objectives about friend and content discovery, i.e. finding known persons registered in the network, making new acquaintances with users with common interests, finding interesting content from new sources or hidden among other less relevant data.

Finally, multi-agent systems are a powerful tool for simulating the behaviour of online social networks, in the same way they have been used for simulating the behaviour of persons in real social environments for a long time. In fact, multi-agent systems have proved to be very effective in the simulation of social networks, both during their initial creation and development and during their further operation. They allow to describe the behaviors of individuals, mimicking the actions of human users in similar contexts, and to analyze the associated emerging behaviour of the network as a whole. This way, multi-agent systems can provide precious insights for further improvement of existing social platforms.

II. RELATED WORK

Various solutions are being proposed to overcome the centralized architecture of the most widespread social networking platforms. Many of these proposals follow a federated approach, allowing users registered on a certain server to create relationships with users of other servers. Others are full-fledged peer-to-peer systems, usually based on a distributed hash table (DHT).

Federated social networking systems allow users registered on a certain server to create relationships with users of other servers. The best known examples are **Diaspora**[1] and **StatusNet**[2]. Diaspora servers communicate by means of an ad-hoc federation protocol and the standard Salmon protocol[3] for comments. StatusNet (formerly known as Laconica) adheres to the OStatus standard protocol for the interconnection of various servers and uses a number of existing protocols for interoperability with other networks.

Various social networking systems are being developed on the basis of peer-to-peer communications and DHT indexing. Among these, **PeerSoN** [2][3] is a prototype designed to provide encryption, decentralization and direct data exchange in the field of social networks. A DHT is used to trace the user's network presence and for obtaining the index of the user's recent content. **LotusNet** [4] is a model of a social network to be built over Likir. Likir itself is a secured DHT, which requires a user to be authenticated according to an IBE (Identity-Based Encryption) scheme, before participating in the network. **Safebook** [5] is based on a DHT and a network of socially close peers, defined Matryoshka. Peers in a user's Matryoshka are trusted and support the user by anonymizing communications and replicating content and profile information. **Persona** [6], though not being a distributed social network, uses an interesting Attribute-Based Encryption protocol for protecting access to users' content. It allows each user to assign credentials to various groups of “friends”, for accessing protected content.

Not many existing social networking platforms are based entirely on multi-agent systems. Among the research works, MAgNet [7] is a multi-agent system built using JADE [8] and FOAF [9]. It is a prototype application providing social oriented services to mobile users, i.e. defining groups of users and arranging group events. In [10], authors discuss the advantages of using multi-agent technologies for building social platforms. They also underline some existing issues, mainly in terms of overlay infrastructure, navigability of the social network, existence of specific ontologies.

A larger number of systems exploit multi-agent technology for augmenting existing social platforms. For example, in [11], an agent-based photo searching and recommender system for flickr.com is proposed. In [12], authors propose an approach for finding an expert in a social network. A user's profile is not supposed to be completely available, and instead is learned by an agent, by evaluating exchanged messages and the user's referrals. In [13], authors present a model of a recommender system based on social networks, autonomous agents and trust relationships. The aim is to both reach information not available in close nodes and filter information to be processed. The system is analyzed with varying network density, preference heterogeneity and knowledge sparsenesss. In [14], the problem of automatic trust negotiation is contextualized to multi-agent systems. Agents are used to negotiate and build trust among users, disclosing data and privacy policies incrementally and reciprocally. This is especially useful for connecting users in a social network, disclosing only the minimal possible set of profile attributes.

Another widespread application of multi-agent systems in the field of social networks is simulation. In [15], for example, a model of social network based on the notion of “circles” is simulated over a multi-agent system. In [16], there is an example of a simulation platform specifically designed for studying social networks. More in general, Ascape, NetLogo, MASON, Repast and Swarm are among the best known platforms for agent simulation, often used to study emerging behaviours and features of both online and offline social networks.

III. Resilient microblogging

While many authors argue for the distribution and openness of social networking and micro-blogging services, few usable implementations exist, either in the field of federated networks or as fully distributed solutions. Considering the existing or proposed solutions, we therefore present a new system, which we named Blogracy[4]. Essentially, it is an anonymous and uncensurable microblogging platform, built incrementally over BitTorrent, a popular and resilient file-sharing service.

The architecture of the application is modular and is build around two basic components: (*i*) an underlying module for basic file sharing and DHT operations, possibly exploiting an existing implementation, and (*ii*) an OpenSocial container, i.e., a module providing the services of the social platform to the local user, to be accessed through a web interface. Additionally, the system supports autonomous agents for providing (*i*) recommendations of both users and content, (*ii*) personalization of results, (*iii*) trust negotiation mechanisms. In the following

1 http://joindiaspora.com/

2 http://status.net/

3 http://www.salmon-protocol.org/

4 http://www.blogracy.net/

paragraphs we will describe the most distinguishing features realized in Blogracy over this extensible architecture.

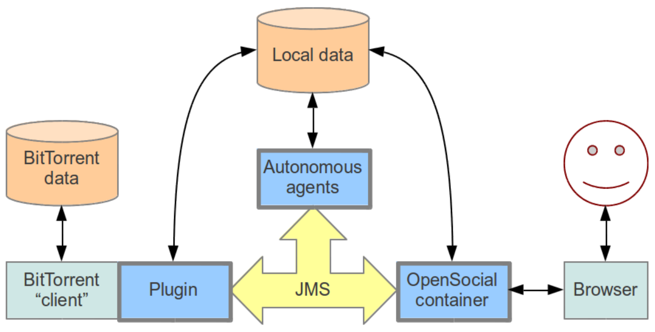

Figure 1. Blogracy architecture

For its basic operation, Blogracy exploits a peer-to-peer file-sharing mechanism and two logically separated DHTs. Users in Blogracy have a profile and a semantically meaningful activity stream, which contains their actions in the system (e.g., add a post, tag a picture, comment a video). One DHT maps the user's identifier with his activity stream, which also contains a reference to the user's profile and references to user generated content (e.g., posts, comments). These references are keys of the second DHT, which are then resolved to the actual files. The files are delivered using the underlying peer-to-peer file-sharing mechanism.

Among the features of public online information systems, and in particular in the case of micro-blogging and social networking applications, anonymity or pseudonymity are often a requirement. But, even under anonymity or pseudonymity, users' content need to be verified for authenticity and integrity. Blogracy uses a key-based identity scheme [17], where a user's public key is used directly to represent the user. This way, all content produced by the user can be easily verified against his public key, which is also his own main identifier. Moreover, for assuring anonymity at the lower network level, various anonymizing technologies exist, varying from simple proxies to complex mix-net schemes, and can be integrated into the platform.

For publishing confidential information, accessible only to a restricted circle of contacts, Blogracy supports attribute-based encryption. Similarly to Persona, Blogracy privacy model uses attribute credentials for protecting access to sensible content, creating a sort of very flexible "circles", i.e., parametrized roles to be assigned to users for granting a certain set of access rights. The encryption scheme is based on the CP-ABE protocol (Cyphertext-Policy Attribute-Based Encryption) [18].

Once users can be distinguished by their ID, i.e., the hash of their public key, it is also possible to associate additional information with them, including personal profile and personal activity stream. The activities of a user are represented as a flow, which friends and followers are interested into and want to subscribe to. In Blogracy, personal activities are included into a standard ActivityStream[5] feed, which is eventually signed to avoid tampering. Activity Streams is an open format specification for the syndication of activities taken in social web applications and services. In Blogracy, the personal feed is eventually signed to avoid tampering and then shared using the underlying file-sharing platform.

5 http://activitystrea.ms/

Clearly, an application that does not provide explicit representation for the user's profile and contacts should not be considered a social networking application. Essentially, in Blogracy users define and manage a list of other users, represented by their IDs. A user is not required to publish his profile, nor the network of his social relations. However, if he does, the profile, containing partial or full information, can be retrieved as any other shared file and its magnet-uri can be also reported in the user's feed. In case privacy needs to be added, cryptography shall be used. At the current stage, for exporting profiles and contacts, Blogracy adopts Portable Contacts[6] with OpenSocial[7] extensions, a format which has some benefits from the interoperability point of view, being quite simple and well supported by existing large social networks and mail systems. It also allows to associate tags with each user, thus matching the basic data structure managed by Blogracy.

One of the technical issues of a peer-to-peer microblogging application is data availability; in fact, popular content will quickly gain lots of seeds, while posts published by peripheral users, with few contacts and sparse online presence, will instead suffer poor availability to the extent that it is possible that the publisher remains the only seed for his own new posts. In some systems focused on distributed data storage, like Freenet, the problem is addressed through multiple replication of all published resources. However, in modern peer-to-peer networks, the hostile behavior of some nodes has to be taken for granted; pollution and other kinds of attacks cannot be underestimated. What we foster, instead, is a replication system based on acquaintances. Essentially, an introducing user is responsible to introduce the invited as smoothly as possible. This kind of mechanisms is thoroughly analyzed in [19][20][21], with special regards to (*i*) content replication in peer-to-peer storage and (*ii*) the problem of peers with low availability in completely decentralized systems. In fact, using some kind of fallback strategies for sharing non popular resources may improve the system performance regarding data availability.

Another important issue is interoperability with other existing online social networking and micro-blogging platforms. In principle, since Blogracy handles users' feeds in the form of Activity Streams, it can also manage similar feeds obtained in other ways, seamlessly integrating content from web blogs and from the peer-to-peer network. Interoperability with more traditional news-feeds, web-based micro-blogging posts, and content distributed over the peer-to-peer network is thus guaranteed, provided that the stream semantics is correct. On the other hand, resources distributed through Blogracy can be easily replicated over the web. Since the actual system architecture has a web interface, for user operation, it is relatively simple to host a Blogracy instance on a remote node and configure it for public access, acting as a gateway for Blogracy public content.

Finally, apart from requesting updated feeds at startup, followers should be timely notified that one of their followees updated some resource. Traditionally the strategies are: (*i*) pull,

6 http://portablecontacts.net/
7 http://opensocial.org/

i.e., the observer periodically checks the observed resource for updates or (*ii*) push, i.e., the update is automatically announced to the observer. Apart from relying on the DHT, Blogracy benefits from the peer-to-peer messaging facility provided by the file-sharing protocol. In fact, for their basic operation, file-sharing systems need to keep track of the peers that are currently seeding or downloading a certain file (sometimes collectively defined as a "swarm"). So, advertising about a new feed is simply a matter of contacting the peers that are sharing the superseded version of the user's feed.

As described in the previous sections, Blogracy relies on the BitTorrent protocol for basic file-sharing, and uses a DHT mechanism for indexing the users' feeds. Specifically, we implemented the system exploiting Vuze, a popular BitTorrent client (formerly known as Azureus) implemented in Java and available as open source software. In particular, the specific DHT of Vuze (known as DDB) has a set of generic primitive queries that fit our purposes better than the Mainline DHT of other BitTorrent applications. Moreover, the Vuze platform has a modular architecture, where functionality can be added with plug-ins. The main application exposes to the plug-ins only a restricted interface, which is nonetheless sufficient for our purposes.

IV. Intelligent, pervasive social networking

The Blogracy system itself relies only on users' nodes for its operation. Thus users need to perform background tasks on their own, in a distributed way. On the basis of the experience gained developing AOIS [22], we are integrating a layer of autonomous agents into the system, for assisting the user in finding new interesting content and connections and for pushing the local user's activities to followers.

In particular, a personal assistant (PA) monitors the local user's actions in the platform and learns the user's profile, beyond information provided explicitly. The PA receives the user's queries, forwards them to the available information finders (IF) and presents the results to the user. Moreover, a PA provides the local user with recommendations about possibly interesting content and connections available in the network. Another task performed by the PA is the personalization of results. In fact, as a social network becomes larger and more richly interconnected, users unavoidably face some form of information overflow. A personal agent, on the basis of a user's profile, can arrange presented data in a way to give evidence to the most interesting bits.

An Information Finder (IF) is an agent that searches information on the repository contained into the node where it lives, on the basis of an automatic TF-IDF indexing and explicit hashtags associated with local posts. It provides this information both to its user and to other trusted users. An IF receives users' queries, finds appropriate results and filters them on the basis of its user's access policies.

An information pusher (IP) is an agent that monitors the changes in the local repository and pushes the new information to the PA of interested subscribers who are currently connected. The IP can forward content produced both by the local user or by remote acquaintances to other contacts, according to privacy preserving policies and to recent queries made by other users.

Over the Blogracy OpenSocial container, we are also integrating some functionalities for pervasive online social networking, specifically for realizing locality and proximity groups. For this purpose, each node of the social network will hosts multiple agents, with different levels of agency. Some of the more important agents are (*i*) the Neighborhood Manager agent (NM), which cooperates with lower level agents to discover the users in its neighborhood; (ii) the Trust Negotiator agent (TN), that is involved in the decisions regarding privacy and data access and (iii) the OpenSocial agent, that provides a bridge towards the underlying Blogracy modules.

A user may own multiple nodes (e.g., an instance on the smart-phone and an instance on his home computer) and since the actual location of the user is important for our application, the nodes in the different device negotiate which should be considered active (i.e., which one determines the user location): (*i*) the nodes determine which is the device that registered an explicit user action or (*ii*) they ask the user to select the device he is currently using.

Apart from the personal circles defined by each user, we also have two additional kinds of groups: (*i*) Proximity groups and (*ii*) Location groups. Proximity groups are centered on each member of the social networking system and represents physical closeness to such member. Proximity groups are extremely fluid, in the sense that users can physically move and consequently the set of users belonging to a Proximity group varies in time. Each user configures the sticky-ness of his Proximity group, i.e., how long the other users are considered part of it after they are no longer physically close to him. Although a Proximity group may be entirely public, for privacy reasons it is safer to consider only Proximity groups that are subset of other groups (or to the set union of all groups, i.e., only "friends" are part of a Proximity group). The Neighborhood Manager agent informs the OpenSocial agent when users enter and leave the Proximity group and the latter notifies the OpenSocial container about it.

On the other hand, a Location group (i) is associated with the users in the proximity of a given location (e.g., a classroom or a museum room), (ii) has a host, i.e., a node that both identifies and supports the group and (iii) is associated with a location profile, which can be either hosted on the central server or on the device itself. In fact, a location, although logically different from a regular user, works in the same way and a Location group is essentially a Proximity group for the location.

A generic Trust Negotiation protocol may be needed since users joining a proximity or location group are not necessarily connected a priori in the social network, and they may need to acknowledge their profile attributes before practical social interaction. Such a negotiation requires the controlled exchange of credentials and policies, without disclosing unnecessary sensible information, yet establishing trust if possible. In [14] we already presented a generic library supporting zero-knowledge proof for attribute verification, which facilitates the creation of trust in similar situations.

Agents present different degrees of autonomy and intelligence. For example, agents such as the lower level agents are mostly reactive agents that inform the NM agent when a new node is discovered. The NM agent itself has some degrees of autonomy and intelligence: (*i*) it aggregates information from the agents that discover new peers, (*ii*) informs the OpenSocial of the state of neighborhood, (*iii*) tries to present a consistent view, merging the data from the different sources and (iv) it configures the discovering agents according to high level criteria, such as battery consumption and hardware availability. The OpenSocial agent is basically a gateway to the OpenSocial container translating the other agents requests for the OpenSocial container and The TN agents is a true agent that performs potentially complex negotiations on his user's behalf and depending from the configuration may work in entire autonomy.

V. CONCLUSIONS

In this paper we proposed a novel peer-to-peer social networking platform that leverages existing, widespread and stable technologies such as DHTs and BitTorrent. Although the primitives offered by those technologies were created with other goals in mind, however, they could be used with minor modification in our system. In particular, we introduced a key-based identity system and a model of social relations for distributing resources efficiently among interested readers.

In fact, we designed Blogracy as a micro-blogging social networking system, and we gave priority to the features more important for micro-blogging, such as: (i) anonymity and resilience to censorship; (ii) authenticatable content; (iii) semantic interoperability using activity streams and weak semantic data formats for contacts and profiles; and (iv) data availability.

After having implemented and tested all the core features of Blogracy, we proved that a peer-to-peer architecture can be functional for both sharing files and advertising new social activities. Moreover, by adhering to the OpenSocial standard, the system can be integrated with other existing social platforms.

Apart from the core functions of the system, we are experimenting with more advanced features, which nevertheless are essential to provide a smooth experience to users. In fact, differently from centralized systems, the nodes of Blogracy are fully responsible for the platform operation. Friend discovery and content recommendation have to be realized in a completely distributed fashion, on the basis of trust agreements among users.

Finally, the realization of pervasive features on top of Blogracy will require the dynamic management of open location and proximity groups, with possibly complex trust negotiation protocols among autonomous agents.

REFERENCES

[1] T. Berners-Lee, "Long Live the Web: A Call for Continued Open Standards and Neutrality," in *Scientific American Magazine*, December 2010.

[2] S. Buchegger and A. Datta, "A case for P2P infrastructure for social networks – opportunities & challenges," in *Proceedings of Sixth International Conference on Wireless On-Demand Network Systems and Services (WONS 2009)*, Snowbird, UT, USA, feb. 2009, pp. 161–168.

[3] S. Buchegger, D. Schiöberg, L. Vu and A. Datta, "PeerSoN: P2P social networking: early experiences and insights," in *Proceedings of the Second ACM EuroSys Workshop on Social Network Systems.* Nuremberg, Germany: ACM, 2009, pp. 46–52.

[4] L.M. Aiello and G. Ruffo, "LotusNet: Tunable privacy for distributed online social network services," *Computer Communications*, Elsevier B.V., vol. 35, no. 1, pp. 75-88. 2010.

[5] L.A. Cutillo, R. Molva and T. Strufe, "Safebook: a Privacy Preserving Online Social Network Leveraging on Real-Life Trust," in *IEEE Communications Magazine*, vol 47, n.12, ser. Consumer Communications and Networking, 2009, pp 94-101.

[6] R. Baden, A. Bender, N. Spring, B. Bhattacharjee and D. Starin., "Persona: an online social network with user-defined privacy", in *Proceedings of the ACM conference on Data communication (SIGCOMM '09),* Barcellona, Spain: ACM, 2009, pp. 135-146.

[7] M. Basuga, R. Belavic, A. Slipcevic, V. Podobnik, A. Petric and I. Lovrek, "The MAgNet: Agent-based Middleware Enabling Social Networking for Mobile Users," in *Proceedings of the 10th International Conference on Telecommunications ConTEL 2009* / Podnar Zarko, Ivana; Vrdoljak, Boris, editor(s). Zagreb, Croatia: IEEE, 2009. 89-96.

[8] F. Bellifemine, G. Caire, A. Poggi, G. Rimassa, "JADE: a Software Framework for Developing Multi-Agent Applications. Lessons Learned," in *Information and Software Technology Journal 50 (2008)*, pp. 10-21.

[9] D. Brickley and L. Miller, "FOAF vocabulary specification," http://xmlns.com/foaf/0.1/, 2005.

[10] F. Bergenti, E. Franchi and A. Poggi, "Agent-based Social Networks for Enterprise Collaboration," in *Proceedings of the 20th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 2011.

[11] A. Gursel and S. Sen, "Improving Search In Social Networks by Agent Based Mining," in *IJCAI'09, Proceedings of the 21st international jont conference on Artifical intelligence*, Morgan Kaufmann Publishers Inc. San Francisco, CA, USA, 2009, pp. 2034-2039.

[12] B. Yu and M. P. Singh. "Searching social networks," in AAMAS '03: Proceedings of the second international joint conference on Autonomous agents and multi-agent systems, New York, NY, USA, 2003. ACM Press, pp. 65–72.

[13] F.E. Walter, S. Battiston and F. Schweitzer, A Model of a Trust-based Recommendation System on a Social Network, Journal of Autonomous Agents and Multi-Agent Systems, vol. 16, no. 1 (2008), pp. 57-74.

[14] F. Bergenti, L. Rossi and M. Tomaiuolo, "Towards Automated Trust Negotiation in MAS," *in Proceedings of the 10th Workshop "Dagli Oggetti agli Agenti" (WOA 2009).*

[15] L. Hamill and N. Gilbert, "Simulating large social networks in agent-based models: A social circle model," in *Emergence Complexity Organization* (2010) Volume: 12, Issue: 4, Publisher: Emergent Publications, pp. 78–94.

[16] F. Bergenti, E. Franchi and A. Poggi, "Selected Models for Agent-based Simulation of Social Networks," in *3rd Symposium on Social Networks and Multiagent Systems (SNAMAS '11)*, pp. 27–32.

[17] N. Li. "Local Names in SPKI/SDSI," in *Proceedings of the 13th IEEE workshop on Computer Security Foundations (CSFW '00).* Cambridge, UK, July 2000: IEEE Computer Society Press, pp. 2–15.

[18] J. Bethencourt, A. Sahai and B. Waters, "Ciphertext-Policy Attribute-Based Encryption," in *IEEE Symposium on Security and Privacy*, 2007. Oakland, California, USA, May 2007, pp.321-334.

[19] K. Rzadca, A. Datta and S. Buchegger, "Replica placement in P2P storage: Complexity and game theoretic analyses," in *Proceedings of the International Conference on Distributed Computing Systems*. Genova, Italy: IEEE Computer Society, 2010, pp. 599–609.

[20] J. Douceur and R. Wattenhofer, "Competitive hill-climbing strategies for replica placement in a distributed file system," in *Distributed Computing*, ser. Lecture Notes in Computer Science, J. Welch, Ed. Berlin / Heidelberg: Springer, 2001, vol. 2180, pp. 48–62.

[21] S. Bernard and F. Le Fessant, "Optimizing peer-to-peer backup using lifetime estimations," in *Proceedings of the 2009 EDBT/ICDT Workshops*. New York, NY, USA: ACM, 2009, pp. 26–33.

[22] E. Franchi, A. Poggi and M. Tomaiuolo, "Developing Applications with HDS," in *Proceedings of the 12th Workshop "Dagli Oggetti agli Agenti" (WOA 2011)*, pp. 117-122.