Service Provisioning Framework for a Self-Organized Grid

Amos Brocco
Department of Informatics
University of Fribourg, Switzerland
Email: amos.brocco@unifr.ch

Béat Hirsbrunner
Department of Informatics
University of Fribourg, Switzerland
Email: beat.hirsbrunner@unifr.ch

***Abstract*—Efficient, scalable and robust grid service provisioning is one of the cornerstones of next generation grid systems. In agreement with this vision, we propose a flexible grid service provisioning framework built on a two layer architecture. A lower layer provides membership management and basic communication upon a self-organized peer-to-peer overlay maintained and optimized using a fully distributed bio-inspired algorithm. An upper layer exploits these facilities to deliver high-level grid services such as resource discovery and monitoring. We evaluate the proposed solution on different scenarios and assess its qualities concerning performance, reliability, fault-resilience and network communication efficiency.**

I. Introduction

A Grid system enables the sharing of geographically dispersed computing or data resources, with the goal of supporting computationally intensive tasks. An essential factor to fully exploit these resources is the possibility to locate them in the grid. In this respect, grid systems need to provide efficient, scalable and robust resource discovery services. Furthermore, such architectures need to be flexible, autonomic, and self-manageable, in order to reduce deployment costs, increase reliability, and meet dynamic users' needs [1]. Therefore, research on enabling technologies to support the next-generation grids needs to take these aspects into account.

To support resource discovery, traditional grid systems, such as [2], [3], rely on centralized indexing services, known as grid resource information services [4]. Centralized solutions limit the scalability of grids and represent communication bottlenecks as well as single points of failure. A partial solution to this problem is found in hierachical system designs, with the grid divided in different groups, each one managing local indices [5]. Nonetheless, the inherent problems of centralized designs remain, although at a smaller scale.

A different approach to the problem focuses on the implementation of solutions originating from P2P networks, which inherently provide both flexibility and fault-resilience. These systems are classified as structured and unstructured approaches. The first category groups systems based on deterministic topologies [6], [7], that organize the information in a distributed hash table according to a key-node match. In such systems, the topology itself embeds important semantic information that enables very efficient and deterministic resource discovery, with low network overhead and response time. On the contrary, in this context range search and multidimensional queries tend to be a complex task, which has nevertheless been partially addressed, for example in [8]. Additionally, structured overlays may not be well suited for dynamic conditions and suffer from excessive churn rates [9]. Unstructured solutions [10], [11] do not enforce strict topologies, and are more tolerant toward nodes joining and leaving the network. Resources are typically located using flooding mechanisms, by first querying neighbors and then propagating queries progressively throughout the network. These approaches allow free text queries but suffer from excessive network communication overhead, unless some structure is introduced (i.e. *superpeers* [12]) or optimized flooding strategies are employed.

Our research aims at providing an efficient resource discovery platform, by combining the aforementioned advantages of both structured and unstructured systems and overcoming some of their limitations. By taking into account the different functional requirements between grid and P2P systems, we propose a flexible and self-managed framework to support diverse grid operations. In this respect, we describe a self-structured grid service provisioning framework built on a two layer architecture consisting of a self-organized peer-to-peer overlay and a grid service provider. The former provides membership management as well as basic group communication primitives that enable the latter to deliver high-level grid services such as resource discovery and monitoring, or to support load balancing tasks, etc. Efficient low-level communication is achieved by reorganizing logical connections between nodes in order to reduce average path length in the overlay. Optimization of the network is accomplished with a completely decentralized approach based on a collaborative bio-inspired algorithm. The proposed layered architecture promotes a clear separation of concerns and enables high level access to grid resources while hiding the complexity of the underlying network.

The rest of this paper is organized as follows: Section II presents a brief overview of grid service provisioning architectures. Section III introduces the middleware architecture, whereas Section IV details the operation of the network communication layer and the overlay management algorithm. Section V discusses the service provisioning layer and the services available to applications. Sections VI and VII present the evaluation scenarios, respectively the results obtained. Finally, Section VIII summarizes the work presented in the paper and provides some insights to future developments.

II. Related Work

In this section, a brief overview of grid service and communication frameworks based on unstructured P2P overlays is presented. In [13], the authors underline the differences between grids and P2P systems, and the need for different performance analysis strategies when solutions devised for the latter are applied on grids. The focus is then put on resource discovery forwarding strategies that are more tailored for the characteristics and design goals of grids. UMM [14] proposes a two layer approach separating the task of maintaining an overlay and disseminating the information in an efficient way. Node connections are optimized to reduce latency and increase available bandwidth; furthermore, to reduce overhead, information is forwarded on the network according to an optimized scheme that tries to avoid message duplication. Another project, called SAXONS [15], maintains an overlay with low latency, high bandwidth paths, as well as small distances that aims at supporting internet services through optimized overlay communication. Finally [16] proposes an example of service-oriented framework, SP2A, based upon a peer-to-peer overlay and focusing on resource sharing in grids. Our work follows the path traced by the aforementioned approaches, but maintains an optimized self-structured overlay using a bio-inspired algorithm.

III. Framework Architecture

The proposed framework architecture is composed of two functional layers: a low-level network communication layer, and a higher level service provisioning layer (Figure 1).

On one side, the network communication layer is responsible for membership and communication management, by ensuring node connectivity with the rest of the network, respectively by carrying out basic communication tasks. On the other side, the upper layer exposes grid oriented services, namely resource discovery and querying, to the grid application, and hides the complexity of the network management.

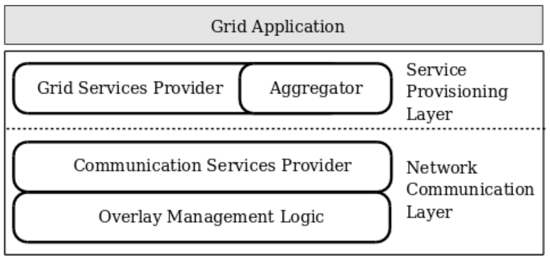
Grid Application
Grid Services Provider
Aggregator
Service Provisioning Layer
Communication Services Provider
Overlay Management Logic
Network Communication Layer

Fig. 1. Framework Architecture

Applications interact with the upper layer by submitting resource discovery queries to the grid service provider and by retrieving the corresponding results. Conversely, the latter exploits the communication facilities of the lower layer to spread the query over the network. The service provisioning layer is divided into a grid services provider and an aggregator component: the first provides support for simple tasks like resource discovery queries, while the latter enables complex multi-task queries and implements caching.

IV. Network Communication Layer

The network communication layer provides low level connectivity between grid nodes, as well as basic information delivery mechanisms. This layer is divided into two components: an overlay management logic, and a communication services provider. In this section the details of both layers are discussed.

A. Overlay Management Logic

The overlay management logic provides connectivity between nodes by mean of an optimized topology which keeps reduced distances between each pair of nodes thus minimizing the average path length. The management algorithm, called BLÅTANT-S, is an improved version of our previous algorithm BLÅTANT-R [17] that reduces both the complexity and the generated traffic of the latter, while retaining its performance characteristics. The algorithm is fully distributed, and is based on different kind of lightweight mobile agents traveling across the network whose behavior is inspired by ant colonies.

Ant algorithms are part of the field of swarm intelligence [18] optimization algorithms. These algorithms mimic the behavior of ant colonies in order to solve optimization problems, such as routing problems in dynamic networks. In a similar way, the proposed overlay management algorithm autonomously organizes the topology exploiting the collaborative behavior of different ant agents. This behavior is adaptive with respect to changes to the underlying network and fault tolerant.

Each node n_i on the grid maintains a set of logical neighbors N_i, and a fixed-size cache table with information about other nodes, such as their estimated distance and their neighbors. This table is updated using the information carried by incoming ant agents, and provides a local partial view of the network. Furthermore, the number of neighbors cannot exceed a user-defined limit, to prevent hub formation.

Topology Optimization Rules. According to its own partial information about the network, each node performs local optimization of its connections by creating new logical links and removing existing ones. This process is controlled by two rules exploiting a user defined parameter D, such that the resulting average path length is between D and $2D-1$.

Connection Rule. Let n_i and n_j be two non-connected nodes in an overlay G, and $d_G(n_i, n_j)$ the minimal routing distance from n_i to n_j in G. A new link between n_i and n_j is created if the following condition holds:

$$d'_G(n_i, n_j) \geq 2D - 1 \tag{1}$$

where $d'_G(x, y)$ is defined as $min(d_G(x, y), d_G(y, x))$, and the logical connection is created by adding n_i to N_j, respectively n_j to N_i.

The Connection Rule reduces the average path length between each pair of nodes to a value less than $2D - 1$.

Conversely, as the number of connections per node is bounded, the Disconnection Rule aims at removing links that are not deemed necessary to achieve the desired average path length. A detailed proof of these rules is given in [17].

Disconnection Rule. Let n_i and n_j be two connected nodes in an overlay network G, $i \neq j$. Let $G' \leftarrow G \setminus \{n_i\}$ and N_i the set of neighbors of n_i. Node n_i is disconnected from $n_j \in N_i$ if:

$$\exists\, n_k \in N_i, k \neq j, |N_j| > |N_k| \;:\; d^*_{G'}(n_j, n_k) + 1 \leq D \quad (2)$$

where $d^*_G(x, y)$ is defined as $max(d_G(x, y), d_G(y, x))$, and the disconnection consists of removing n_i from N_j, respectively n_j from N_i.

Ant Species. Different tasks of the algorithm are performed using various species of ant agents. These species differ from each other by the information they carry, and the behavior they trigger on each visited node:

– *Discovery Ants* wander across the network building up a fixed-size vector of visited node identifiers. This information is processed by each visited node to update the local view.
– *Construction-Link Ants* carry connection requests from a node wanting to join the overlay. If the recipient cannot accept the request (because the maximum number of neighbors has been reached), the ant is forwarded to the neighbor with the least degree.
– *Optimization-Link Ants* perform connections between two nodes as result of the Connection Rule.
– *Unlink Ants* disconnect two nodes according to the Disconnection Rule.
– *Update Neighbors Ants* notify a node when one of its neighbors has updated its neighbors set. This ensures that each node is able to recover from abrupt disconnection of a neighbor by connecting with its last known neighbors.
– *Ping Ants* are used to keep connections between nodes alive.

Discovery Ants are generated periodically on each node according to a uniform probability distribution, and are disposed after a given number of hops traveled in the overlay; other species of ants are instantiated as necessary to fulfill the particular task. As the optimization task depends on the information gathered by Discovery ants, running the algorithm with a null population will prevent any improvement of the topology.

Pheromone Trails. Real ants use a stigmergic (i.e. indirect) communication mechanism which involves leaving chemical *pheromone* trails in the environment. This chemical traces can be sensed by other individuals in the colony and their concentration indicates the desirability of a given path. With time, unless new chemical is left by the insect, the trail completely evaporates. In our system, we emulate this behavior, and in that respect pheromone concentrations are represented as numerical values $\tau \in [0, 1]$ stored on each node. Ants executing on a node can both read the actual concentration of a trail, and *reinforce* it by increasing its value up to a maximum of 1.

Each node periodically simulates *evaporation* by lowering the value of a trail τ according to an update function $\tau \leftarrow \tau * \psi$, and $\psi < 1$. We distinguish between incoming β trails, and outgoing γ trails. When an ant travels from node n_i to a neighbor n_j, the corresponding trail $\gamma_i[n_j]$ on n_i is reinforced. Conversely, when the ant arrives on n_j, pheromone trail $\beta_j[n_i]$ is reinforced. Discovery Ants wandering on the network preferably chose paths with lower γ pheromone concentration, ensuring a fair coverage of the network. Furthermore, abrupt node disconnection can be detected by monitoring the concentration of β trails: when values approach a lower threshold a recovery procedure is started. Accordingly, in absence of application traffic, Ping Ants ensure that pheromone concentrations do not completely evaporate when nodes are still connected to the overlay.

B. Communication Services Provider

On top of the network overlay management, the communication services provider delivers basic data packet transmission services to the upper layers, similarly to an electronic mail system. In particular, three different information transmission services are provided: unicast (point-to-point message transmission), multicast (one-to-many message transmission), and broadcast (one-to-all message transmission). To limit the amount of traffic generated by broadcast communication, a selective and limited flooding strategy may be employed. Selective flooding means that, at each step, the query is forwarded only to a random subset of all neighbors. With limited flooding each node avoids forwarding queries that have already been processed in the past, which implies that nodes keep track of received queries. Accordingly, broadcast communication is provided with a *best-effort* policy: although the depth of flooding can be specified, depending on the actual overlay topology, only a subset of the nodes may be contacted. The efficiency of this broadcasting strategy is validated by our experimental results which are presented in Section VII.

Actual unicast or multicast communication can be targeted to either precise IP addresses, or to logical names: in the latter case, because of the lack of a centralized name resolution service in the overlay, the system will fall back to a broadcast.

V. Service Provisioning Layer

The Service Provisioning Layer provides an interface between grid applications and the pool of resources shared in the grid. Interaction between the service provisioning layer and the overlay management is asynchronous and based upon a custom message passing protocol and callback notifications.

On one side, the service provisioning layer exploits the basic communication infrastructure offered by the overlay management layer to provide high-level services, such as resource discovery or monitoring. In particular, resource discovery is supported by broadcast and multicast communication, whereas resource monitoring typically relies on point-to-point communication with the queried node.

On the other side, this layer also provides a mechanism to match incoming queries from other nodes that are fetched from the lower layer. Activities of the grid services provider related to complex resource discovery queries are supported by an aggregator component. Furthermore, this layer also exposes communication services that enable best-effort communication within nodes: this can be used, for example, to broadcast notification messages to individual node administrators.

A. Grid Services Provider

The Grid Service Provider implements the logic required to parse high-level queries and broadcast the appropriate information on the network. This component is not concerned with scheduling of jobs on remote nodes nor with actual communication with remote resources: as these tasks typically depend on complex scheduling and resource access policies, they are left to the grid application.

B. Aggregator

The aggregator component provides support for complex queries that may require the composition of different services. For example, to support a load-balancing activity, not only does the application require a particular resource profile, but also it needs to choose the best match according to the available bandwidth. In this case, the aggregator will decompose the task by first performing a resource discovery, and then profiling the available bandwidth on the connection to matching nodes.

The aggregator also improves response time by providing cached results from previous resource discovery queries. The cache is also used to profile searches and detect similarity between queries in order to issue proactive queries that will enhance results and reduce latency for popular requests.

VI. Evaluation

To assess the suitability of the proposed architecture to support grid service provisioning, we evaluated its behavior in different usage scenarios. Accordingly, we conducted experiments to test the scalability of the system, as well as its reliability and efficiency for resource discovery tasks. In this section the details of the considered scenarios are discussed.

Simulation Timing. For practical reasons all experiments are run in a discrete time simulator, the implementation remains fully distributed. Timing is dictated by migrations of the population of ants: at each iteration any ant may travel at most for one hop in the overlay (migrating from a node towards another node). Although communication is reliable, the transport layer does not ensure a FIFO behavior, as the order of multiple messages originating from the same source node is not ensured at recipients.

Overlay Management. Unless otherwise specified, in all evaluation scenarios the main algorithm parameters are kept constant. The choice of parameter values, not discussed here due to space limitations, does not significantly affect the qualitative behavior of the presented results. The pheromone decay ratio ψ is set to 0.991. Each node generates a new Discovery ant with a probability of 1% every 150 iterations: this ant may subsequently travel for at most 50 hops in the overlay. Each Discovery ant carries the identifiers of at most the last 20 visited nodes. Furthermore, in all simulations the optimization parameter D is set to 5, which should result in an average path length between 5 and 9.

Simulation Scenarios. Evaluation of the framework is performed, with different goals, on different network scenarios. In particular we consider an expanding network scenario and several dynamic scenarios with different connection/disconnection rates.

All topologies are constructed starting from a pool of 10 nodes (referred to as *well-known* connection points), to whom a number of additional nodes connect. Each node may accept a maximum of 8 connections. All scenarios are evaluated on an overlay composed of 1281 nodes. The only exception is in the expanding network scenario, used to evaluate the scalability of the overlay, where the number of nodes is progressively increased from 10 to around 10000 by adding a node every 5 iterations. Within this scenario, the performance of the overlay management algorithm is also evaluated by varying the Discovery ants birth probability from 0% (i.e. no optimization), to 1%, and 5%.

To assess the reliability of the overlay and the generated network overhead under typical conditions, a set of dynamic scenarios are then considered: during simulations the network is periodically modified by adding and removing nodes. To evaluate the reliability of the network, the considered connection and disconnection intervals are of 20, 50, and 100 iterations. When removing a node, two equiprobable policies are considered: proper disconnection, where the leaving node notifies its neighbors and reconnects them to preserve connectivity, and abrupt disconnection. As the rate of connections and disconnections are equal the size of the network remains almost constant. Finally, to evaluate resource discovery and group communication a dynamic network scenario with a connection and disconnection period of 20 iterations is used.

Resource Discovery and Group Communication. The efficiency of resource discovery is evaluated by issuing queries and measuring the success rate as well as the forwarding strategy costs. Every 25 iterations, 10 queries are started from random nodes and broadcasted on the network to search for a resource shared by other nodes. Each query has a limited time-to-live, travels up to a maximum distance from the starting node in the overlay, and is transmitted according to a selective forwarding policy.

Several forwarding strategies are evaluated by varying the forwarding distance (TTL) as well as the maximum number of neighbors to which the query is forwarded at each step (FW).

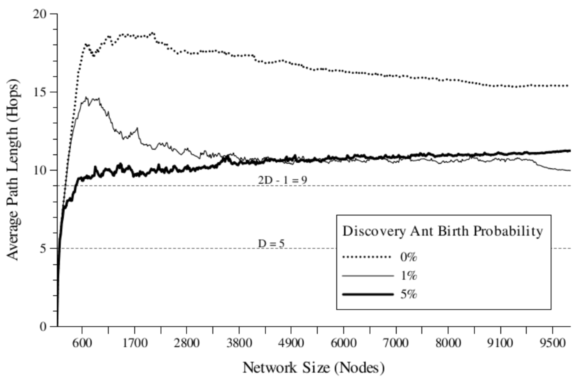

Fig. 2. Average Path Length in Expanding Network

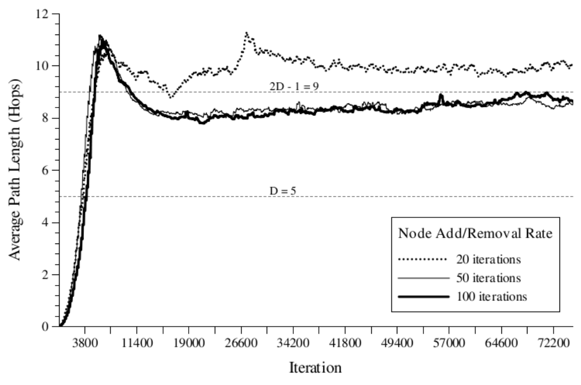

Fig. 3. Reliability in Dynamic Scenarios

For each possible resource about 27 replicas are available on the network. Once a match is found, resource discovery is considered as successful, and the query is stopped. Forwarding is also stopped when a node receives a query that has already been processed before.

Group communication efficiency is assessed by determining the overall coverage (percentage of the nodes that received the message) as well as the total number of message forwardings. Every 25 iterations, 10 broadcast messages are sent by random nodes. As with resource discovery queries, we experimented with different strategies by varying the TTL and FW.

Communication Costs. To determine the transmission cost resulting from the traffic generated by the algorithm, an estimation of the size of each data packet used by the framework has been made. In particular, for the overlay management part, the estimated size of each ant is as follows:

– *Discovery Ant*: 420 bytes *plus* 24 bytes/visited node;
– *Construction-link Ant*: 444 bytes;
– *Optimization-link Ant*: 420 bytes;
– *Unlink Ant*: 420 bytes;
– *Update Neighbors Ant*: 420 bytes *plus* 24 bytes/neighbor.
– *Ping Ant*: 420 bytes;

For the resource discovery task, query and reply sizes are considered as follows:

– *Resource discovery query*: 1024 bytes;
– *Resource discovery query reply*: 456 bytes;

All estimations are based on the minimal size of a UDP packet and the actual payload. A resource discovery query reply is sent to the origin node for each match found.

VII. Results

In this section we present and discuss the results of the considered evaluation scenarios. The first set of results mainly concerns the overlay management algorithm, whereas the second part concentrates on the efficiency of high level grid services such as resource discovery and group communication.

A. Scalability in an Expanding Network

Figure 2 shows the average path length in the expanding network scenario. Without optimizations, the construction protocol is able to keep an average length of 15 hops. In contrast, optimization tasks are able to keep up with the growth of the network, and bound path lengths in the overlay to values close to 10. Although a larger population of ants does not seem to improve the result, the number of edges in the resulting network (having an average of 9500 nodes) is significantly lowered: in average 46000 with 1%, 39000 with 5%, and 21000 without optimization.

B. Reliability and Traffic Estimation in Dynamic Scenarios

Overlay reliability results aim at assessing the ability of the network to control dynamic conditions with high connection and disconnection rates. As shown in Figure 3, the algorithm is able to maintain an average path length close to 8.6 hops with rates of 50 and 100 iterations, and of 10 hops with 20 iterations. Although not shown in the figure, in all scenarios the network remains fully connected.

After the initial construction phase, the average number of edges in the overlay is 6700. With a join-leave period of 50 iterations, according to packet size estimations provided in the previous section, at each iteration the algorithm generates a total average traffic of 20 KBytes, which reduces to a bandwidth consumption of about 3 bytes per link.

C. Resource Discovery Efficiency

Resource discovery efficiency is evaluated both in respect of the success rate of single queries, as well as the traffic generated by the forwarding algorithm. Figure 4 shows the results of different forwarding strategies, with varying TTL and FW, concerning the average cost and success rate of one query. The histograms detail both the cost of maintaining the overlay (averaged for a rate of 10 queries every 25 iterations, for a total of 10000 queries) and the actual resource discovery query cost. Clearly, better result can be obtained at the expense of generating more traffic. The overlay management counts for a negligible part of the traffic; moreover, increasing the query traffic makes the overlay management even more efficient because less Ping Ants are needed.

Figure 5 shows group communication cost and performance. The best coverage (67% of an average of 1256 nodes) is obtained with a TTL of 9 and FW equal to 8, resulting

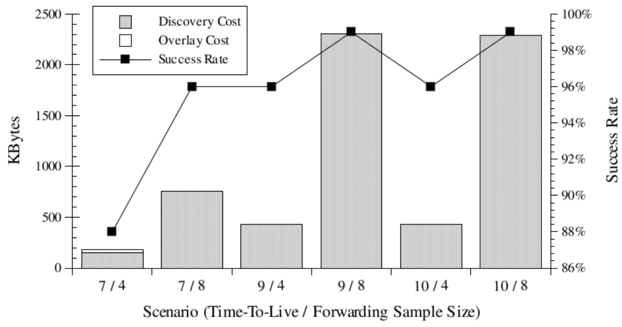

Fig. 4. Resource Discovery Efficiency

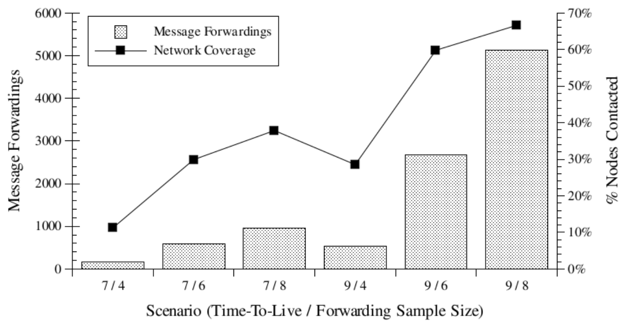

Fig. 5. Group Communication Efficiency

in an average of 5136 forwardings. It is interesting to note how the 9/6 scenario obtains similar coverage (60%) with about half the number of forwardings: as the average number of neighbors per node is between 5 and 6, increasing the FW value does not provide significant advantages. Additional experiments showed that group communication is influenced by the dynamics of the network, with the coverage improving in more stable topologies.

VIII. Conclusions

In this paper we introduced a framework for grid service provisioning composed of two functional layers: a self-organized peer-to-peer overlay management layer, and a grid services interface layer. This layered approach aims at separating different concerns: low-level network and membership management, and high-level grid services. The overlay management layer maintains an optimized topology through the collaborative behavior of different mobile agents inspired by the behavior of insect colonies. The proposed algorithm bounds the distance between each pair of nodes, by creating additional links and rearranging existing ones. This solution provides an adaptive, reliable and robust communication platform to connect grid nodes and ensure efficient communication. The upper service provisioning layer hides the complexity of the network by providing grid applications with an interface to the resources shared on the grid. In particular, the layer supports resource discovery and monitoring, as well as group communication. Simulations and experiments on different scenarios demonstrated the qualities of the framework. More specifically, we assessed the scalability and fault resiliency of the overlay management, as well as the efficiency of both resource discovery service and group communication.

Future work will focus on improved query forwarding and message broadcasting strategies, as well as on the integration of this framework within the SmartGrid [19] platform.

IX. Acknowledgments

The authors would like to thank Dr. Apostolos Malatras and Fulvio Frapolli for their useful feedback. This research is financially supported by the Swiss Hasler Foundation in the framework of the “ManCom Initiative”, project Nr. 2122.

References

[1] A. Andrzejak, A. Reinefeld, F. Schintke, T. Schtt, C. Mastroianni, P. Fragopoulou, D. Kondo, P. Malecot, G. Cosmin Silaghi, L. Moura Silva, P. Trunfio, D. Zeinalipour-Yazti, and E. Zimeo. Grid architectural issues: State-of-the-art and future trends, May 2008.

[2] D. Thain, T. Tannenbaum, and M. Livny. Distributed computing in practice: the condor experience. *Concurrency - Practice and Experience*, 17(2-4):323–356, 2005.

[3] R. Buyya, D. Abramson, and J. Giddy. Nimrod/g: An architecture for a resource management and scheduling system in a global computational grid. In *HPC ASIA2000*. IEEE, 2000.

[4] K. Czajkowski, S. Fitzgerald, I. Foster, and C. Kesselman. Grid information services for distributed resource sharing. *10th IEEE Int. Symposium on High Perf. Dist. Computing, 2001.*, 2001.

[5] M L. Massie, B. N. Chun, and D. E. Culler. The ganglia distributed monitoring system: design, implementation, and experience. *Parallel Computing*, 30(7), 2004.

[6] I. Stoica, R. Morris, D. Karger, F. M. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *SIGCOMM '01*, volume 31, pages 149–160, New York, NY, USA, October 2001. ACM Press.

[7] A. Rowstron and P. Druschel. Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. *Lecture Notes in Computer Science*, 2218, 2001.

[8] C. Schmidt and M. Parashar. Flexible information discovery in decentralized distributed systems. *12th IEEE International Symposium on High Performance Distributed Computing*, June 2003.

[9] S. Rhea, D. Geels, T. Roscoe, and J. Kubiatowicz. Handling churn in a dht. In *ATEC '04*, Berkeley, CA, USA, 2004. USENIX Association.

[10] M. Ripeanu. Peer-to-peer architecture case study: Gnutella network. In *Peer-to-Peer Computing, 2001.*, August 2001.

[11] I. Clarke, O. Sandberg, b. Wiley, and T. W. Hong. Freenet: A distributed anonymous information storage and retrieval system. *Lecture Notes in Computer Science*, 2009, 2001.

[12] B. Beverly Yang and H. Garcia-Molina. Designing a super-peer network. *19th Int. Conf. on Data Engineering*, March 2003.

[13] A. Iamnitchi and I. Foster. On fully decentralized resource discovery in grid environments. In *Int. Workshop on Grid Computing*, 2001.

[14] M. Ripeanu, A. Iamnitchi, I. Foster, and A. Rogers. In search of simplicity: A self-organizing group communication overlay. *SASO '07*, pages 371–374, July 2007.

[15] K. Shen. Structure management for scalable overlay service construction. In *NSDI'04*, pages 21–21. USENIX Association, 2004.

[16] M. Amoretti, F. Zanichelli, and G. Conte. Sp2a: a service-oriented framework for p2p-based grids. In *MGC '05*, pages 1–6, New York, NY, USA, 2005. ACM.

[17] A. Brocco, F. Frapolli, and B. Hirsbrunner. Bounded diameter overlay construction: A self organized approach. In *IEEE SIS*, April 2009.

[18] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm intelligence: from natural to artificial systems*. Oxford University Press, Inc., New York, NY, USA, 1999.

[19] Y. Huang, A. Brocco, B. Hirsbrunner, M. Courant, and P. Kuonen. Smartgrid: A fully decentralized grid scheduling framework supported by swarm intelligence. In *7th Int. Conf. on Grid and Cooperative Computing*. GCC2008, October 2008.