DyMRA: Dynamic Market Deployment for Decentralized Resource Allocation*

Daniel Lázaro, Xavier Vilajosana, and Joan Manuel Marquès

Universitat Oberta de Catalunya
{dlazaroi,xvilajosana,jmarquesp}@uoc.edu

Abstract. The workload supported by Virtual Organizations (VO) is limited by the quantity of available resources. VOs with scarce resources or peer-to-peer based VOs — due to the dynamicity of available resources — may need extra resources to carry out a given task. Conversely, many Internet-connected computers have surplus bandwidth, storage and computational resources. We face those tradeoffs by enabling VOs to collect and aggregate surplus resources and provide them with availability guarantees to other VOs. This paper presents DyMRA, a decentralized resource allocation system based on markets that allows inter-VO resource allocation. DyMRA is specially designed for dynamic and peer-to-peer environments, where the autonomy of participants to disconnect resources at any time and its decentralized nature requires the capacity to dynamically reallocate resources and services that manage the overall system. DyMRA is built on top of LaCOLLA, a peer-to-peer middleware that allows a group of users to share resources in a collaborative manner. We present the design, architecture and validation of our proposal.

1 Introduction

Internet has fostered the proliferation of virtual communities that are generally formed by users that have common goals and need to collaborate to achieve their technological or business objectives. Virtual communities are managed as virtual organizations[1] (VO) that require computational resources to satisfy the needs of members' applications. Besides, these requirements may typically vary over the lifetime of the virtual organizations due to dynamicity and spontaneous load surges.

Generally, VO resources are supplied by one of the following sources. a) resources may physically belong to the VO. b) members may contribute resources in benefit of the community, as is the case of non-profit organizations such as seti@home [1]. And c), users may join resources in a cooperative way that results in a benefit of all the participants. An example of such kind of systems is the peer-to-peer file sharing system Tribler [2].

Whilst the resources within a VO may be sometimes insufficient to satisfy QoS requirements under unexpected load surges, high level of dynamicity of its

* Work supported by MCYT-TSI2005-08225-C07-05 and Grid4All(IST-2006-034567).

[1] For the purpose of this paper, we consider that a virtual organization (VO) is a group of individuals or institutions who share resources and services for a common goal.

R. Meersman, Z. Tari, P. Herrero et al. (Eds.): OTM 2007 Ws, Part I, LNCS 4805, pp. 53–63, 2007.

members or unpredictable failures, many other Internet-connected computers have surplus bandwidth, storage and computations resources. This opens challenging opportunities to promote inter-VO resource allocation. In other words, a VO could aggregate their surplus resources and offer them to other VOs.

In this paper we address Inter-VO allocation of resources by means of decentralized markets that promote the creation of many local ad hoc trading sites that can be accessed by any VO. Markets have proven their ability to allocate resources efficiently [3] and, more importantly, because they provide mechanisms through which the need may be correctly elicited and quantified; and indeed, they promote incentives to resource owners to provide or trade their resources.

We developed DyMRA, a decentralized resource allocation system based on markets that allows inter-VO resource allocation. DyMRA is specially designed for dynamic and peer-to-peer environments, where the autonomy of participants to disconnect resources at any time and its decentralized nature requires the capacity to dynamically reallocate resources and services that manage the overall system.

DyMRA markets are created at will and run as services within the VO. The choice of a decentralized markets approach in the form of many local ad hoc markets is motivated by the need to deal with dynamic communities and scalability issues that would be limited by a centralized approach.

One issue not addressed in this paper due to space limitations is that of resource specification and bidding language. Many interesting approaches [4,5] present efficient bidding languages that fulfill our requirements. For the purpose of this paper, we consider resources as processing time, storage, and applications that provide a stateless service, like an efficient codec or a parallelizing compiler. We deal with heterogeneity by using standard interfaces, implemented as WS, to access the resources. These belong to a VO and we assume that they can be disconnected or fail at any time. This dynamic behavior introduces a complexity that, added to the decentralized behavior of markets, forced us to design a system that allows us to decouple services from physical resources. Furthermore, we had to pay special attention to availability guarantees.

DyMRA is built on top of LaCOLLA [6,7], a middleware available at [8]. LaCOLLA is a peer-to-peer middleware that allows a group of users scattered across the Internet to share resources in a cooperative manner and that allows the deployment of stateless services using the resources provided by the members of the VO. LaCOLLA guarantees that services deployed are always available (if enough resources are provided). Therefore, DyMRA components are deployed as services in LaCOLLA middleware.

2 Scenario

The scenario presents three different VOs A, B and C that provide general purpose functionalities such as a file sharing services and communication services to its members. VO A is an online gaming community where few members contribute regularly their computational resources to the community; instead

members pay a subscription fee to obtain the services. B is a scientific community and its purpose is the sharing of knowledge and technical documents amongst its members; generally its members contribute with their resources to the community. Finally, VO C is a photo sharing community where members usually contribute with their resources.

This paper focuses on the allocation of resources provided by virtual organizations to other ones and fits perfectly in this scenario. We assume that VOs provide management logic to control the resources within the community.

External allocations may be needed due to spontaneous load surges or when resources cannot be provided by the VO. At a time, the VO A may require more resources than available to match the required quality of service. The VO monitoring service will trigger the buyer service (termed Prospector) to allocate the needed resources. The Prospector searches for markets that trade in the required resources and places a bid on them. Markets are services exposed by VOs that aim to trade in some of their resources. Seller agents (from B or C VOs for example) are triggered to sell resources when the monitoring service determines a surplus of resources within the VO according to some VO policies. Sellers create or place a bid on Markets depending on the suitability of the Market to trade in their resources. When a DyMRA component fails or is disconnected due to the inherent dynamism of VO members (i.e. someone switches off her computer) automatically and transparently to participants the service is reallocated to another suitable node within the VO. After the market clears and winning Seller services and Prospector services are notified, the resources are added to the Pool of external resources of the buying community.

DyMRA addresses this kind of scenarios by providing services to automatically allocate external resources into a VO. The main contributions of DyMRA are twofold; first, the components of DyMRA are deployed as services inside a VO and can, hence, be reallocated when its current location fails or disconnects, keeping the functionality available. Second, DyMRA proposes to distribute markets amongst virtual organizations that place our approach in a design space between a decentralized and a centralized architecture, which we believe responds better to the targeted environment.

3 Requirements

- **Interoperability:** VO services may be exposed as standard interfaces that enable interoperation between VOs.
- **Group self-sufficiency:** The execution of services and the deployment management should be performed using only the resources contributed to the VO by its members.
- **Decentralization and self-organization:** In case of connections, disconnections and failures, the system should keep functioning (it shouldn't have a single point of failure) and should reorganize without requiring any external intervention, getting to a consistent state as soon as the available resources and VO stability allow it.

- **Individual autonomy:** The VO's members should be free to decide which actions to carry out, what resources and services to provide, and when to connect or disconnect.
- **Market availability:** Market services should always be available (if needed) as long as there are enough resources to execute them in the VO.
- **Location transparency:** Market services don't have to worry about other market service's location. The system resolves them transparently, and services access each other using a location-independent identifier.

4 Related Work

Economic based resource allocation within the context of Virtual Organizations, Grid Computing and large scale peer-to-peer systems has been extensively studied [9,10,11]. However, as far as we know, the issue of addressing inter-VO resource allocation is an emergent field of study. Recently in [12] a novel architecture for inter-VO resource allocation in Grids is presented. Their proposal is suited between a centralized and a decentralized approach and proposes a configurable mediator process (executing within the VO) that allocates resources from external providers. Our approach goes one step beyond and we provide, on one hand, transparent reallocation of market services in a dynamic environment, and, on the other hand, we rely on the members of collaborating communities to mutually provide resources.

Another feature that we addressed is that of dynamic deployment of services. We acknowledge some systems that perform this task in a decentralized and self-organized way, as is our target. Snap [13] nodes form a Distributed Hash Table (DHT) which stores the code and data of the services. Replicas of a service are created on demand and stopped when demand decreases. Another system called Chameleon [14] deploys services in a cluster of nodes communicated through a DHT, while trying to maximize its "utility" (calculated from a value assigned to each service and its performance in a given node).

5 Architecture

The architecture of DyMRA consists of a series of components which are in charge of the trading process. These components are:

- **Prospector:** when external resources are needed, it is in charge of finding a suitable market and obtaining the desired resources.
- **Seller:** it is in charge of offering the aggregated surplus resources of the VO in a suitable market.
- **Pool service:** it controls the access of the VO members to the external resources acquired by the VO, acting as a mediator.
- **Sale Handler:** it controls the external access to a set of resources sold to another VO, acting as a mediator.

- **Accounting service:** it monitors the resources available in a VO. Following a policy determined by the VO, it starts the acquisition of external resources or the cession of own resources to other VOs when convenient.
- **Market:** it mediates the trading of resources between VOs.
- **Market Directory:** Contains an index of existing markets and their locations.

The system is built upon a middleware called LaCOLLA [6] which allows a small group of computers connected through internet to participate in collaborative activities and sharing their resources (i.e. provides virtualization of resources), while tolerating high levels of dynamism. This middleware also allows the deployment of services within a VO (or group) [7]. When a service is deployed, the system guarantees that it will always be available, placing it in a suitable location chosen among the resources of the VO, and reinstantiating it in case of failure.

The components of DyMRA are deployed as services inside a VO (except the Market Directory), and can, hence, be reallocated when its current location fails or disconnects, keeping the functionality available. The communication between VOs is done through markets, which are also services, existing in a specific VO. To access a market, it must be discovered through the Market Directory (MD). Markets contact the MD to publish their location and characteristics, and the MD keeps them as a soft state. In case a market ceases to exist, the MD will delete the information about it after its time-to-live (TTL) expires. If a market is reallocated, it will inform the MD of its new location.

The MD is not part of a VO, but an external service which is known and can be accessed by all groups. Its implementation is out of the scope of this paper, but there are many possibilities. It could be a centralized index, but it could also be implemented in a decentralized way if each VO deployed a "MD node" service, and each one of these services act as a node of a DHT, thus distributing the information stored among the VOs. Anyway, this doesn't affect the design of our architecture.

To help understand the functionality of each of the components presented and the overall behavior of the system, we will explain in detail how the trade of resources is done at the buyer VO and at the seller VO (shown at fig. 1), and how the posterior access to the traded resources is managed (fig. 2).

5.1 Trading Process

Buying resources

1a. The Accounting service detects that the resources of a certain type (e.g. storage) available in the VO are below a certain threshold defined by the VO policy. According to a given policy, it determines the resources needed and other factors such as the price that should be paid for them. With this information, it contacts the Prospector and asks it to acquire such resources.

2a. The Prospector looks for a suitable market in the Market Directory.

3a. The Market Directory sends the Prospector a list of markets which suit the specified needs.
4a. The Prospector chooses one of the markets of the list. In case that there is no suitable market, it proceeds to the creation of a new one. Once it has the adequate market located, the Prospector sends its bid. A generic bid describes the type of resource to bid for, the price per unit offered and the number of units required amongst others.

Selling resources

1b. The Accounting service detects that the resources of a certain type (e.g. storage) available in the VO are above a certain threshold defined by the VO policy. According to a given policy, it determines that these resources can be leased to another group, and fixes the price that should be paid for them. With this information, it contacts the Seller and asks it to sell the surplus resources.
2b. The Seller looks for a suitable market in the Market Directory.
3b. The Market Directory sends the Seller a list of markets which suit the specified needs.
4b. The Seller chooses one of the markets of the list. In case that there is no suitable market, it proceeds to the creation of a new one. Once it has the adequate market located, the Seller sends its offer.

Agreement

5. The market makes an agreement between the buyer and the seller. A scheduled double auction is used to match winning bids and offers. The winners are

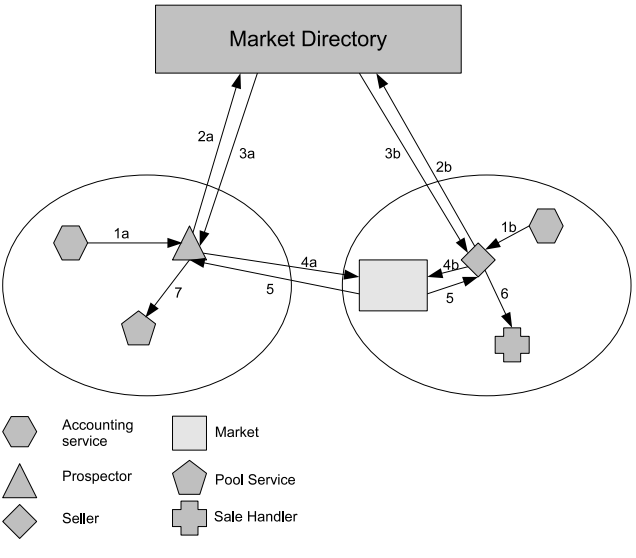

Fig. 1. Interaction among components in the trading process

selected by calculating the price where supply balances demand and matching the highest buy bids above the price with the lowest sell offers below the price. After this, it notifies the sale to both the Prospector and the Seller.

6. The Seller starts a Sale Handler, which is deployed in its VO. This Sale Handler keeps the information about the leasing conditions, and mediates the use of the resources according to these conditions.
7. The Prospector informs the Pool service of its group about the resources bought and the agreement conditions, as well as the location of the Seller of the resources.

When a Prospector or a Seller finds that there is no market available that suits its needs, it proceeds to the creation of a new one. As stated before, the market is implemented as a service. Hence, the component (Prospector or Seller) creates a new service in its VO, which is a market with the desired characteristics. This market registers itself in the Market Directory, and therefore can be accessed by buyers or sellers from outside the VO.

5.2 Accessing the Resources

1. Whenever a client needs to use a resource, the system checks the VO policies to determine whether it must depend only on local resources or should use external resources. In the latter case, the client contacts the Accounting service.
2. The Accounting service checks the resources currently available to the VO. Following the VO's policy, it determines what resources the client must use, whether these are internal or external. In the former case, it tells the client which resource to use. Otherwise, it tells him to contact the Pool service.
3. The client contacts the Pool service, as if it was a local resource.
4. The Pool service chooses which of the external resources available to the VO should be used, and contacts its corresponding Seller. It sends the id of the sale it wants to use.

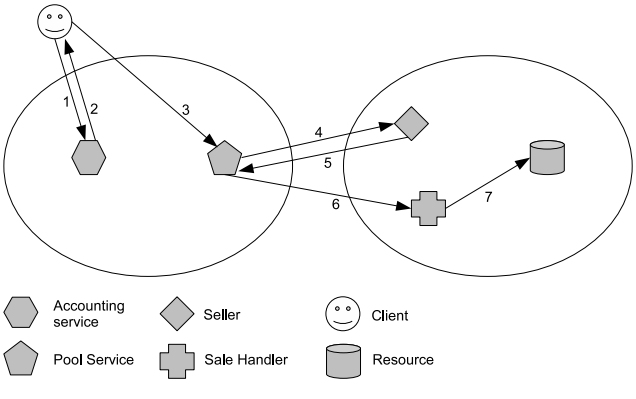

Fig. 2. Interaction among components in the access process

5. The Seller tells the Pool service the location of the Sale Handler that manages the specific agreement.
6. The Pool service contacts the Sale Handler, according to the conditions of the agreement, which may include, for example, symmetric key cryptography. It basically resends the request of the client.
7. The Sale Handler checks that the request of the Pool service does not violate the conditions of the agreement. After this, it uses the resources of the VO to fulfill the request of the Pool service.

As stated before, the services can change their locations due to failures or disconnections. This is not a problem inside a VO, as the system guarantees that clients, as well as other services, can contact any active service. To access external resources, though, the Pool must contact the Seller of another VO, whose location might have changed from the moment when the agreement was made. This can be solved in more than one way. A solution would be to use the Market Directory to store also the location of the Seller of each VO. This information would be maintained in a soft state, just like the one about markets, with the Sellers explicitly publishing their locations in the Directory. The Pool could then contact the MD to get the current location of the Seller, in case it cannot reach it in its previous location. This would solve the problem, but implies relying in an external entity (even though, as seen before, the MD can be implemented cooperatively by the VOs). A solution that only depends on the two VOs that need to communicate would be that both the Pool and the Seller keep the location of those Sellers and Pools, respectively, they have a deal with. In case one of these services is reallocated, it would notify all its "business partners" about its new location. Although it would be less probable, contact can still be lost if both Pool and Seller are reallocated at the same time. To further diminish this probability, these services could be replicated inside the VO. In the worst case, if all the replicas of both services fail together and the Pool of one VO can't contact the Seller of the other VO, the deal is broken, and both VOs will have to go back to the market.

6 Validation

This section presents an implementation of the proposed mechanism and its first validation. These preliminary results demonstrate the viability of our proposal and encourage us to refine it. Currently we are working on a further and exhaustive validation.

We implemented a prototype of the proposed architecture to test its usefulness. The Prospector, Seller, Pool, SaleHandler and the Market have been implemented as deployable services over the LaCOLLA middleware. The Market provides generic operations that allow different mechanisms to be implemented. For our testing purposes we developed a double auction [15] protocol that enables buyers and sellers to submit bids for multiple units of a single resource (i.e storage capacity, cpu capacity and applications).

The MarketDirectory has been implemented as a centralized index, but, as mentioned above, it can be easily substituted with a decentralized approach [16,17]. For our testing purposes, the market directory stores pairs of $< key, value >$ where the key identifies the type of traded resource and the value refers to the identifier of the market where it is traded in.

The objective of our test is to validate the trading process described above. One of the main objectives of our proposal is to provide good availability in environments of high dynamism and churn. Hence, availability has been the main focus of our tests.

We executed a process which periodically tried to buy resources, and another that tried to sell resources. The necessary services (Prospector, Pool, Seller) where active inside the VO, while there was a MarketDirectory available in a static location. Markets, though, according to our proposal, are created on demand. When a Prospector or a Seller wants to access a Market, but there in't any available, it proceeds to create and activate one. When this happens, it is counted in our tests as a failed attempt. For simplicity, Markets have been assigned a limited lifespan, after which they resolve the auction and send the results to the clients. This implies that, periodically, a Prospector or a Seller will have to create a Market, thus decreasing the perceived availability. Markets could also be permanently active, which would increase the availability of the system. There is, though, a trade off between the obtained availability and the resources spent to keep the market active.

The LaCOLLA middleware offers the ability to simulate users' activity and system dynamism (connections, disconnections, failures) in order to conduct tests and validate its functioning. We measured the availability of markets in function of the levels of dynamism of the system. Specifically, we evaluated two different levels of dynamism. In the less dynamic (from now on, called G1) each component had a probability of failure per iteration of 0,0005, and a probability of ordered disconnection of 0,0025. In the more dynamic of the two (G2), the probability of failure per iteration was 0,005, while the probability of disconnection was 0,008. Tests lasted 500 iterations.

The data we analyze is the number of bids that arrive to the market, in relation to the number of bids issued by the group. This depends exclusively of the mechanisms of our system, in contrast to the number of matches, which depends on supply and demand. Note once again that this number decreases because markets have a limited lifespan and are created on demand, which results in a failed access when a market must be created. That doesn't mean that, in a real situation, the bid cannot be issued, only that it will have a bigger delay.

Fig. 3 shows the availability (percentage of succesfully issued bids) obtained in 20 executions, for both G1 and G2. We see that, as expected, the availability is higher in G1, decreasing in G2 because of the higher level of dynamism.

Fig. 4 shows the cumulative distribution function for both G1 and G2. For G1, 50% of the executions obtain an availability of 70% or higher, which must be considered noting that markets are activated on demand, and we count it as

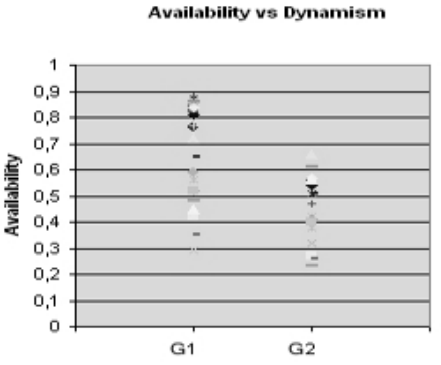

Fig. 3. Availability vs level of Dynamism

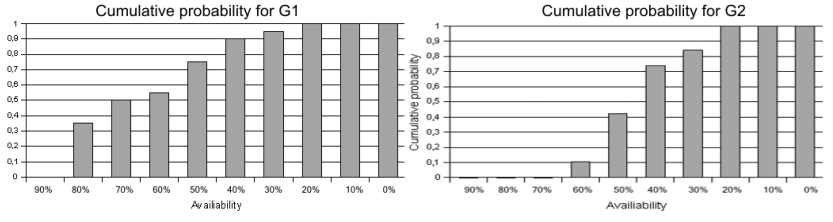

Fig. 4. Cumulative probability of availability levels for G1 and G2

unavailable when activation is needed. For G2, availability is low because of the high level of dynamism.

7 Conclusions

The paper proposes DyMRA, a framework for inter-VO resource allocation. The key aspect of DyMRA is that of market decentralization, that allows allocations of resources amongst different VO in spite of markets' failures. Markets and mediator components such as buyer agents and seller agents are exposed as mobile services within the VO that allows the utilization of inherently centralized mechanisms such as auctions into a decentralized environment without introducing bottlenecks or single points of failure. Furthermore the paper presents the preliminary results of evaluating our proposed architecture. Our future work includes the complete development of the DyMRA components, such as a decentralized Market Directory, and the set of mechanisms to control the access to external allocated resources. Besides, we aim to consider duration of the allocations of resources (lease times) that would permit the application of our framework in a real environment.

References

1. http://setiathome.berkeley.edu/
2. Pouwelse, J., Garbacki, P., Wang, J., Bakker, A., Yang, J., Iosup, A., Epema, D.H.J., Reinders, M., van Steen, M., Sips, H.: Tribler: A social-based peer-to-peer system. Concurrency and Computation: Practice and Experience 19, 1–11 (2007)
3. Shneidman, J., Ng, C., Parkes, D.C., AuYoung, A., Snoeren, A.C., Vahdat, A., Chun, B.: Why markets could (but don't currently) solve resource allocation problems in systems. In: HOTOS 2005. Proceedings of the 10th conference on Hot Topics in Operating Systems, Berkeley, CA, USA, USENIX Association, p. 7 (2005)
4. Cavallo, R., Parkes, D.C., Juda, A.I., Kirsch, A., Kulesza, A., Lahaie, S., Lubin, B., Michael, L., Shneidman, J.: Tbbl: A tree-based bidding language for iterative combinatorial exchanges. In: Multidisciplinary Workshop on Advances in Preference Handling (IJCAI) (2005)
5. Boutilier, C., Hoos, H.H.: Bidding languages for combinatorial auctions. In: Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, pp. 1211–1217 (2001)
6. Marquès, J.M., Vilajosana, X., Daradoumis, T., Navarro, L.: Lacolla: Middleware for self-sufficient online collaboration. IEEE Internet Computing 11(2), 56–64 (2007)
7. Lázaro, D., Marquès, J.M., Jorba, J.: Decentralized service deployment for collaborative environments. In: CISIS 2007. Proceedings of the 1st International Conference on Complex, Intelligent and Software-Intensive Systems, pp. 229–234. IEEE Computer Society Press, Los Alamitos (2007)
8. http://dpcs.uoc.edu/lacolla/
9. Lai, K., Huberman, B.A., Fine, L.: Tycoon: A Distributed Market-based Resource Allocation System. Technical Report arXiv:cs.DC/0404013, HP Labs, Palo Alto, CA, USA (2004)
10. Catnets Consortium: Deliverable d3.1: Implementation of additional services for the economic enhanced platforms in grid/p2p platform: Preparation of the concepts and mechanisms for implementation (gmm) (2005)
11. Buyya, R., Abramson, D., Venugopal, S.: The grid economy (2004)
12. Amara-Hachmi, N., Vilajosana, X., Krishnaswamy, R., Navarro, L., Marques, J.M.: Towards an open grid marketplace framework for resources trade. In: Meersman, R., Tari, Z. (eds.) OTM 2007. LNCS, vol. 4804, pp. 1322–1330. Springer, Heidelberg (2007)
13. Gavalda, C.P., Lopez, P.G., Andreu, R.M.: Deploying wide-area applications is a snap. IEEE Internet Computing 11(2), 72–79 (2007)
14. Adam, C., Stadler, R.: Implementation and evaluation of a middleware for self-organizing decentralized web services (2006)
15. Bao, S., Wurman, P.R.: A comparison of two algorithms for multi-unit k-double auctions. In: ICEC 2003: Proceedings of the 5th international conference on Electronic commerce, pp. 47–52. ACM Press, New York (2003)
16. Ghodsi, A.: Distributed k-ary System: Algorithms for Distributed Hash Tables. PhD dissertation, KTH—Royal Institute of Technology, Stockholm, Sweden (2006)
17. Castro, M., Druschel, P., Kermarrec, A.M., Rowstron, A.: One ring to rule them all: service discovery and binding in structured peer-to-peer overlay networks. In: EW10: Proceedings of the 10th workshop on ACM SIGOPS European workshop: beyond the PC, pp. 140–145. ACM Press, New York (2002)