Life Science Grid Middleware in a More Dynamic Environment

Milena Radenkovic and Bartosz Wietrzyk

School of Computer Science and IT, University of Nottingham,
Jubilee Campus, Wollaton Road, Nottingham, NG8 1BB, UK
{mvr, bzw}@cs.nott.ac.uk

Abstract. This paper proposes a model for integrating a higher level Semantic Grid Middleware with Web Service Resource Framework (WSRF) that extends the prototype presented in [1] informed by issues that were identified in our early experiments with the prototype. WSRF defines generic and open framework for modeling and accessing stateful resources using Web Services and Web Service Notification standardizing publish/subscribe notification for Web Services. In particular we focus on using WSRF to support data integration, workflow enactment and notification management in the leading EPSRC e-Science pilot project. We report on our experience from the implementation of our proposed model and argue that our model converges with peer-to-peer technology in a promising way forward towards enabling Semantic Grid Middleware in mobile ad-hoc networks environments.

1 Introduction

The Grid and Web Services technologies are currently converging and many Grid middleware projects are in transition from pre-Web Service versions (e.g. Globus Toolkit 2 [2], LCG2 [3]) to the new ones based on Web Services. The two major European Grid initiatives, namely UK's National Grid Service (NGS) [4, 5] and EU's Enabling Grids for e-Science (EGEE) [6] have chosen them as the interface for services they provide. Although there is a widespread agreement within the Grid community about the long-term wisdom of adopting a Service Oriented Architecture (SOA) and Web Service technologies as the basis for building a robust and stable Grid infrastructure, there is a short-term problem with such an approach. SOA is still very much work in progress. There are currently many proposed specifications but very few mature standards, which have gained general acceptance and are supported with robust tooling. While standardized and widely adopted specifications are essential for the production deployment, it is also important for the communities to explore the emerging standards and estimate their value before they are adopted in the production environment [7]. Stable and promising Web Services Resource Framework (WSRF) [8, 9] implementations are being evaluated by the National Grid Service (NGS). There is a commitment from NGS that WSRF implementations will be deployed and tested on the NGS production level service [4, 5].

Web Service technology is a major step from the human-centric Web, providing communication between a human and an application, to the application-centric Web

R. Meersman et al. (Eds.): OTM Workshops 2005, LNCS 3762, pp. 264 – 273, 2005.
© Springer-Verlag Berlin Heidelberg 2005

providing communication between applications. Web Services are platform and programming language independent which is extremely important for integrating heterogonous systems and development of loosely coupled distributed systems. They do not only standardize the message exchanges but also the descriptions of interfaces and service discovery [10]. Web Services in their basic form are already widely accepted and adopted by both commercial and academic institutions. The Web Service standards are used in many ongoing e-Science projects providing either Grid or Semantic Web middleware, they are even considered to be ‘an actualization of the Semantic Web vision’ [11].

Web Services from their definition are stateless, i.e. implement message exchange with no access or use of information not contained in the input reference [12]. Even though it is sufficient for many applications, there are many which require the notion of state for their communication. In these cases, the lack of any standards for expressing the state forces designers to use custom solutions, which are highly incompatible between particular systems, making their interoperability and integration increasingly difficult.

WSRF standardizes the design patterns and message exchanges for expressing state, i.e. data values that persist across, and evolve because of, Web Service interactions [12]. WSRF is considered an ‘instruction set of the Grid’ [13]. Besides WSRF there is another important standard defined. It is Web Service Notification (WSN) [14] which covers publish/subscribe notification for Web Services.

In this paper, we focus on these two standards that we believe are very important for the future of Web Services, reporting on development of our prototype work initially introduced in [1]. In order to investigate how they can support an existing, complex middleware project, we took myGrid as our example. MyGrid [15] is one of the leading EPSRC e-Science pilot projects that focus on the development of the open source Semantic Grid middleware for Bioinformatics supporting *in silico* experiments. MyGrid is building high-level services for data and application integration like resource discovery, distributed query processing and workflow enactment. Additional services are provided to support scientific method such as provenance management, change notification and personalization. As this project will finish shortly a proper exit strategy is essential. We believe that integration with WSRF and WSN are a very important part of it, increasing myGrid’s interoperability and taking advantage of deployed services and resources, in particular provided by NGS. We also want to improve myGrid’s scalability by providing a more distributed architecture.

In this paper we propose a novel model for integrating myGrid with WSRF/WSN. We report on our experience from the integration we made discussing its advantages and challenges. We envisage that our approach will converge with the peer-to-peer technology paving the way of Semantic Grid Middleware into mobile ad-hoc network (MANET) environment. The novelty of our approach comes from using peer-to-peer technology not only to provide self-organization and scalability but also to increase the reliability of storage and message delivery [16].

This paper is organized as follows. Section 2 presents an overview of myGrid architecture and proposes a model for integrating myGrid with WSRF and WSN. Section 3 evaluates our model, reports on the experiences form the integration we made and presents our peer-to-peer approach for utilizing mobile ad-hoc environments. Section 4 gives conclusions.

2 Model for Integrating MyGrid with WSRF/WSN Standards

This section defines our novel model for integrating myGrid with WSRF and WSN standards. We begin with describing the current myGrid's architecture. MyGrid is an EPSRC e-Science pilot project aiming at development of Open Source Semantic Grid middleware for Bioinformatics supporting *in silico* experiments. MyGrid is building high-level services for data and application integration like resource discovery, distributed query processing and workflow enactment. Additional services are provided to support scientific method such as provenance management, change notification and personalization [15].

In myGrid the *in silico* experiments are depicted as workflows comprising an interconnected set of inputs, outputs and processors, which are either local routines or Web Services [17]. The enactment of workflows, which are XML documents, is performed by the enactment engine. All the data including not only input, output and workflows but also connections between them and particular people together with the structure of involved institutions is stored in the myGrid Information Repository (MIR). All the entities within myGrid have their own unique IDs – Life Science ID (LSID) issued by the central authority. The real time inter-component event-driven notification is provided by the myGrid Notification Service (MNS) [18]. MyGrid adapts the service-oriented approach i.e. not only helps to integrate, possibly third party, Web Services, but all its components have Web Service interfaces.

Its key components, relevant for modeling with WS-Resources due to having notion of state include MIR entities, workflow enactments, and enactment services. MyGrid notification infrastructure can be adapted to the WS-Notification [14] specification. All the WS-Resource Qualified Endpoint References (WSRQER) of myGrid's WS-Resources will comprise the address of the appropriate Web Service and LSID.

2.1 MyGrid Data

Currently all MIR entities have types. Every type is associated with an XML schema of an XML document describing attributes of an entity. Entities are stored in the relational database, and accessed through the Web Service interface using document call format. The supported operations are presented in Table 1. We map the MIR entities on WS-Resources. Every single entity becomes a WS-Resource, every entity type - a resource type and every entity's attribute - a WS-Resourve's property. We use the XML schema for MIR information model as a source of types for resource properties documents for WSRF. The only change involves replacing the attributes containing LSIDs as references to other entities with their complete WSRQERs. The mapping between the current and WSRF operations are shown in Table 2. The WSRF data exchanges are simpler than their current equivalents, because the WS-Resource type is designated by the URL part of the WSRQER specified in the SOAP header and the LSID is passed as the ResoureProperty of the WSRQER.

As it can be clearly spotted, Table 1 contains two more entries than Table 2. The StoreEntity operation was omitted because WSRF does not provide any standardized message exchanges for creating WS-Resources. According to the WS-Resource Factory pattern [12] we keep the operations StoreEntity and GetEntityCollection in

Table 1. Operations currently supported by MIR

Name	Input	Output	Description
StoreEntity	XML document with all entity properties; entity type	Generated LSID	Stores a new entity into MIR and generates an LSID for it.
GetEntity	Sequence of one or more pairs of entity type and an LSID	Sequence of XML documents with all entity properties	Retrieves properties of documents of given LSIDs.
GetEntity Collection	Entity type; optionally a sequence of restrictions comprising an attribute name, a condition and a value	Sequence of entities' LSIDs	Retrieves a sequence of LSIDs for entities of a given type, according to given criteria (similar to the SQL *where* clause).
DeleteEntity	Sequence of one or more pairs of an entity type and an LSID	Status	Deletes one or more entity element from MIR.
UpdateEntity	Sequence of pairs of an XML document describing entity's properties and an LSID; entity type	Status	Updates properties of one or more entities of a given type (not all types are updatable).

Table 2. Mapping between current and new WSRF operations

Current operation	WSRF operation	Inputs	Outputs	Standard	Comments
GetEntity	GetResource Property	Property name	Property value	WS-Resource Properties	It is possible to specify properties to query.
	GetResource Properties	Sequence of property names	Sequence of property values		
Delete Entity	Delete	None	None	WS-Resource Lifetime	We can allow either immediate or delayed destruction.
	SetTermination Time	Termination time	New termination time and current time		
Update Entity	SetResourceProperties/UpdateResourceProperties	Sequence of pairs comprising a property name and a new value	None	WS-Resource Properties	We do not have to provide all attributes to update only a subset of them.

their current form in the Resource Factory and Discovery (RFAD) service, only changing their names to Create and Query and their output from LSIDs to WSRQERs. RFAD is responsible for creating and discovering WS-Resources and knows where all data resources are hosted. This approach allows every single resource to be stored on a different machine. In a large deployment, there could be a few public RFAD services running on specialized machines and several machines hosting WS-Resources having its own RFAD service, only for the use by public RFAD services, see Figure 1.

In our model, every WS-Resource replacing a MIR entity provides NotificationProducer interface as describe in WS-ResourceProperties [19] and WS-ResourceLifetime [20]. The notification process would be performed in a standard pattern as described in WS-BaseNotification [21], providing notifications about destruction of WS-Resources and changes of their properties.

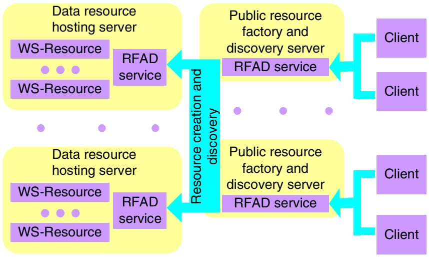
Data resource hosting server
WS-Resource
RFAD service
WS-Resource
Data resource hosting server
WS-Resource
RFAD service
WS-Resource
Resource creation and discovery
Public resource factory and discovery server
RFAD service
Public resource factory and discovery server
RFAD service
Client
Client
Client
Client

Fig. 1. Proposed data resources architecture

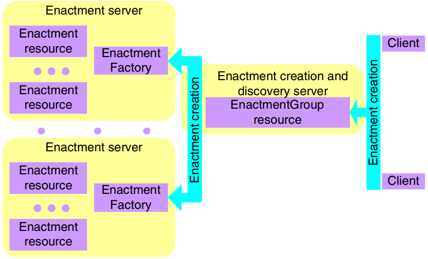

Fig. 2. Proposed workflow enactment architecture

2.2 Workflow Enactment

In myGrid workflows are depicted as XML documents written in SCUFL, which can be enacted by the Freefluo enactment engine. An important part of this process is collection of provenance, which comprises intermediary results and technical metadata including type of processors, status, start and end time and descriptions of the performed operations. This information can be useful in investigating how the erroneous or unexpected results have been achieved [17].

In the current release of myGrid Freefluo is integrated with Taverna and must be run locally from it, what is inconvenient for a user in case of long lasting enactments. It is planned to decouple them, by providing Freefluo as a Web Service hosted on a remote machine and manually putting its URL into the Taverna's configuration file. Using only one enactment server for a deployment can cause a bottleneck. Even if we use more, the current approach does not support any load balancing, because a user chooses one explicitly.

In our model shown in Figure 2 every enactment is virtualized as a WS-Resource (Enactment resource), which could be running on any of the available enactment servers. Its properties include: StartTime (the time stamp of the resource creation), Operation (reference to the Operation data resource [22] containing input parameters and workflow description in the SCUFL language), Status (statuses of the processors' invocations and intermediate results) and Topics (notification topics including only one topic ResourceTermination for enactment finish, error or cancellation). Enactment resource provides following operations: GetResourcePoropoerty and GetResourceProperties from WS-ResourceProperties for acquiring the resource's properties, Destroy from WS-ResourceLifetime for the immediate cancellation of the enactment, Subscribe from WS-BaseNotification for subscribing to notifications about the finish, error or cancelling of the enactment.

The EnactmentGroup resource acts as a WS-ResourceGroup [23] of all Enactment resources to allow their seamless discovery. It provides one custom operation Create, which will create an Enactment resource, taking as a parameter a WSRQER to the Operation data resource [22], referencing all the input data. It returns a WSRQER to the newly created Enactment resource. The EnactmentGroup selects an enactment server, on which the Enactment resource should be created, providing some load balancing. Internally the resource creation is done by invoking Enactment Factory

Web Service on the enactment server. Client can query at any time the status of the resource using WSRQER either from the Create operation or WS-ResourceGroup's Entry Properties. When the Enactment resource is terminated, either by the successful or erroneous end of the enactment or a client's cancellation, the OperationInstance data resource [22] is created, which holds all the provenance data.

2.3 Notification Infrastructure

In our model, any notification producer can manage its own subscriptions, so in a simple deployment no notification service would be necessary. However to provide more scalability notification brokers can be introduced. In WS-Notification sense, a broker implements both notification consumer and producer interfaces, so it can decouple notification consumer and producer forming even a very complex structure, where one broker receives notifications form another and passes it further [24]. The broker is subscribed to the subset of notification producer's topics and exposes them as his property. Depending on the particular deployment, a broker can have one of two objectives: aggregating topics managed by different WS-Resources to allow their seamless discovery or distributing the task of message delivery to increase its speed and decrease network congestion.

The role and the interface of a notification broker in WSRF is very different from the current myGrid Notification Service [18]. On one hand notification brokers are much more lightweight not supporting complex features like quality of service negotiation; on the other hand they are more generic allowing building complex structures.

3 Discussion

3.1 Overview of the WSRF/WSN Toolkits for MyGrid

We choose Apache WSRF (formerly Apollo) [25] for WSRF and Pubscribe (formerly Hermes) [26] for WSN both developed by The Apache Software Foundation. We choose Pubscribe because it is built on top of Apache WSRF and Apache WSRF as it is the only one offering all together Java API, dynamic creation of WS-Resources, callbacks for modification of WS-Resources and free, Open Source status. That is described in more detail in [1].

3.2 Evaluation of the Model

Our model changes the centralized myGrid Information Repository (MIR) into a set of cross-referenced WS-Resources, which can be hosted on different machines possibly administrated by different institutions. We introduce a distributed and scalable enactment infrastructure supporting load balancing. That all provides more flexible architecture, which can be easily extended after the system is deployed, so it can seamlessly grow with the increasing number of users, fully transparently to them.

It is possible because in WSRF WS-Resources are addressed by WS-Resource Qualified Endpoint References (WSQERs) containing URLs of Web Services describing where a Web Service part of a WS-Resource is located, so it is possible to host them at different network locations, in a transparent to a user way. Moving existing WS-Resources will be even easier, when the WS-RenewableReferences specification [9] becomes available. It will allow seamless updating WSQERs when they become outdated.

In our model, the notification infrastructure is more scalable, distributed and lightweight than originally in myGrid. The notification brokers can be used to form any topology either to aggregate topics published by various WS-Resources or to distribute the process of delivering messages. As every WS-Resource can manage its own subscribers, the notification brokers are optional and not required for simple deployments. Because the notification infrastructure is compliant with WSN, it is compatible with the third-party infrastructure for delivering messages and notification clients that are available now or in the future.

Introduction of WSRF and WSN provides one coherent and logical interface for operating on user's data, workflow enactment and notification infrastructure. That decreases the design effort needed to integrate various myGrid components or in future to integrate myGrid with third party tools, and UK's National Grid Infrastructure [4].

As the data model of WS-Resources is declared in WSDL descriptions of their Web Services, it becomes explicit for the clients. It makes evolution of the data model easier and the service consumers can even automatically adapt to its changes. It allows the gradual modification of the data model to fulfill changing users' requirements in fully transparent way even after deployment.

The WS-Resources implemented using Apache WSRF [25] and Pubscribe [26] have the form of servlets, which can be deployed to any servlet container like for example Jakarta Tomcat [27]. Their deployment comprises installing third party products (Java 2 Standard Edition, Jakarta Tomcat or an alternative servlet container, MySQL and LSID Launchpad), deployment of WAR files and modification of appropriate configuration files. Therefore the integration with WSRF/WSN does not make the deployment of myGrid [28] more demanding.

3.3 Experience from the Integration

Working on the implementation of our proposal we identified following challenges. To take the full advantage of WSRF the LSID references must be replaced with WSQER. That means making major changes in both the existing data model and the code. Using Apache WSRF, which currently is the only WSRF toolkit that fulfilled our requirements demands a lot of coding effort. It is mainly due to the extensive plumbing between the XMLbeans used to handle properties and a means of persistence, which in case of myGrid is the mySQL database supported with Hibernate [29]. There is also an important performance issue when using Apache WSRF or Java WS Core from GT4 to access data stored in a database. When a resource is created or accessed it becomes a Java object residing in the server's memory, until the service is shut down or the object is destroyed i.e. permanently removed. In case of huge databases it means a very inefficient use of the system's memory.

3.4 Towards Self-organizing Grids

Currently myGrid is meant to be deployed on the fixed network infrastructure. Such an infrastructure is not always available for a scientist doing for example a field research. Deploying myGrid on ad-hoc, possibly mobile, networks would mean facing some challenges. Firstly, the naming scheme depends on the DNS infrastructure, which must be pre-configured, preventing self-organization. Secondly, the state of WS-Resources is available only when a machine hosting it is on-line, what heavily limits the reliability in mobile ad-hoc network (MANET) environments.

As our future research we are going to address these issues. We plan to alleviate the need of DNS by using Distributed Hash Tables (DHTs) basing on the approach suggested in [30]. We also plan to increase the reliability in MANET environments by providing distributed caching of the WS-Resource (WSR) state [16]. The state data will be available even when the WSR itself is off-line.

This approach will lead to more scalable and reliable data storage offering easy deployment with minimal administration effort, due to lack of centralization, cooperative caching and automatic configuration. Similar approach can be used to simplify the deployment and increase the reliability of remote access to scientific devices monitored or controlled over the network. Such an approach can be particularly useful in areas where the research takes place, which offer unreliable or none pre-planned network infrastructure.

We also consider using overlay networks and application level routing to increase the reliability of the notification delivery. The WSN approach is vulnerable to a

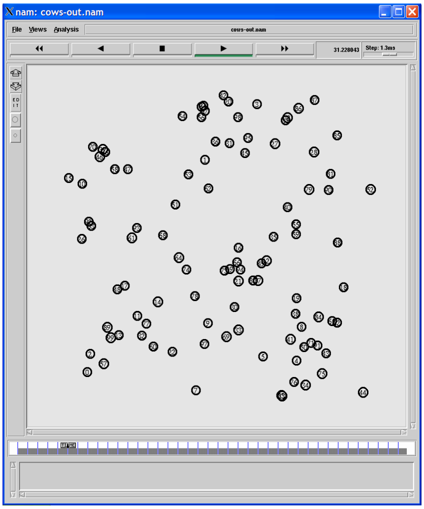

Fig. 3. Network simulation of 99 wireless Grid services and one user in 750m x 750m topology. User moves at the speed of 3-4 m/s starting from the bottom left corner and never stops. Services move at the speed of 1-4 m/s making stops for up to 20s.

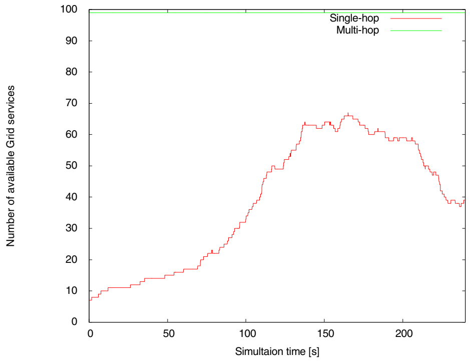

Fig. 4. Grid services available for the user – both within the radio range of her transceiver (single-hop) and over multiple hops

broker or notification producer going off-line. Using application level routing it is possible to distribute the responsibility for the message delivery among the forwarding nodes [16].

Currently to test our approach we are designing and developing a large scale simulation of application level DHTs using the *ns-2* network simulator [31] and the CanuMobiSim framework [32]. We want to examine the influence of dynamics of queries and the underlying network workload on the reliability of the storage and message delivery and performance of the queries. The example simulation is shown on Figure 3. Currently it comprises randomly generated movements of 99 Grid services and a user placed in a topology of 750m x 750m which in future will communicate using an application level DHT protocol. Services are wireless sensors placed on monitored animals. Figure 4 shows how many services were available for the user during the simulation. While the number of services within the range of user's wireless transceiver was limited, all the services were available in case of using multi-hop communication. That justifies using multi-hop peer-to-peer communication for such scenarios.

4 Conclusions

We proposed a novel model of integrating myGrid and WSRF that helps myGrid benefit in terms of scalability and interoperability. We also adapted the myGrid Notification Service, to the WSN specification. In this paper we define and evaluate our model and report on our experiences from the implementation of our proposal. Finally we argue that our approach will converge with the peer-to-peer technology to utilize mobile ad-hoc network environments. The myGrid data resources are already fully implemented and we are planning to implement the proposed workflow enactment architecture.

We proved that WSRF/WSN standards are well suited for the complex higher level middleware but applying these standards to the existing projects can lead to a significant coding effort. The approach presented here is universal and can be applied for standardization of other existing higher level middleware projects.

References

1. Wietrzyk, B., Radenkovic, M.: Semantic Life Science Middleware with Web Service Resource Framework. In: Proc. Fourth All Hands Meeting (2005)
2. Globus Toolkit 2.2.[Online]. Available: http://www.globus.org/gt2.2/
3. (2005) LHC Computing Grid Project (LCG) Home Page.[Online]. Available: http:// lcg.web.cern.ch/LCG/
4. Geddes, N. GOSC and NGS Status.[Online]. Available: http://www.gridpp.ac.uk/ gridpp12/GOSCNGS-status-Jan2005.ppt
5. Geddes, N., Richards, A. Grid Operations Support Centre (GOSC).[Online]. Available: http://www.nesc.ac.uk/talks/507/NESC_BBSRC_GOSC_241104.pdf
6. Loomis, C., Hahkala, J., Orellana, J. EGEE Middleware Architecture and Planning. EGEE [Online]. Available: https://edms.cern.ch/document/476451/
7. Atkinson, M., et al.: Web Service Grids: An Evolutionary Approach. OMII, (2004)

8. WSRF - The WS-Resource Framework.[Online]. Available: http://www.globus.org/wsrf/
9. Czajkowski, K., et al.: The WS-Resource Framework. Oasis, (2004)
10. Cerami, E.: Web Services Essentials. O'Reilly & Associates, Inc., Sebastopol, Canada (2002)
11. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American (2001)
12. Foster, I., et al.: Modeling Stateful Resources with Web Services. IBM, (2004)
13. Priol, T.: Objects, Components, Services for grid middleware: pros & cons. In: Proc. European Grid Conference (2005)
14. Graham, S., et al.: Publish-Subscribe Notification for Web services. IBM, (2004)
15. Stevens, R. D., Robinson, A. J., Goble, C. A.: myGrid: personalised bioinformatics on the information grid. Bioinformatics 19 (2003) i302-i304
16. Fall, K., et al.: Reliability in MANETs with Overlays. In: Proc. Peer-to-Peer Mobile Ad Hoc Networks - New Research Issues (2005)
17. Oinn, T., et al.: Taverna, lessons in creating a workflow environment for the life sciences. In: Proc. GGF10 (2004)
18. Krishna, A., Tan, V., Lawley, R., Miles, S., Moreau, L.: myGrid Notification Service. In: Proc. UK e-Science All Hands Meeting 2003 (2003)
19. Graham, S., Treadwell, J.: Web Services Resource Properties 1.2. Oasis, (2004)
20. Srinivasan, L., Banks, T.: Web Services Resource Lifetime 1.2. Oasis, (2004)
21. Graham, S., et al.: Web Services Base Notification. (2004)
22. Alpdemir, N., Ferris, J., Greenwood, M., Li, P., Sharman, N., Wroe, C.: The myGrid Information Model. University of Manchester, Design note Manchester (2004)
23. Maguire, T., Snelling, D.: Web Services Service Group 1.2. Oasis, (2004)
24. Graham, S., et al.: Web Services Brokered Notification. (2004)
25. Apache WSRF.[Online]. Available: http://ws.apache.org/wsrf/
26. Pubscribe.[Online]. Available: http://ws.apache.org/pubscribe/
27. (2005) The Jakarta Site - Apache Jakarta Tomcat.[Online]. Available: http://jakarta.apache.org/tomcat/
28. Nick Sharman, K. W., Tom Oinn, Kevin Glover, Nedim Alpdemir, Chris Wroe: The myGrid Installation Guide. (2005)
29. Bauer, C., King, G.: Hibernate in Action. Manning, Greenwich, USA (2004)
30. Ratnasamy, S., Francis, P., Handley, M., Karp, R., Shenker, S.: A Scalable Content-Addressable Network. In: Proc. SIGCOMM (2001)
31. The Network Simulator - ns-2.[Online]. Available: http://www.isi.edu/nsnam/ns/
32. Stepanov, I. CANU Mobility Simulation Environment (CanuMobiSim).[Online]. Available: http://canu.informatik.uni-stuttgart.de/mobisim