Mobile-to-Grid Middleware: An approach for breaching the divide between mobile and Grid environments

Umar Kalim, Hassan Jameel, Ali Sajjad, Sungyoung Lee

Department of Computer Engineering, Kyung Hee University
Sochen-ri, Giheung-eup, Yongin-si, Gyeonggi-do, 449-701, South Korea
{umar, hassan, ali, sylee}@oslab.khu.ac.kr

Abstract. In this paper we present an architecture of a middleware layer[1] that enables users of mobile devices to seamlessly and securely access distributed resources in a Grid. It lays the ground work for an application toolkit that addresses issues such as delegation of the job to the Grid service, interaction with heterogeneous mobile devices, support for offline processing, secure communication between the client and the middleware and presentation of results formatted in accordance with the device specification by outsourcing computationally intensive tasks with high storage and network bandwidth demands.

1 Introduction

Grid [1] computing is based on an open set of standards and protocols that enable coordinated resource sharing and problem solving in dynamic, multi-institutional virtual organizations [2]. With Grid computing, organizations can optimize computing and data resources by pooling them for large capacity workloads, share them across networks and enable collaboration. Though the concept of Grid computing is still evolving, yet there have been a number of achievements in the arena of scientific applications [3], [4], [5]. Extending this potential of the Grid to a wider audience, promises increase in productivity, particularly for users of mobile devices who are the prospective users of this technology.

Wireless environments and mobile devices bring different challenges when compared to wired networks and workstations. Although mobile devices promote mobile communication and flexible usage, yet they bring along problems such as unpredictable network quality, lower trust, limited resources (power, network bandwidth etc) and extended periods of disconnections [6]. If such resource limited mobile devices could access and utilize the Grid's resources then they could implicitly obtain results from resource intensive tasks never thought of before.

[1] This research work has been partially supported by Korea Ministry of Information and Communications' ITRC Program joint with Information and Communications University.

The classical client server approach assumes that the location of computational and storage resources is known [7]. This approach has to evolve in order to provide transparent access to distributed resources. Also considering the limitations [8] of interaction among mobile devices and grid nodes as well as the complexity of the Grid protocols, there is an emerging consensus [8] to develop a middleware layer which will mediate and manage access to distributed resources. Besides the clear separation among the key functionality, the introduction of a middleware layer can offer potential technical advantages. Among them are reduced communication cost, reduced network bandwidth usage, the possibility of using remote interfaces and the support for off-line computation.

In this paper we present an architecture for a middleware (Section 3), enabling heterogeneous mobile devices access to Grid services and implement an application toolkit that acts as a gateway to the Grid. This middleware provides support for seamless delegation of jobs to the Grid, secure communication between the client and the Grid (the same level of security that RSA provides by using smaller key sizes), offline processing, adaptation to network connectivity issues and presentation of results in a form that is in keeping with the resources available at the client device.

2 Mobile-to-Grid Middleware

Considering the constraints of mobile devices and how operations take place within a Grid environment [1], demands for computational as well as network bandwidth resources are intense. These demands for resources make it difficult for the developers to implement practical applications for the users of mobile devices. The problems [8] of mobile and wireless environments aggravate the dilemma. Hence there is a need for a middleware which could operate on behalf of the client and interact with the Grid services in such a manner that the client application is only required to participate primarily at only two instances; firstly before submitting the job and secondly when collecting the results so that the client application is not obliged to steer the process.

There are a number of areas that need to be addressed, namely, job delegation to the Grid, management of the request, handling of disconnections in the wireless environment, the security aspects between the client and the middleware layer, formalization of results depending upon the client's device specification and managing all this information efficiently etc.

To instantiate a job, the client must be able to instruct the Grid service. However, direct interaction with the Grid service results in exhausting demands for storage and computational resources on the client device. Similarly if the request to a Grid service requires continuous steering from the client, this puts strenuous demands on the computational resources and network bandwidth.

If the job submitted by the user is expected to complete in a long duration of time, conventionally the user is bound to maintain an active connection to obtain the results. Also if the network connection goes down (due to power loss, being out of range etc) the user would lose his connection and would have to start mediating with the Grid service from scratch which would result in loss of precious time and resources.

The Grid security infrastructure is based on public key scheme mainly deployed using the RSA algorithm [9]. However key sizes in the RSA scheme are large and thus computationally heavy on handheld devices such as PDA's, mobile phone's etc [10]. Also a user accessing the Grid services must use credentials which are compliant with the X.509 proxy credentials of GSI [9]. Transferring user credentials to access the Grid services without creating a security hazard and demanding relatively large computational and bandwidth resources is another issue that needs to be addressed.

The introduction of a middleware layer allows the user to reduce computational activities at the client device, minimize usage of network bandwidth, save battery power, permit offline processing etc. This is achieved as the middleware acts as a broker and interacts with the Grid on behalf of the client.

3 Detailed Architecture

The primary steps that may occur while the client device accesses the Grid services may be explained as follows. Firstly the client application discovers and connects with the middleware. Then the device, after authentication, submits its device specification along with the job request. The middleware then locates the relevant Grid service and after authorization forwards the request. The client may then request some status information (regarding the job or the service). If the client wishes to disconnect (and collect the results later), the middleware would facilitate a soft state registration and to which would later help in the reintegration. After disconnection all the requests are served locally (with the cached information). Requests that result in updates at the middleware service are logged for execution at reconnection. Upon reconnection pending instructions are executed and information updates at the client end are made to maintain consistency. Considering these steps the details of the modules involved (shown in Figure 1) are mentioned below.

3.1 Discovery Service

The discovery of the middleware by mobile devices is managed by employing a UDDI registry [11], [12]. Once the middleware service is deployed and registered, other applications/devices would be able to discover and invoke it using the API in the UDDI specification [11] which is defined in XML, wrapped in a SOAP [7] envelop and sent over HTTP.

3.2 Communication Interface with the Client Application

The interface advertised to the client application is the communication layer between the mobile device and the middleware. This layer enables the middleware to operate as a web service and communicate via the SOAP framework [13].

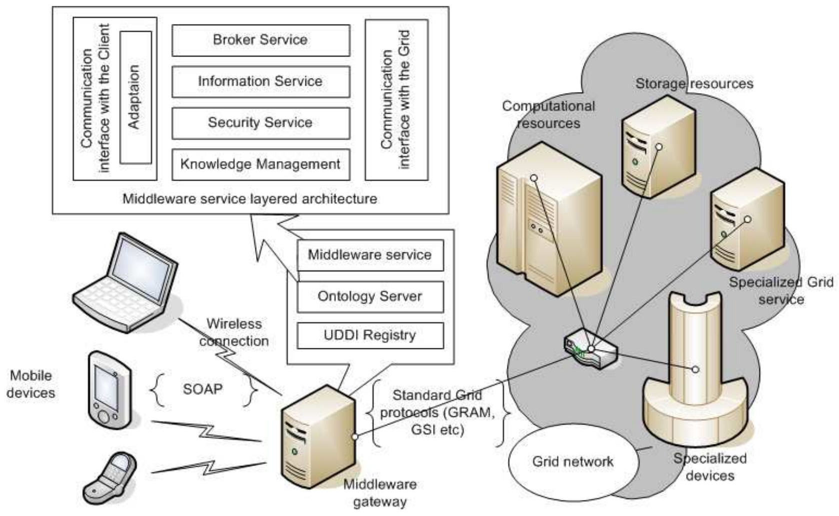

Fig. 1. Deployment model and the architecture

Adaption to Disconnected Operations The advertisement of the mobile-to-Grid middleware as a web service permits the development of the architecture in a manner that does not make it mandatory for the client application to remain connected to the middleware at all times while the request is being served.

We focus on providing software support for offline processing at the client device. For this we consider two kinds of disconnections; intentional disconnection, where the user decides to discontinue the wireless connection and unintentional disconnection, which might occur due to variation in bandwidth, noise, lack of power etc. This is made possible by pre-fetching information or meta-data only from the middleware service. This facilitates in locally serving the client application at the device. However, requests that would result in updates at the middleware service are logged (so that they may be executed upon reconnection).

To establish the mode of operation for the client application, a connection monitor is used to determine the network bandwidth and consequently the connection state (connected or disconnected). Moreover, during execution, checkpoints are maintained at the client and the middleware in order to optimize reintegration after disconnection.

3.3 Communication Interface with the Grid

The communication interface with the Grid provides access to the Grid services by creating wrappers for the API advertised by the Grid. These wrappers include standard Grid protocols such as GRAM [14], MDS [15], GSI [16] etc which are mandatory for any client application trying to communicate with the Grid services. This enables the middleware to communicate with the Grid, in order to accomplish the job assigned by the client.

3.4 Broker Service

The broker service deals with initiating the job request and steering it on behalf of the client application. Firstly the client application places a request for a job submission. After determining the availability of the Grid service and authorization of the client, the middleware downloads the code (from the mobile device or from a location specified by the client e.g. an FTP/web server). Once the code is available, the broker service submits a "createService" request on the GRAM's Master Managed Job Factory Service (via the wrapper) which is received by the Redirector [14]. The application code (controlled by the middleware) then interacts with the newly created instance of the service to accomplish the task. The rest of the process including creating a Virtual Host Environment (VHE) process and submitting the job to a scheduling system is done by GRAM. Subsequent requests by the client code to the broker service are redirected through the GRAM's Redirector.

The Status monitor (a subset of the broker service) interacts with GRAM's wrapper to submit FindServiceData requests in order to determine the status of the job. The Status monitor service then communicates with the Knowledge Management module to store the results. The mobile client may reconnect and ask for the (intermediate/final) results of its job from the status monitor service.

3.5 Knowledge Management

The knowledge management layer of the system is used to manage the relevant information regarding both the client and Grid applications and services. The main function of this layer is to connect the client and Grid seamlessly as well as to introduce the capability of taking intelligent decisions, based on the information available to the system.

Also, the results to be presented to the client are formatted (or scaled down) here considering the device profiles maintained at the ontology server.

3.6 Security

The Grid Security Infrastructure is based on public key scheme mainly deployed using the RSA algorithm [9]. However key sizes in the RSA scheme are large and thus computationally heavy on handheld devices such as PDA's, mobile phone's, smart phones etc [10]. We employ the Web Services Security Model [17] to provide secure mobile access to the Grid. This web services model supports multiple cryptographic technologies.

The Elliptic Curve Cryptography based public key scheme can be used in conjunction with Advanced Encryption Standard for access to the Grid. This provides the same level of security as RSA while the key sizes are a smaller [10].

Communication between the user and middleware is based on security policies specified in the user profile. According to this policy different levels of security can be used. e.g. some users might just require authentication, and need not want privacy or integrity of messages. Both ECC and AES have smaller key sizes as compared to RSA [10] which means faster computation, low memory, bandwidth and power consumption with high level of security. It may be noted that we emphasize on providing security on Application layer, which also gives us the flexibility to change the security mechanism if the need arises.

4 Information Service

This module interacts with the wrapper of the GLOBUS toolkit's API for information services (MDS [15]). It facilitates the client application by managing the process of determining which services and resources are available in the Grid (the description of the services as well as resource monitoring such as CPU load, free memory etc. Detailed information about grid nodes (which is made available by MDS) is also shared on explicit request by the client.

5 Multiple Instances of the Middleware Gateway

In case multiple instances of the middleware gateway are introduced for scalability, some problematic scenarios might arise. Consider a client that accesses the Grid via gateway M_1, but disconnects after submitting the job. If the client later reconnects at gateway M_2 and inquires about its job status, the system would be unable to respond if the middleware is not capable of sharing information with other instances. This can be achieved in the following manner.

We define a Middleware Directory Listing which maintains the ordered pairs (ID, URI) which will be used for the identification of the middleware instance. Also, we define an X service as a module of the middleware which facilitates the communication between any two middleware instances. After reintegration of the client at M_2, client C sends the ID of the middleware instance, where the job was submitted (i.e. M_1), to the X service. The X service determines that the ID is not that of M_2. The X service then checks the Middleware Directory Listing to find the URI corresponding to M_1. The X service then requests the job-ID submitted by C. Upon a successful response the X service communicates with the X service of M_1 using the URI retrieved. After mutual authentication, X-M_2 sends the job-ID along with the clients request for fetching the (intermediate/final) results to X-M_1. If the job is complete, the compiled results are forwarded to client. In case the job isn't complete yet, the client continues to interact with middleware service X-M_1 (where the job was submitted). Note that X-M_2 acts as a broker for communication between C and M_1. Also, if the C decides to disconnect and later reconnect at a third middleware instance M_3, then M_3 will act as a broker and communicate with M_1 on behalf of C. As all the processing of information is done at the middleware where the job was submitted, the other instances would only act as message forwarding agents.

6 Related Work

Various efforts have been made to solve the problem of mobile-to-Grid middleware. Signal [18] proposes a mobile proxy-based architecture that can execute jobs submitted to mobile devices, so in-effect making a grid of mobile devices. A proxy interacts with the Globus Toolkit's MDS to communicate resource availability in the nodes it represents. The proxy server and mobile device communicate via SOAP and authenticate each other via the generic security service (GSS) API. The proxy server analyzes code and checks for resource allocation through the monitoring and discovery service (MDS). After the proxy server determines resource availability, the adaptation middleware layer component in the server sends the job request to remote locations. Because of this distributed execution, the mobile device consumes little power and uses bandwidth effectively. Also their efforts are more inclined towards QoS issues such as management of allocated resources, support for QoS guarantees at application, middleware and network layer and support of resource and service discoveries based on QoS properties.

In [19] a mobile agent paradigm is used to develop a middleware to allow mobile users' access to the Grid and it focus's on providing this access transparently and keeping the mobile host connected to the service. Though they have to improve upon the system's security, fault tolerance and QoS, their architecture is sufficiently scalable. GridBlocks [20] builds a Grid application framework with standardized interfaces facilitating the creation of end user services. They advocate the use of propriety protocol communication protocol and state that SOAP usage on mobile devices maybe 2-3 times slower as compared to a proprietary protocol. For security, they are inclined towards the MIDP specification version 2 which includes security features on Transport layer.

7 Future Work

Some devices may not be able to efficiently process SOAP messages. Therefore we intend to provide multi-protocol support in order to extend the same facilities to such devices. However our first and foremost goal is to complete the implementation of the architecture using Java's support for web services and devices using 802.11b wireless interface cards. Our main focus will be on handling security, providing support for offline processing, presentation of results depending upon the device specification and interfacing with the Grid nodes. Along with this implementation we intend to validate our approach by using non-Markovian stochastic Petri nets to analyze the model.

8 Conclusion

In this paper we identified the potential of enabling mobile devices access to the Grid. We focused on providing solutions related to distributed computing in wireless environments, particularly when mobile devices intend to interact with grid services. An architecture for a middleware is presented which facilitates implicit interaction of mobile devices with grid services. This middleware is based on the web services communication paradigm. It handles secure communication between the client and the middleware service, provides software support for offline processing, manages the presentation of results to heterogeneous devices (i.e. considering the device specification) and deals with the delegation of job requests from the client to the Grid.

References

1. Foster, I., et al.: The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Proc. 4th Global Grid Forum, Open Grid Services Infrastructure working group, Global Grid Forum. (2002), http://www.gridforum.org/Meetings/ggf4/default.htm.
2. Foster, I. et al.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations, Int'l J. Supercomputer Applications, vol. 15, no. 3, (2001), pp.200-222.
3. EU DataGrid. http://eu-datagrid.web.cern.ch/eu-datagrid/
4. Particle Physics Data Grid. http://www.ppdg.net
5. BioGrid. http://www.biogrid.jp
6. Forman, G. and Zahorjan, J.: The Challenges of Mobile Computing, IEEE Computer, vol. 27, no. 4, (April 1994).
7. Tanenbaum, A. S., et. al.: Distributed Systems Principles and Paradigms. Prentice Hall, 1st edition, pp. 42, 43. ISBN 81-7808-789-8.
8. Wen, Y.: Mobile Grid. http://pompone.cs.ucsb.edu/ wenye/majorexam/writeup.pdf
9. Von Welch, Foster, I., Carl Kesselman, et al.: X.509 Proxy Certificates for dynamic delegation. Proceedings of the 3rd Annual PKI R&D Workshop, (2004).
10. Vipul Gupta, Sumit Gupta, et al.: Performance Analysis of Elliptic Curve Cryptography for SSL. Proceedings of ACM Workshop on Wireless Security - WiSe 2002 pp. 87-94, Atlanta, GA, USA, (September 2002), ACM Press.
11. Hoschek, H.: Web service discovery processing steps. http://www-itg.lbl.gov/ hoschek/publications/icwi2002.pdf
12. UDDI specification. www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm
13. SOAP Framework: W3C Simple Object Access Protocol ver 1.1, World Wide Web Consortium recommendation, (8 May 2000); www.w3.org/TR/SOAP/
14. GT3 GRAM Architecture, www-unix.globus.org/developer/gram-architecture.html
15. Czajkowski, K. et al.: Grid Information Services for Distributed Resource Sharing. Proceedings of the Tenth IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press, (August 2001)
16. Welch, V. Siebenlist, F. Foster, I. et al.: Security for grid services. HPDC, 2003.
17. Della-Libera Giovanni, et al., Security in a Web Services World; A Proposed Architecture and Roadmap, (2002), A joint security whitepaper from IBM Corporation and Microsoft Corporation. (April 7, 2002), Version 1.0
18. Hwang, J. Aravamudham, P.: Middleware Services for P2P Computing in Wireless Grid Networks. IEEE Internet Computing vol. 8, no. 4, (July/August 2004), pp. 40-46
19. Bruneo, D. et al.: Communication Paradigms for Mobile Grid Users. Proceedings 10th IEEE International Symposium in High-Performance Distributed Computing.
20. Gridblocks project (CERN) http://gridblocks.sourceforge.net/docs.htm