Enabling Computational Grids Using JXTA-technology

Henri Mikkonen
Helsinki University of Technology
Henri.Mikkonen@hut.fi

Abstract

The purpose of this paper is to investigate three existing JXTA-Grid -projects and compare them with "real Grids", basically meaning the Grids that use Globus Toolkit as the Grid middleware. Along with an overview of the JXTA, Globus, JNGI, OurGrid and Personal Power Plant technologies, the scope embraces on analyzing their suitability for different purposes. Security and usability issues are also taken into consideration. As the Grids based on the Globus toolkit are widely used in the academic research community, the analysis show that JXTA-Grids suit best to Home-Grids that enables utilization of computational power owned by large scale of digital home equipment. Globus software is far too heavy for this purpose as JXTA-Grids are too light nowadays to be used in a scientific Grids.

KEYWORDS: JXTA, Globus Toolkit, Peer-to-Peer, Grid computing

1 Introduction

Grid technologies have become common in the academic world for enabling computing intensive distributed services over the Internet. Their purpose is to harness idle computing, storage and network resources. One of the barriers before the Grid technologies can become common cross the Internet is in usability: the Grid middlewares are complex software and their installation may be too demanding for an average Internet user.

By contrast Peer-to-Peer (P2P) technologies have become common in the Internet. Some of those technologies allow more complex opportunities than just file-sharing, which is probably their most popular feature. One good example is JXTA, an open source P2P-technology introduced by Sun Microsystems. JXTA enables developers to create different kinds of distributed services and applications.

This paper investigates means to enable relatively small-scale computational Grids using JXTA technology. While related technologies and existing projects are briefly presented and reviewed, this paper concentrates on comparing these JXTA-Grids to the Grids which use the de-facto middleware, Globus Toolkit [11]. The focus is on comparing the installation and usage of the software without forgetting other important issues such as security.

Chapter 2 gives some background information about JXTA and the Grid, especially a middleware called Globus Toolkit. Chapter 3 continues by presenting three existing JXTA-based distributed computing projects while chapter 4 analyses them more deeply and compares them to the "real" Grids. The last chapter includes conclusions and proposals for future work.

2 Background

This section gives some background information needed for the rest of the paper. First subsection gives a brief introduction to JXTA before the next one takes the Grid and a middlware called Globus Toolkit into consideration.

2.1 JXTA

JXTA (JuXTApose) [16] was introduced in 2001 by Sun Microsystems. It is a P2P-infrastructure that aims at hiding the complexity of the underlying networks and making all kinds of devices interoperate. Its three design principles are: *interoperability* with other existing P2P systems and communities, *platform independence* (i.e. minimal requirements for software) and *ubiquity* (i.e. minimal requirements for hardware).

JXTA can be seen as a set of protocols defined by XML-messages. The fundamental concepts for the protocols are (1) *peer* (a participant, referred by a unique ID), (2) *peer group* (a collection of peers with its own membership policy, also referred by a unique ID), (3) *advertisement* (an XML-file describing all existing resource), (4) *message* (between peers) and (5) *pipe* (message transfer mechanism). The pipe can be a point-to-point pipe between two peers or a propagate pipe from one peer to two or more peers. Peers have one output and input pipe for the communication with the others.

The protocol set consists of six following items:

- *Peer Endpoint Protocol (PEP)*. A peer (A) uses PEP to determine the routing information to another peer (B). If no direct route from A to B exists or the old known route is unavailable, PEP constructs a new route with the information known by other peers.
- *Peer Rendezvous Protocol (RVP)*. Peers can be rendezvous peers[1] or just listeners of them within a peer group. RVP allows messages to be sent to all the listeners.
- *Peer Resolver Protocol (PRP)*. This protocol is used for sending generic resolver queries to other peers and receiving responses for them. PRP uses RVP for propagating messages.

[1] These peers enable the propagation service

- *Peer Information Protocol (PIP)* is used to obtain status information about other peers. PRP is used for the queries.
- *Peer Discovery Protocol (PDP)*. Peers uses PDP to publish advertisements (with PRP) and discover them from the other peers.
- *Pipe Binding Protocol (PBP)* is used to establish a virtual communication channel or pipe between two or more peers. PRP is used for sending pipe binding requests.

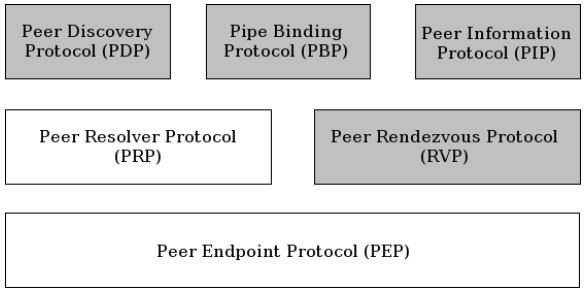

Figure 1: JXTA Protocol Hierarchy as described in [12].

The PEP and PRP are mandatory core protocols while the other four protocols are optional and a peer can implement only the ones it requires. The protocol hierarchy is illustrated in Figure 1: white boxes are mandatory core protocols and grey boxes optional standard services protocols [12].

2.2 The Grid

The term "Grid" refers to electricity power grids that consist of many different power plants providing electricity together to their users. A computational grid can be considered in the same way: several computers of different types (from a single workstation to a big cluster) are connected together and their computational power can be exploited by a grid-user.

Several computational grids with different technologies and for different purposes have been deployed over the years. A good example of national-scale Grid is Finnish M-Grid [18] as NorduGrid [19] can be mentioned as an example of a bit larger-scale Grid. However, Ian Foster [7] has defined the Grid with the following three-point checklist:

- A Grid coordinates resources that are not centrally managed.
- A Grid uses open, standard protocols and interfaces.
- A Grid provides different, non-trivial types of QoS (Quality of Service).

The second item of the list and especially the word "standard" is the most restrictive. As the standardization is assigned to the Global Grid Forum [10], their specifications and documents are used for defining the Grid [9].

Not all of the existing implementations that advertise themself as Grid middleware satisfy all the three items in the checlist. However, a middleware called Globus Tookit can be seen satisfying them and actually it can currently be seen as the de facto standard in Grid middlwares. It is next taken into consideration.

The Globus Toolkit

Ian Foster et al published Grid Security Architecture (GSA) in 1998 [8]. It is a general solution for the Grid security problem illustrated in Figure 2. As it can be seen, a physicist at site A makes an analysis request for some data at site C. Site C contacts resource broker at site D as the analysis program needs high computational power. Site D locates free resources at sites E and G where the computation is executed. During the computation these sites need to access some parameter values at site F. All the resources have their own local security mechanisms and policies, see Kerberos at site C and SSL at site D, for example.

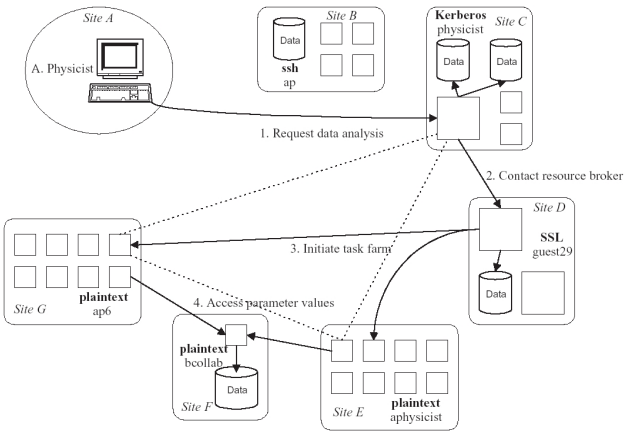

Figure 2: Grid security problem as described in [8].

The GSA's solution to fulfill the requirements is based on *proxies*: user proxy and resource proxy. The definitions 5.1 and 5.2 in [8] determine that:

- A user proxy is *a session manager process given permission to act on behalf of a user for a limited period of time*.
- A resource proxy is *an agent used to translate between interdomain security operations and local intradomain mechanisms*.

Grid Security Infrastructure (GSI) [2] is an implementation of GSA used in the Globus Toolkit. Users and resources are identified with X.509v3 certificates [13]. User proxy, on the other hand consists of short-live proxy certificate [23] and the corresponding private key. The proxy certificate is digitally signed by the owner user with the private key associated with his "real" certificate issued by a Certificate Authority (CA). User proxy represents the owner user in GSI and it can generate sub-proxys by digitally signing new proxy certificates. The signature can be generated without user intervention since the proxy is not encrypted. This is

compensated with proxy's short lifetime[2]. The whole certificate chain (from the owner user to the proxy) must be verified during the proxy validation.

Before submitting a job, users search for relevant resources using Monitoring and Discovery System (MDS) [3]. MDS has indexed information about available software, types of CPUs and free disk space for example. The current state of a resource can also be queried directly from the resource to get the most recent information. After appropriate resources have found, client program makes the scheduling (or use suitable services for doing it) and describes the job in the Globus Resource Specification Language (RSL). The RSL-file is then transmitted to the participating peers that run Globus Gatekeeper with Grid Resource Allocation and Management (GRAM) service. The file also includes a definition how the results are delivered to the submitter.

3 Existing projects

This section briefly describes three different JXTA-based distributed computing projects: JNGI, OurGrid and Personal Power Plant. Table 1 in the next page summarizes job submit, distribution and computation with overviewed technologies and the Globus Toolkit.

3.1 JNGI

JNGI [15] is a Java-project that defines a P2P-based distributed computing framework [24]. The fundamental building block of the framework is JXTA's peer group that is a collection of peers with its own membership policy. Peers are grouped to three following groups according to functionality: *monitor group*, *worker group* or *task dispatcher group*. A single node can belong to several groups of different types in the framework.

Peers that are joining to the framework first discover a monitor group. According to current situation, the monitor group redirects the peer to one worker group that is associated with one task dispatcher group.

Every job has an unique ID that is created by a submitter worker. A worker contacts a task dispatcher with its available resource information along with possible jobs that it wants to be performed by the other peers. The task dispatcher distributes not only the job for different tasks among the worker group but also definition, current state and possible results of the job to other task dispatchers among the task dispatcher group. This enables the job submitter to query any task dispatcher within the group for information about the job if the task dispatcher used for submitting the job has left the framework. The job submitter need to query for the results between some intervals as there is no lasting connection between submitter and dispatcher. Also, task dispatchers do not notify the submitters without a retrieval request from them.

[2]By default, proxy is valid for 12 hours. It can be defined in more detail for corresponding to requirements

3.2 OurGrid

OurGrid [1] aims at building the simplest form of an exchanged based economic model for Grids. The assumption is that there are atleast two peers in its environment that are willing to share their resources in order to obtain more resources from the other/others later on. The more peer has shared resources earlier, the higher rank it has to use resources shared by other peers.

The OurGrid resource sharing protocol is used for communication, access gaining and consuming, and providing resources in a P2P network. The system uses JXTA protocols for basic P2P-functionalities like peer discovery and message broadcasting. The three essential parts in OurGrid are: *client*, *consumer* and *provider*, every peer having both consumer and provider modules.

Client program first determines the characteristics required for its job. It includes them to a request message that is sent to a consumer module. Consumer module broadcast the request to all provider modules in the P2P-network. Consumer waits a certain time for replies from suitable and willing providers. The list of them is then sent back to client program to schedule tasks onto them. Scheduled tasks are transmitted to providers via consumer module, who waits for the task results from each participant provider. The report containing job result (i.e. all the task results) is finally sent to the client program.

3.3 Personal Power Plant

Personal Power Plant (P3) is a JXTA-based middleware that advertises itself to enable mutual and equal transfer of computational resources [20]. Peers running P3 middlware form an own base peer group that is a subgroup of always existing JXTA's base group.

A peer can run *host* daemon to share resources and/or *controller* tool to access resources shared by the others. To distribute a job to resource providers, a user first creates a distinct peer group for the job, *job group*, with the controller tool. After joining the group itself, controller publishes an advertisement of his job within the group. It includes a description of the job.

Peers running a host daemon discover the advertisements of job groups made by controllers. According to their policy, they decide whether they want to join a job group and contribute to processing the job. Instead of pre-defined policy, a Graphical User Interface (GUI) can be used for selecting jobs, but it requires user intervention. However, if a host decide to join, it discover and obtain a Java Archive (JAR) from the controller. The archive contains compiled Java application code.

Currently, the application can choose from three different types of parallel programming libraries: object passing, message passing or master-worker library. In addition to the job management subsystem and three parallel programming libraries, P3 also contains a WWW-based job monitor.

	Job submit	Resource discovery	Computation	Job results
JNGI	Worker	Task dispatcher	Worker	Must be queried
OurGrid	Client	Consumer	Provider	Sent by the consumer
P3	Controller	Controller	Host	Sent by hosts
Globus	Client	MDS	GRAM	Several options

Table 1: Summary of submit, distribution and computation of jobs in JNGI, OurGrid, P3 and the Globus Toolkit

4 Analysis

This section compares JNGI, OurGrid and P3 with the Globus Toolkit. First, their suitability to different scenarios is analyzed. The section also takes some security and usability issues into consideration.

4.1 Suitability

The three-point checklist described in section 2.2 defines among other things that a Grid provides different, non-trivial types of QoS. On the other hand, one of the assumptions for OurGrid environment is that "the applications that will be executed using OurGrid need no QoS guarantees" [1]. This assumption seems to subsist in the two other JXTA-Grids too, since they have not taken QoS issues into consideration. This is quite natural as P2P-networks usually consist of "normal PCs" that are turned on, and more imporantly, the P2P-application is running only when they are being used. In practise, QoS cannot be guaranteed, because availability of peers cannot be precisely predicted.

Globus Grids, on the other hand, usually consist of Linux workstations and clusters that are always turned on. In addition to the current workload, their availability depends on network connections, whose QoS can be guaranteed. Certain CPU-time and other resources can also be guaranteed during the job scheduling and thus the QoS can be guaranteed in its entirety.

JXTA-Grids seem to fit best to solving problems that can be divided into a bag of tasks. This means that as every task is independent from the others, they can be easily distributed among different peers which can process the task by itself. P3 is the only JXTA-Grid middleware that enables parallel processing in a way that computing peers can interact during the computation. In the Globus Toolkit, this is a matter of course by utilizing libraries like MPI (Message Passing Interface) provided by the operating system.

Currently Globus is only implemented to Linux operating system. Microsoft Windows is however more popular and even though some interoperability with some Windows Grid implementations (e.g. Condor-G [22]) and Globus do exist, platform-dependency reduces the amount of potential use scenarios. JXTA-Grids are all implemented in Java, which is basically platform independent. In comparison with Globus, they are also much lighter software that also raises the amount of potential devices where the software could be installed.

4.2 Security

As it was described in section 2.2, security in Globus is based on X.509 certificates. Certificates are widely used but on the other hand, they are quite a demanding alternative from a user's point of view. Some users may not have a proper understanding over the protection of a private key associated with a certificate. By default, private key is stored in a password-protected file that is also protected by the filesystem. If the file is exposed, it is vulnerable to dictionary attacks, as the amount of incorrect attempts cannot be limited. Don Davis [6] for example has argued that safe private key management is far too demanding for an average Internet user.

The security of user proxies can also be called into question even though they have limited lifetime. They can be set in a way that they can only be used in certain resources. However, that is not always possible as some resources may dynamically use other resources which cannot be predicted. This leads users to delegate their rights to be used anywhere with a single proxy. Any misbehaving resource that obtains the proxy can thus impersonate as the user that the proxy represents.

By default, JXTA-Grids do not seem to have comparable authentication and authorization architectures. Identification is just based on JXTA's unique peer id. Actually only P3 and OurGrid could take advantage of stronger authentication since resource providers cannot select its customers in JNGI. However, JXTA do not prevent the usage of certificates, for example. They would basically be the only reasonable authentication solution in JXTA-Grids too because users cannot have distinct user accounts to every resource providing peer. Depending on the nature of the P2P-network and their distributed jobs, strong authentication or other security features may not even be necessary.

4.3 Usability

One of the main problems that prevents Grid-middlewares like Globus to become common among average Internet users is the complex start-up process. Even if a user just wanted to use, not share resources in Globus-based Grid, he would first have to have a user account in every resource he would want to use. In addition to that, he would not only need to have a certificate signed by a CA (Certificate Authority) but also install specific software to generate a certain digital identifier (user proxy, see 4.2) for accessing resources. Users that want to share their resources need to install and configure more space-consuming software packages. All this may be straightforward for an expert user, but insuperable for an average Internet user.

JXTA-based middlewares also requires the installation of some software package, but the configuration process is essentially simpler and thus possible for wider scale of users. However, as it was said in the earlier subsection, JXTA enables different security solutions but raising the security level bring more configuration parameters and thus complicate the start-up process.

Globus-resources can be accessed with command prompt tools, GUIs, portals (e.g. GridBlocks [17]) and APIs (e.g. CoG Kit [5], currently implemented in Java and Python). At all events, user must supply his user proxy to the used method to enable required authentication and authorization features.

Current JXTA-Grids do not have such requirements for identification, but on the other hand they can only be utilized with the client program or tool. Even though P3 has a WWW interface for monitoring the jobs, job submit must still be done by a controller tool. Usage of such a tool can be made simple enough for average users but the peer id that in this case also means user id is confined to the used device. Portals, on the other hand, would enable users to access their personal services with any device that can access WWW pages. For example, a user could submit some long-lasting job with his personal workstation but retrieve the results directly to his PDA via portal.

5 Conclusions

Computational capacity is growing fast as more and more home equipment includes microchips. While wireless LANs keep becoming more general, they offer a way for all digital equipment to interact in an effective way. As one of the JXTA's design principles is ubiquity, it should be able to be used with any kind of devices that allows additional software to be installed. JXTA-Grids would then offer a way to harness all these digital equipment and build "Home-Grids". This concept is not yet specified, but Home-Grids would provide more computational capacity without investments in new hardware.

Grids based on Globus Toolkit, on the other hand, are widely used and have appeared to be powerful in academic scientific world. Currently, they are however too complex to be used by an average Internet user. In fact, most of their features would not even be needed in Home-Grids even though they are essential in scientific Grids. This is natural, because Globus aims at fulfilling the three-point checklist for the Grid that was described in section 2.2. The third item of the list, QoS guarantees, is not an essential feature for Home-Grids. In fact, that item could be replaced with a requirement for ubiquity and then the becoming list could be a starting point for constructing a checklist for the Home-Grid.

5.1 Future work

A Home-Grid concept requires much more investigation and analysis than the brief overview in this paper. Along with the detailed requirement analysis, a prototype implementation for an operative Home-Grid is proposed for future work. Some features of the overviewed JXTA-Grid projects can certainly be exploited in the implementation.

The next step from the definition of Home-Grid concept is proposed to investigate possiblities to integrate it with existing Grid-technologies, like Globus Toolkit. Actually some work for JXTA and Globus integration has already been started, for example a Jxta-Grid project "to help define a joint technology / framework that uses both P2P and Parallel computing technologies" [14], a Globus Alliance's CoG Kit project to enable "A JAXTA-based Grid Broker Service" [4] and a mechanism that allows Globus job submission to JXTA-networks [21].

References

[1] N. Andrade, W. Cirne, F. V. Brasileiro, and P. Roisenberg. Ourgrid: An approach to easily assemble grids with equitable resource sharing. In D. G. Feitelson, L. Rudolph, and U. Schwiegelshohn, editors, *JSSPP*, volume 2862 of *Lecture Notes in Computer Science*, pages 61–86. Springer, 2003.

[2] R. Butler, V. Velch, D. Engert, I. Foster, S. Tuecke, J. Volmer, and C. Kesselman. A National-Scale Authentication Infrastructure. *IEEE Computer*, 33(12):60–66, 2000.

[3] B. Clifford. Globus Monitoring and Discovery (SLIDES). In *The GlobusWORLD Event, Session 9c*, Boston, Massachusetts, Feb. 2005. Online: www.globusworld.org/2005Slides/Session%209c.pdf.

[4] Cog Kit JXTA Project's homepage: A JAXTA-based Grid Broker Service. Online: http://www-unix.globus.org/cog/projects/jxta/. Referenced on 9th March 2005.

[5] CoG kit homepage. http://www-unix.globus.org/cog/. Referenced on 9th March 2005.

[6] D. Davis. Compliance Defects in Public-Key Cryptography. In *Sixth USENIX Security Symposium Proceedings*, pages 171–178, San Jose, CA, July 1996.

[7] I. Foster. What is the Grid? A Three Point Checklist. *Grid Today*, 1(6), July 2002. http://www.gridtoday.com/02/0722/100136.html.

[8] I. Foster, C. Kesselman, G. Tsudik, and S. Tuecke. A Security Architecture for Computational Grids. In *5th ACM Conference on Computer and Communications Security*, pages 83–92, 1998.

[9] W. Gentzsch. The Grid: Defining the Future of the Internet. *Grid Today*, 3(33–35), Aug. 2004. http://www.gridtoday.com/04/0816/103659.html.

[10] Global Grid Forum. http://www.gridforum.org/. Referenced on 9th March 2005.

[11] The Globus Toolkit. http://www.globus.org/toolkit/. Referenced on 9th March 2005.

[12] H. Helin. JXTA. Lecture slides of 582449: Peer-to-Peer Computing course in University of Helsinki, Finland, Fall 2004. Online:

www.cs.helsinki.fi/u/hhelin/opetus/p2p03/slides/jxta-lecture.pdf.

[13] R. Housley, T. Polk, W. Ford, and D. Solo. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 3280, April 2002.

[14] Jxta-Grid0 Project's homepage. http://jxta-grid.jxta.org/. Referenced on 9th March 2005.

[15] JNGI Project's homepage. http://jngi.jxta.org/. Referenced on 9th March 2005.

[16] JXTA Project's homepage. http://www.jxta.org/. Referenced on 9th March 2005.

[17] J. Karppinen, M. Niinimaki, J. White, and T. Niemi. Gridblocks – web portal and client for distributed computing. In *Proceedings of ICEIS 2004, 6th International Conference on Enterprise Information Systems*, Porto, Portugal, Apr. 2004.

[18] M-Grid: Material Sciences National Grid Infrastructure. http://www.csc.fi/proj/mgrid/. Referenced on 9th March 2005.

[19] The NorduGrid Collaboration. http://www.nordugrid.org/. Referenced on 9th March 2005.

[20] K. Shudo, Y. Tanaka, and S. Sekiguchi. P3: P2P-based Middleware Enabling Transfer and Aggregation of Computational Resources. In *5th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid05)*, Cardiff, United Kingdom, May 2005. To appear.

[21] M. Silander, M. Pitkanen, and M. Niinimaki. Running Grid Jobs in Peer-to-Peer Networks. In *5th IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid05)*, Cardiff, United Kingdom, May 2005. To appear.

[22] D. Thain, T. Tannenbaum, and M. Livny. Distributed Computing in Practice: The Condor Experience. *Concurrency and Computation: Practice and Experience*, 17(2–4):323–356, 2005.

[23] S. Tuecke, V. Welch, D. Engert, L. Pearlman, and M. Thompson. Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. RFC 3820, June 2004.

[24] J. Verbeke, N. Nadgir, G. Ruetsch, and I. Sharapov. Framework for peer-to-peer distributed computing in a heterogeneous, decentralized environment. In M. Parashar, editor, *GRID*, volume 2536 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2002.