CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2001; volume (number): 000–000

Features of the Java Commodity Grid Kit

Gregor von Laszewski, Jarek Gawor, Peter Lane, Nell Rehn, and Mike Russell
Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, 60439, U.S.A.

Corresponding Author: Gregor von Laszewski, gregor@mcs.anl.gov, phone: + 630 378 0837 fax: + 630 252 5986

Journals Production Department, John Wiley & Sons, Ltd.,
Chichester, West Sussex, PO19 1UD, U.K.

SUMMARY
In this paper we report on the features of the Java Commodity Grid Kit. The Java CoG Kit provides middleware for accessing Grid functionality from the Java framework. Java CoG Kit middleware is general enough to design a variety of advanced Grid applications with quite different user requirements. Access to the Grid is established via Globus Toolkit protocols, allowing the Java CoG Kit to communicate also with the services distributed as part of the C Globus Toolkit reference implementation. Thus, the Java CoG Kit provides Grid developers with the ability to utilize the Grid, as well as numerous additional libraries and frameworks developed by the Java community to enable network, Internet, enterprise, and peer-to-peer computing. A variety of projects have successfully used the client libraries of the Java CoG Kit to access Grids driven by the C Globus Toolkit software. In this paper we also report on the efforts to develop serverside Java CoG Kit components. As part of this research we have implemented a prototype pure Java resource management system that enables one to run Grid jobs on platforms on which a Java virtual machine is supported, including Windows NT machines.

KEY WORDS: Grid Computing, Globus Toolkit, Peer-to-Peer Information Service, Portal, Java

1. INTRODUCTION

Over the past few years, international groups have initiated research in the area of computational Grids to provide scientists with new modalities required by state-of-the-art scientific application domains. The term “Grid” emerged in the past decade and denotes an integrated distributed computing infrastructure for advanced science and engineering applications. The Grid concept is based on coordinated resource sharing and problem solving in dynamic multi-institutional virtual organizations [12]. Besides access to a diverse set of remote resources among different organizations, Grid computing is required to facilitate highly flexible sharing relationships among them, ranging from client-server to peer-to-peer computing. High-end applications using such

Received 23 July 2001
Revised 23 July 2001

computational Grids include data-, compute-, and network-intensive applications. Examples range from nanomaterials, structural biology, and chemical engineering to high-energy physics and astrophysics [11, 25, 38]. Many of these applications require the coordinated use of real-time large-scale instrument control and experiment handling, distributed data sharing among hundreds or even thousands of scientists, petabyte distributed storage facilities, parameter studies, and teraflops of compute power. All of these applications have in common a complex infrastructure that is difficult to manage. Researchers therefore have been developing services, and portals using these services, to facilitate these complex environments and to hide much of the complexity of the underlying infrastructure. The Globus project provides a small set of useful services, including authentication, remote access to resources, and information services to discover and query such remote resource. Unfortunately, these services may not be compatible with the commodity technologies used for application development by the software engineers and scientists. Instead, users prefer accessing the Grid from a higher abstraction level than the C Globus Toolkit provides. This includes, for example, Java [34], CORBA [36], Python and IDL [38].

To overcome these difficulties, the Commodity Grid project is creating what we call *Co*mmodity *G*rid Toolkits (CoG Kits) that define mappings and interfaces between Grid services and particular commodity frameworks. Technologies and frameworks of interest currently include Java, Python, CORBA, Perl, and Web services.

In this paper we concentrate on the features of the Java CoG Kit. It provides convenient access to Grid functionality through client-side and a limited set of server-side classes and components, all implemented in pure Java. Additionally, the Java CoG Kit provides unique advanced services currently not available in the C Globus Toolkit reference implementation, services that are essential for designing computing portals.

2. WHY JAVA FOR GRID COMPUTING?

Besides the observations made above, several additional factors make Java a good choice for Grid computing: Java is a modern, object-oriented programming language that makes software engineering of large-scale distributed systems much easier. Thus, it is well suited as a basis for an interoperability framework and expose Grid functionality at a higher abstraction level than is possible with the C Globus Toolkit. Numerous factors such as platform compatibility, a rich set of class libraries, and related

frameworks make Grid programming easier. Such libraries and frameworks include JAAS [19], JINI [9], JXTA [16], JNDI [24], JSP [17], EJBs [20], and CORBA/IIOP [27]. We have depicted in Figure 1 a small subset of the Java technology that can be used to support various levels of the Grid architecture [12]. The Java CoG Kit builds a bridge between existing Grid technologies and the Java framework while enabling each to use the other's services to develop Grid services based on Java technology and to expose higher-level frameworks to the Grid community while providing interoperability [30].

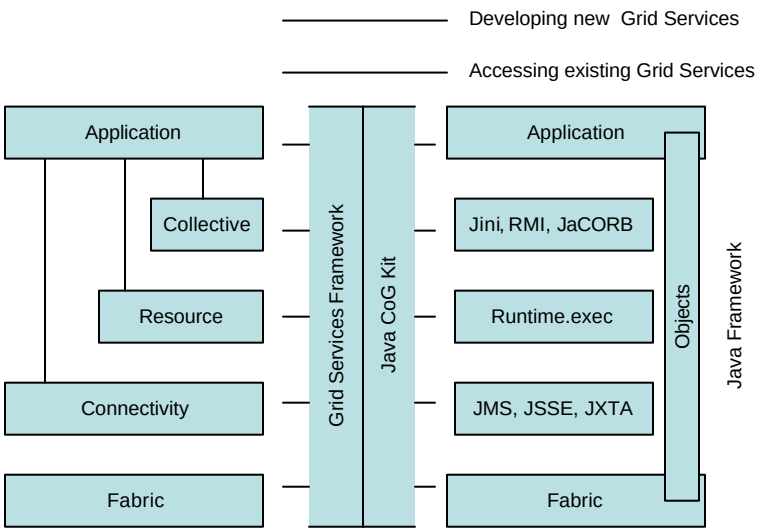

Figure 1: The Java CoG Kit allows to access Grid Services from the Java Framework and enable application and Grid developers to use a higher level of abstraction for Grid service development.

Furthermore, Java is quite well situated as a development framework for Web applications. Accessing technologies such as XML [18], XML schema [3], SOAP [4], and WSDL [7] will become increasingly important for the Grid community. We are currently investigating these and other technologies for Grid computing as part of the Commodity Grid projects to prototype a new generation of Grid services.

Because of these advantages, Java has received considerable attention by the Grid community in the area of application integration and portal development. For example, the EU DataGrid effort recently defined Java, in addition to C, as one of their target implementation languages. Additional motivation for choosing Java for Grid computing can be found in [15].

3. OVERVIEW OF THE JAVA COG KIT

The Java CoG Kit is general enough to be used in the design of a variety of advanced Grid applications with quite different user requirements. The Java CoG Kit integrates Java and Grid components and services within one toolkit, as a *bag* of services and components. In general, each developer chooses the components, services, and classes that ultimately support his development requirements. The goal of the Java CoG Kit is to enable Grid developers to use much of the Globus Toolkit functionality and to have access to the numerous additional libraries and frameworks developed by the Java community, allowing network, Internet, enterprise, and peer-to-peer computing. Thus, we have decided to expose Grid functionality as part of the Java CoG Kit [34] while implementing Globus Toolkit protocols and functionality in pure Java. Based on the advanced features of Java, the Java CoG Kit does not provide a simple one to one mapping between the C Globus Toolkit and Java CoG Kit API. Since we strive to be only protocol compliant, however, it is possible to access the more sophisticated Java events and exception handling rather than using the archaic C-based methods. Based on the protocol portability, the following Grid services can be accessed through the client-side Java CoG Kit:

- An information service compatible to the Globus Toolkit Metacomputing Directory Service (MDS) [10] implemented with JNDI.
- A security infrastructure compatible to the Globus Toolkit Grid Security Infrastructure (GSI) implemented with the iaik security library [5].
- A data transfer compatible with a subset of the Globus Toolkit GridFTP and/or GSIFTP.
- Resource management and job submission to the Globus Globus Resource Access Manager (GRAM) [8].
- Quality of service compatible with the Globus Toolkit General-purpose Architecture for Reservation and Allocation (GARA) [13].
- A certificate store based on the myProxy server [22].

Additionally, the Java CoG Kit contains a set of command-line scripts that provide convenient access to Globus Toolkit-enabled production Grids from the client. This includes support for MS Windows batch files, which are not supported by the C Globus Toolkit. Additionally, we provide as part of an enhanced version of globusrun the submission of multiple GRAM jobs. Other useful services include the ability to access Java smart card or iButton technology [1] to perform secure authentication with a possible multiple credential store on a smart card or an iButton.

Besides these elementary Grid services and tools, several other features and services currently not provided by the C Globus Toolkit are included explicitly or implicitly within the Java CoG Kit. We provide services that use the Grid Object Specification (GOS) as defined by the Grid Information Services Working group of the Global Grid Forum [33]. A binding from GOS to the RFC2252 [37] also is provided that can be used to create schemas for the Metacomputing Directory Service [10] of the Globus Project.

The Java Webstart [2] and signed *applet* technologies provide developers with an advanced service to simplify code startup, code distribution, and code update. Java Webstart allows the easy distribution of the code as part of downloadable jar files that are installed locally on a machine through a browser or an application interface. We demonstrated the use of Webstart within the Java CoG Kit by installing sophisticated Graphical User Interface (GUI) applications on client machines. Component frameworks, such as JavaBeans, and the availability of commercial *integrated development environments* (IDEs), enables the Grid developer to use IDEs as part of rapid Grid prototyping while enabling code reuse in the attempt to reduce development costs.

4. PORTAL DEVELOPMENT WITH THE JAVA COG KIT

We have shown recently (at SC2001 in Denver) that our proposed framework for developing collaborative scientific problem solving environments and portals, based on this combined strength of the Java and the Grid technologies, is well supported by the Java CoG Kit. In the past, we proposed portal architectures similar to the one depicted in Figure 2, in which the Java CoG Kit is used as an elementary middleware to integrate Grid services within portals and applications.

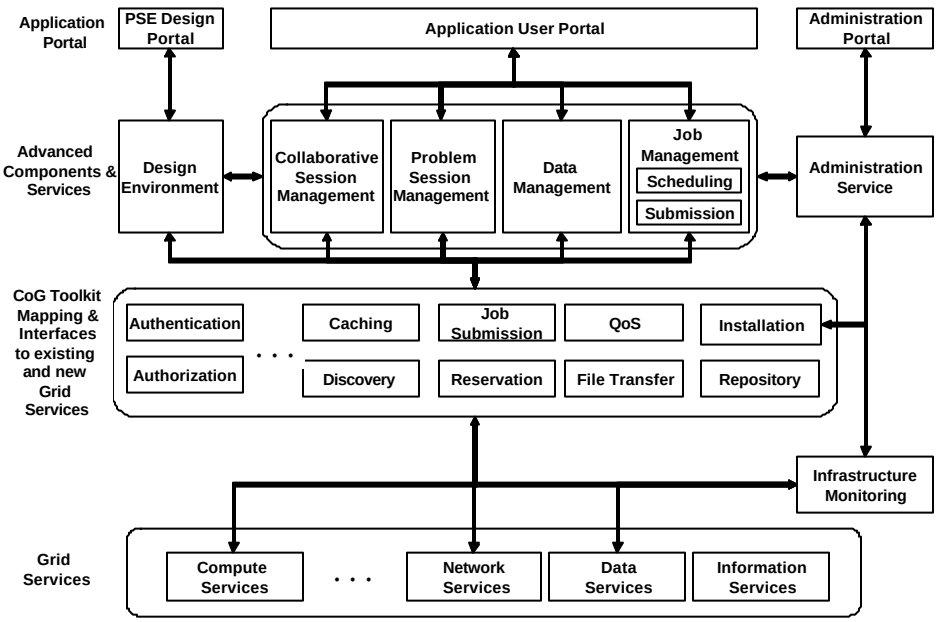

Figure 2: The architecture of an application portal that can be developed with current and future CoG Kit components. Additional components may also be provided by other projects.

We expect that services currently considered as advanced services will be integrated in future releases within the Java CoG Kit or as extension packages. Additionally, it is possible to implement several core Grid services, currently provided as part of the C Globus Toolkit, in pure Java while exposing the service through the newly proposed Web Services Framework by W3C. This has been demonstrated for file transfer and for job execution. The availability of these services and protocol handlers in pure Java will make future portal development and the integration with existing production Grid far easier. We have provided example programs using advanced GUI components in Java as part of the Java CoG Kit. These examples include a setup component for the Java CoG Kit, a form-based job submission component, a drag-and-drop-based submission component similar to a Windows desktop, an information service browser and search queries. We hope that the community will contribute more components so that the usefulness of the Java CoG Kit will increase.

5. CURRENT AND FUTURE WORK

Most recently, we have prototyped a GRAM service in pure Java. We have demonstrated that it is possible to install a personalised GRAM service on a local machine in 2.5 seconds, in addition to the time required to submit the (at the moment unoptimized and large) jar files of 2 Mbytes. One unique feature of this service is the integration with an information service allowing querying the status of a resource on a peer-to-peer basis. To reflect its dual purpose, we call this service InfoGram [32]. This service also includes a logging mechanism to allow for better fault-tolerant behavior. Information caching within the server as well as the logging capabilities enables a fast reply upon client requests. We are currently improving this service and strive for integration within the next release of the Globus Toolkit.

To allow simple access from clients, we extended the Globus Toolkit Resources Specification Language (RSL) with a number of new tags. The *info tag* is followed by a keyword that can be set in a configuration file, defines a mapping between the keyword and the command to be executed. If it is set to (info=all), all commands are executed. Commands can be selectively queried while concatenating multiple info tag queries, for example, (info=Memory)(info=CPU) return information about the memory usage and the CPU usage. The *response tag* defines the behaviour with respect to the information caching. Thus, with (response=immediate) the commands associated with the info tag are executed immediately regardless of the time to live. This will also update the cached values. Using (response=cached), which is default, will return the information from the cache value if it is valid; otherwise it will update the cache first. Using (response=last) will return the value stored last in the cache without updating it. The *format tag* defines the format in which the information is returned. The supported formats are LDIF and XML. Nevertheless, it is straightforward to support other formats such as DSML. The *filter tag* specifies a serverside filter to be invoked before the information is returned to the client. Such a filter is indicated with the filter keyword in the configuration file. We are adding a tag that specifies a *timeout* value and a behavior via an *action* tag upon reaching this timeout. As an example, the RSL (executable=command)(timeout=1000)(action=cancel) would cancel the command specified through the RSL, while (action=exception) would throw an exception if the command has not completed is execution.

This peer-to-peer information and job submission service supports the simplification of the architecture bound to the delivery of an integrated job submission and information service. Querying the information is handled by clients much as the execution of jobs. Moreover, this peer-to-peer information service can easily be integrated into the Globus Toolkit MDS information service architecture. Other advantages are the immediate availability of an information service on the Windows operating system, thus reducing the porting and maintenance effort dramatically. An additional benefit of the combination of the job submission service and the information service is the use of the same code for authentication. Other benefits will be introduced while providing authorization mechanisms as part of this service, which can (ideally) be supported by the Java platform.

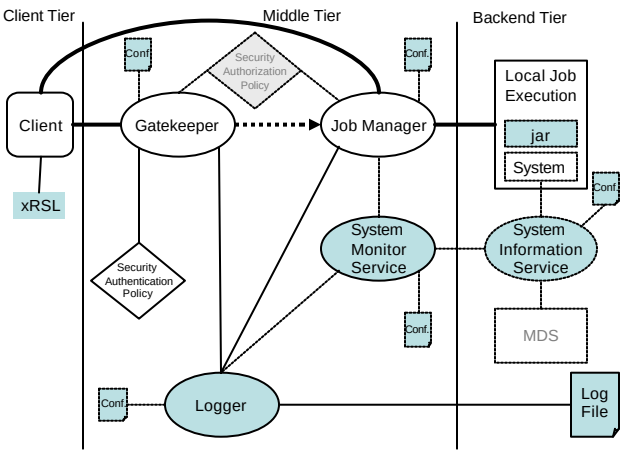

Figure 3: Modifications to the architecture of the Java GRAM protocol, client, and server usage.

Figure 3 illustrates the modifications we have made to the original GRAM service while at the same time providing protocol compatibility.
At present, we also explore the use of the InfoGram Service as part of sporadic Grids. Sporadic Grids are integrating production Grids with sporadically available resources such as a computer at a beamline or a computer donated for a short period of time to a compute cluster. The current prototype service together with the Java framework allows our technology to be reused within sporadic Grids, enabling an easier integration with

current production Grids. The architecture design of the InfoGram service can enable a SETI@home type of service, which can be used to integrate machines running MS Windows machines. Besides executing processes outside of the JVM, we enhance the security model for Grid computing while reusing Java's security model to, for example, restrict access to machine resource and prevent Trojan programs.

6. COMMUNITY USE OF THE JAVA CoG KIT

The user community served by the Java CoG Kit is quite diverse. The Java CoG Kit allows

- *middleware developers* to create new middleware components that depend on the Java CoG Kit;
- *portal developers* to create portals that expose transparently the Grid functionality as part of a portalservice; and
- *application developers* to use of Grid services within the application portal.

Projects currently using the Java CoG Kit for accessing Grid functionality include the following:

- *CoGBox* [29] provides a simple GUI for much of the client-side functionality such as file transfer and job submission.
- *CCAT* [31] provides an implementation of a standard suggested by the Common Component Architecture Forum, defining a minimal set of standard features that a high-performance component framework has to provide, or can expect, in order to be able to use components developed within different frameworks.
- Grid Portal Development Kit (GPDK) provides access to Grid services by using Java Server Pages (JSP) and JavaBeans using Tomcat, a Web application server.
- JiPANG (Jini-based Portal AugmeNting Grids) [28] is a computing portal system that provides uniform access layer to a large variety of Grid services including other PSEs, libraries, and applications.
- The NASA IPG LaunchPad [21] uses the Grid Portal Development Kit based on the Java CoG Kit. The tool consists of easy-to-use windows for users to input job information, such as the amount of memory and number of processors needed.

- The NCSA Science Portal [14] provides a "personal Web server" that the user runs on a workstation. This server has been extended in several ways to allow the user to access Grid resources from a Web browser or from desktop applications.
- The Astrophysics Simulation Code Portal (ASC Portal) [25] is building a computational collaboratory to bring the numerical treatment of the Einstein theory of general relativity to astrophysics.
- TENT [26] is a distributed simulation and integration system used, for example, for airplane design in commercial settings.
- ProActive [6] is a Java library for parallel, distributed, and concurrent computing and programming. The library is based on a reduced set of rather simple primitives and supports an active object model. It is based on the RMI Java standard library. The CoG Kit provides access to the Grid.
- DISCOVER [23] is developing a generic framework for interactive steering of scientific applications and collaborative visualization of data sets generated by such simulations. Access to the Grid will be enabled through the CORBA and Java Commodity Grid Kits.
- The Java CORBA CoG Kit provides a simple Grid domain that can be accessed from CORBA clients. The domain is provided as pure Java prototype. Future implementations in C++ are possible.

A regularly updated list of such projects can be found at [35]. We encourage the community to notify us of additional projects using the Java CoG Kit, so we can continue to update the Web page. The role of the Java CoG Kit for some of these projects is depicted in Figure 4

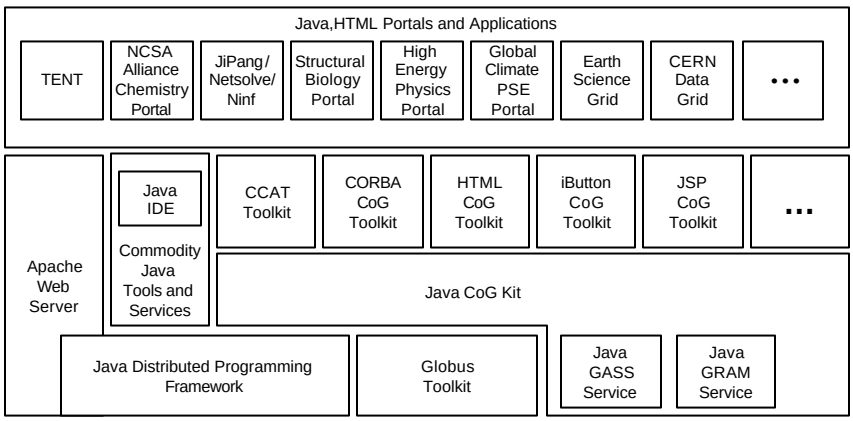

Figure 4: The Java CoG Kit builds a solid foundation for developing Grid applications based on the ability to combine Grid and Web technologies.

7. AVAILABILITY

The Java CoG Kit monitors closely the development within the Globus Project to assure that a level of interoperability is maintained. The CoG Kit development team continues to keep track of projects that reuse the Java CoG Kit and documents the requirements of the community, in order to feed this information back to the Globus development team and to develop new features within the Java CoG Kit. The Java CoG Kit is available for download [35] and depends on the most recent production version of Java and Globus (at the time of submission JDK 1.3.1 and Globus v1.0, and Globus v2.0 beta). For up-to-date release notes, readers should refer to the Web page at http://www.globus.org/cog. New releases are announced to the mailing list at cog-news@globus.org.

8. CONCLUSION

Commodity distributed-computing technologies enable the rapid construction of sophisticated client-server applications. Grid technologies provide advanced network services for large-scale, wide area, multi-institutional environments and for applications that require the coordinated use of multiple resources. In the Commodity Grid project,

we bridge these two worlds so as to enable advanced applications that can benefit from both Grid services and sophisticated commodity development environments. The Java Commodity Grid Project is creating such a bridge for the Java framework. We provide an elementary set of classes that allow the Java programmer to access basic Grid services, as well as enhanced services suitable for the definition of desktop problem solving environments. Additionally, we provided the Globus Toolkit with an independent set of client tools that was able to identify several problems in passed Globus Toolkit releases. Thus, our team has greatly contributed to the improvement of the Globus Toolkit.

Our future work will involve the integration of more advanced services into the Java CoG Kit and the creation of other CoG Kits, with CORBA and Python being short-term priorities. We hope to gain a better understanding of where changes to commodity or Grid technologies can facilitate interoperability and of how commodity technologies can be exploited in Grid environments.

ACKNOWLEDGMENTS

This work was supported by the Mathematical, Information, and Computational Science Division subprogram of the Office of Advanced Scientific Computing Research, U.S. Department of Energy, under Contract W-31-109-Eng-38. DARPA, DOE, and NSF support Globus Toolkit research and development. We thank Ian Foster, Geoffrey C. Fox, Dennis Gannon, and Jay Alameda for the valuable discussions during the course of the ongoing CoG Kit development. This work would not have been possible without the help of the Globus Project team. Globus Toolkit and Globus Project are trademarks held by the University of Chicago.

REFERENCES

1. Anonymous. iButton Web Page, 2001, http://www.ibutton.com, http://www.ibutton.com/ibuttons/java.html.
2. Anonymous. Java Web Start, a deployment framework, 2001, http://java.sun.com/products/javawebstart/.
3. Anonymous. XML SChema, Primer 0 - 3, 2001, http://www.w3.org/XML/Schema.

4. D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. F. Nielsen, S. Thatte and D. Winer. Simple Object Access Protocol (SOAP) 1.1, 2000, http://www.w3.org/TR/SOAP/.
5. R. Butler, D. Engert, I. Foster, C. Kesselman, S. Tuecke, J. Volmer and V. Welch Design and Deployment of a National-Scale Authentication Infrastructure. *IEEE Computer*, *33* (12). 60-66.
6. D. Caromel. The ProActive Java Library for Parallel, Distributed, Concurrent, and Mobile Computing, 2001, http://www-sop.inria.fr/oasis/proactive/.
7. E. Christensen, F. Curbera, G. Meredith and S. Weerawarana. Web Services Description Language (WSDL), W3C, 2001, http://www.w3.org/TR/wsdl.
8. K. Czajkowski, I. Foster and C. Kesselman, Co-allocation Services for Computational Grids. in *Proc. 8th IEEE Symposium on High Performance Distributed Computing*, (1999), IEEE Press.
9. W. K. Edwards *Core Jini.* Prentice Hall Computer Books, 2000,
10. S. Fitzgerald, I. Foster, C. Kesselman, G. v. Laszewski, W. Smith and S. Tuecke. A Directory Service for Configuring High-performance Distributed Computations. in *Proc. 6th IEEE Symp. on High Performance Distributed Computing*, 1997, 365--375.
11. I. Foster, J. Insley, G. v. Laszewski, C. Kesselman and M. Thiebaux Distance Visualization: Data Exploration on the Grid. *IEEE Computer*, *32* (12). 36--43.
12. I. Foster, C. Kesselman and S. Tuecke The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International Journal of High Performance Computing Applications*, *15* (3). 200-222, www.globus.org/research/papers/anatomy.pdf.
13. I. Foster, A. Roy and V. Sander, A Quality of Service Architecture that Combines Resource Reservation and Application Adaptation. in *Proc. 8th International Workshop on Quality of Service*, (2000).
14. D. Gannon. Indiana NCSA Science Portal, 2000, http://www.computingportals.org/CPdoc/gannon.doc.
15. V. Getov, G. von Laszewski, M. Philippsen and I. Foster Multi-Paradigm Communications in Java for Grid Computing. *Communications of ACM, 44* (10). 119-125, http://www.globus.org/cog/documentataion/papers/.
16. L. Gong. JXTA: A Network Programming Environment, SUN Microsystems, 2001, http://java.sun.com/people/gong/papers/jxta-ieeeic.pdf.
17. M. Hall *Core Servlets and JavaServer Pages (JSP).* Prentice Hall PTR/Sun Microsystems Press, 2000,

18. S. Holzner *Inside XML*. New Riders Publishing, 2000,
19. J. Jaworski, P. J. Perrone and V. S. R. R. Chaganti *Java security handbook*. Sams, Indianapolis, Ind., 2000,
20. D. M. John Crupi, Deepak Alur *Core J2EE Patterns: Best Practices and Design Strategies*. Prentice Hall PTR/Sun Microsystems Press, 2001,
21. W. E. Johnston, D. Gannon and B. Nitzberg, Grids as Production Computing Environments: The Engineering Aspects of NASA's Information Power Grid. in *Proc. 8th IEEE Symposium on High Performance Distributed Computing*, (1999), IEEE Press, http://www.ipg.nasa.gov/.
22. J. Novotny, S. Tuecke and V. Welch, An Online Credential Repository for the Grid: MyProxy. in *10th IEEE International Symposium on High Performance Distributed Computing*, (2001), IEEE Press, 104-111.
23. M. Parashar. The DISCOVER Project, 2001, http://www.caip.rutgers.edu/TASSL/new.htm.
24. S. S. Rosanna Lee *JNDI API Tutorial and Reference: Building Directory-Enabled Java Applications*. Addison-Wesley, 2000,
25. M. Russell, G. Allen, G. Daues, I. Foster, T. Goodale, E. Seidel, J. Novotny, J. Shalf, W.-M. Suen and G. V. Laszewski, The Astrophysics Simulation Collaboratory: A Science Portal EnablingCommunity Software Development. in *Tenth IEEE International Symposium on High Performance Distributed Computing (HPDC10)*, (San Francisco, 2001), http://www.cactuscode.org/Showcase/Publications.html.
26. A. Schreiber. TENT- Documentation, 2001, http://www.sistec.dlr.de/tent/docs/.
27. J. Siegel *CORBA 3 Fundamentals and Programming*. John Wiley & SonS, 2000,
28. T. Suzumura, S. Matsuoka and M. Nakada. JiPANG - A Jini based Computing Portal, http://www.gridforum.org/Documents/Drafts/jipang-ggf1-010225.pdf.
29. B. Temko. CoG Box, 2000, http://www.globus.org/cog.
30. S. Verma, M. Parashar, J. Gawor and G. von Laszewski. Design and Implementation of a CORBA Commodity Grid Kit. in Lee, C.A. ed. *Second International Workshop on Grid Computing - GRID 2001*, Spinger LNCS 2242, Denver, 2001, 2-12, http://www.caip.rutgers.edu/TASSL/CorbaCoG/CORBACog.htm.
31. J. Villaca, M. Covindaraju, D. Stern, A. Whitaker, F. Breg, P. Deuskar, B. Temko, D. Gannon and R. Bramley. CAT: A High Performance Distributed

Component Architecture. in *Proc. 8th IEEE Symp. on High Performance Distributed Computing*, 1999.

32. G. von Laszewski. InfoGram: A peer-to-peer information and job submission service, Argonne National Laboratory, 2001, http://www.globus.org/cog.
33. G. von Laszewski, S. Fitzgerald, P. Vanderbilt, P. Lane and B. Didier. GOSv3: A Data Definition Language for Grid Information Services, Argonne National Laboratory and Pacific Northwest Laboratory, 2001, http://www-unix.mcs.anl.gov/gridforum/gis/reports/gos-v3/gis-wg-021-002.html.
34. G. von Laszewski, I. Foster, J. Gawor and P. Lane A Java Commodity Grid Kit. *Concurrency and Computation: Practice and Experience*, *13* (8-9). 643-662, http://www.globus.org/cog/documentation/papers/cog-cpe-final.pdf.
35. G. von Laszewski and K. Jackson. The CoG Kit Web Pages, 2001, http://www.cogkits.org.
36. G. von Laszewski, S. Verma, J. Gawor and M. Parashar. A CORBA Commodity Grid Kit *submitted*, 2001, http://www.globus.org/cog.
37. M. Wahl, A. Coulbeck, T. Howes and S. Kille. Lightweight Directory Access Protocol (v3): Attribute Syntax Definitions, 1997, http://www.faqs.org/rfcs/rfc2252.html.
38. Y. Wang, F. D. Carlo, D. Mancini, I. McNulty, B. Tieman, J. Bresnahan, I. Foster, J. Insley, P. Lane, G. v. Laszewski, C. Kesselman, M.-H. Su and M. Thiebaux A high-throughput x-ray microtomography system at the Advanced Photon Source. *Review of Scientific Instruments*, *72* (4). 2062-2068.

