FORSCHUNGSZENTRUM JÜLICH GmbH
Zentralinstitut für Angewandte Mathematik
D-52425 Jülich, Tel. (02461) 61-6402

Interner Bericht

High-Performance Computer Management Based on Java

Volker Sander, Dietmar Erwin, Valentina Huber

FZJ-ZAM-IB-9806

April 1998

(letzte Änderung: 28.04.98)

Preprint: "Proceedings of the HPCN ’98, 21.4. - 23.4.1998, Amsterdam, pp. 526 - 534"

High-Performance Computer Management Based on Java

Volker Sander, Dietmar Erwin, Valentina Huber

Forschungszentrum Jülich GmbH
Zentralinstitut für Angewandte Mathematik
D-52425 Jülich
V.Sander@fz-juelich.de, D.Erwin@fz-juelich.de, V.Huber@fz-juelich.de

Abstract. Coupling of distributed computer resources connected by a high speed network to one virtual computer is the basic idea of a metacomputer. Access to the metacomputer should be provided by an intuitive graphical user interface (GUI), ideally WWW based. This paper presents a metacomputer architecture using a Java based GUI. The concept will be discussed with regard to security, communication, scalability, and the integration into existing frameworks.

1 Introduction

Combining supercomputers and high speed networks with low latency leads to the innovative model of a virtual heterogenous computer: a metacomputer. Building a metacomputer requires a software for interacting with different operating systems and different resource management systems. In addition, a uniformed graphical user interface is needed, supporting a single sign-on environment, to establish a homogeneous view of the metacomputer. The WWW based batch interface to Cray Systems at the Research Centre Jülich ([BHB0138]) is an example of a uniformed GUI by using a WWW-browser.
This paper presents a WWW-accessible three-tier architecture ([San97]), developed in cooperation between the Research Centre Jülich and the Paderborn Center for Parallel Computing (PC^2) for a management system of a metacomputer, called High Performance Computer Management (HPCM). The goal of the project is to deliver software that allows users to submit jobs to remote high performance computing resources without having to learn details of the target operating system, or administrative policies and procedures at the target site. HPCM is one of the projects of the Northrhine Westfalia metacomputing research cooperation ([BMR96]).

2 Basic Architecture

The fundamental idea of HPCM architecture is to define a kind of a three-tier model, consisting of:

– a graphical user interface
– one or more management daemons
– coupling modules on the compute servers

Access to the metacomputer is established by a lightweight client based on a WWW browser. The already existing CGI based interface to Cray Systems at the Research Centre Jülich is converted to a Java applet running within the Java Virtual Machine (JVM) of the user's preferred WWW browser. The Java applet loaded while accessing the metacomputer's HTTP address is generating a context specific graphical interface by using user related informations, e.g. his authorization, or the state of his transactions.

The central component of the HPCM architecture is the management daemon (MD) that creates the view of a virtual computer. This MD executes on the HPCM server machines with the task to receive the user's requests from the Java applet, to interpret their semantics, and to submit them to the related computing instances. To implement this functionality, the MD needs a global view of the metacomputer's components, i.e. joined users and compute servers. Consequently the MD is the administrative instance of the metacomputer which is responsible to maintain the metacomputer related data. Due to the fact that the MD is accessing resources on behalf of the user, there is a need for a security policy. The authentication scheme must also support a single sign-on environment, to encapsulate the computing instances from the user.

A coupling module (CM) interfaces with each compute server. The operating system of each participating platform needs to be adapted to the specified communication protocol and its semantics. This is done as a daemon process on the compute server. It translates the abstract management daemon requests to the underlying management software, e.g. NQS, NQE, DQS, CCS, LSF or CODINE ([KaNe94]). Figure 1 is illustrating the basic architecture.

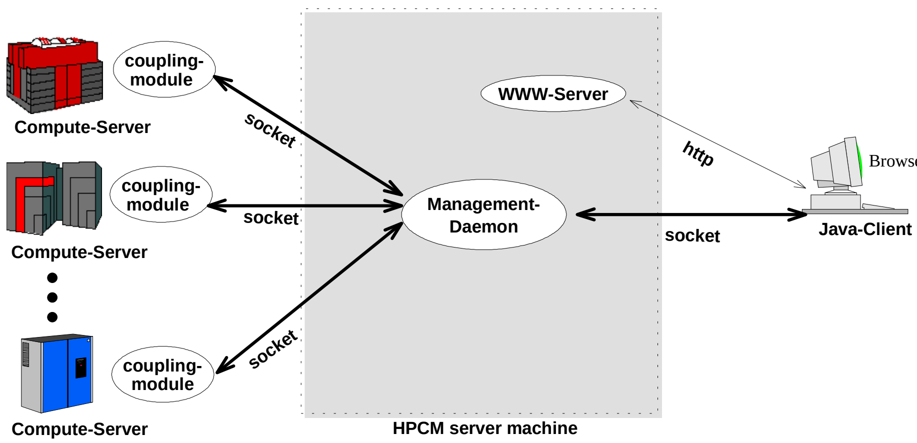

Fig. 1. HPCM overview

The advantage of this architecture is the high degree of system independance.

– The GUI relies only on WWW browsers and needs no special software at the user's workstation.
– The MD has to be developed only once.

Only the CM must be adapted for each new family of HPCM compute servers.

3 The Java Applet

Access to the metacomputer should be provided without the need for distributing and installing special software at the user's desktop. Due to the fact that WWW browsers are available everywhere, a WWW based solution satisfies this requirement. To implement the necessary functionality a Java applet seems to be the right design decision:

– Java applets are supported by popular WWW browser,
– Java applets are offering a standardized object oriented concept,
– the local CPU can be used for an adapted authentication scheme,
– the interface is context specific without reloading all informations,
– functions implemented by an applet are available everwhere within the Internet except for some firewall configurations.

As a consequence of the design decision several problems have to be solved. For security reasons Java applets loaded over the net are prevented from reading and writing local files and from making network connections except to the host the applet was loaded from. Consequently the Java applet must be loaded from the same host the management daemon is executing on.
To implement a context sensitive GUI within the applet, there is the need for accessing user related as well as environment specific information. This information could be used to tailor the interface, taking the user's authorization, the actual state of his transactions, and his preferrences into account. One way to implement this feature is to use the Java File System (JFS). This requires an additional daemon on the HPCM-server: the Java-Daemon.
Another problem arises for developers outside the US. The communication channel between the applet and the HPCM-server must be secure, ideally by using SSL-classes as provided by the Java Cryptographic Extension (JCE) or by Phaos's SSLava, an SSL 3.0 / Java Toolkit. But those APIs fall under US export restrictions (ITAR) although the SSLeay toolkit written in C is freely available around the world. Unfortunally, Java developers outside the US have to obtain another solution for their security needs.
However, the authentication scheme used for the applet communication must support message authentication as well as strong message encryption for passwords. To ensure that the user's password is read by the authorized applet, there is a need for signed applets within this concept. But signed applets are only supported by the upcomming versions of WWW browsers and those are not yet available on every desktop computer. Therefore the applet needs to be validated by an explicit authentication step based on the browser's SSL capabilities ([FKK96]) [1] .

[1] The SSL Protocol - Version 3.0 draft-ietf has beed deleted.

The authentication scheme selected for the communication between Java applet and HPCM-Server is divided into two independant parts:

- The functionality of signed applets is achieved by using an SSL based http connection. For firewall configurations, there is a need for establishing an SSL-proxy which is forwarding the endpoint communication through the firewall. Later versions of HPCM will use signed applets which will lead to a model, where an applet based GUI is exempted from the security restrictions mentioned above.
- The communication channel between applet and server is encrypted by a freeware library of Systemics Inc. called Cryptix^{TM} ([HaPa97]). The source distribution of Cryptix^{TM} contains a pure Java implementation of nearly all major crypthographic algorithms. Cryptix^{TM} was selected due to the fact that it is developped within the european union and is therefore not falling under the ITAR restrictions. Beside the source code availability the main advantage of these classes is that they support strong encryption (128 Bit), which is a requirement for authenticating the user by its password.

  But there is also a disadvantage, mainly caused by the different Java implementation strategies of the public browsers. Most of them will not let applets define classes whose names start with "java.". To use Cryptix^{TM} within an applet, those classes must be installed locally, accessible through the CLASSPATH-environment-variable. Therefore the installation of a local class archive on the user's desktop is a required step. Unfortunally Netscape's Navigator requires a zip-archive whereas Microsoft's Internet Explorer is looking for a cab-archive. Consequently two archives have to be maintained and distributed through the HTTP server.

4 The HPCM-Server

Using a Java applet for implementing the GUI of a metacomputer also influences the design of the HPCM-Server. First, applets need a HTTP daemon to be loaded from. Since applets are not able to communicate with any host except the one the applet was loaded from, the management daemon and the HTTP daemon must execute on the same host. Secondly, the user and cluster related information is maintained by another daemon: the Java-Daemon. The final architecture is illustrated by the figure 2.

5 Communication Protocol

One of the major design decision was related to the communication protocol used between the HPCM components. Combining object-oriented methods within a distributed environment leads to the question, whether the Common Object Request Broker Architecture (CORBA [Vino97]) is offering an appropriate framework.
Obviously CORBA is addressing major aspects of the HPCM-environment:

- CORBA interconnects objects written in a variety of programming languages, including C, C++, Java, Smalltalk, and Ada,

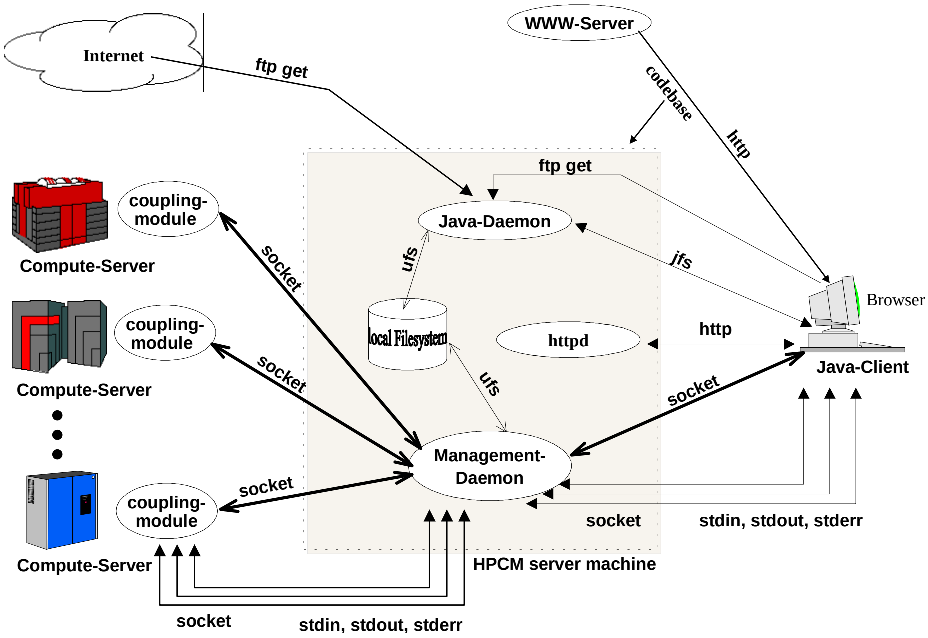

Fig. 2. HPCM Architecture

– the Naming Service allows clients to find objects based on names,
– the Trading Service allows clients to find objects based on their properties,
– the Security Service defines an authentication scheme.

Therefore one the HPCM-design goals is the usage of CORBA for implementing object relations. But non-Intranet applets communicating in a CORBA compliant manner are actually a problem. The public Internet browsers do not support CORBA implementations in a sufficient manner right now. Netscape's Navigator 4.0, for example, will integrate Visigenic's VisiBroker later on, to support CORBA/IIOP applications and faster Java support across platforms ([Net97]). But the dependencies between used Internet browser and its supported Java mechanisms and limitations would hinder a rapid implementation of the HPCM modules. Therefore the actual HPCM implementation is using a socket based communication:

– The heterogeneity of the HPCM architecture is supported by data encoding mechanisms similar to the ASN.1 standard ([ISO8824]).
– Each remote message invocation is simulated by a request-response-sequence, exchanging the manipulated data structures (objects).

6 Scalability and Reliability

To avoid a single point of failure and to design a scalable system more than one HPCM-Server has to be supported.

This request could be fullfilled by at least three methods:

– URLs typically specify a particular file on a particular machine, addressed by its internet hostname. Address resolution is performed as usual by using the Domain Name Service (DNS). A closer look to the Requests For Comments ([RFC1034] and [RFC1035]) reveals that the address resolution step of a hostname supports a list of addresses (A records). This address list can be used on the client site to contact the most suitable host, for example by submitting a multicast message expecting a load average indicator as a response. The consequence of this DNS property is that the hostname resolution could be used as the technique for replication and load balancing. Consequently it is a function of the WWW browser or the resolver library of the client site. Unfortunately this function is not implemented within the current versions of the available browsers. In fact, browsers were fixed to prevent this feature due to the DNS spoofing attack ([DFW96]).
– The concept of Universal Resource Names (URNs, [RoNi95] and [DaMe97]) is based on the idea of separating the resolution system from the way names are assigned. URLs identify a protocol at a specified host, URNs are refering to Internet accessible resources. These abstract resource names are resolved by a DNS based location broker, using a new NAPTR record which enables the usage of different hosts for the same resource. Unfortunally this is only a draft standard with no sufficient support by the available browsers.
– Another solution could be established by a proxy server. The URL is pointing to a proxy server which delegates the request to a second level HTTP daemon. Unfortunally this concept has the disadvantage, that the Java applet must communicate with the proxy server. The HPCM design requires an immediate communication between applet and management daemon. But the idea of a proxy server might be implemented without receiving the applet from the proxy server. The HTML-document used for addressing the applet is not necessarily a file load during the HTTP-GET-operation. It is possible to use CGI scripts or servlets, producing the related HTML output. Combining this feature with the *CODEBASE* Parameter of the HTML *APPLET* tag offers an alternative solution. The location (URL) of the applet to be loaded is specified by the *CODEBASE* parameter. Consequently, the URL specified for accessing the metacomputer is not necessarily addressing the same host as the applet provider. The user is accessing the HTML-page of a WWW server whose *APPLET* tag is created dynamically. The *CODEBASE* parameter is pointing to the HPCM-Server host, used for the upcoming communication.

Scalability of the HPCM architecture is established by using a dynamically generated *CODEBASE* parameter, addressing the HPCM-Server to be used. However, URNs needs to be supported within the design in the future

7 Integration of Distributed Resources

Besides the scalability of the design, a secure framework for integrating distributed resources and a specification for using applets within this framework is needed. We decided to use the DCE framework ([OSF91]):

– The Distributed File Service (DFS) is offering a secure global filesystem with a replication scheme and without any explicit mount operation. A distributed filesystem is one of the most important resources for building a metacomputer, especially if data access is provided in a UNIX manner.
– Security is guaranteed by the Kerberos protocol version 5 ([SNS88] and [KNT91]).
– DCE is divided into independent administrative domains, called cells. An application is able to access services from a remote cell - cross cell authentication - as well as a cell offers replicated services distributed among a WAN.
– The concept of delegation enables a secure migration of a user's credentials. This is basically needed for a single sign-on architecture regarding the security policies.
– CORBA and DCE are not really exclusive concepts. There are CORBA implementations based on DCE (environment specific inter-ORB-protocols). Consequently the usage of DCE is not perventing the usage of CORBA in later HPCM releases.

The usage of the DCE framework must also consider the integration of the applet. Solutions based on protocol tunneling, as descriped in the DCE Web project ([LeZu95]), require a DCE capable browser. That is contradictory to the fundamental prerequisite of the concept: the general availability of a browser at the user's desktop computer. Therefore another solution must be implemented. The Java applet is authenticating the user by a password. This password might be submitted strongly(!) encrypted via Cryptix^{TM} - e.g. not crackable by using a dictionary - to the management daemon which is using the decrypted user password for a DCE login. If this step succeeds, the MD creates a login context for the user's Java session. Remember that the whole communication channel used within this session must be secure. Otherwise, security could not be guaranteed. Within this solution another benefit of Java can be utilized. If the management daemon's login context of the user expires, the applet will prompt the user to authenticate again without any reload actions to be done.

The effort and impacts for establishing a Northrhine Westfalia metacomputer-DCE-cell was analyzed by a cooperation between the Regional Computing Centre of the University of Cologne (RRZK) and the Center for Parallel Computing (ZPR) of the University of Cologne ([LuWi97]). It is planned to integrate the HPCM-implementation into the DCE-framework in the middle of 1998.

The current implementation of HPCM is using a table for translating the HPCM user credentials (uid,gid) to those of the compute servers. Consequently this table must be maintained for each participating computing instance.

Another issue system accounting. HPCM is following a concept of soft accounting which is implemented as a layer above the native accounting methods of the compute server. Jobs submitted by the user are accounted on the destination system and therefore follows the rules definied at the executing institution. Each coupling module is pushing the actual accounting information to the MD once a day. On receipt of the message, the management daemon is updating its soft accounting base. This information base will be used to inform the user about possible accounting problems at job submission.

8 Summary

Access to High Performance Computing resources has traditional been complicated for the scientist and tends to be architecture-specific. The added value promised by a metacomputer concept could only be exploited by the users, if the specified architecture handles not only the integrative aspect of the distributed resources. A uniform graphical user interface is essential for the success of a metacomputing concept. The World Wide Web, especially Java based, is offering a fundamental primitive for implementing such a GUI. WWW browsers are available on nearly every desktop computer. Consequently, there is no need for maintaining a special GUI at the user level. This point is essential since a distributed metacomputer will be accessed by a large amount of users, located at different sites and maintaining a software module at every desktop computer would cause an enormous administrative effort.

The High Performance Computer Management project is integrating distributed computing resources, regarding the demand for a uniform graphical user interface. The design of HPCM is following a three-tier architecture. The Java applet is responsible for reading the user's requests, the management daemon is establishing the view of a virtual computer, and the coupling module is adapting the operating system of the compute server, especially the batch subsystem. Within the architecture, the communication is addressing the security problems of a distributed metacomputer environment.

On the way to implement the metacomputing concept, the DCE framework will be used to establish distributed resources, especially a global filesystem. Right now, the HPCM implementation has shown that many system specific details of a heterogenous environment can be encapsulated. The abstraction level handled by the management daemon allows a straight forward integration of different plattforms besides the existing coupling modules for CRAY T3E and Parsytec computers.

References

[BHB0138] W. Frings, and R. Knecht (eds.), *The Cray Systems at Research Centre Jülich, Volume 1, handling of Jobs and Data*, ZAM User Manual No. 0138 (1997)

[BMR96] A. Bachem, B. Monien, and F. Ramme, *Der Forschungsverbund NRW-Metacomputing 'Verteiltes Höchstleistungsrechnen"*, http://www.uni-paderborn.de/pc2/nrw-mc/html_rep/html_rep.html (1996)

[DaMe97] R. Daniel and M. Mealling, *Resolution of Uniform Resource Identifiers using the Domain Name System*, Internet-Draft "'NAPTR proposal"' (1997)

[DFW96] D. Dean, W.W. Felten, and D.S. Wallach, *Java Security: From HotJava to Netscape and Beyond*, 1996 IEEE Symposium on Security and Privacy (1996)

[FKK96] A.O. Freier, P. Karlton and P.C. Kocher, *The SSL Protocol - Version 3.0*, Internet-Draft (1996)

[ISO8824] International Organization for Standardization, *Information Processing Systems - Open System Interconnection - Specification of Abstract Syntax Notation One (ASN.1)*, ISO8824 (1987)

[HaPa97] H. Passon, *Auf Nummer Sicher*, iX, Magazin für professionelle Informationstechnik, Oktober (1997)

[KaNe94] J.A. Kaplan and M.L. Nelson, *A Comparison of Queueing, Cluster and Distributed Computing Systems*, NASA Langley Research Center, NASA TM 109025 (1994)

[KNT91] J.T. Kohl, B.C. Neuman, and T.Y. Ts'o, *The Evolution of the Kerberos Authentication Service*, EurOpen Conference, pp. 295- 313 (1991)

[LeZu95] S. Lewontin and M.E. Zurko, *The DCE Web Project: Providing Authorization and Other Distributed Services to the World Wide Web*, OSF Research Institute (1995)

[LuWi97] M. Luig and R. Wimmershoff, *Aufbau und Management einer Metacomputing-Infrastuktur*, http://www.uni-koeln.de/RRZK/Projekte/DCE/docs/bericht97.ps (1997)

[Net97] Netscape, *The Networked Enterprise: Communicator Roadmap*, http://search.netscape.com/comprod/at_work/white_paper/vision/communicator.-html#mercury (1997)

[OSF91] Open Software Foundation, *A Distributed Computing Environment Framework . An OSF Perspective* (1991)

[RF1034] P. Mockapetris, *Domain Names - Concepts and Facilities*, Request for Coments (RFC) No. 1034 (1987)

[RF1035] P.Mockapetris, *Domain Names - Implementation and Specification*, Request for Coments (RFC) No. 1035 (1987)

[RoNi95] K.E. Rowe and C.K. Nicholas, *Reliability of WWW Name Servers*, Conference Proceedings of The Third International Worl-Wide Web Conference, Germany (1995)

[San97] V. Sander, *A Metacomputer Architecture Based on Cooperative Resource Management*, Proceedings of High Performance Computing and Networking Europe 1997 (HPCN 1997), Wien, 28. - 30. April (1997)

[SNS88] J.G. Steiner, C. Neumann and J.I. Schiller, *Kerberos: An Authentication Service for Open Network Systems*, Usenix Conference Proceedings (1988)

[Vino97] S.Vinoski, *CORBA: Integrating Diverse Applications Within Distributed Environments*, IEEE Communications Magazine, Vol. 14(2), (1997)