Confidential

Case for Support:
The Use of Java in a Multi-Language Environment for High Performance Applications
(Draft 15/1/99)

Proposal to the EPSRC under the

Responsive Grant Programme for IT and Computing

by

Vladimir Getov[†] **and Mike Ashworth**[§]

[†] Harrow School of Computer Science
University of Westminster
Northwick Park, Harrow HA1 3TP

[§]CLRC Daresbury Laboratory
Warrington WA4 4AD

January 1999

1 Previous Track Record

1.1 Previous Work at Westminster

The Computer Systems Performance Engineering Group (CSPEG) was established at the beginning of 1996 in accordance with the new strategic plan of the University of Westminster in order to focus resources on research areas of strength and innovation. The group is fortunate to have the services of Roger Hockney as a Visiting Professor who has already made significant contributions to the numerical analysis of parallel kernels and performance evaluation of supercomputers. A sure sign of future promise for the research group has been the quick attraction of external research and collaborative funding from different sources. The EPSRC grant (GR/L38868) received last year facilitates the development of a portable high-performance FFT kernel, in collaboration with the University of California, San Diego and IBM's T.J.Watson Research Centre. In addition, the group is involved in several international projects with collaborative research grants in the area of Performance Engineering from both CEC and British Council.

Meanwhile, a modest internal funding under the NFF initiative has been allocated to support research activities in the area of high performance Java programming. In this field CSPEG has conducted pioneering research since the beginning of 1997 with some initial results in developing Java wrappers to legacy parallel libraries in C and/or Fortran such as MPI and ScaLAPACK [3]. The draft JavaMPI binding has been particularly successful as a "proof-of-concept" message-passing interface for Java and the group is now one of the leading participants in a proposal for the development and implementation of a standard MPI for Java specification [2]. The University of Westminster is well-placed to build on this prior and current work and to establish strong research activities in the area of high performance distributed computing in Java.

1.2 Previous Work at Daresbury

The Computational Engineering Group at CLRC's Daresbury Laboratory is concerned with the promotion, development and exploitation of high performance computing in advancing engineering simulation in collaboration with academia and industry. Simulation is fast replacing experiment in the design process in many areas resulting in reduced development times, improved product efficiency, more extensive and detailed optimisation, and improved design for hazard mitigation and reduced environmental impact. The Group is involved in a wide range of technical activities, including :

- Parallel algorithms and applications development
- Benchmark development and systems evaluation
- Software prototyping and standards evaluation
- Technology transfer to academia and industry
- Education and training
- Supporting and coordinating the exploitation of local and national HPC systems

One of the key developments within the Computational Engineering Group in recent years has been the HP-PIPES code. HP-PIPES is an advanced simulation code developed through the HP-PIPES (High Performance Parallel Computing In Process Engineering Simulations) project (European Union ESPRIT programme project 8114). The objective of the project was to demonstrate the application of high-performance computing and computational fluid dynamics to support the next generation approach to industrial research and development in the process industries, with emphasis on the use of Information Technology for simulation and prediction.

The core code is an advanced computational fluid dynamics code designed to exploit the power of high-performance and parallel systems. It models the incompressible, turbulent Navier-Stokes equations together with an enthalpy equation describing heat transfer in three-dimensional duct and pipe geometries. This is a basic requirement common to chemical reactors, combustion chambers and a host of other process engineering applications. The code is highly modular, allowing the development of modules which model the specific features of particular applications. Using this code, simulations have been carried out of the complex physics and chemistry involved in a Titanium dioxide chemical reactor and a power station combustion chamber. These modules describe phenomena such as chemical reactions within a high temperature flow, particle agglomeration, radiative heat transfer and convection. From these modules, realistic models of the reactor or boiler combustion chamber have been constructed in collaboration with industrial partners.

The code has been implemented using Fortran 77 and the Message Passing Interface (MPI) standard to enable the utilization of parallel processing on a wide range of high performance platforms. A number of static and dynamic load balancing strategies were used to ensure an even distribution of the computational load across processors.

Results from the HP-PIPES applications have been disseminated widely in the form of a multimedia CD-ROM developed in the follow-on APEX (Awareness and Promotion Exercise) project (European Union ESPRIT programme project 20259). The APEX CD is aimed at promoting the use of HPC and CFD in the process industries. Around 625 CDs have been distributed to industrialists and academics in industrially-related fields using a range of dissemination and promotion techniques including exhibitions, mailshots and advertisements.

1.3 Research Experience

The principal investigator for Westminster, *Dr Vladimir Getov* is a Senior Lecturer in the School of Computer Science at the University of Westminster and Head of the Computer Systems Performance Engineering Group there. Dr Getov has considerable experience in the performance modelling and analysis of parallel and distributed systems with a proven research record of over 50 published papers. His current research interests include the development of portable performance optimisation strategies and multi-language high-performance programming environments. He has been a member of the PARKBENCH Committee since its creation in 1992 and is now actively involved in the work of the Java Grande Forum.

The principal investigator for Daresbury, *Dr Mike Ashworth* is a High Performance Computing (HPC) Consultant working in the Computational Engineering Group at CLRC's Daresbury Laboratory. Dr Ashworth has 16 years experience in supercomputing and parallel processing in application areas from pulsar research to climate modelling. He now specialises in the application of HPC to CFD problems, both industrial and environmental, and in the benchmarking and optimization of software for high-performance applications. Dr Ashworth is now actively involved in the further development of and dissemination of results from the HP-PIPES code.

In addition to the research activities at Westminster and Daresbury, we propose to collaborate with *Professor Geoffrey Fox* and other colleagues from his Northeast Parallel Architectures Center at Syracuse University in the USA. He is well-known as the leader of the Caltech Hypercube Concurrent Supercomputer project during the last decade. Professor Fox has been the initiator and, since the beginning of 1998, the academic coordinator of the Java Grande Forum which provides internationally the primary focus for the exchange of ideas, research results, and solutions on the use of Java in high performance computing.

2 Description of Proposed Research

2.1 Background

With the recent availability of high-performance low-cost networking components, it is now becoming practical and cost-effective to assemble a network of workstations and/or PCs into a single distributed-memory computing resource. Such systems are frequently referred to as NOW systems (Network of Workstations). Another example is the Beowulf class systems which use particularly cost-effective PC technology [1]. Whether the departmental compute server is a traditional mainframe or supercomputer, a purpose-built distributed memory parallel computer or a NOW or Beowulf system, it is advantageous to adopt a client/server approach to application design. In this way the user interface is a GUI running on the user's workstation handling control tasks and other non-performance critical book-keeping activities (the client) while the high-performance kernel executes on the compute server. Moreover, with this model the client/server approach is not limited to local area computing. The client may be further divided into a Web server and remote client enabling computing and application resources to be accessed from anywhere on the Internet.

The World Wide Web was conceived as a means of accessing information and that is still its major function, but it is increasingly evolving towards a means of providing and accessing services. Likewise, networking, both local-area and wide-area, is no longer about providing connectivity, but about providing services. The client-server based model of distributed computing is growing beyond the simple file-server and compute-server concepts to encompass a global web-based metacomputer.

Most Internet resources today present data as pages written in HTML format, a document markup language which allows simple text formatting, access to data in various forms (image, audio, video, etc), and, crucially as this gives the Web its web-like structure, hypertext links to other pages. Limited services, such as email forms, database searches etc., are provided using CGI, which allows an incoming client to execute programs made available on the server.

A major sea change is happening on the Internet, due in no small part to the emergence of the Java programming language. Java is a modern, object-oriented language which has been designed from the start to facilitate the provision of services over the Internet. The main difference from CGI, is that Java applets (small Java applications embedded in Web pages) execute not on the server but on the client, enabling applications to make better use of low bandwidth communications. Java is compiled to a machine-independent bytecode. When a Java-enabled client accesses a Java applet, the bytecode is downloaded and runs on the client's own hardware using an interpreter known as the Java Virtual Machine. Java is expanding rapidly and comes complete with a wide range of class libraries and toolkits for Web access, thread based computing and Graphical User Interfaces (GUI).

The Java language has several built-in mechanisms which allow the parallelism inherent in scientific programs to be exploited. Threads and concurrency constructs are well-suited to shared memory computers, but not large-scale distributed memory machines. Although sockets and the Remote Method Invocation (RMI) interface allow network programming, they are too client/server oriented to be suitable for SPMD-style scientific programming. Thus, codes based on RMI would potentially underperform compared to platform-specific implementations of standard communication libraries like MPI. Nevertheless, as a programming language, Java has the basic qualities needed for writing high-performance applications. With the maturing of compilation technology, such applications written in Java will doubtlessly appear. Fortunately, rapid progress is being made in this area by developing optimizing Java compilers, such as the IBM High-Performance Compiler for Java (HPCJ), which generates native code for the RS6000 architecture [6]. Since the Java language is fairly new, however, it lacks the extensive scientific software of other languages like Fortran 77 and C.

Although the popular image of the Web has been purely as a global information provider, this has not been its only manifestation. Increasingly, organisations are setting up an *Intranet* - a corporate Web server providing information and services which are targetted at, and indeed are only accessible by, people within the organisation, or a part of it. The Intranet may contain information for staff ranging from accounts to telephone directories and from minutes of meetings to the date of the Xmas dinner. Furthermore, in an industrial research and development environment, for example, the Intranet may enable access for everyone in a group or department to the technical designs and specifications of a new product together with the project specifications, modification records, test procedures, test results, marketing information etc.

It is in this environment that we see the major benefits of integrating Java with the traditional high-performance programming environment based on Fortran and C using a client/server model. This vision emphasises the strengths of each component: Java allows easy access to Web-based computing and facilitates the construction of GUIs; Fortran and C allow applications best to exploit high-performance hardware.

Therefore, a key research goal is to develop and evaluate a new type of multi-language multi-paradigm programming environment for high-performance applications which makes use of innovative Web technologies and GUIs based on Java and provides a simple Application Programming Interface (API), high performance, and a high level of portability and flexibility for large-scale applications.

2.2 Programme and Methodology

2.2.1 Project aims and objectives

The proposed project will develop a methodology for building application-specific virtual programming environments based on a set of intelligent tools for automatic generation of flexible and efficient interfaces between different languages, such as Java, Fortran, and C, and different programming paradigms such as message-passing, threads, and shared memory. The emphasis will be on the development of tools which allow techniques to be applied generically in a range of scientific and engineering applications.

The project will demonstrate the applicability of this new programming style by applying it in the case of HP-PIPES - an advanced CFD application.

Measureable objectives may be defined by the following milestones and deliverables:

- Report on a standard Java-MPI interface specification
- Implementation of a set of intelligent tools for multi-language bindings generation
- Report on different variants of multi-language programming environments
- Report on strategies and guidelines for multi-language software development, performance optimisation, and tuning
- Report describing the design of a Java-based interface to the HP-PIPES code, including tested and functional Java-Fortran-MPI code examples
- Implementation of the full interface described above
- Report describing the evaluation of the capabilities, utility and performance of the Java-HP-PIPES code in a number of virtual execution environments

2.2.2 Methodology

Our general approach is to combine Java and legacy code written in Fortran and/or C into multi-language programming environments where Java serves as a front-end wrapper for large-scale codes and existing native libraries. We believe that this approach is practical in promoting the use of modern, portable, maintainable object-oriented software while still maintaining the benefits of past investment in legacy code. If Java is to gain acceptance as a high performance language it has to interface with existing native libraries. Building on top of our initial work [3], we will develop a set of intelligent tools for automatic code generation of Java bindings for various legacy libraries. Using the automatically created multi-language interfaces, we can build different kinds of multi-language programming environments for high performance Java computing in a flexible and elegant way.

One of the primary goals of our approach will be to gain portability by using Java, without sacrificing performance from highly optimized legacy code. However, the binding of legacy code to Java has certain limitations. In particular, for security reasons, applets downloaded over the network may not load libraries or define native methods. One possible solution to this problem is to extend the functionality of the IceT virtual collaborative environment [5] by providing the ability to spawn remote Java applications on machines running an IceT daemon and then soft-loading any missing software components such as multi-language bindings to remote Java processes. In such a virtual environment both processes and data would be allowed to migrate and to be transferred throughout owned and unowned resources, under flexible security measures imposed by the users. In this case an application programmer will be given the best of both worlds – their codes would enjoy enhanced portability and wider accessibility to resources similar to pure Java applications and would be able at the same time to derive high performance levels exclusive to conventional languages such as C and Fortran.

Application of the programming environment to the HP-PIPES code will be incremental. Initial work will concentrate on developing and testing example components using simple test programs linking Java and Fortran to gain experience and demonstrate proof-of-concept. Demonstrations will begin single-threaded and move towards multi-threaded Java interfaced to parallel message-passing Fortran. Experience gained from these tests will enable the design of a complete interface for the HP-PIPES code featuring a GUI with input controls to set-up and steer the computation and Web-based visualization of the resulting flow fields. The initial phase will be completed with a report outlining the techniques to be employed, results from the test programs and the Java-HP-PIPES design concepts. Implementation of the interface will follow the design document with testing at appropriate stages and modification of the specification as necessary.

The final Java-HP-PIPES code will be evaluated for performance and ease-of-use in a number of execution environments. Key execution components are the client workstation running a Web browser, the Web server containing the main Java elements and the back-end compute server providing high-performance computing resources. Systems currently available at the Daresbury Laboratory include a 26-processor IBM SP2 and an 8 processor Beowulf-class system consisting of a cluster of PCs running Linux and MPI.

2.2.3 Timeliness and novelty

Java and related Web-based computing technologies are making a new and significant impact in many areas of computing. The time is right to exploit these technological advances to make a positive contribution to the productivity of high-performance codes in a R&D environment. The novelty of this approach is clear. The acceptance of such a novel paradigm by what is seen at times to be a somewhat conservative user-base may be problematic. For this reason we place a great emphasis on the dissemination and exploitation activities in this project.

2.2.4 Programme of work

The detailed programme of work is as follows:

Task 1: Components of the multi-language environment.

- Intelligent generator of bindings
- Java MPI interface specification and binding generation
- Environment integration

Personnel: Vladimir Getov, RAw.

Task 2: Application development methodology.

- Methodology for multi-language software development
- Balanced design and implementation of specific kernels from the HP-PIPES code
- Methodology for performance optimisation and tuning

Personnel: Vladimir Getov, RAw.

Task 3: Advanced CFD application - HP-PIPES.

- Proof-of-concept using Java-Fortran-MPI demonstrator codes
- Design of a Java based interface to the HP-PIPES code
- Implementation of interface
- Performance evaluation in a range of virtual execution environments

Personnel: Mike Ashworth, RAd

2.3 Relevance to Beneficiaries

The multi-paradigm programming environment proposed in this project will enable the exploitation of cost-effective networked computing resources in industry and academic R&D environments and the take-up of new technologies in Web-based computing. The emphasis on the development of tools, methods and interfaces will encourage the design of efficient, maintainable, portable and reliable object-oriented software while still maintaining the benefit of past investment in software development of existing high performance codes.

2.4 Dissemination and Exploitation

Dissemination of techniques, methods and results will be through published journal papers, conference papers and presentations, and through workshops and working groups of the Java Grande Forum.

HP-PIPES was developed in collaboration with ICI, and their subsidiary Tioxide, and EDP, the portuguese electricity generator. It is currently being extended to include a Reynolds-stress turbulence model (in addition to the existing k-ϵ), and curvilinear coordinate and multi-block capabilities. With these additional modelling features and the new capabilities and added-value engendered by the Java interface, HP-PIPES will be highly placed as a state-of-the-art technology demonstrator in attracting future collaborative projects with partners in the chemical engineering, process engineering and power generation industries.

A letter of support, a copy of which is attached to the proposal, has been received from Tioxide indicating their support for the aims of the proposed research and their interest in making use of its results.

The techniques developed during the course of this project are readily applicable to other application codes, for example marine CFD, in which the Computational Engineering Group at Daresbury has an increasing involvement, leading to potential exploitation in coastal engineering, offshore industries, fisheries management, marine pollution monitoring, weather forecasting and climate research to name but a few.

2.5 Justification of Resources

The proposed research involves three main work packages as described in the programme of work. As the diagrammatic workplan shows, 2 people will be required to complete the work, in addition to the proposers. Funding is sought for full-time research assistants at both Westminster (RAw) and Daresbury (RAd) for the full two year duration of the project.

RAw will be responsible for the design and development of the multi-language environment and the application development methodology (Tasks 1 and 2). RAd will be responsible for the development of the HP-PIPES code within the multi-language framework, necessitating familiarity with the fundamentals of the CFD code as well as a clear understanding of the development methodology.

Each RA will require a PC or workstation with facilities for compilation and execution of Java codes and use of leading-edge Java related libraries and utilities. At present, due to the leading work being undertaken by Sun in this area, it is anticipated that Sun workstations will provide the best and most effective working platforms.

There is a clear requirement for close collaboration between the two institutes, with Daresbury requiring results of the methodology and tool development at Westminster, and Westminster benefitting from feedback and benchmark results from Daresbury. For that reason we have included a generous allocation for travel between the two sites.

The primary focus for the exchange of ideas, research results, and solutions on the use of Java in high performance computing is the Java Grande Forum [8]. In this rapidly evolving area, it is essential that researchers are continually monitoring state-of-the-art developments taking place elsewhere. Moreover, we are in an excellent position to continue feeding new research results into the Java Grande development process. The Java-MPI standardisation activities and the methodology for multi-language application development are the areas of our particular strength. In order to benefit from further interactions and closer collaboration with work at the Northeast Parallel Architectures Center, Syracuse University, as well as the International Java Grande Forum as a whole a travel element has been included allowing for 2 short visits from UK to Syracuse and 2 short visits from Syracuse to UK.

Other meetings, such as the next workshops in the series on Java for High Performance Network Computing [7], are being planned within the timescale of the project. We therefore request funding for two researchers to travel to European Java-related conferences or workshops and for two attendances of the Java Grande Forum meetings and workshops. Presentation of results at these workshops forms a key part of the dissemination process.

References

[1] P. Beaugendre, T. Priol, G. Alleon and D. Delaveux, A Client/Server Approach for HPC Applications within a Networking Environment, in *High-Performance Computing and Networking*, eds. P. Sloot, M. Bubak and B. Hertzberger, Springer, 518-525 (1998).

[2] B. Carpenter, V. Getov, G. Judd, T. Skjellum, G. Fox, MPI for Java: Position Document and Draft Specification Report, JGF-TR-03, Java Grande Forum, November, 1998.

[3] V. Getov, S. Flynn-Hummel, and S. Mintchev, High-performance parallel programming in Java: Exploiting native libraries, *Concurrency: Practice & Experience*, to appear (1998).

[4] R. Gordon. *Essential JNI: Java Native Interface.* Prentice–Hall, 1998.

[5] P. Gray and V. Sunderam, The IceT environment for parallel and distributed computing, in *Scientific Computing in Object-Oriented Parallel Environments*, Y. Ishikawa, R. Oldehoeft, J. Reynders, and M. Tholburn, eds., *LNCS* 1343 (1997).

[6] IBM Corp., High-performance compiler for Java: An optimizing native code compiler for Java applications, http://www.alphaWorks.ibm.com/formula/ (1997).

[7] First UK Workshop on Java for High Performance Network Computing, http://www.cs.cf.ac.uk/hpjworkshop/

[8] Java Grande Forum, http://www.javagrande.org/

3 Part 3: Diagrammatic Workplan

Code	Task	Project month							
		1 3	6	9	12	15	18	21	24
T 1.1	Intelligent generator of bindings	■	■	■					
T 1.2	Java MPI interface specification and binding generation		■	■	■				
T 1.3	Environment integration					■	■	■	
T 2.1	Methodology for multi-language Java software development		■	■	■				
T 2.2	Balanced design and implementation of specific kernels	■	■	■	■	■	■		
T 2.3	Methodology for performance optimisation and tuning			■	■	■	■	■	■
T 3.1	Proof-of-concept using Java-Fortran-MPI demonstrator code	■	■						
T 3.2	Design of a Java based interface to the HP-PIPES code		■	■					
T 3.3	Implementation of interface			■	■	■	■	■	
T 3.4	Performance evaluation in a range of virtual environments						■	■	■