A two-tier distributed electronic design framework

Tom Kazmierski and Neil Clayton
Department of Electronics And Computer Science
University of Southampton,
Southampton, SO17 1BJ, United Kingdom

Abstract

We present the concept of a distributed, web-based electronic design framework. The salient feature of our system is the extension of the client-server architecture to two-tiers, with the web server serving client requests whilst acting as client to the tool servers. In the sample application of the framework, developed in Java, any of the servers can be based on Linux, MS Windows or Sun-SPARC server. The web server that has been used to demonstrate the framework for on-line access to VAMS (a VHDL-AMS parser) and Avant! HSPICE is currently available for Linux but has been developed with a truly platform independent implementation in mind.

1. Introduction

In a keynote speech at DAC 2000 [1,2] entitled "The Internet: the next IC Design Environment" a concept was outlined to allow companies to create global design groups to complete complex system-on-chip designs. The Internet is a great enabler and multiplier of people resources [1,3,4]. It has been anticipated [5] that the $6.5 billion plus marketplace for design and engineering software will grow at a rate of approximately 10% annually over the next five years, while the marketplace for collaborative software, although smaller in comparison ($1 billion), will expand at a much greater rate of more than 25% per annum over the next five years. There exist a number of distributed frameworks and web-based tools motivated by the popularity of the Internet. Several Web-based tools concern power consumption analysis, such as PowerPlay [6]. The WELD system developed at Berkeley [7] is a comprehensive distributed framework for IC design. An important result reported by the WELD team indicates that a distributed frameworks system based on a Java front end, data manager, a server wrapper package, Java client packages, proxy services and a distributed workflow system, all built with scalability and extensibility in mind, is feasible.

A more recently proposed constraint management methodology called CCM (Collaborative design Constraint Management) [8] provides automated support for conflict detection and resolution as an integral part of the design process. Another recent proposals introduce the concept of task-oriented programming with distributed components to create configurable environments for networked design projects [9] and a distributed design process management system whereby designers receive constraint-based feedback that enables them to apply search heuristics in their collaborative work [10]

2. Two-tier client-server architecture

The aim of this contribution is to provide a framework that harnesses this concept, building upon legacy design tool frameworks. In particular it is the ease with which users may upload and download files in a secure and supervised manner and view graphical and textual responses with little or no understanding of the underlying systems. It is this abstraction that the Tool Framework aims to achieve and it is our belief that there will be interest in continuing this project from the academic community and from commercial tool providers as a case study for future developments. Our aim was to investigate the feasibility of a secure, session-based CAD design web framework to which the addition of some sort of subscriber system should be straightforward (figure 1). We have developed a sample framework to provide on-line access to VAMS [4] (a VHDL-AMS parser) and Avant! HSPICE [11]. The successful elements of the network's testing and prototyping process were then used in conjunction with industry standard design techniques (i.e. Unified Modeling Language) in order to devise complete system architecture. Where possible, standard code components have been used in order to increase productivity and decrease time spent debugging basic routines. The basic code modules were then combined and developed into a complete framework. All the code for the framework is written in Java and so is platform independent, allowing any of the servers to be Linux, MS Windows or Sun-SPARC.

The multi-user nature of this framework requires the web server to have some 'memory' of what actions users have already been undertaken so that it can prompt for the next stage. HTTP is by definition a stateless protocol and as

such maintains no such ‘memory’. Java provides a session tracking API, which makes the best use of the facilities available, such as cookies[1], URL rewriting or hidden fields[3] to maintain this ‘memory’. Simple servlet code was written to implement this session tracking which could then be integrated later in the project. The session-tracking API states that it is the web server’s responsibility to maintain a collection of session objects that are created as different users make requests.

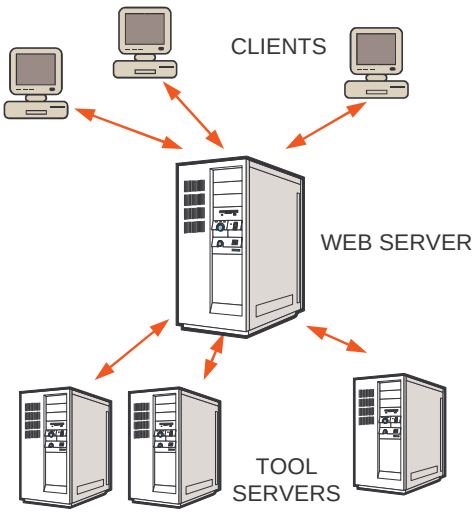

Fig. 1. Two-tier client-server architecture.

It is in these objects that programs may store information and objects that help maintain information regarding state.

The nature of the framework also requires that files (e.g. netlists) be uploaded from a user’s computer to the web server for processing. The file must be streamed through Java StreamReader classes in order to write out to the file system. Code has been written to do this and it was thoroughly tested. The two servlets above can be combined to give a system, which will uniquely record a user's identification when writing the uploaded files to the webserver file system. A system has been developed to store data about when files were written and last accessed in order to allow the webserver to recover in the event of a crash and delete expired files.

The RMI was used as the method of choice for communicating with the tool servers (figure 3). The Java Remote Method Invocation (RMI) distributed object model provides an abstracted interface allowing programmers to develop distributed Java programs with the same syntax as for non-distributed programs [12]. It is this model that provides transparency to the developer such that systems can simply be developed as if the tools reside on the web server. The architecture presents an interface to the developer that allows the code that defines behavior and code that implements the behavior to remain separate and run on separate JVM’s (e.g. separate computers). It is necessary to register the remote java services with a Java RMI registry on the remote machine so that when connection is made to a TCP socket the correct service can be tasked to handle the connection.

Java classes have been developed in order to test the feasibility of this method for communication between the web server and the tool severs. RMI provides a programming method that allows code development as if the remote Java classes are resident locally. This level of transparency is clearly excellent; the only consideration is that any objects that are passed to the remote classes must be serializable.

Java provides a number of built-in classes that enable graphics programming. These are the abstract window toolkit (AWT) and the swing class that provides a number of reusable graphics components. A very simple method of graphing data in java is available, and indeed this should not be overlooked as a method as it provides a very simple device for communicating numerical data. Having researched this area, a number of scientific classes (JSci) produced at the University of Durham [13] were found to be available. These provide further graphical components that were designed specifically for graphing numerical data.

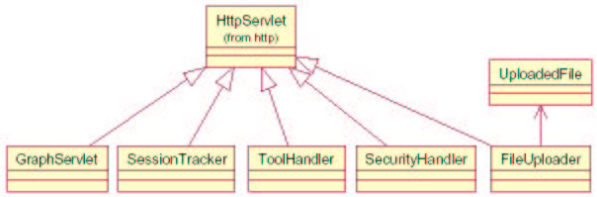

Fig 2. UML Class Diagram – Servlet Architecture.

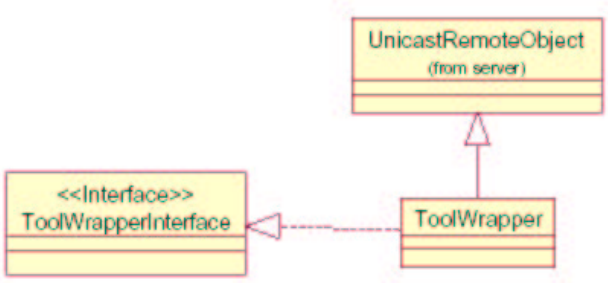

Fig. 3. UML Class Diagram – RMI Class Architecture.

3. VHDL-AMS Parser

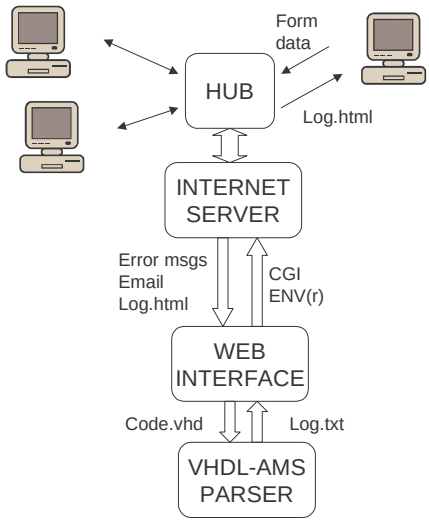

Fig. 4 Conventional CGI architecture in VHDL-AMS compiler.

The Southampton VHDL-AMS Compiler, a tool for validating code written in VHDL-AMS (an analog mixed signal design language) was added to the framework in order to demonstrate the ease with which multi-tool operation can be achieved. The tool requires no access restriction and therefore no use of the logon handler. The tool simply accepts a file as an input, specified as a command-line parameter, and returns a textual log file detailing the outcome of the validation (figure 4).

Due to the design of the framework the addition of the tool was straightforward, utilizing the same uploading system as for the HSpice tool (Section 3) and a customized version of the remote tool controller configured for the changed command line format. It is clear that a generic version of the tool controller could be provided such that tool settings and command-line format could be prescribed in a local text file. There is no requirement for advanced graphical tools or any form of data processing, thus a version of the Hspice controller can be used as VHDL controller.

Due to the fact that it is not possible to specify the name of the output file that the parser creates at runtime it was necessary to use simple session directories on the tool server to allow simultaneous multi-user operation without causing any conflict with respect to file creation and downloading. This system took extra coding to implement but provides support for a range of tools that share this form of command line control. Figure 5 shows a screenshot of the front end to the VHDL-AMS parser that the framework uses.

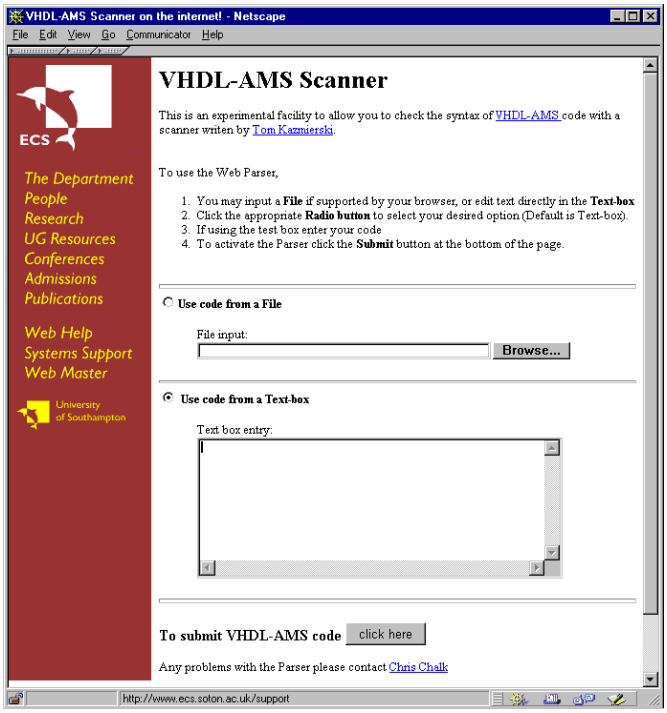

Fig. 5. Screenshot of VHDL-AMS parser front-end.

3. Star HSPICE

The Hspice controller servlet is invoked by the user from the dynamically generated HTML and is passed the name of a file that has previously been uploaded. It is responsible for operating as the server end of this process and the client end of the communication process with the tool server. The result files are then copied back from the to tool server to the user's session directory and then deleted from the tool server in order to maintain centralized control.

A class has been developed in order to execute commands, a relatively simple process which spawns an operating system thread and executes the specified command within the Java security sandbox. This was integrated with the file transfer class described above so that a file could be transferred to the remote machine and read using the Linux 'cat' command (chosen as an example command). This was then extended such that a file could be uploaded to the remote tool server, HSpice could be invoked by the web server to run on the tool server and files could be transferred back to the web server using the vector transfer method.

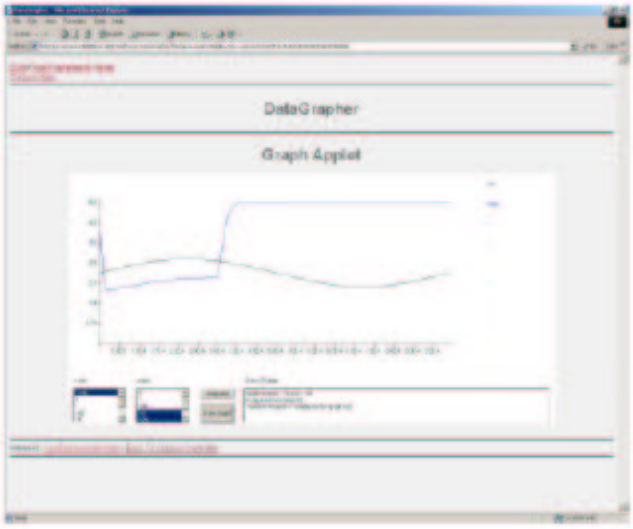

Fig. 6. GraphApplet form showing HSPICE results.

4. Security

Java is designed specifically for network-based computing and security measures are an integral part of Java's design. Few other distributed solutions can make this claim. From their inception, most other distributed solutions utilize a traditional execution model [14]. The business end of he Java security model [15] is conveniently described by using the metaphor of the 'Sandbox', which comprises a number of cooperating system components, ranging from security managers that execute as part of the application, to security measures designed into the Java Virtual Machine (JVM) and the language itself. The sandbox ensures that a distrusted – and possibly malicious - application cannot gain access to system resources. Also fundamental to the Java security model is the concept of a Java Protected Domain. Java Protected Domains enable the use of "permissions" by the user or can use a pre-configured default setting. This type of capability serves to extend Java's existing fine-grained control by allowing multiple and unique permissions for individual applications. Collectively, these and other integral features provide a highly secure and flexible security model for the Java platform. The security of the web-server [16] must be considered and in particular the security of the requests made to the web-server and those made to the tool servers. Using a Java based approach and a Java based web server it should be a simple matter to implement the industry standard Secure Sockets Layer (SSL) technology which using public key cryptography provides a range of security services, such as server authentication, client authentication (optional), integrity and confidentiality. Communication between applets and the web-server can be controlled by using the 'java key' system to 'sign' the applet. Data is then only provided to an applet with the correct signature.

In our implementation the HTTP protocol provides the ability to use simple name/password authentication – called basic authentication – in order to restrict access to preset domains within a web server. With this technique, the web server maintains a database of usernames and passwords and identifies certain resources as protected. When a user requests access to a protected resource, the server responds with a request for the client's username and password. At this point the browser usually pops up a dialog box where the user enters the information. Basic authentication is very weak. It provides no confidentiality, no integrity, and only the most basic authentication. The problem is that passwords are transmitted over the network, thinly disguised by a well known and easily reversed Base64 encoding. It is due to both this limitation and the need for a great deal of web server configuration that a custom authentication technique, to which up-to-date security techniques may easily be applied, has been chosen for this application. This is a method whereby a servlet handles the access restriction, governing which servlets may then be accessed.

5. Conclusion

This contribution presents a novel, a platform independent framework supporting distributed tool servers that is sufficiently abstracted to make the integration of other tools a simple task. The technological approach uses state-of-the-art methods; taking where appropriate from the rapidly advancing field of e-commerce development solutions.

The level of functionality provided easily exceeds that of the minimum requirement and indeed fulfils the requirement for extra functionality including the addition of the VHDL-AMS parser to demonstrate the ease with which extra tools may be incorporated. It should be noted that this level of both functionality and flexibility is a consequence of the design process used. This process involved a rapid stage of prototyping and feasibility assessment in order to determine how a system could work leading to a complete UML [17] specification of the system in order to comply with industry 'best practice' in the area of software design.

In summary, this project provides a state-of-the-art Java based solution to the problem of providing a framework through which to control a number of legacy circuit design tools. The flexibility and extendibility of the framework have been demonstrated and methods for using Java applets to provide customizable tools for data processing have been illustrated. The framework has been designed in such a manner as to be dependent on the web

server for the majority of its standard security features a fact that allows the latest advances in e-security, pioneered through electronic banking and commerce to be integrated seamlessly with the improvement in web servers. Where necessary custom security measures have been introduced.

6. References

[1] Sangiovanni-Vincentelli A.,"The Internet: the next IC Design Environment", *Key note presentation, DAC'2000*, June 2000, Los Angeles.

[2] Newton A R. "Impact of Web Technologies on EDA System Architectures", *ISPD 1998*, Monterey CA, 6th April 1998, Department of Electrical Engineering and Computer Sciences University of California at Berkeley.

[3] Rogers JL, Salas AO. "Towards a more flexible Web-based framework for multidisciplinary design", *Advances in Engineering Software 30(1999)*, 439-444, 1 Jan 1999.

[4] Kazmierski T J, Chalk C D. "Web-based VHDL-AMS design environment", *Proceedings of the IEEE/VIUF International Workshops on Behavioral Modeling & Simulation*, October 1998.

[5] "The Electronics Industry Supply Chain: Who Does What?", Panel, Organizer and Chair: Rita Glover, *Proc. 38th DAC*, June 18-22, 2001, Las Vegas.

[6] D. Lidsky, J. M. Rabaey, "Early Power Explo ration - a World Wide Web Application", *Proc. 33rd DAC*, Las Vegas, June 1996.

[7] F. L. Chan, M. D. Spiller, A. R. Newton "WELD – An Environment for Web-Based Electronic Design", *Proc. 35th DAC*, June 1998

[8] G. Konduri, A. Chandrakasan, "A Framework for Collaborative and Distributed Web-based Design", *Proc. 36th DAC*, June 1999.

[9] F. Brglez, H. Lavana, "A Universal Client for Distributed Networked Design and Computing ", *Proc. 38th DAC*, June 18-22, 2001 Las Vegas.

[10] J. A. Carballo, S. W. Director," Application of Constraint-Based Heuristics in Collaborative Design", *Proc. 38th DAC*, June 18-22, 2001 Las Vegas.

[11] "Star-Hspice Manual". *Release 1999.2, June 1999*, Avant! Corporation, Fremont CA 94538.

[12] "Fundamentals of RM", *Sun Microsystems 2000*. Sun Microsystems Inc, *www.java.sun.com*.

[13] Hale M., "JSci – A science API for Java", *http://jsci.org.uk.*

[14] Tanenbaum AS. "Distributed Operating Systems", *Prentice Hall International Publishing 1995*, ISBN 0-13-143934-0.

[15] McGraw G, Felten EW. "Secure Computing With Java: Now And The Future", *Sun Microsystems Whitepaper 1997*, Sun Microsystems Inc. *www.java.sun.com*.

[16] Kossakowski KP, Allen J. "Securing Public Web Servers", April 2000, *http://www.cert.org/security-improvement/practices/p082.html*.

[17] Fowler M, Scott K. "UML Distilled", 2nd Ed., *Addison-Wesley 1999*, ISBN 020165783X.