
F*iscal 1997 Project Portfolio Report*

A Summary of Significant Accomplishments

Sun Microsystems Laboratories (SunLabs)
Fiscal year ending June 30, 1997

SMLI TR-97-62

© Copyright 1997 Sun Microsystems, Inc. All rights reserved except as expressly modified hereunder. The SML Technical Report Series is published by Sun Microsystems Laboratories, a division of Sun Microsystems, Inc. Printed in U.S.A.

This document is protected by copyright and distributed under licenses restricting its use, copying, distribution, and decompilation. No part of this document may be reproduced in any form or by any means without prior written authorization of Sun or its licensors, if any.

[Ce produit ou document est protégé par un copyright et distribué avec des licences qui en restreignent l'utilisation, la copie, la distribution, et la décompilation. Aucune partie de ce produit ou document ne peut être reproduite sous aucune forme, par quelque moyen que ce soit, sans l'autorisation préalable et écrite de Sun et de ses bailleurs de licence, s'il y en a.]

TRADEMARKS

Sun, Sun Microsystems, the Sun logo, Java, JavaScope, JavaScript, JavaSoft, JavaSpec, JavaStar, JDK, ShowMe TV, Solaris, SpecTcl and SunTuner are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries. All SPARC trademarks, including UltraSPARC, are trademarks or registered trademarks of SPARC International, Inc. SPARCstation is licensed exclusively to Sun Microsystems, Inc. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd. FireWire and the FireWire logo are trademarks of Apple Computer, Inc., used under license. Netscape Navigator is a trademark of Netscape Communications Corporation.

For information regarding the SML Technical Report Series, contact Jeanie Treichel, Editor-in-Chief <jeanie.treichel@eng.sun.com>. For distribution issues, contact Amy Tashbook Hall, Assistant Editor <amy.hall@eng.sun.com>.

O*ne of the greatest pains to human nature is the pain of a new idea.*

—Walter Bagehot

Table of Contents

Foreword	vii
Introduction	1
Asynchronous Processor Design	5
CAD Research	9
Conceptual Indexing	13
Distance Learning	17
Forest	23
High-Speed Networking.....	31
Internetworking	35
Java Topics	39
Java Wallet	43
Kanban Compilation	49
Media Systems Group	51
Smart Cards and TCO	57
Speech Applications.....	59
SunTest.....	65
Tcl and Tk	69
Venturi	73
Appendix A—Patents Issued FY97	75
Appendix B—Technical Reports Issued FY97	77
Glossary	79

Foreword

A Year of Collaboration

This sixth full year of SunLabs operations has contained many changes and significant accomplishments. I am personally most proud of the multitude of ways in which the people of SunLabs have provided value to Sun Microsystems, our corporate sponsor and patron.

In FY97, SunLabs grew by absorbing activities in networking and computer security from our parent CTO organization. The addition of an emerging Boston Center for Networking to the SunLabs East Coast operation at Chelmsford, MA, provides valuable additional technical strength and leadership from Phil Rosenzweig. This year's expansion also provided our first international employees in Holland and the United Kingdom. SunLabs at present has its employees spread over its main California and Massachusetts locations with additional individual contributors in New York City and two European countries.

Prior years have seen many physical and organizational transfers of Labs staff into the company. In contrast, this past year was one of intensive collaboration and cooperation with other Sun organizations without explicit formal transfer of administrative responsibility. I have loaned SunLabs people to other Sun organizations where their particular talents and skills provide unique assistance. Furthermore, a number of Labs projects have been directly focused on quite near-term technical problems of importance to the company. While I worry that the Labs' long-term perspective will be compromised, I am convinced that SunLabs has delivered extremely valuable know-how to the company this year. A partial listing of collaborative staff and areas includes:

Nicole Yankelovich and her team in SunLabs East have provided the leadership for the development of the Java Speech APIs.

Ted Goldstein, on loan to JavaSoft, led the development of the Java Electronic Commerce Framework and was the key Sun participant in the creation of JavaCard, likely to be a unifying force in the fragmented world of smart cards.

Randy Smith, on loan to JavaSoft, led the development of the Java Collaboration API.

Bob Bosnyak has been assisting Sun Microelectronics (SME) with clock distribution issues in Sun's next generation of processor chips.

Dave Ungar and Mario Wolczko moved part-time into SunSoft quarters to assist with the development of Java virtual machines.

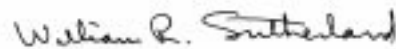
Steve Heller and his group in Chelmsford have also contributed to virtual machines with improved garbage collection technology.

Bob Sproull has organized a group of JavaSoft partners to define a network printing API for Java clients.

On a different tack, this past year SunLabs started two experimental businesses in an attempt to transfer to the company not only technology but also marketplace-validated business concepts. Within the Labs, we started SunTest to deliver Java testing tools and SunScript to provide commercialized versions of Tcl/Tk. These two business experiments have provided a 50% hit rate. In the Spring of 1997, we concluded that the commercial outlook of SunScript did not warrant further business continuation. SunLabs' support of Tcl/Tk research will continue with an emphasis on coupling Tcl and Java. SunTest is continuing with promising commercial prospects and has provided the technical basis and tools for the 100% pure Java certification program. 100% pure Java is a key part of the overall Sun approach to its Java-based activities.

SunLabs is no more than the collective talent of its staff. In FY97, Guy Steele, Distinguished Engineer, was recognized by his industry peers with the first ACM SIGPLAN Programming Languages Achievement Award. This is a gratifying addition to his already noteworthy list of professional accomplishments. Bob Sproull, Sun Vice President and Fellow, was elected this year to membership in the National Academy of Engineering. All of us celebrate the professional recognition of these two colleagues.

Finally, once again I sadly dedicate this annual report to a departed colleague. In December of 1996 we lost our friend, mentor, and colleague Dr. Charles Molnar. Charlie was an inspiration to our young professionals and a source of wisdom and clear thinking to all of us whatever our age and experience. Charlie, we do and will miss you!



William R. (Bert) Sutherland

Director, Sun Microsystems Laboratories
Vice President, Sun Microsystems, Inc.
bert.sutherland@eng.sun.com

Introduction

SunLabs nurtures our portfolio by seeking to balance our technology on several dimensions. Projects should start, proceed, and end in a reasonable time horizon. Some results are usable soon, while others have a longer gestation period. Some projects focus on software, some on hardware, and some on methodology and technique. We collaborate with other activities at Sun as a way of enhancing our efforts and maximizing the transfer of ideas into other areas of the company. New projects entering our portfolio must have innovative ideas as a basis, relevance to Sun's goals, and each needs a technical champion to drive the project to effective results. Our goal is to see the results of our research flow out into practical use in Sun.

The Fiscal 1997 Portfolio is comprised of the following projects:

- **Asynchronous Design**

To design and apply high-performance asynchronous circuits.

ivan.sutherland@eng.sun.com

bob.sproull@east.sun.com

- **Computer-Aided Design (CAD)**

To investigate ways to improve the design methodology used by Sun in the design of its ASICS and full custom chips.

tom.mcwilliams@eng.sun.com

- **Conceptual Indexing**

To organize indexed material into a conceptual taxonomy to make connections between terms and related terminology for efficient text-based online information retrieval.

william.woods@east.sun.com

- **Distance Learning**

To create and investigate technologies and learning paradigms that support real-time interaction within a distributed group.

randall.smith@eng.sun.com

- **Forest**

To explore the development and use of persistent programming systems, with a particular focus on their application to the construction of software development environments.

mick.jordan@eng.sun.com

- **Internetworking**

To explore and develop new, highly scalable, reliable, multicast network protocols and services to enable timely and bandwidth-efficient delivery of mission-critical data to very large numbers of clients by Java™-based “push” applications.

phil.rosenzweig@east.sun.com

- **Java Topics**

To research special projects for Java, including storage management and garbage collection.

steve.heller@east.sun.com

- **Java Wallet**

To create electronic commerce clients and servers using Java technology.

ted.goldstein@eng.sun.com

- **Kanban**

To apply self-styled dynamic optimization techniques to the execution of Java programs.

neil.wilhelm@eng.sun.com

- **Media Systems Group**

To research problems in the area of system architecture and the support of networked digital media.

duane.northcutt@eng.sun.com

- **OpeNet**

To design and implement a high-performance, open, light-weight, vendor-independent ATM network control platform to enable development of universal network services applications.

raphael.rom@eng.sun.com

- **Security**

To investigate and develop a scalable, flexible, platform-independent network security solution to audit and maintain security on systems throughout an Intranet. (Project brought into SunLabs at the end of FY97; therefore there is no report included in the FY97 Project Portfolio.)

linda.mccarthy@eng.sun.com

- **Smart Cards and TCO**

To analyze the economics of information systems and develop infrastructure measurement systems to track and charge for usage of shared resources.

diana.neiman@eng.sun.com

- **Speech Applications**

To understand how to build a robust, effective environment for speech applications.

nicole.yankelovich@east.sun.com

paul.martin@east.sun.com

- **SunTest**

To develop and market software testing tools based on the technology developed in FY96 by the PrimaVera/ADL group.

alberto.savoia@eng.sun.com

- **Tcl and Tk**

To develop tools and infrastructure that allow Tcl and Tk to be used for a variety of scripting applications, including Internet/Java applications and applications with special security needs.

john.ousterhout@eng.sun.com

- **Venturi**

To develop circuit design techniques for embedded CMOS digital serial link transceivers to increase speed and robustness, and testability, and to lower power, area, and pin count.

neil.wilhelm@eng.sun.com

Asynchronous Processor Design

Ivan E. Sutherland

ivan.sutherland@eng.sun.com

Overall Objective

Our objective is to explore asynchronous concepts and technology in microprocessor systems. The knowledge gained may permit Sun to use asynchronous techniques earlier, with greater confidence, and more effectively than competitors.

Objectives for FY97

Following a major review in May 1995, we have focused on speed over complexity. Our objective has been to find simple circuit architectures for pipeline structures that can provide speed and ease of understanding. We seek to demonstrate our understanding through design, fabrication, and testing of simple chips running at high speed.

Description

Some of the advantages that asynchronous systems may offer include higher speed, lower power, better modularity, reduced electromagnetic radiation, easier upgrade, and greater tolerance to manufacturing variation.

At Sun Microsystems Laboratories, high speed guides our efforts. To be useful to Sun, systems must go fast. Speed requires concurrency. We seek designs in which fine grain concurrency enhances performance. In modern computer systems, one can devote only a handful of gate delays to each cycle; concurrent operation on a gate-by-gate basis is essential to competitive performance. We find, as have other workers in both hardware and software, that concurrency is hard to grasp. The human mind seems to prefer sequential algorithms. Much of our progress has come from insights related to concurrency.

Two years ago we started designing, building, and testing a series of integrated circuit chips to give our ideas the ultimate verification of working silicon. We continue this effort.

Our chip design efforts keep us from glossing over any of the practical or theoretical problems of making asynchronous chips work. In planning each chip we must consider not only how it should work, but also how to test it for failures. The hardest failures to detect, of course, are low probability transient failures from unexpected causes. We design our chips to demonstrate the absence of such failures, so that successful tests enhance our confidence in their reliability. So far we have learned most of the lessons of each chip during the design phase. This may indicate only that we are too conservative in our designs, because each new chip provides a fresh opportunity to learn from unexpected behavior. Working chips confirm that we can achieve competitive speeds while maintaining the flexibility of asynchronous design.

We are pleased with the analytical side of our efforts. Our efforts to understand the details of delay in asynchronous pipelines have given us powerful command of such systems. We can now predict both the transient and steady state performance of asynchronous pipelines from the properties of their components.

Accomplishments

1. Over the past two years we have improved the raw speed of our simple pipeline structures approximately threefold through circuit and architectural improvements while holding the integrated circuit technology constant.
2. We have demonstrated the ability to design modular pipeline chips in an asynchronous style.
3. We have evolved a ring test technique for high-speed pipeline structures. In such a pipeline ring, data elements circulate around the ring millions of times, thus revealing the “steady state” behavior of the underlying structures.
4. We have designed, built, and tested five chips in FY97, gaining confidence in our understanding of the underlying structures.
5. One of our chips, the “asP* FIFO ring,” operates at nearly 1 billion data items per second, making it, we believe, the fastest working asynchronous FIFO on the planet. We reported this experiment at Async97.¹ (See the Publications section of this report.)

6. On the basis of our experiments, we are confident that our asynchronous control structures can drive a data path consisting of latches and logic as fast as it can go. We are confident that asynchronous systems can be every bit as fast as synchronous ones, while retaining the additional flexibility offered by asynchronous control.
7. We have calibrated our C++ event simulator against SPICE simulations permitting us to simulate very complex structures with modest amounts of computer time. Such simulations proved useful in constructing test procedures for our recent chips.
8. We have developed a remarkable understanding of the details of FIFO behavior. Because logic gates tend to operate more slowly when multiple inputs change simultaneously, the maximum throughput of an asynchronous pipeline is slightly less than one might predict from its behavior at lower throughputs.
9. We have produced a large body of intellectual property for Sun. We believe that ideas become intellectual property only when written down. During this past fiscal year, our written output includes:
 - Over 209 technical memos completed and filed in Sun Microsystems Laboratories internal archives.
 - Five invention disclosures sent to Sun Legal department.
 - Five new patent filings.
 - Approximately 20 patents pending.
 - Three patents issued (See Patents section.)
 - One paper, presented at Async 97 (See References.)

References

¹Sutherland, I.E. and J. Ebergen. "A Way to Predict FIFO Throughput." Presented at Async97, Eindhoven University, the Netherlands (April 1997). [SML 97-0108].

Publications

Molnar, C.E., I.E. Jones, W. Coates, and J. Lexau. "A FIFO Ring Oscillator Performance Experiment." *Proceedings from Async97*, Eindhoven University, The Netherlands (April 1997).

Sproull, R.F. (ed.), et al. "AND/IF 1.0." Sun Microsystems Laboratories (March 1997). [SML 97-0208].

Sutherland, I.E. "A View of the Task You Face: A Report to the NRC Committee on Cryptography." *Sun Microsystems Laboratories Perspectives 96-2* (August 1996).

Patents

Title: Cascaded Multistage Counterflow Pipeline Processor for Carrying Distinct Data in Two Opposite Directions
Inventors: I.E. Sutherland, C.E. Molnar, R. F. Sproull, I.W. Jones
Issued: 11/05/96 (5,572,690)

Title: System for Fast Switching of Time Critical Input Signals
Inventor: I.E. Sutherland
Issued: 01/07/97 (5,592,103)

Title: Counterflow Pipeline Processor with Instructions Flowing in a First Direction and Instruction Results Flowing in the Reverse Directions
Inventors: R.F. Sproull, I.E. Sutherland
Issued: 02/04/97 (5,600,848)

CAD Research

Thomas M. McWilliams

tom.mcwilliams@eng.sun.com

Overall Objective

The overall objective of the Computer-Aided Design (CAD) Research group is to improve the tools and processes available to the rest of Sun for the design and implementation of integrated circuit chips. We work with design teams to understand their needs, develop innovative technologies not available from commercial CAD tools, and implement and deploy tools incorporating successful new ideas.

Objectives for FY97

The CAD Research group was newly formed at the beginning of FY97. The major objectives for this first year were to:

- Build up an initial project research staff.
- Understand Sun's current design tools, processes, and CAD issues.
- Understand CAD industry directions, tools, and challenges.
- Identify the most pressing problems faced by Sun's hardware engineers.
- Identify promising areas for CAD research.

Description

Sun's hardware design teams use CAD tools to automate various aspects of the hardware design process. Most of the tools used at Sun are purchased from CAD vendors. Internally-developed tools fill in the gaps in commercial offerings, connect tools from separate vendors, and provide a strategic advantage over using only commercially available tools.

Over the past year our main goal has been to identify promising areas of research. We would like our future research to 1) solve problems that are

relevant and important to Sun's hardware design community, and 2) develop technology or methods that will *not* be available commercially. To that end, we have studied a number of the most widely used commercial CAD tools in order to understand their capabilities, limitations, and direction of evolution. Concurrently, we have had meetings with design teams to discuss their use of tools, their perception of problems, and their needs for better tools as driven by the relentless evolution of technology. From these studies we have identified a number of promising areas for active research. A description of these areas follows.

Integration of Logical and Physical Descriptions in ASIC Design

Current ASIC design processes separate the logical and physical aspects of the design. Current processes place primary emphasis on the logical design. The physical design is often an afterthought left to a separate physical design team employed by the ASIC vendor. As VLSI technology scales into the deep sub-micron regime, on-chip interconnect delay will dominate logic delays¹ and such a separation of logical from physical design becomes unworkable.

In order to design future high-performance ASICs, Sun must develop a new design process that allows the designer to conveniently specify physical design constraints. How best to do this is an open research topic. Current CAD vendors tend to focus on GUI-based floor planners, which have had limited success at addressing this issue.

Addressing Design Complexity

The amount of effort required to design increasingly complex chips has been growing sharply, creating pressure on improving productivity. The view of how best to address this problem is through re-use of intellectual property. In the past, a chip would be designed once and re-used in many designs. In the future, complete systems will fit on a few very large chips, preventing this natural re-use of chips. Approaches for re-using IP in the design of large chips should be developed and refined to reduce development costs. How best to do this is a promising research area.

CAD Utilization of Large Symetric Multi Processor (SMP) Servers

A large segment of Sun's business is selling systems to be used in the CAD area. Traditionally, most CAD programs run on a single processor.

Research is needed to adapt these programs to run efficiently on large multiple processor (MP) servers, thus shortening execution time and speeding the development process. Improved algorithms could be helpful for internal use and also for showing the CAD industry ways to use these large servers effectively.

Accomplishments

1. After examining a number of existing hardware description languages, we have noted a number of flaws in the most commonly used languages. We have proposed a new hardware description language, named JHDL, that is based on Java and avoids those flaws.
2. We have investigated new ideas in timing verification and produced a proposal for research in that area.
3. We have studied the methodologies and tools provided by the commercial CAD industry. This has included: 1) attending a number of industry conferences; 2) meeting with CAD vendors to understand their future plans; 3) sponsoring and/or attending training courses on widely-used commercial CAD tools including Verilog, Synopsys Synthesis, HLD Floor Planner, and the Motive Timing Verifier.
4. We have studied the needs and problems faced by Sun's hardware designers, meeting with SMCC and SME staff to discuss their future needs. We have participated in the weekly meetings of several ASIC development projects. We have run the Motive Timing Verifier on the design of one of the ASIC designs.
5. Hired initial CAD project research staff.

References

¹Wilhelm, N.C. "Why Wire Delays Will No Longer Scale for VLSI Chips." *Sun Microsystems Laboratories Technical Report SMLI TR-94-44* (August 1995).

Conceptual Indexing

William A. Woods

william.woods@east.sun.com

Overall Objective

The objective of the Conceptual Indexing project is to improve people's ability to find information in online material. Our approach is to use semantic relationships among concepts and natural language processing and knowledge representation techniques to deal with the differences in terminology between the way queries are expressed and the way that desired information is worded.

Objectives for FY97

Our goal for 1997 was to deliver a version of the system (Nova) that can be used by others to build applications, to produce a technical report describing the technology, and to publicize the technology within Sun. A secondary goal was to begin exploring new applications.

Description

The Conceptual Indexing project has developed techniques for indexing and organizing information in structured conceptual taxonomies that facilitate browsing and retrieval of specific information in response to specific information needs. These taxonomies are structured networks of concepts and relationships among concepts that can be used to relate the terminology of a request to terminology used in information that may satisfy the request. Our goal is to understand the use of such structures in applications such as online documentation, online information access, hypertext publishing, catalog indexes, and information access over networks, and to develop a technology for doing this efficiently.

Conceptual Indexing refers to a technology for organizing facts, ideas, words, phrases, and descriptions into a structured taxonomy that can be used as an organizing structure to improve the performance of information retrieval systems and also as a structure to support human browsing and navigation in "conceptual space." We have implemented a conceptual

indexer that will automatically extract words and phrases from text and organize them into a taxonomy that can be browsed by a user and also used by an information retrieval algorithm to make connections between terminology used in a request and related terminology used in material to be found. We have also implemented a “dynamic passage retrieval” algorithm that can use this taxonomy to find specific passages in response to specific requests.

Accomplishments

This year, we have added a new application of conceptual indexing, developed by Jacek Ambroziak, that automatically conceptually indexes Web pages as users browse the Web. Called “BrowseGuide,” this system attaches to a Web browser and not only indexes the pages viewed, but automatically indexes the pages one step ahead of what has already been seen. An active window attracts the user’s attention to any passages that it finds that are relevant to the user’s specification of interests. This provides a kind of “peripheral vision” that attracts the user’s attention to items of interest that would otherwise be beyond the user’s field of view. BrowseGuide also provides an excellent conceptually-organized bookmarking capability for every page the user visits, including all of the conceptual content of those pages.

In addition, we began an internal project for Sun’s internal information systems group to evaluate the capabilities of search engines for use with Sun’s internal Web pages. To this end, we collected a set of queries from a survey of Sun employees and began the process of evaluating the results of a candidate search engine for a subset of those queries. We also plan to use these queries to evaluate our own technology.

We have continued work on a “gisting browser” that assists Web searchers when they encounter pages that are not in their own language. Specific attention this year was given to the language identification module and to dealing with the character sets of different languages.

Finally, we began a project this year to explore the use of conceptually-indexed knowledge to support the activities of a community of users with a common interest. Specifically, we collected a number of knowledge sources of relevance to Java programmers and made them available with conceptually-indexed access. We plan to use this system in the coming year to explore the role of information search in the activities of

programming and debugging, and to explore ways that an information community can cooperate to share in the generation of collectively useful information.

References

Publications

Adams, G. and P. Resnik. "A Language Identification Application Built on the Java Client/Server Platform." Presented at ACL 97/EACL 97 workshop: *From Research to Commercial Applications: Making NLP Technology Work in Practice*, Madrid, Spain (July 1997).

Kuhns, R.J. "A Survey of Information Retrieval Vendors." *Sun Microsystems Laboratories Technical Report SMLI TR-96-56*.

Woods, W.A. "Beyond Search Engines." Presented at Sun's West Coast CTO Conference: *Managing the Next Generation of the Technology Enterprise*, Menlo Park, Calif. (April, 1997). (Conference slides are available at: <http://www.sun-cto.com/CTOConf97>).

Woods, W.A. "Conceptual Indexing: A Better Way to Organize Knowledge." *Sun Microsystems Laboratories Technical Report SMLI TR-97-61*.

Distance Learning

Randall B. Smith

randall.smith@eng.sun.com

Overall Objective

The overall objective of this project is to investigate distance learning methodologies, and to build and demonstrate technologies that support these methodologies. Our particular focus is on real-time communication within a small, distributed group.

Our approach has two components: a learning paradigm and a technology platform that supports experiments with the learning paradigm.

The technology platform, called Kansas, is a shared, 2D virtual reality that supports audio and video links among participants. Because it is written on top of the Self language, it is easy to make deep changes to the system. Programmability is an important feature because exact future needs cannot be completely determined in advance. (See “The Importance of a Programmable Environment” below.)

The learning paradigm is called Distributed Tutored Video Instruction (DTVI), in which a small group of participants gather in a virtual space to study videotaped lecture material. Based on years of study of the non-virtual analogous paradigm (Tutored Video Instruction [TVI]), we believe we can show that DTVI is as effective or more effective than attending a real lecture.

Objectives for FY97

- Compare TVI with DTVI (its distributed counterpart) in classes at both California Polytechnic State University at San Luis Obispo (CalPoly) and California State University at Chico (Chico), using:
 - Formal statistical comparison of student grades.
 - Statistical comparison of process differences.
 - Interviews with students, faculty and facilitators.

- Fully integrate digital audio and video into the Kansas “world kit,” and use it to create a digital DTVI platform.
- Technology transfer: help, establish, support, and advise collaboration technology projects in JavaSoft™.

Description

The Vision: Small Group Real-time Interaction at a Distance

We are preparing for a global shift in the world's network. When latency drops and bandwidth rises beyond critical thresholds, people from around the world will be able to interact in real-time, with audio/video links in a shared, computationally-rich space. When this happens, everyday life will change profoundly. In particular, corporate and public education will be transformed. We are creating new ways to teach and learn that are appropriate to this emerging interactive network.

The DTVI Learning Paradigm

One way to make a new learning environment is to virtualize existing approaches. TVI has been shown to be more effective than conventional classroom lectures. In TVI, a small group studies a videotape of the lecture with a facilitator (or “tutor”). In DTVI (“D” for “Distributed”), audio and video links among participants link them together while they study the tape from their desktops.

The Flexible Classroom Paradigm

Computer-created virtual spaces can easily become crowded. When a small group is limited to a single common space, all audio is heard by everyone, and each participant is essentially face-to-face with everyone else. In contrast, real world classroom sessions often break up into periods of pairwise or individual work. The flexible classroom paradigm uses a spatial metaphor in the computer to allow participants to “travel” through a space encompassing the total virtual classroom. This supports subgroup formation, so that several smaller sessions can proceed simultaneously without mutual interference. Freedom to “travel” among groups is useful for a facilitator or teacher who wishes to offer help, and is useful for a student who needs to seek out help.

The Importance of a Programmable Environment

We are entering new, unexplored areas, and the ability to make new functionality has proven valuable. For example, we needed to create new user interfaces and to add facilities that redirect bandwidth among video streams. Furthermore, in a flexible media like Kansas (based on Self), a programmer can work with an instructor *in Kansas* to build new kinds of learning environments (such as construction kits or simulations), for later use by students.

Accomplishments

1. CalPoly and Chico each collected a year's worth of student data using analog and (at Chico) digital platforms, involving over 300 students in eight different courses. The data consists of:
 - Performance data (grades).
 - Extensive process data (videotapes of sample sessions were reviewed, and participant statements were counted and categorized).
 - Extensive interviews with students, facilitators, and faculty.

The resulting 20-page interim report presents the primary finding on the data collected to date: so far we find no significant difference in student performance (as measured by grades) between DTVI and TVI.

The interim report also statistically compares, analyzes, and discusses the group interaction process as revealed by the process and interview data.

2. The Kansas "shared world construction kit" was enhanced by adding:
 - Support for user capability objects that govern per-object access to user interface elements.
 - Improvements to audio and video streaming and video display.
 - New "small computer" objects to give users information about and control over each participating host.
3. The resulting enhanced Kansas system was used to build a digital DTVI platform which, in addition to the specially designed software, consists of ten workstations, cameras, audio headsets, an interface for

session recording, and a specially situated 100-baseT network. This platform was installed at California State University at Chico and used in two classes during the Fall 1996 semester.

- A session start-up, tear-down interface was constructed.
 - The new capability was used to create four levels of user types: student, facilitator, coordinator, and wizard.
 - A separate audio world holds all the audio control objects for each participant. The facilitator and coordinator can roam freely through this world to access any station's audio control.
4. Student feedback regarding the new digital platform was elicited with an assessment questionnaire and a series of interviews. The digital platform was modified in response to student feedback:
 - Network audio was extended to include full stereo and arbitrary left-right placement of each channel.
 - A new audio user interface for each student allows independent pan of voice and VCR.
 - Better audio electronics were provided.
 - A video image size adjustment facility was added, and performance was tuned.
 - Added a system to allow anonymous polling and per-image labels to show participant names.
 5. Initiated the JAMM (Java Applets Made Multi-User) project in JavaSoft, to investigate the feasibility of using the Java programming language to support Kansas-like application screen sharing. The effort involved two students for six months, and resulted in a paper (to be published). JAMM technology could be a future product base for distance learning.
 6. Wrote invention proposals for video conferencing camera off-screen self-detection, and for capability objects.

References

Publications

Begole, J., C. Struble, C. Shaffer, and R.B. Smith. "Transparent Sharing of Java Applets: A Replicated Approach," to appear in *Proceedings of UIST'97 Conference*.

Smith, R.B. "Kansas: a Dynamically Programmable Multi-user Virtual Reality," to appear in *1997 UNESCO Workshop on Learning Environments*.

Smith, R.B. "Reusability Lessons from Reality (...and how we tried to capture them in Self)." *Proceedings of OOPSLA 96 Conference*, San Jose, Calif.

Smith, R.B. and D. Ungar. "A Simple and Unifying Approach to Subjective Objects." *TAPOS*, 2(3) (1996).

Smith, R.B., M. Wolczko, and D. Ungar. "From Kansas to Oz: Collaborative Debugging When a Shared World Breaks." *Communications of the ACM* 40(4) (April 1997) .

World Wide Web

<http://www.sunlabs.com/research/distancelearning/kansas.html>

<http://www.sunlabs.com/research/distancelearning/dtvi.html>

Forest

Mick Jordan

mick.jordan@eng.sun.com

Overall Objective

The Forest project began as an investigation into the use of *persistent programming systems* as the basis of the construction of software development environments. This objective has both broadened and narrowed during the project history. It has broadened into pursuing persistent programming for a wide variety of application areas, and narrowed by focusing on the development of *orthogonal persistence* for the Java programming language.

As a consequence, Forest has effectively split into two sub-projects: first, the development of orthogonal persistence for Java (OPJ), and second, the use of OPJ as the infrastructure for building a software development environment. The two projects are inextricably linked since a characteristic of orthogonal persistence for Java is the persistent binding between object instances (data) and the methods (code) that define their behavior.

Objectives for FY97

- To develop and release a prototype system that provides orthogonal persistence for Java.
- To develop a prototype environment to support large-scale, multi-user development of Java applications and related content such as HTML, using orthogonal persistence for Java as infrastructure.
- To investigate the use of orthogonal persistence for Java in other application domains.

Description

Modular System Building

The goal of the Modular System Building project is to construct an environment that effectively supports the large scale development of software and other computer-generated content, for example, HTML documents. Specifically, we aim to lower the barriers to component reuse by providing the illusion of a global component infrastructure that has the properties of stability, reliability, and availability.

We believe that the constraints imposed by the current de facto integration technology of file systems and file formats are a significant barrier to realizing such an environment. Our goal is to demonstrate that an infrastructure of typed, persistent objects is far superior and can promote a paradigm shift in the way in which software is developed and distributed.

Our approach is to integrate authoring tools, configuration management, system building, and the mechanisms for collaboration into a single, but extensible, framework based on fine-grain persistent objects. To provide scalability and reliability we reject the traditional approach to system building exemplified by *make*-based tools, which assumes a world of inherently mutable objects, and choose to follow the approach pioneered by Vesta.¹ The key ideas of this approach are as follows:

- System building is modeled as a function that generates a set of *derived* objects by transforming a set of immutable *source* objects.
- A system is described using modularization and abstraction techniques similar to those provided by programming languages like Java. This allows the description to scale to very large systems, even beyond the scope of a single organization.
- The evolution of a system is modeled as a sequence of atomic changes to the set of source objects and the associated build functions. Each step in this evolution results in a completely specified, immutable snapshot of the system that provides the ability to reliably generate the derived objects. This characteristic can be used to facilitate high availability through caching and also to control storage utilization.

In Forest, we aspire to a very fine grain snapshot, so that the grain is essentially invisible to the user during normal development, yet provides stability and rollback when needed. To achieve this requires a very efficient and non-intrusive persistent object technology. Since the majority of the tools in the environment manipulate graph-structured data structures, the candidate technologies are object-oriented databases and persistent programming languages. Since we cannot expect to rewrite these tools in their entirety, we have chosen to pursue orthogonal persistence for Java as the implementation language because of its ability to incorporate existing Java code unmodified.

We are building a prototype of this environment, code-named JP, to validate the approach and measure performance. To build this system we are using an earlier prototype, JP-lite, developed in FY96 in Tcl/Tk and based on file system technology, as the development environment.

Orthogonal Persistence for Java

Orthogonal persistence for Java provides the basis for the Modular System Building project. However, it is also widely applicable in other domains where there is a need to store complex object structures that outlive the execution of a single program. The project is a collaborative research effort with the University of Glasgow. Originally called Persistent Java, or PJava, the project now goes by the code name PJama, to avoid conflicts with Java trademarks.

Application programmers exploit the object-oriented capabilities of Java to model the form and behavior of the service that the application provides. The provision of *persistence* allows these models to have whatever lifetime is appropriate for the application, from microseconds to years. The requirement of *orthogonality* ensures that persistence is available to all entities in the Java language. Without this capability, a substantial amount of the application development effort must go into explicitly managing alternate forms of persistence, such as file formats or relations in a relational database. Orthogonal persistence for Java implies providing persistence for class instances (metadata) and also for threads, since these are standard features of the language.

Achieving the goals of orthogonal persistence with adequate performance requires that it be provided as part of the Java platform. In practice, this means modifications to the virtual machine layer to provide the illusion of

a window (cache) onto the potentially very large object space in the persistent store. The approach taken by PJama₀, the first of a series of successively more capable implementations, is to augment the virtual machine with a persistent object cache, separate from the normal Java heap, and arrange matters so that the virtual machine is minimally affected by the object cache.

Accomplishments

Much of the project's effort in FY97 was focused on developing and publicizing the initial prototype of orthogonal persistence for Java. We also developed some of the infrastructure for JP, in particular, the configuration management system.

1. Development of the PJama₀ prototype:

- The very first prototype came online just prior to the start of FY97. During Q1, several experiments were conducted with the prototype to assess its performance and functionality, for example, exploiting orthogonal persistence for servlets in the alpha release of the Java Web Server. The experiments were publicized in the paper, "Early Experiences with Persistent Java."
- A release of the prototype was made internally within Sun and at Glasgow in September. Development continued, particularly in the area of object cache management, culminating in a version that was deemed suitable for external release. This external release version was made available under a signed license agreement to academic research groups, but not to the world at large.
- Further work on object cache management and multi-threading support has resulted in a stable prototype that is capable of managing persistent stores in the hundreds of megabytes size range.
- Integrating with version 1.1 of the Java Development Kit has occupied a considerable amount of time and effort owing to its increased size and complexity. We had hoped that a version of PJama₀ for JDK™ 1.1 would be available in Q3FY97 but this has been delayed until Q1FY98. We appear to have reached the limits of development with this line of virtual machine technology.

Choosing a new virtual machine platform will be a key task in FY98.

- A number of talks and demonstrations of PJama₀ were given during the year. SunLabs demonstrated PJama₀ at OOPSLA'96 in Q2 and Glasgow demonstrated PJama₀ at ICDE'97 in Q4.
- SunLabs sponsored the First International Workshop on Persistence and Java, organized by Glasgow and held in Scotland in September 1996. The proceedings were published as *Sun Microsystems Laboratories Technical Report SMLI TR-96-58*.

2. PJama Spin-off Projects:

Several spin-off projects were started during FY97:

- Based on the success of the experiment with the Java Web Server, a project was started in Q3 to encapsulate all persistent server states as persistent Java objects, instead of files. Unfortunately, owing to the delay in completing the JDK 1.1 version of PJama₀, this project was put on hold until FY98.
- A project was started to port PJama₀ to the JavaOS platform, an implementation of the Java platform on bare hardware. The integration with PJama₀ offers several attractive possibilities: first, to exploit the object caching of PJama₀ to mitigate the memory limitations of a network computer; second, to provide a cheap and efficient server platform for persistent object stores.
- We began an investigation into the performance characteristics of persistence technologies for Java objects, including PJama₀ and its major competitors. We are particularly interested in establishing the total cost of an entire system, given that many technologies are based on heavily layered implementations.
- Certain areas of the Java Language Specification pose subtle problems for orthogonal persistence—for example, class initialization and finalization, the semantics of *transient*, and the stability of hashcoding. We are investigating these and proposing solutions.

3. Development of the JP environment:

- We continued to use and evolve the Tcl/Tk-based JP-lite prototype over the year, using it to build our internal and external Web sites, as well as supporting its own development and all of

our work in Java. This provided a constant reminder of the limitations of the file system infrastructure.

- The basic organizational model of JP, that of a federation of projects containing versioned packages of source objects, was implemented in Java. Tools to import from and export to a file system were also written. A benchmark, originally developed in C++ for the ObjectStore object-oriented database, that measures the performance of the configuration management system, was converted to JP and provided a useful stress test for PJama₀.
- During Q4 we began an experiment in the persistence of user interface components, using the Swing set that will become part of the Java Foundation Classes (JFC).
- To support a geographically-distributed federation of JP stores, we plan to use Java Remote Method Invocation (RMI). Currently, RMI requires foresight by the programmer that an object will be accessed remotely, a requirement that is remarkably similar to the foresight required by many persistence mechanisms. We are conducting an experiment that supports more orthogonal distribution, leveraging the capabilities of PJama₀.

References

¹Levin, R. and P.R. McJones. "The Vesta Approach to Precise Configuration of Large Software Systems." Digital Systems Research Center, TR105 (June 1993).

Publications

Atkinson M.P., L. Daynès, M.J. Jordan, T. Printezis, and S. Spence. "An Orthogonally Persistent Java." *ACM SIGMOD Record* (December 1996).

Jordan, M.J. "Early Experiences with Persistent Java." *Proceedings of the First International Workshop on Persistence and Java. Sun Microsystems Laboratories Technical Report SMLI TR-96-58.*

Jordan, M.J. and M. Van De Vanter. "Modular System Building with Java Packages." *Proceedings of the 8th International Conference on Software Engineering Environments, Cottbus, Germany*(April 1997).

Forest

World Wide Web

<http://www.sunlabs.com/research/forest>

High-Speed Networking

Raphael Rom

raphael.rom@eng.sun.com

Overall Objective

To explore new opportunities and become a knowledge center in the core emerging technologies of high-speed networking technologies, and to investigate value-added synergy between computing and communication platforms.

Objectives for FY97

Develop initial concepts into tangible technologies and lay the groundwork for subsequent advanced research in value-added networking functions. In addition, the group should make its knowledge and capability available to all of Sun.

Description

The High-Speed Networking group is designing and implementing OpeNet, a high-performance, open, vendor-independent ATM network control platform (also known as a connection management platform). The existence of such a platform is crucial for the success of the ATM market, and will enable the development of many universal network services applications (network management, datagram service, backbone switching, and others). The OpeNet design seeks to achieve its goals by devising efficient algorithms and by using underlying switching capabilities efficiently.

Accomplishments

During FY97, the basic components of OpeNet have been implemented. The software design is based on streams, and is implemented within the Solaris™ kernel. It currently runs on Solaris 2.4, Solaris 2.5, and the recently announced Solaris 2.6. The code consists of several disjoint modules such as route computation, database maintenance, routing information distribution, connection management, etc., all integrated into a

single unit. These modules can be used independently of OpeNet, such as within PNNI implementations or within other signaling platforms.

To ensure vendor independence, OpeNet is implemented over a canonical switch rather than a specific one. To demonstrate the OpeNet software, an emulator was developed that includes the OpeNet code, a switch emulation module, and a Tcl/Tk™ graphical user interface. The actual OpeNet code runs over the emulated switches and the state of the network is made visible through the interface. The emulator itself can emulate the signaling for fairly large networks: a SPARCstation™ 5 simulates 30 nodes and an UltraSPARC™ 140 simulates over 160 nodes! Beyond correctness testing, the emulator assists in determining bottlenecks of signaling computation and serves as a crucial debugging tool.

Implementing OpeNet on real switches requires that a clear interface between the switch and the controller be defined. To that end, we are developing an extension to the GSMP protocol that includes quality of service specification. Such an interface is generic and will fit switches as well as routers and telecommunication cross-connects. This development is being done in cooperation with other data communication and telecommunication companies.

As groundwork for subsequent development of networks and switches, we devised a management architecture that allows the retrieval and setting of network information both in-band and out-of-band, and supports both datagram transactions as well as data streams. This architecture takes full advantage of the OpeNet architecture and the GSMP switch interface.

Interest in OpeNet activity has given rise to a broad collaboration with several data communications and telecommunication companies, as well as several universities. The latter include Columbia University (Prof. Lazar), UCSC (Prof. Garcia-Luna), UCLA (Prof. Gerla), the Technion (Prof. Cidon and Prof. Sidi), and the University of Pretoria (Prof. Roos). These collaborators will help in the development of OpeNet by implementing the software in their facilities, by adding additional capabilities, and by enhancing its functionality.

The High-Speed Networking group actively represents Sun in many national and international forums. The group maintains active membership in the ATM forum, the CIF (Cells in Frames) group, and the 60GHz

working group. In all these groups, active membership entails making technical contributions and representing Sun's interests.

Members of the group have been providing general assistance to the rest of Sun in many areas of networking, such as analyzing flow control of Ethernet switches, designing a buffer management scheme, designing an RSVP implementation, and providing critical review of networking products.

References

Publications

Cidon, I., A. Khamisy, and M. Sidi. "Analysis of Queueing Displacement Using Switch Port Speedup." *Proceedings of Infocom'97*, Kobe, Japan (April 1997).

Levine, B. and R. Rom. "Approaches for Reliable Concast Support in ATM." *Proceedings of the DIMACS Workshop on Architecture and Algorithmic Aspects of communications networks*, New Jersey (January 1997).

Patents

Title: A Method and Apparatus for Explicit Rate Flow Control in ATM Networks
Inventors: C. Buckley, I. Cidon, A. Khamisy, R. Rom
Issued: 03/96 (8/619,040)

Title: Method and Apparatus for Implementing Self-Organization in a Wireless Local Area Network
Inventor: R. Rom
Issued: 05/07/96 (5,515,509)

Title: Systems and Method for Traversing ATM Networks Based on Forward and Reverse VC Labels
Inventors: I. Cidon, T. Hsiao, P. Jujjavarapu, A. Khamisy, R. Rom, M. Sidi
Issued: 11/26/96 (5,579,480)

World Wide Web

<http://www.sunlabs.com/research/hsn>

Internetworking

Phil Rosenzweig

phil.rosenzweig@east.sun.com

Overall Objective

Our overall objective is to develop a competency center in Boston in Internet Protocol (IP) networking that develops new ideas in network protocol technology into product opportunities for Sun. The group, called the Boston Center for Networking (BCN), was started in December 1996 as an advanced development effort under the Chief Technology Officer and became part of Sun Microsystems Laboratories in April 1997.

The group's first project is in reliable multicast protocols and related services. It addresses the problem of how to broadcast content over IP networks with reliability, conservation of bandwidth, and timely delivery.

Objectives for FY97

- Investigate requirements of the network infrastructure for multicast-aware applications and discover current technology barriers.
- Develop the architecture, interfaces, and functions for a set of services that support multicast-aware applications.
- Develop an enhanced, reliable, multicast protocol that will be highly scalable for one-to-many delivery.

Description

Emerging applications that deliver data from one sender to many receivers, such as news, real-time financial information, or software distribution, are currently being deployed using "push" technology based on HTTP protocols. Since HTTP is point-to-point, or unicast, it inherently does not scale to large numbers of receivers. We expect that reliable multicast will replace HTTP as the protocol for delivery of push content. It will also be used as the supporting protocol for collaborative applications, such as a shared whiteboard, which require many concurrent senders and receivers.

Slingshot is a set of libraries and services for building multicast applications. It enables building applications that broadcast data from a “sender” to “receivers” over “channels” with distributed control over content mix. It includes a dynamic filtering mechanism that uses Java classes that are sent into the network for the purpose of interpreting and directing the data downstream from the server (hence the name *Slingshot*).

Slingshot supports multiple concurrent reliable multicast transports through a “federated” interface. This provides isolation to applications and allows third parties to plug in other reliable multicast transport implementations. These will be selectable by applications based on type of service needs.

Slingshot can be used by next generation applications as the transport for delivery of content to very large constituencies. As compared to unicast (point-to-point protocols), reliable multicast enables broadcasting to groups of receivers, ensuring bandwidth conservation and timely delivery. The Slingshot reliable multicast transport protocol is being designed for high scalability, aiming for at least two orders of magnitude more receivers than other proposed protocol implementations. Slingshot includes new services for multicast address allocation and channel management (channels are streams of data running over the transport). Slingshot software is designed for implementation in end-systems and depends only on unreliable multicast in the network infrastructure. All code resides within senders or receivers. It will be completely implemented in Java.

Accomplishments

1. Architecture:

- In looking at multicast-aware application requirements, we discovered the need to support various types of data transmission. These include bulk data transfer, live data, resilient streams, shared data, and hybrid. Each has specific requirements for the number of senders, when data can be sent, whether or not the data is real-time, consistency, ordering of data as it is received, and degrees of reliability. It became obvious that no single transport protocol would be appropriate for all needs. This influenced the decision to construct a framework to support multiple transport protocols which would be selectable by applications.

- To support multiple protocols, we needed to devise a general purpose interface from the application to the transport manager and another from the individual transport protocol stack up to the transport manager. This would include the ability to transfer data and information for selecting and configuring transport parameters.
- Similar to transport, an architecture for allocating multicast addresses was developed which would facilitate more than one concurrent scheme for providing applications with multicast addresses.
- Layered above the transport and address allocation services is channel management. We determined that the needs for overall control of multicast transmission include services for advertisement and subscription, authentication and encryption, dynamic filtering, high-level reliability, and managing receiver groups. The decision to layer channels cleanly over the transport makes Slingshot lightweight in that the channel layer is optional for minimally-scoped applications, but provides robust services for large-scale applications if needed.

2. Reliable Multicast:

- We have looked at many reliable multicast protocols proposed in the academic and Internet communities. In the Internet standards world, the topic of reliable multicast currently resides in the Reliable Multicast Research Group of the Internet Research Task Force.
- We have developed a reliable multicast transport protocol which is designed for a single sender and many receivers. It attempts to address a number of the challenges of large-scale reliable multicast: ack/nak implosion, efficient retransmission of lost data, and congestion control. Since reliable delivery of data packets implies acknowledgment of receipt, transmission of control information must be minimized, retransmission of lost data must be localized as close to the receivers as possible, and an end-to-end flow control mechanism is required to keep data flowing smoothly even though there may be a wide variety of topologies and link speeds connecting the receivers to the network.

3. Collaborations:

- We are cultivating several collaborations with large customers and other vendors in the development of Slingshot. The objectives are to prototype and pilot multicast-aware applications in real-world environments and to explore further advances in reliable multicast protocol design.

Java Topics

Steve Heller

steve.heller@east.sun.com

Overall Objective

The Java Topics group explores advanced technologies we believe will be useful for the Java programming language nine months out and beyond. Our focus in FY97 was the development and deployment of soft real-time memory management systems for Java.

Objectives for FY97

- Study the automatic memory management literature.
- Build the infrastructure necessary for memory management research.
- Consult within Sun on a broad range of Java topics.

Description

Because the Java language depends on automatic storage management, we need to understand implementation techniques that will perform well in many settings, from interactive applications to high-load servers, to batch computations. Garbage collection (GC) has been studied since the 1950s, and there has been a steady stream of algorithmic improvements. We view “memory management” as larger than GC per se. Issues of object layout, interaction with the virtual machine (VM), and interaction with compilation systems are all important.

Java Topics set out to develop and deliver soft, real-time memory management systems for the Java program. To facilitate the research, a version of JavaSoft’s Java Virtual Machine was modified to support exact GC.

Supporting exact GC requires the identification of all pointers to objects. Some approaches identify this set “conservatively,” precluding many interesting and efficient techniques for memory management. The VM developed by the Java Topics group, the exact VM, was designed to allow

flexibility both in the layout of objects and stacks and in the algorithm used for GC. In this way, the path is paved for studying the complete Java memory management problem.

An important goal of this work is to produce a set of memory management interfaces to support pluggable GC as well as flexibility in other aspects of memory management.

Accomplishments

1. Prepping the Java virtual machine for GC research:
 - Our main task in FY97 was the design and implementation of the exact VM. Now operational, the exact VM supports exact and inexact GC as well as a variety of other features.
 - Objects were abstracted to facilitate changing object layout. Handles, optional in the exact VM, provide a level of indirection for all object accesses, making object relocation easier, but execution less efficient. As the execution speed of Java VMs improve, the ability to point directly to objects will be ever more valuable.
 - The exact VM also supports the read and write barriers necessary for various GC strategies. Similarly, stack layout has been abstracted to facilitate experimenting with different strategies.
 - The Java Native Method Interface is used to track references to Java objects from outside Java. To track references to Java objects from within the VM itself, a more efficient interface called the Low Level Native Interface was developed.
 - Finally, we have begun to specify a GC interface and already have a host of collectors that plug into the exact VM.

2. Java Consulting:
 - The Java Topics group also consulted internally at Sun on language semantics, security, virtual machines, Java on the Macintosh, internationalization, numerics, architectural support for GC, and other issues.

Guy Steele was awarded the first ACM SIGPLAN Programming Languages Achievement Award at the ACM SIGPLAN Conference on

Programming Language Design and Implementation, Las Vegas, June 17, 1997. Guy has had a profound influence on many programming languages throughout his career.

References

Publications

Agesen, O. "Design and Implementation of Pep, a Java Just-In-Time Translator." *Theory and Practice of Object Oriented Systems*, 3(2), (1997).

Agesen, O., S.N. Freund, and J.C. Mitchell. "Adding Type Parameterization to the Java Language." *ACM Conference on Object-Oriented Programming Languages, Systems, and Applications*, Atlanta, Georgia (October 1997).

Agesen, O. and S. Sankar. "Compiler." David Hemmendinger, Anthony Ralston, Edwin Reilly (Editors). *Encyclopedia of Computer Science*, 4th Edition, to be published in 1998 by International Thompson Computer Press.

Gosling, J., B. Joy, and G. Steele. *The Java Language Specification*, Massachusetts. Addison Wesley (1996).

Java Wallet

Ted Goldstein

ted.goldstein@eng.sun.com

Overall Objective

The Java Wallet project is creating a software framework for electronic commerce that leverages the special capabilities of Sun's Java platform. The work reported here was performed in close collaboration with JavaSoft.

Objectives for FY97

- Create the Java Electronic Commerce Framework (JECF).
- Create a security model that models the limited trust relationships found in all business-to-business relationships.
- Publish the JECF and its security model.
- Create a payment system based on an off-the-shelf Smart Cardsmart card-based stored-value payment system.
- Explore the utility of the JECF and its security model in smart cards.

Description

Computers and commerce have been closely associated for many years. Typically, the transactions are entities that have a direct business relationship, such as a merchant-bank relationship or an electronic data interchange (EDI) relationship. The Internet holds the promise for a very different sort of electronic commerce relationship that we may think of as spontaneous business relationships. Spontaneous business relationships exist in the conventional world. Walking into a store, a buyer need only present currency or another trusted document to conduct a transaction. The physical presence of the buyer and the seller and the reputation of the store are usually all that is needed to create trust between the buyer and the seller. Local governmental authorities act to reinforce the trust

environment with laws and courts to regulate commerce and provide dispute resolution.

But the Internet has no indisputable location. Documents have no inherent authentication. What is needed is a completely new set of mechanisms to recreate both the authenticity and trust environment. Cryptography begins to provide a technology for providing authenticity, but cryptography by itself does not create a trusted environment. Two more elements must come together to create it. There need to exist well-known, trusted guarantors of commerce, such as banks and governments. There must also be a way to represent these trusted relationships to the user to create a trusted environment for commerce.

The Java programming language and application programming interfaces (APIs) provide a reliable, secure platform to deliver applications on Internet and Intranet networks. Without Java, users who download applications from these networks could be vulnerable to dangerous software viruses. All of the popular personal computer operating systems are vulnerable to such attacks. Of course, Java executing on conventional personal computers cannot totally prevent virus attacks. But if users are careful and load only Java software and software from totally reliable sources, they then lower their chances of software virus infection. The ultimate solution is an entirely Java operating system running directly on a processor.

Java provides safety and security by restricting the use of potentially dangerous operations. Unfortunately, many beneficial applications, such as electronic commerce applications, are also prevented from executing these operations. The limitations of Java motivate a new security model called the Gateway that extends the current Java security model. This new model is the core of an entire application framework for financial applications called the Java Electronic Commerce Framework (JECF). The JECF and the Gateway model meet the requirements of electronic commerce and improve the Java security model by providing:

- A common graphical user interface
- A secure encrypted database
- An interface to strong cryptography

- A structure for purchasing applications

Internet Electronic Commerce Requirements

Internet electronic commerce has many demanding security requirements:

- Transactions must be private. Only the intended participants in the transaction may know the details of the transaction.
- Transactions must be authentic. The identities of the buyer and the seller have to be true and accurate. Both participants in the transaction must agree to the terms.
- Transactions must be auditable. Auditing detects fraud and mistakes. Auditability must be managed against the needs of users' right to privacy. Auditing also assists in tuning efficiency.
- Financial transactions must obey the law. It is very difficult to define the venue of Internet electronic commerce transactions. A transaction that is legal at the seller's location may be illegal at the buyer's location. Taxes must be paid at one or both ends of a transaction. Cryptographic control laws must be obeyed. Consumers will want privacy. All of these requirements are in conflict. World governments will be spending many years learning to cope with the conflicting demands of the Internet. Electronic commerce systems must have the flexibility to adapt.
- Existing systems must be extended to accommodate new applications. The financial industry and the computer industry continually develop new payment instruments, financial services, and technologies. Users can take advantage of these new applications only if the infrastructure can accept change.
- Payment services must cooperate. Just as with physical world commerce, Internet customers want to use cash, checks, credit, debit, and discounts to buy things. Therefore, multiple payment mechanisms must work together.
- Financial services must cooperate. Few customers use just one bank or broker, but instead have relationships with many financial institutions and many merchants. Financial goal analysis and

government taxation require a total and complete integration of a customer's entire financial picture. Financial applications must communicate to integrate a financial portfolio. Trusted third-party financial applications can greatly assist users with managing their finances and reaching their financial objectives. Therefore, even home banking and broker systems must work together.

Current software protocols and conventional financial applications do not yet satisfy these requirements. Financial technology companies and institutions have made great strides in electronic commerce protocol and application design, but there is still need for significant improvement to allow the Internet to meet users' requirements for purchasing and financial services.

By employing cryptography and digital signatures, protocols such as Secure Electronic Transactions (SET), Secure Key Internet Protocol (SKIP) and Secure Socket Layer (SSL) enable privacy and authenticity of network communication. These protocols, however, do not enable inter-application communication, cooperation, or auditability.

Today, when users want to add functionality to a consumer financial application, they have to discard the existing application. As more and diverse applications appear on the market, discarding applications and re-entering data ceases to be a viable technology migration strategy. End-users need a system that enables data sharing between applications from different vendors.

Soon, conventional home banking software, home broker software, and electronic wallets will cooperate. To cooperate, these packages must share data with each other. These packages should not try to lock the user into a single application because customers will choose home banking packages that communicate with their home broker software and their electronic wallets.

Limited Trust

To understand the need for the Gateway model, consider the trust relationships in business. All business relationships have defined limited trust. By definition, businesses in a relationship do not have complete access to all aspects of each party's affairs. (Such devotion is reserved for relationships of the heart.) You trust your lawyer for some procedures, and

your doctor for others. Every business-to-business contract specifies roles, responsibilities, rights, and deliverables among the participants. Sometimes rights are transferable, but often they are not.

The business community uses a broad variety of control mechanisms to enforce contracts. Employees, security guards, access badges, locks, and laws provide infrastructure for the physical world to enforce and limit contractual rights. These mechanisms are not infallible. It is still possible to break these mechanisms and violate the contract. When these physical world mechanisms fail, government and the court system mediate the disputes. To quote an old saying, "Locks keep honest people honest; laws discourage dishonest people from being dishonest."

The electronic commerce community needs an equivalent set of software locks and infrastructure to keep honest people honest and to discourage dishonest people. Implementing electronic commerce requires a comprehensive suite of functions such as public key operations, database transactions, smart card exchanges, and elaborate protocol communication. Without software protection mechanisms, rogue software can steal money from users. Even legitimate software may inadvertently violate the contractually-defined privacy of a user's financial portfolio or other records. The Gateway model allows for such a lock to be placed at the point of call of an API. This means that software from different vendors can cooperate in the same address space using an API calling mechanism, without compromising each application's security integrity.

Java Commerce targets developers of secure online spontaneous electronic commerce for both the enterprise and the marketplace that desire a scalable, platform-independent environment. The Java Electronic Commerce Framework extends the core Java platform.

Accomplishments

1. JECF was released in an alpha version with the following components:
 - The JECF can be extended using components called *cassettes*. Cassettes are installed into and become an extension of the JECF. The JECF is persistent on the client system. Installed cassettes are persistent on the client as well.

- Cassettes can extend the functionality of the JECF in three ways: as 1) operations, 2) protocols, and 3) instruments. Operations are high level business processes such as purchasing, frequent flyer mile redemption, or checkbook balancing. Protocol objects implement communication protocols such as SET or OFX. Instruments represent the actual credit card or banking account.
 - The JECF uses the Gateway security model to limit the trust that any software component must place in another component. The Gateway model is an object-oriented design pattern that allows a lock to be placed at the point of call of an API. This means that software from different vendors can cooperate in the same address space using an API calling mechanism without compromising each application's security integrity.
 - The Java Wallet provides a trustable graphical user interface for manipulating all of the above elements.
2. Java Card is Java code on a smart card:
- Java Card 1 is a subset of Java. The specification is a proof of concept that a subset of the Java language can execute a reasonable Java subset in an incredibly limited environment of 12K of electrically-erasable memory and 512 bytes of RAM on an 8-bit CPU.
 - Java Card 2, currently in process, defines a complete smart card application environment.

References

Publications

Goldstein, T.C. "The Gateway Security Model." International Association of Cryptography Conference, *Financial Cryptography '97*, conference addendum. (See: <http://java.sun.com/commerce/doc.gateway.ps>)

World Wide Web

<http://java.sun.com/commerce>

Kanban Compilation

Neil Wilhelm

neil.wilhelm@eng.sun.com

Overall Objective

The objective of the Kanban Compilation project is to deliver dynamic optimization compiler technology to Sun in a usable form.

Objectives for FY97

The objective for FY97 was to initiate contact with the operating companies of Sun and assist them with Java Virtual Machine implementations and issues.

Description

In executing object-oriented programs and executing binaries from other architectures (binary translation), an important common difficulty becomes apparent: little is known beforehand about the program being executed. The difficulty with object-oriented programs arises from such constructs as virtual method calls, because the method invoked cannot be known with certainty until runtime. The difficulty with binary translation arises from such things as virtual method calls, of course, but also from such simple things as the difficulty of separating executable code from data. Thus, optimization must be performed dynamically (at runtime) when and as the properties of the executing program are understood.

The Kanban Compilation project started from the technology developed by the Self project: dynamic optimization for the Smalltalk-like Self language.

Accomplishments

1. The Kanban group took up part-time residence at SunSoft:
 - The group has helped SunSoft define its Java Virtual Machine (VM) product strategy.

- The group has worked on two specific Java VM product issues: the VM and the toolkit.
2. The group collaborated with JavaSoft on Java VM portability.
 3. The group has worked with Sun Microelectronics (SME):
 - Assisting SME with Java VMs and the SPARC™ architecture.
 - Assisting SME with the Java chips architectures.
 4. Student interns assisted with the project:
 - A summer 1996 intern from the University of Chicago.
 - A winter 1997 intern from the University of Queensland, Australia.
 5. Recruiting:
 - Two new staff members joined Kanban in November 1996 and July 1997.
 6. Intellectual property:
 - Nine patent applications filed.
 - Three applications in progress.

References

Publications

Cramer, T., R. Friedman, T. Miller, D. Seberger, R. Wilson, and M. Wolczko. "Compiling Java Just-in-Time." *IEEE Micro* (June 1997).

Smith, R.B. and D. Ungar. "A Simple and Unifying Approach to Subjective Objects." *Theory and Practice of Object Systems*, 2(3) (1997).

Smith, R.B., M. Wolczko, and D. Ungar. "From Kansas to Oz: Collaborative Debugging When a Shared World Breaks." *Communications of the ACM* (April 1997).

Ungar, D., H. Lieberman, and C. Fry. "Debugging and the Experience of Immediacy." *Communications of the ACM* (April 1997).

Media Systems Group

J. Duane Northcutt

duane.northcutt@eng.sun.com

Overall Objective

The focus of the Media Systems Group has been on development of architectural and system-level (e.g., network and operating system) support for digital media such as audio and video. This has been done in the belief that workstations are evolving into primarily communications (as opposed to computational) devices, and that the inclusion of richer media in the networked computing environment enhances the communication of ideas among people, across both time and space.

The Media Systems Group and its precursors have performed work in several broad areas. These include: developing a large-scale system to explore the integration of digital media and workstations (i.e., the High Resolution Video (HRV) Workstation); laying the technical groundwork for SMCC's Sun™ MediaCenter™ video-on-demand server product; and developing numerous media-related subsystems to support the group's research. Some of these projects have become commercial products, e.g., SBus video digitizing board (SlicVideo), RS232-controlled TV tuner with stereo audio, and closed-caption decode (SunTuner™). The Media Systems Group also has a long history of significant contributions to the development of media-related APIs within Sun, as well as contributing to Sun's network, operating system, and hardware support for digital media. Recently, the Media Systems Group has also participated in a series of experiments in various aspects of electronic commerce.

Objectives for FY97

- Redirect the Agorics experiments toward products. After two years of research in the area of electronic commerce with our partners from Agorics, Inc., it has become time to advance these experiments to the next stage of development. The objective for this year was to determine the best way to transition the learning obtained to-date in this area into something that will directly contribute to the development of products.

- Enhance the ability of Sun platforms to support multimedia applications. In keeping with the project's general mission statement, a goal for this year was to continue to provide new technology to make Sun products more effective in the creation, storage/retrieval, transport, and presentation of rich, interactive, networked media.
- Investigate key emerging media technologies in advance of Sun's needs. Another of this year's goals was to select areas of potential future interest to Sun's product organizations and gain direct, firsthand experience with the relevant technologies in advance of when decisions or commitments have to be made by product groups. This is best done by developing early prototypes using early access releases of components and subsystems that may be considered for inclusion in Sun products.
- Identify and explore new potential product directions. A general goal taken from the Labs' charter is to define and develop innovative new concepts that might lead to new product directions for Sun's operating companies. While it was expected that this would be done from the group's media systems perspective, it was not part of the goal to remain within the narrow confines of media systems architecture.

Accomplishments

The key technical accomplishments of the Media Systems Group for FY97 include:

1. JavaMedia API's hand-off:

The group's work on the JavaMedia API was brought to an end this fiscal year, and a significant effort was made to ensure that the work, in all of its complexity, would be successfully transferred to JavaSoft.

Throughout this fiscal year, members of the group continued to consult with JavaSoft to ensure a smooth transition of responsibilities and the successful accomplishment of the initial goals set for this effort.

2. Agorics electronic commerce experiments wrap-up:

At the beginning of this fiscal year, the group's final experiments in electronic commerce were completed, and an effort was made to transition this work into a more product-focused phase.

To this end, some of the technology developed in the course of this effort was licensed to our partners at Agorics Inc., some was shifted to other groups in SunSoft and SunLabs.

3. Creation of an MPEG1 encoding station:

As a logical follow-on to the group's work on SMCC's Sun Media Center product, the group created a set of tools to allow arbitrary videotapes to be encoded in the MPEG1 audio/video coding format for use with the Media Center.

A result of this effort was the creation of a tool based on standard Sun products that, at the push of a button on a Web page, digitizes, encodes, and transfers movies from videotape to the Media Center.

Prior to the creation of this tool, it was necessary to either purchase an extremely expensive MPEG encoder unit, or pay a substantial amount of money to a service bureau to convert analog video into a form suitable for use with the Sun Media Center.

The group cooperated with SMCC's Interactive Services Group to see that this utility was bundled with a developer's kit distribution for the Sun Media Center.

4. Digital video (DV) explorations:

As it was becoming clear that the emerging DV coding standard that is being using in new camcorders and VCRs might become significant to Sun, the group launched a project to develop technical experience in the area.

In the course of this effort, a prototype Java-based desktop video editing package was developed. This tool supports the creation of professional-quality video presentations, and represents an example

of an important authoring tool for creation of content for the Web (or the Sun Media Center).

This prototype editor runs on Sun's current desktop machines and has advanced features (e.g., automatic segmentation, drop-frame timecode, frame-accurate drag-and-drop-based non-linear editing capabilities), which heretofore were thought to require special substantial amounts of specialized (and expensive) hardware.

5. Investigation of the IEEE1394 interconnect:

In conjunction with the digital video work described above, the group developed a high-performance driver architecture and Solaris device drivers for various (PCI-based) IEEE1394 interface boards.

In the course of this investigation, much valuable, practical information was gained about Firewire and its current implementations. In addition, much insight was gained into the current state of audio/video devices which support this interface (i.e., a limited number of camcorders and VCRs), by connecting them to Sun workstations and creating the necessary (data and control) connections to make use of them in a realistic environment.

6. Development of a prototype digital video security application:

With the help of Sun's internal Corporate Security Group, a new project was defined to develop and prototype an all-digital security video application with the intent to replace the people-intensive, analog-based, security video systems currently in use on Sun's campuses.

For this project, Java-based client programs were developed to allow the viewing of current (multicast-based) video output from digital security cameras, as well as to allow operators to search and navigate through archived video on tertiary store for events of interest.

7. Implementation of a network-attached video device (NetCam):

In support of the Digital Security Video effort, the Media Systems Group developed a network-attached device that can compress/decompress audio/video information and send/receive it over a 10/100Mbps Ethernet connection.

This device was designed to be a small (i.e., < 300cc), low cost (i.e., <\$200), low power (i.e., < 10W) device, with a small code footprint (i.e., < 64KB) that includes a multi-threaded real-time executive, complete Internet protocol stack, and application code.

Multiple applications have been developed for the NetCam, including the Digital Security Camera (which transmits compressed data only when significant scene changes occur), the Video Sender (which continuously transmits compressed audio and video), and the Video Receiver (which receives data, decompresses it, and generates baseband audio and video as output).

The applications developed so far all make use of IP multicast and the RTP/RTCP protocols for media transport and device control, and are therefore compatible with Sun's ShowMe™TV™ product and the MBONE teleconferencing tools.

8. Acceleration of color-space conversion:

As different forms of color-space conversion are found in almost every form of video application, high performance color-space conversion is critical for overall system performance when dealing with video. We have worked with different groups within SMCC, SunSoft, and SME to enhance the functionality and performance (through both hardware and software) of color-space conversion functions on Sun platforms.

Members of the group worked with SMCC to add color-space conversion support to the FFB2 framebuffer, and with SunSoft XIL Group, the SMCC Video and Imaging Software Group, and the SME VIS Group to add accelerated support for YUV4:2:2 subsampled data conversion to our platforms.

9. Empirical validation of the SMART scheduler:

Continuing with the group's earlier work on processor scheduling, a Stanford student who has been collaborating with the group for some time has implemented the SMART scheduler within Solaris 2.5.1. Very encouraging results came from running a realistic application mix on the SMART-enhanced Solaris OS and measuring the effectiveness of the scheduler (as well as its overhead), with respect to the standard SVR4 scheduler.

10. Introduction of the Network Terminal (NewT) project:

The group initiated a new project to explore limits of “thin-client” architecture, based on a new partitioning of functionality between clients and servers.

References

Publications

Nieh, J. “The Design, Implementation and Evaluation of SMART: A Scheduler for Multimedia Applications,” to appear in *SOSP '97*. [SML 97-0098] (See: <http://suif.stanford.edu/~nieh>)

Nieh, J. and M.S. Lam. “SMART UNIX™ SVR4 Support for Multimedia Applications.” *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, Ottawa, Canada (June 1997). [SML 96-0277] (See: <http://suif.stanford.edu/~nieh>)

Patents

Title: Software-based Encoder for a Software Implemented End-to-End Scalable Video Delivery System
Inventors: N. Chadda, J.D. Northcutt, G.A. Wall, J.G. Hanko
Issued: 04/15/97 (5,621,660)

Title: Method for Simulating the Parallel Processing of Video Data
Inventor: J.G. Hanko
Issued: 05/06/97 (5,627,966)

Title: Method and Apparatus for Providing Collection Browsers
Inventors: H.A. McIntosh, E. Priyadarshan, A. Ruberg, T. Shea
Issued: 05/13/97 (5,630,042)

Smart Cards and TCO

Diana Neiman

diana.neiman@eng.sun.com

Overall Objective

To investigate smart card technology as a distributed budgeting system for shared resources.

Objectives for FY97

- Analyze published research data on the topic of total cost of ownership (TCO) of computing facilities. Understand all components and cost drivers and report on findings.
- Investigate the potential of smart cards as a means to distribute budget in a closed economy to reduce the number of accounting transactions for internal services, and thus lower cost.

Description

The primary purpose of any internal charging system is to optimize the provision and use of internal resources. If users perceive such resources to be free, it is likely that demand will exceed value. In a market-based internal economy, the user explicitly makes a choice to acquire a product or service and knowingly accepts the costs of that transaction. However, in today's "after the fact" billing systems, the user is protected and separated from the consequences of his actions. This project is exploring how to replace this very costly billing system with a low cost alternative that places authority with the user at the point of sale through the use of smart cards.

Accomplishments

1. TCO Presentation:
 - Completed review and analysis of Gartner, IDC, Forrester, Zona, Deloitte & Touche, and KPMG reports.

- Developed customized TCO discovery and analysis tool with Interpose.

2. Smart Cards:

- Internally published the project plan, and invited comments from interested Sun parties. Unique management accounting possibilities have emerged and will be tested in the coming months.
- Hired small technical team to leverage the work of JavaCard and develop the banking and payment modules necessary for the project.
- Demonstrated remote access to Sun's corporate network using a smart card for authentication instead of the digital token currently used.

Speech Applications

Nicole Yankelovich

Paul Martin

nicole.yankelovich@east.sun.com

paul.martin@east.sun.com

Overall Objective

To develop within Sun a center of expertise on speech technologies; to build a robust, effective environment for speech application development; and to help define Sun platform-level requirements needed to support sophisticated speech interaction.

Objectives for FY97

The major objectives for FY97 were to initiate the specification of a Java Speech API and to move existing natural language and other speech application software development tools to the Java platform.

Description

The Speech Applications project is a multi-year effort encompassing several activities. One of the most important of these is tracking speech recognition, speech synthesis, and other related technologies. By establishing relationships with speech researchers and vendors, we can help to place Sun in a position to take advantage of emerging speech technologies as they gain importance in the marketplace.

Another major activity involves building a testbed for the design and study of speech applications. In this area, we initially focused on the needs of users who might benefit from access to online data while they are away from their computers. For these users, we prototyped a suite of conversational applications which allows them to call up their Sun workstation and verbally interact with tools such as electronic mail and calendar, and data feeds such as stock quotations, currency exchange information, and national weather forecasts.

To construct these speech applications, we initially created a prototype development framework called SpeechActs, in which multiple speech-

driven applications can be integrated. This framework supports a variety of speech recognizers and synthesizers, and includes a natural language component and a unified grammar language. In FY97, many of the tools within this framework have been converted to the Java environment. With these Java-based tools, we have created platform-independent applications, as well as ones that are not specific to a particular speech vendor's products.

A key activity in the design of all speech applications is ensuring a high degree of usability—a challenging task given today's imperfect speech recognition and synthesis systems. The group's research on speech user interfaces involves a series of user studies during the life-cycle of each application development project. The user-centered design approach begins with natural dialog pre-design studies in which human-human conversations are analyzed as the basis for the computer-human dialog design. After a preliminary software implementation, laboratory studies help the team find and correct usability problems. For applications intended for long-term use, field studies are conducted to determine the effectiveness of the speech application over time in real-life settings. Based on these studies, the group has developed and published guidelines for creating effective speech applications.

Accomplishments

After constructing the backbone of the SpeechActs framework in FY94, in FY95 we focused on user testing and feature enhancement, and in FY96, we expanded the scope of our efforts to encompass isolated-word dictation software, speech situated in the office, and multimodal rather than speech-only applications. These new types of applications drove our effort to significantly enhance the SpeechActs Discourse Manager to support database queries that take context into account and to handle various forms of ambiguous user input. In addition, over the five years of the project, our numerous user studies have added to our understanding of how to build effective, fluent speech applications.

In FY97, in conjunction with JavaSoft, we launched an effort to specify a Java Speech API (JSAPI), and we redesigned and reimplemented components of the SpeechActs environment in Java to form the basis of a Java Speech Toolkit for developers wishing to create applications on top of JSAPI. Also in FY97, we developed a technique to use examples to restrict speech recognition grammars.

1. Java Speech API:

- The Java Speech API (JSAPI) specification defines a platform-independent interface to both speech synthesis and speech recognition. The specification, being developed in close collaboration with JavaSoft, is a standard extension to the Java platform. It includes a Java Synthesis Markup Language for cross-platform control of speech synthesizers, and the Java Grammar Format for controlling speech recognizers. Seven external partners (Apple, AT&T, Dragon, IBM, Novell, Philips, and Texas Instruments) are participating with Sun in defining the JSAPI specification.

2. Java Speech Toolkit:

- *Prototype JSAPI Implementation.* As the API specification develops, this prototype implementation of speech recognition and synthesis is updated to test out the API and allow development of experimental applications.
- *Natural Language Tools: Unified Grammar and Swiftus.* These components, derived from the SpeechActs environment, allow developers to write recognizer-independent grammar specifications and then, using the Swiftus natural language processor, translate spoken text into sets of feature-value pairs that are easy for an application program to interpret.
- *Speech UI Scripting Language (Nicobol).* The scripting language is designed to give non-programmers control over the user interface portions of a class of speech applications built on top of JSAPI.

3. Java-based Speech Applications:

- *Hands-Free Instruction Manual: Wiring for NetDay.* This Java application, written on top of the prototype JSAPI implementation, is designed to teach NetDay volunteers the networking skills necessary to wire a school for access to the Internet. Speech input and output free the users' hands to practice skills as they step through the spoken, written, and video instructions. The application runs within the HotJava browser and integrates other Java Media components, such as the Java Media MPEG player.

- *Speech-Enabled Drawing*. In this Java-based graphical editor, users employ a combination of mouse, keyboard, and speech input to create two-dimensional drawings. Speech is used to streamline the process of changing multiple attributes (e.g., the user wants a “wide, red border” or “14 point bold italic text in green,”) and it is used to issue commands while the user’s hands are busy in the drawing area.
4. Automatic Generation of Lexical Features:
- As part of a project to create a speech-operated online catalog, in collaboration with Lands’ End, we have developed a technique to automatically constrain speech recognition grammars based on a set of examples taken from the catalog database. This allows users to refer to catalog items in flexible ways (e.g., “Show me women’s tailored pants in a size medium”), while constraining the grammar enough to rule out nonsensical combinations of catalog attributes (e.g., “Casual cashmere diaper bag”).

References

Publications

Hunt, A. “Java Speech API: A White Paper.” (April 1997).
(See: <http://java.sun.com/marketing/collateral/speech.html>)

Martin, P. “The ‘Casual Cashmere Diaper Bag’: Constraining Speech Recognition Using Examples.” *Proceedings of the Association of Computational Linguistics*, Madrid, Spain (July 1997).
(See: <http://www.sunlabs.com/research/speech/SpeechActsPapers.html>)

Martin, P., F. Crabbe, S. Adams, E. Baatz, and N. Yankelovich. “SpeechActs: A Spoken Language Framework.” *IEEE Computer*, 29(7) (July 1996).
(See: <http://www.sunlabs.com/research/speech/SpeechActsPapers.html>)

Yankelovich, N. “How do Users Know What to Say?” *ACM Interactions*, 3(6) (November/December 1996).
(See: <http://www.sunlabs.com/research/speech/papers/Interactions.html>)

Demonstration Video

“Hands-Free Instruction Manual: Wiring for NetDay,” a Java Speech API demonstration.

World Wide Web

<http://www.sunlabs.com/research/speech>

SunTest

Alberto Savoia

alberto.savoia@eng.sun.com

Overall Objective

SunTest's overall objective is to generate revenues and help the adoption of the Java programming language by developing and marketing Java testing tools and services.

Objectives for FY97

- Release a Java GUI testing tool
- Release a Java API testing tool
- Release a Java test coverage tool
- Develop and deliver Java testing classes

Description

SunTest was started on July 1, 1996, when it became clear that Java had a great chance of being widely adopted, and that Sun Microsystems Laboratories' PrimaVera/ADL technology and personnel could greatly help Sun and Java by refocusing their software testing tools efforts and expertise on Java.

SunTest's first priority was to develop a suite of products that could be sold to generate revenues to support and grow the business unit.

Accomplishments

1. Designed, developed, and released three new Java testing tools:
 - JavaStar™, a tool for automating the testing of Java applications through their graphical user interface (GUI). JavaStar works by capturing and playing back user interactions with the application

under test and comparing the results against a “golden” reference set of results. Any differences between a given test run and the reference test run are marked as a potential bug or regression.

- JavaSpec™, a tool for testing Java applications and libraries through their programmatic interface. JavaSpec allows the user to specify test data as well as pre- and post-conditions for each method of the class under test. JavaSpec then generates a self-checking test that executes the method under test with various combinations of test data and evaluates all pre- and post-conditions to ensure that the implementation of the method matches the test specification.
 - JavaScope™, a test coverage analyzer. JavaScope provides Java code with instruments to enable the collection of profiling information (e.g., how many times this line of code was executed). The profiling information can then be used to determine how well the tests developed with JavaStar and JavaSpec to cover the application under test.
2. Established SunTest as the leading Java testing company in the market:
 - Designed, developed, and released three Java testing tools: JavaSpec, JavaStar, and JavaScope.
 - Used a variety of media and methods (including Internet, magazine advertisements, trade shows, etc.) to communicate to the market that SunTest is *the* place to look for Java testing tools.
 - Designed, developed, and delivered over 20 Java testing courses.
 3. Helped JavaSoft’s 100% Pure Java certification effort:
 - Designed and developed the process, tools, and documentation for the 100% Pure Java certification program.
 - Worked with Java ISV to promote 100% Pure Java certification program.

References

World Wide Web

<http://www.sun.com/suntest>

Tcl and Tk

John K. Ousterhout

john.ousterhout@eng.sun.com

Overall Objective

This project is using the Tcl scripting language and the Tk toolkit to explore scripting issues for the Internet. Our goal is to develop scripting technology suitable for composing and connecting the key component technologies of the Internet, such as Enterprise JavaBeans™. The issues we are exploring include cross-platform portability of scripts, security, runtime execution efficiency, and how to integrate scripting languages with component technologies such as JavaBeans and OLE.

Objectives for FY97

- Implement native look and feel for the Windows and Macintosh ports of Tk.
- Complete the bytecode compiler and release it to the Tcl community.
- Release version 1.0 of SpecTcl™, the Tcl/Tk GUI builder.
- Release version 1.0 of the Tcl browser plug-in; then implement a mechanism that allows a variety of security policies.
- Develop other Tcl/Tk tools and applications that demonstrate the power of scripting for Internet applications.

Description

Tcl is a simple interpreted scripting language with the interesting property that it is embeddable: its interpreter is implemented as a C library package that can easily be incorporated into C and C++ applications. Each application can extend the basic language features with new application-specific features, so that Tcl scripts can then be written to control the application. Tk is a toolkit for creating graphical user interfaces; it is

implemented as a set of Tcl extensions. With Tk, graphical user interfaces can be created as Tcl scripts without writing any C code.

Tcl and Tk have become widely used in the UNIX[®] community for several reasons. First, Tk provides a very simple and high-level interface that makes it possible to write GUI applications 5-20 times more quickly than with more traditional toolkits based on C or C++. Second, Tcl's embeddability makes it easy to create powerful command languages for a variety of applications, and its interpretive nature makes it easy to create hypertext and other forms of active content. Third, Tcl provides an excellent "glue" language for making a variety of components and applications work together.

Our approach for this project has been to evolve Tcl and Tk from their UNIX origins to make them suitable for Internet scripting tasks. To do this, we have been enhancing them with the same portability and security properties that have made Java such an attractive vehicle for Internet programming. At the beginning of this project Tcl and Tk ran only on UNIX platforms; we have now ported them to run on Windows and Macintosh platforms as well. In addition, we are implementing security mechanisms so that an incoming script can be executed safely even if it comes from an untrusted sender. This makes it possible to send Tcl/Tk scripts around the Internet to carry out interesting tasks such as workflow, remote device control, automatic purchasing, software installation, and network management. In addition to providing the basic infrastructure for Tcl and Tk applications that span the Internet, we are developing tools to simplify the creation of these applications, and also creating a few showcase applications to demonstrate the power of this new programming paradigm. In our future work we will be integrating Tcl with Java and JavaBeans to combine Java's power as a component creation language with Tcl's power as a scripting language.

Accomplishments

1. Alpha and beta versions of Tcl 8.0 and Tk 8.0 were released. These are major new releases with many new features, including the following:
 - Tcl now includes a bytecode compiler and interpreter, which speeds up Tcl script execution by a factor of 2-20x.

- Tk provides native look and feel on the Macintosh and Windows platforms, in addition to Motif look and feel under UNIX. The same script will run on any platform, but will appear differently on different platforms.
 - Tcl includes a new namespace mechanism that makes it possible to encapsulate code and data.
 - Tcl has new facilities for binary I/O, random number generation, and HTTP requests.
 - Tk's font mechanism has undergone a major revision to simplify its interface and provide better platform independence.
2. We released version 1.0 of the Tcl/Tk plug-in module for Web browsers such as Netscape Navigator™ and Microsoft Internet Explorer. We also made alpha releases of version 2.0, which includes a sophisticated and flexible mechanism for security policies. Since its first release in the summer of 1996, there have been more than 100,000 downloads of the plug-in.
 3. We released version 1.0 of SpecTcl, an interactive GUI builder for creating Tk user interfaces. There have been more than 10,000 downloads of SpecTcl.
 4. We developed an embeddable Web server written entirely in the Tcl scripting language, intended for management applications. The server can be embedded in devices or applications to turn them into HTTP servers. Management tools can then be implemented as Web pages that communicate with the embedded server.
 5. We released a beta version of WebTk, an authoring system for Web pages. WebTk is written entirely in Tcl/Tk scripts.
 6. We experimented with Internet commerce tools that allow SpecTcl and the plug-in to be purchased via the Web using credit cards.
 7. We began work on internationalizing the Tcl and Tk scripting language so that they support character sets other than ISO-8859. Our mechanism is based on the UTF-8 encoding.
 8. Usage of Tcl and Tk scripting language continues to increase. The base Tcl distributions are now downloaded from the Sun FTP site at a

rate of about 5000-8000 downloads per week, up a factor of 3 from 18 months ago. There are now at least 11 Tcl/Tk applications in production use within Sun's internal information systems.

9. We obtained official approval from the applicable architecture review committees for the Tcl and Tk scripting language to be used in Sun products.

References

Publications

Ousterhout, J.K. "Scripting: Higher Level Programming for the 21st Century."

(See: <http://sunlabs.sun.com/people/john.ousterhout/scripting.html>)

Ousterhout, J.K., J.Y. Levy, and B.B. Welch. "The Safe-Tcl Security Model." *Sun Microsystems Laboratories Technical Report SMLI TR-97-60*.

World Wide Web

<http://www.sunscript.com>

Neil Wilhelm

neil.wilhelm@eng.sun.com

Overall Objective

Investigate high-speed (greater than 1Gb/s) CMOS digital serial link (DSL) circuits.

Objectives for FY97

- Understand and improve the subsystems of DSL transmitters and receivers.
- Assist product groups with DSL questions.

Description

Improvements in VLSI and ASIC CMOS processes have made possible integrated serial digital communications links with data rates of 1 Gb/s or more. The increasing transistor densities have made serial links cheap in area. It remains for such serial links to be reduced to practice and to increase their data rates well beyond 1 Gb/s.

These fast, integrated serial links have a wide range of applications. These range from network/cluster communications to simply squeezing more bandwidth from limited pincounts.

Accomplishments

1. The bias phase-locked loop (PLL) test chip submitted to MOSIS for fabrication at the end of FY96 has returned and been characterized:
 - The bias PLL test chip demonstrated that, by setting bias with an external *frequency* reference, bias control can be made process and temperature independent.
 - A new charge pump design was demonstrated in this chip.

- The delay element used in the bias PLL's voltage-controlled oscillator (VCO) is an improvement over previous designs.
2. The Venturi group worked with one of Sun's ASIC vendors on several DSL issues by:
 - Assisting the vendor with a test chip by participating in design reviews and by assisting them with the isolation of analog circuits.
 - Reviewing specifications of their DSL products.
 3. The group has looked at crystal oscillator design and performance:
 - Examined the characteristics of commonly-used commercial oscillators, and,
 - We have been working on crystal oscillator circuits for DSL transmitters.
 4. The Venturi group also has continued participation in the IEEE Fibre Channel jitter subcommittee:
 - Discussions at these meetings helped prompt our work on crystal oscillators for DSL transmitters.
 - We have participated in a test-equipment consortium within Sun to share the cost burden of jitter test equipment.
 - We have provided advice in this area to SMCC.
 5. Intellectual property:
 - Twelve patent applications pending.
 - Five patent applications in progress.

Appendix A—Patents Issued FY97

Title: A Method and Apparatus for Explicit Rate Flow Control in ATM Networks

Inventors: C. Buckley, I. Cidon, A. Khamisy, R. Rom

Issued: 03/96 (8/619,040)

Title: Method and Apparatus for Implementing Self-Organization in a Wireless Local Area Network

Inventor: R. Rom

Issued: 05/07/96 (5,515,509)

Title: Cascaded Multistage Counterflow Pipeline Processor For Carrying Distinct Data in Two Opposite Directions

Inventors: I.E. Sutherland, C.E. Molnar, R.F. Sproull, I.W. Jones

Issued: 11/05/96 (5,572,690)

Title: Systems and Method for Traversing ATM Networks Based on Forward and Reverse VC Labels

Inventors: I. Cidon, T. Hsiao, P. Jujjavarapu, A. Khamisy, R. Rom, M. Sidi

Issued: 11/26/96 (5,579,480)

Title: System for Fast Switching of Time Critical Input Signals

Inventor: I.E. Sutherland

Issued: 01/07/97 (5,592,103)

Title: Counterflow Pipeline Processor with Instructions Flowing in a First Direction and Instruction Results Flowing in the Reverse Directions

Inventors: R.F. Sproull, I.E. Sutherland

Issued: 02/04/97 (5,600,848)

Title: Software-based Encoder for a Software Implemented End-to-End Scalable Video Delivery System

Inventors: N.J. Chadda, J.D. Northcutt, G.A. Wall, J.G. Hanko

Issued: 04/15/97 (5,621,660)

Title: Method for Simulating the Parallel Processing of Video Data

Inventor: J.G. Hanko

Issued: 05/06/97 (5,627,966)

Title: Method and Apparatus for Providing Collection Browsers

Inventors : H.A. McIntosh, E. Priyadarshan, A. Ruberg, T. Shea

Issued: 05/13/97 (5,630,042)

Appendix B—Technical Reports Issued FY97

These Technical Reports may be obtained via the World Wide Web or by sending an email message to Editor@sunlabs.eng.sun.com.

TR-95-49	Charles E. Molnar and Huub Schols	The Design Problem S CPP-A
TR-96-54	Sun Microsystems Laboratories Staff	Fiscal 1996 Project Portfolio Report
TR-96-56	Robert J. Kuhns	A Survey of Information Retrieval Vendors
TR-96-57	Vlada Matena, Yousef A. Khalidi, and Ken Shirriff	Solaris MC File System Framework
TR-96-58	Mick Jordan and Malcolm Atkinson	First International Workshop on Persistence and Java
TR-97-60	John K. Ousterhout, Jacob Y. Levy, and Brent B. Welch	The Safe-Tcl Security Model
TR-97-61	William A. Woods	Conceptual Indexing: A Better Way to Organize Knowledge

Glossary

- ACK/NAK. . . . In reliable multicast, ack/nak implosion is defined as what happens if all receivers in a multicast group acknowledge (or negatively acknowledge) each packet they receive (or didn't receive) back to the sender. This is undesirable because it floods the network with traffic. Various schemes exist to achieve implied acknowledgement without having each receiver explicitly communicate back to the sender in a multicast group.
- ACM Association for Computing Machinery
- API Application Programming Interface
- ASARC Application Software Architecture Review Committee: one of several ARCs (Architecture Review Committees) at Sun. These are bodies that review all non-trivial changes to Sun's product line.
- ASIC Application Specific Integrated Circuit
- ATM Asynchronous Transfer Mode
- BCN Boston Center for Networking
- CAD Computer-Aided Design
- CC Cubic Centimeter
- CIF Cells In Frames
- CMOS Complementary Metal Oxide Semiconductor
- CPU Central Processing Unit
- CTO Chief Technology Officer
- DSL Digital Serial Link
- DTVI Distributed Tutored Video Instruction
- DV Digital Video
- EDI Electronic Data Interchange
- FFB Fast Frame Buffer
- FIFO First In First Out
- FTP File Transfer Protocol: a mechanism for retrieving files over the Internet

Gb or Bg/s	Gigabits
GHz	Giga-Hertz
GC	Garbage Collection
GSMP	Generalized Switch Management Protocol
GUI	Graphical User Interface
HRV	High Resolution Video
HTTP	HyperText Transfer Protocol
HTML	HyperText Markup Language
IDL	Interface Definition Language
IEEE	Institute of Electrical and Electronics Engineers
I/O	Input/Output
IP	Internet Protocol
ISV	Independent Software Vendor
JAMM	Java Applets Made Multi-User
Java	Programming Language
JavaOS	An implementation of the Java virtual machine that is directly hosted on a bare machine or microkernel.
Java VM	Java Virtual Machine
JDK	Java Development Kit
JECF	Java Electronic Commerce Framework
JEPI	Joint Electronic Payment Initiative
JHDL	Java Hardware Description Language
JP	A prototype environment for large-scale development software development in Java, providing configuration management and system building functions, and implemented using orthogonal persistence for Java.
JSAP1	Java Speech API
Mbits/sec	Megabits per second
MBONE	Multicast Backbone
MOSIS	DARPA's MOS Implementation System
MP	Multiple Processor

MPEG	Motion Pictures Expert Group
OFX	Open Financial Exchange
OLE.	Object Linking and Embedding
OPJ	Orthogonal Persistence for Java
OS	Operating System
PCI	Peripheral Component Interconnect
PJama ₀	The first prototype of an extended Java virtual machine that provides orthogonal persistence for Java.
PJava	Persistent Java
PLL	Phase-Locked Loop
PNNI.	Public Network-to-Network Interface
RAM.	Random Access Memory
RMI.	Remote Method Invocation
RSVP	Resource ReserVation Protocol
RTP	Real Time Transport Protocol
RTCP.	Real Time Control Protocol
SET	Secure Electronic Transaction
SIGPLAN	Special Interest Group on Programming Languages
SKIP	Secure Key Internet Protocol
SMART.	Scheduler for Multimedia And Real-Time Applications
SMCC	Sun Microsystems Computer Company
SME	Sun MicroElectronics
SML	Sun Microsystems Laboratories
SMP	Symetric Multi Processor
SPARC	Scalable Processor ARChitecture
SpecTcl	A GUI (Graphical User Interface) Builder for Tcl applications. It will also generate code for Java, Perl, and HTML.
SSL	Secure Socket Layer
SunIR/ENS	Sun Information Resources/Enterprise Network Services
SWAN.	Sun Wide Area Network

Glossary

- Tcl and Tk. . . . Tcl (Tool Command Language): a simple embeddable scripting language. Tk (ToolKit): a graphical user interface toolkit based on the Tcl scripting language.
- TCO Total Cost of Ownership
- TVI Tutored Video Instruction
- VCO Voltage Controlled Oscillator
- VCR Video Cassette Recorder
- VLSI Very Large Scale Integration
- VM Virtual Machine
- WebTk. An authoring system for HTML Web pages, implemented using TCL and TK
- XIL X Imaging Library