6-5

NILE: A Novel Platform-Independent and Efficient Distributed Simulation System for TCAD

Kosuke Yoshitomi, Takaaki Tatsumi, Nobumasa Nakauchi, Mikio Mukai and Yasutoshi Komatsu
ULSI R&D Laboratories, LSI Business & Technology Development Group,
Core Technology & Network Company, Sony Corporation.
4-14-1 Asahi-cho Atsugi-shi Kanagawa-ken 243-0014 JAPAN
TEL: +81-462-30-5719, FAX: +81-462-30-5572
E-mail: yositomi@ulsi.sony.co.jp

Abstract – We have developed a JAVA based novel system called NILE, which is tolerable for practical use and working on a distributed environment for semiconductor process and device simulations. NILE enables the utilization of simulation in practical use from remote sites even if the network bandwidth is narrow. NILE has succeeded to provide a same kind of simulation environment regardless of kinds of platforms. Furthermore NILE enables developing devices for USLI much more efficiently using the central servers.

I. INTRODUCTION

We had reported an automatic simulation system for process parameters control sweeping and analyzing process variation based on process and device simulators called BARAS[1] in 1996. Since then we had many requests from wide spread users even from these of factories. The main needs from various users are platform independent and efficient simulation from distributed users. However, BARAS had difficulty in operating on multiple platforms. Especially the network bandwidth between development laboratories and factories is usually narrow so that it is difficult to use BARAS from remote sites. NILE resolved these problems by using JAVA and a newly designed protocol.

II. NILE: WEB BASED DISTRIBUTED SIMULATION SYSTEM

Fig.1 describes the system overview of NILE. NILE consists of server computers at an operation center and client computers located over networks. Server computers are connected each other with the high speed network such as LAN (Local Area Network) and client machines are connected with LAN or WAN (Wide Area Network) of low speed network. Any kinds of client platforms can be used as long as the web browser and the Java VM (Virtual Machine) run on it. Fig. 2 shows the architecture of NILE. NILE has three servers as can be seen from the figure.

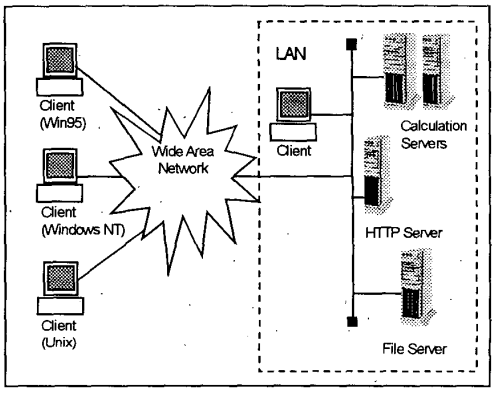


Fig.1 NILE System Overview

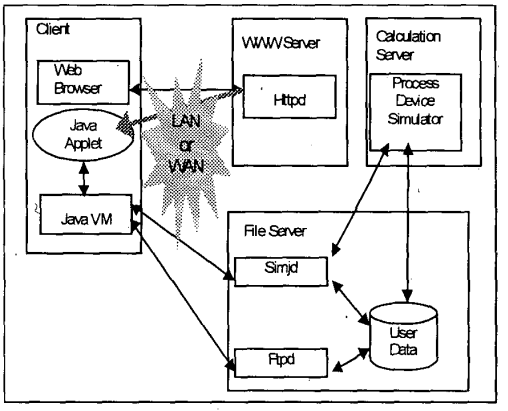


Fig. 2 NILE System architecture

A. WWW server

In order to start the system, user invokes the WWW browser such as Netscape Navigator or Internet Explorer and then download an applet program from the WWW server. This is done from the platform of client machines, usually the PC.

B. File server

A daemon program called simjd (simulation job control daemon), which controls simulation jobs sent from clients, runs on this server. All the simulation data such as input data and calculation results are stored and managed in the database system of File server. The central data handling reduces the data traffic amount between clients and central servers drastically.

C. Calculation Servers

Calculation Servers contain process and device simulators. Input data for the simulation is sent to this server and the result is sent to File server. For achieving efficient TAT, high-performance calculation servers are assigned.

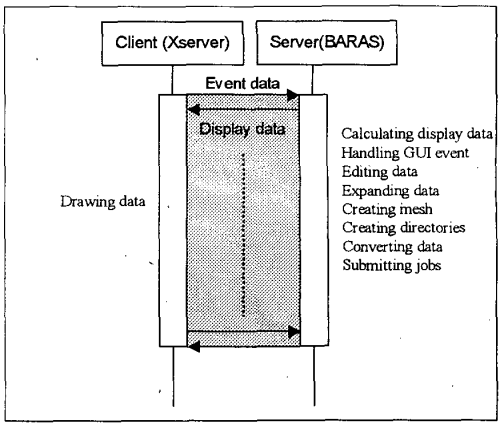
III. Implementation and Evaluation

NILE provides all the functions supported by BARAS such as process control sweeping, statistical variation analysis and output data processing, including the automatic simulation input data generation for them. Fig.3 shows the snap shot of NILE.

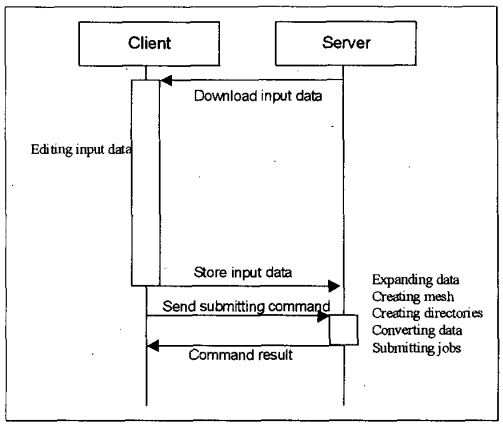
A. Simple protocol between client and simjd.

In simjd, we implemented a simple stateless protocol. Since each command is self-consistent and its size is small (< 1KB), it can be used even if the network speed is low. In addition, drawing data on the display and handling events such as keyboard and mouse operations are performed in the client side. Therefore the client does not need to communicate with the server for user interface displaying.

Fig. 4 describes the comparison of the protocol sequence between NILE and the conventional system. Activation is shown as a tall thin rectangle whose top is aligned with its initiation time and whose bottom is aligned with its completion time. In Fig.4(a), the client of NILE communicates with the server only fourth times. And the server process is activated only when a submitting command comes to the server. On the other hand, in Fig.4(b) the X server software on the client always keeps direct contact with the server because the process to calculate drawing data is performed on the server. In this case, the server is always activated while operating BARAS.



(a) NILE



(b) BARAS with X Window System

Fig. 4 Comparison of protocol sequence between NILE and the conventional system

B. Performance

First we consider about the amount of traffic between the

remote client and the server. In Fig. 5 shows the amount of communication data size in each process such as editing input data and submitting jobs. In each process, NILE communicates with much less data than the conventional X Window System. Especially, while the user is editing input data, it faces with the increase of the amounts of communication for the case of BARAS with X Window System use, but NILE does not need communicate at all. Therefore NILE does not have to keep connection with the server while editing. It is an advantageous when the user uses NILE via telephone line that costs based on the connection time.

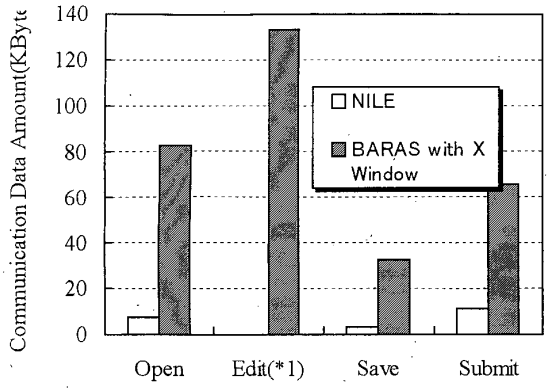


Fig.5 Comparison of communication data amount between newly developed NILE and BARAS with X Window System

Next, we compare another different situation on the start-up and submitting over different bandwidth networks. Table I shows the time to startup and submitting simulations. The startup time is strongly dependent on the network bandwidth for downloading the applet. Deployment of proxy servers at remote sites can avoid this problem. On the other hand, there is not much difference in job submission time since client sends very small data to server. Simjd program consumes much time to prepare creating temporary files, converting data and submitting jobs to the simulator. From this, we can see that by placing only a proxy server near users we can make the system much more efficient.

Finally we consider about the Java. In general, Java program is slower than the native program such as C or C++ Because Java is a dynamically bound object-oriented language and Java VM has interpreter[2]. We compare the performance between NILE and the conventional native application. Table II shows the time of typical operations. In this table, there are not much differences in execution time except time to start up. And we compare the performance between UNIX and Windows. It shows that NILE has a tolerable performance in practical use.

TABLE I
COMPARISON OF PERFORMANCE
AMONG THREE DIFFERENT BANDWIDTHS OF NETWORK

	Bandwidth (Max)	Start-up*	Submit Jobs**
LAN	10Mbps	36sec	64sec
WAN1	128Kbps	3min	66sec
WAN2	33.6Kbps	9min	66sec

OS: Windows98
CPU: MMX Pentium 233MHz (64MB)
Web Browser: Netscape Navigator 4.5J + Java Plug-In
*Applet download time is included.
**Preprocessing time is included.

TABLE II
COMPARISON OF PERFORMANCE
BETWEEN JAVA AND NATIVE PROGRAMS

	Start-up	Start-up Editor	Submit Jobs	Show Result
BARAS*	3.0sec	3.8sec	54sec	3.0sec
NILE**	30.2sec	5.2sec	53sec	3.6sec
NILE***	36.2sec	5.6sec	64sec	4.6sec

*Ultra SPARC 166MHz, 128MB, Solaris2.6
**Ultra SPARC 166MHz, 128MB, Solaris2.6, Sun JDK1.1.7_02 + Appletviewer
***Pentium 233MHz, 64MB, Windows98, Sun JRE1.1.6 + Java Plug-in1.1.1
Network bandwidth is 100Mbps in all cases

C. Multiple Platforms

We implemented a client application of NILE as a Java Applet. Java can be applied for the multiple platforms. NILE can run on the PC and the UNIX platforms which can run Sun Java Plug-in or Java Applet viewer. And we use the Swing toolkit for a graphical user-interface (GUI). Swing has a platform-independent look and feel. Therefore user can obtain a same simulation environment in any platform and location.

D. Software deployment

When user invokes NILE on the client, the web browser downloads an applet. Therefore it does not need to install the software in each client. Furthermore, the users can always employ the newest application.

WWW browsers implement a Java security manager that normally stops applets from reading or writing to local disk drives, call native executable files, connecting to server other than the host, and doing some other potentially dangerous actions. We use an electric signature[3] to the applet so that the WWW browser allows the applet to run with the same full rights as local applications.

IV. CONCLUSION

We have developed a novel platform-independent and efficient distribute TCAD simulation system called NILE for the first time. NILE enables the utilization of simulation in practical use from remote sites even if the network bandwidth is narrow. NILE has succeeded to provide a same kind of simulation environment regardless of kinds of platforms. Furthermore NILE enables developing devices for USLI much more efficiently using the central servers.

REFERENCE

[1]Tatsumi et al., BARAS: Novel and Highly Efficient Simulation System for Process Control Sweeping and Statistical Variation, SISPAD'96, pp149-150, 1996.

[2]Tom R. Halfhill, Al Gallant: How To Soup Up Java, BYTE, May 1998, taken from www.byte.com.

[3]Security and Signed Applets, JDK 1.1 Documentation, Sun Microsystems, 1997.

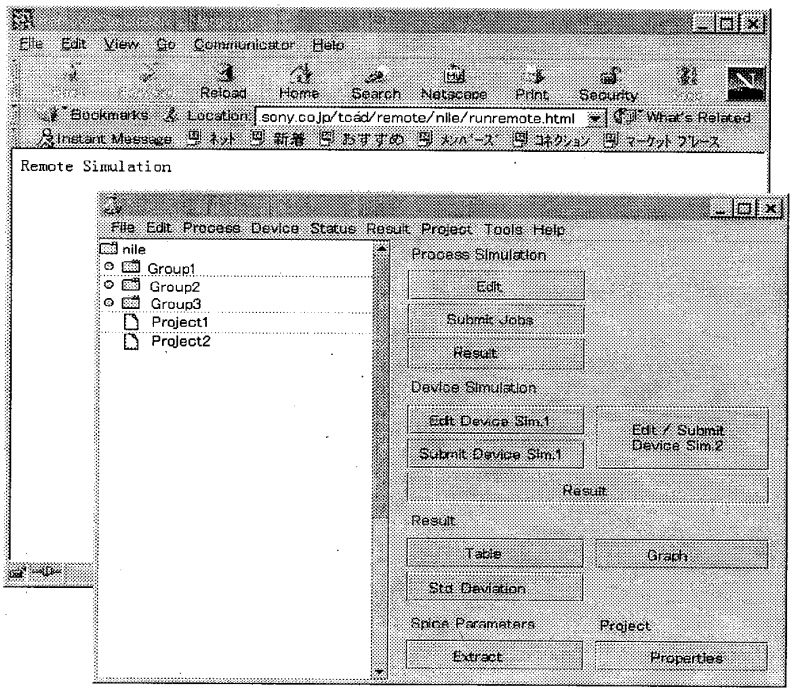


Fig.3 The snap shot of NILE (Main Panel) and WWW Browser