*Proceedings of the 1999 Winter Simulation Conference*
*P. A. Farrington, H. B. Nembhard, D. T. Sturrock, and G. W. Evans, eds.*

# eSCA: A THIN-CLIENT/SERVER/WEB-ENABLED SYSTEM FOR DISTRIBUTED SUPPLY CHAIN SIMULATION

H. Bob Chen
Oliver Bimber
Chintamani Chhatre
Elizabeth Poole
Stephen J. Buckley

IBM T.J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598, U.S.A.

## ABSTRACT

eSCA is a client server, web-enabled architecture and front-end system for IBM's Supply Chain Analyzer (SCA), a new "best-of breed" software tool and methodology for measuring, analyzing, and reengineering complex supply chains. eSCA extends the capability of SCA through a computing network with server-based and web-enabled functionality. eSCA consists of two distinct versions: eManager and Web-enabled SCA. The eManager version provides model catalogs, interactive modeling, seamless model/file transfer, batch experiments, and post-simulation data analysis. The web-enabled version provides a rapid way for users to access SCA through the web. The software development process for eSCA adopted the unified process: use-case driven, architecture-centric, iterative, and incremental. This paper focuses on the eSCA use cases and the application architecture. It is shown that eSCA is a reusable architecture for many different applications.

## 1 INTRODUCTION

IBM's Supply Chain Analyzer (SCA), formerly called Supply Chain Simulator (SCS) (Buckley, Jeffrey 1997, Bagchi et al. 1998, Lin et al. 2000), is a new "best-of breed" software tool and methodology for measuring, analyzing, and re-engineering complex supply chains. As IBM enters the supply chain solution business, it is clear that SCA will play a vital role.

SCA already enjoys considerable success both in the IBM internal supply chain (e.g. IBM's Personal System Group) and in external consulting (e.g. food, electronics, and paper). However there is a need to enhance SCA to next level: client server architecture, web-enabled, model catalogs, and batch experiment management. These requirements motivated us to develop eSCA: a client server, web-enabled system for distributed supply chain simulations. The eSCA system consists of two versions, eManager, a modeler's workbench with a Java user interface, and a web-enabled SCA system. The development process of eSCA adopted the industry open standard process known as the unified process (Jacobson, Booch, and Rumbaugh 1998): use-case driven, architecture-centric, iterative, and incremental. The requirements were captured by use-cases, which drove the development process. The architecture is the blueprint of the development process (Bass, Clements, and Kazman 1998, Harwood 1998) and the entire development process is iterative and incremental.

The salient features of eSCA are:

- It provides a thin client/server-based computing model for SCA, whereby a modeler can conduct modeling and simulation interactively using the combined resources of a computing network.
- It provides a parallel and distributed simulation environment to conduct batch experiments. Users can select fast servers to conduct massive experiments.
- It provides a knowledge-based model catalog whereby a modeler can take advantage of reusable industry-specific models.
- It provides seamless model/file migration between local and remote hosts.
- The eSCA architecture allows easy access to SCA through the web. Using Application Embedded Launch (AEL) technology, eSCA allows web users to perform supply chain modeling and simulation.

## 2 REQUIREMENTS

The requirements stem from the extensive usage of SCA, a standalone desktop/workstation application. It was found that modelers needed to have network access to SCA and its associated data. To evaluate multiple scenarios modelers needed to run several simulations at the same time and therefore needed to utilize remote processors. There was also a need for potentially massive batch experiments, to off load simulation processing from personal computers. As the knowledge base of models gained mass, a centrally -accessible model catalog was needed. And finally, there was a need for post-simulation data analysis. All of these requirements were captured in eSCA Use cases, which are summarized in Figure 1, and further elaborated in Section 2.1 to section 2.5. Note that the users of eSCA are the Modelers (Figure 1), who have presumably had extensive modeling experience using SCA.

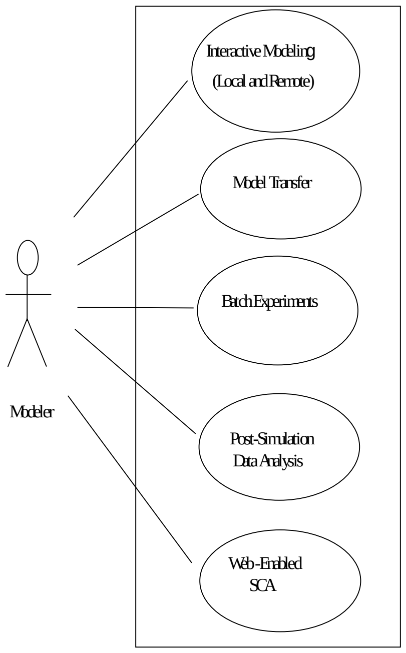


Figure 1: eSCA Use Cases

### 2.1 Interactive Modeling Use Case

#### 2.1.1 Local Modeling

The use case called Interactive Modeling in Figure 1 contains two parts, Local Interactive Modeling and Remote Interactive Modeling. The following step shows the flow of event of the Local Interactive Modeling use case in which the modeler's personal computer is utilized for graphical modeling. This is the most typical modeling use case when the Modeler needs to build a model and has a reasonably powerful and available personal computer.

Flow of the events of the use case is:

1. The use case starts when the modeler selects Interactive Modeling.
2. The modeler selects local modeling.
3. The modeler selects the local host.
4. The modeler scans, navigates the available template models from the model catalog library.
5. The modeler selects the template model similar to the model he/she wishes to build.
6. The modeler invokes the template model.
7. The modeler edits the model and enters his/her own parameter and data.
8. The modeler saves the built model to his/her own working directory.
9. The modeler runs his/her own model to test the model.

#### 2.1.2 Remote Modeling

This section describes the Remote Interactive Modeling use case in which a remote processor is utilized for graphical modeling. This use case can occur when the model is a shared model on a central processor, or when the model complexity requires a more powerful computer than the Modeler's personal computer.

Flow of events of the use case is:

1. The use case starts when the modeler selects Interactive Modeling.
2. The modeler selects remote modeling.
3. The modeler selects a remote host.
4. The modeler scans, navigates the available template models from the remote model catalog library.
5. The modeler selects the template model similar to the model he/she wishes to build.
6. The modeler invokes the template model.
7. The modeler edits the model and enters his/her own parameter and data.
8. The modeler saves the built model to his/her own working directory.
9. The modeler runs his/her own model to test the model.

The model catalog plays a key role in interactive modeling. It is desirable for modelers to have a centralized catalog of reusable models. The eSCA model catalog is organized by industry segment. Models are collected from actual consulting engagements and sanitized. In the future, we plan to provide the ability to search this catalog by keywords.

## 2.2 Model Transfer

The use case called Model Transfer in Figure 1 contains two parts: From Local to Remote and From Remote to Local. From Local to Remote occurs when a locally developed model needs to be simulated on a remote processor, or shared with another modeler. From Remote to Local occurs when a model has been simulated on a remote processor and its results are needed for local analysis. Both of these use cases are critical to a modeler who wishes to run a simulation on a remote server. The transfer of models and files must be seamless to the modeler.

### 2.2.1 From Local to Remote

Precondition of this use case are:

1. The modeler has to finish Interactive User Cases.
2. The Modeler has to have remote access privilege.

Flow of Events of this use case is:

1. The use case starts when the modeler selects Models Transfer.
2. The modeler selects the models from the local host.
3. The modeler selects the remote host.
4. The modeler specifies the working directory in the remote host he/she wishes to transfer the models to.
5. The modeler reviews the both local and remote end.
6. The modeler commits the model transfer.

### 2.2.2 From Remote to Local

Precondition is same as previous one.

Flow of Events of this use case is:

1. The use case starts when the modeler selects Models Transfer.
2. The modeler selects the remote host.
3. The modeler selects the models/files he/she wishes to transfer from.
4. The modeler selects the local host.
5. The modeler specifies the working directory in the local host he/she wishes to transfer the models to.
6. The modeler reviews the both local and remote ends.
7. The modeler commits the models/files transfer.

## 2.3 Batch Experiment Use Case

This section describes the Batch Experiment use case. This use case occurs when a remote processor is used for simulation, perhaps for a very complex model or for parallel execution of multiple scenarios. This use case differs from Remote Interactive Modeling in that no graphics are presented on the modeler's personal computer; only the results of the simulation are desired. In this case, the simulation server is a "black box".

Precondition of Batch Experiment Use Case are:

1. The modeler has to have remote access privilege.
2. The modeler has to either finish use case Model Transfer or interactive modeling
3. Definition of experiment: running an existing model with different parameter and given input data.

Flow of Events of this use case is:

1. The use case starts when the modeler selects Batch Experiment.
2. The modeler specifies the name of the Model (such as MyModels1).
3. The modeler selects the remote host with the best performance in mind.
4. The modeler selects the models to be executed and being transferred already at the remote host.
5. The modeler specifies the working directory in the remote host.
6. The modeler specifies the E-mail address for the notification upon finishing of the experiment.
7. The modeler specifies the unique experiment ID to identify the different experiment for post-data analysis.
8. The modeler commits to run the experiment

## 2.4 Post-Simulation Data Analysis

After simulations have been executed on local or remote hosts, the results are all brought to the local computer to be analyzed and compared. This section captures the Post-Simulation Data Analysis use case.

Precondition of the post-simulation use case are:

1. The modeler has finished Batch Experiment
2. The modeler has finished Model Transfer.

Flow of Events of this use case is:

1. The use case starts when the modeler selects Post-Data Analysis.
2. The modeler scans, navigates, and browses the data from the reports.
3. The modeler reads the data and plots the results.
4. For comparison purpose, for different experiments results, the modeler reads different data as input, and plots the results in an easy to compare, easy to report manner.
5. The above data analysis can be conducted fluidly.

## 2.5 Web-enabled SCA use case

This section describes the Web-Enabled SCA use case. This is similar to Remote Interactive Modeling except that it is initiated from a web browser.

Flow of Events of the Web-enabled SCA use case is:

1. The use case starts when the modeler selects Web-enabled SCA.
2. The modeler starts the Browser his/her choice Netscape/IE.
3. The modeler enters the URL of eSCA.
4. The modeler invokes the SCA from the web
5. The modeler conducts Interactive Modeling through Web.

Having captured the requirements of the eSCA system in this section, we will describe the architecture of the eSCA next.

## 3 ARCHITECTURE

The architecture of eSCA consists of thin clients (Java applications and web browsers), Web Servers, and eServers (Java application servers). Due to SCA's complex graphics and animation capabilities, the most important requirement of eSCA is to deliver the presentation layer (graphics and user interface) to the client while maintaining the simulation engine on the server side.

To satisfy this requirement, we first considered the CORBA architecture. The intention reflects our vision of enterprise simulation and modeling in an industry standard manner. It aligns with the IBM's thin-client network computing model and web computing model. Although it is relatively easy to use CORBA's wrapper technology to wrap the SCA simulation engine as a server, a major effort was required to build a graphical user interface on the client side. To satisfy our resource and schedule constraints, Citrix's ICA (Independent Computing Architecture) (Harwood 1998) was selected as our main

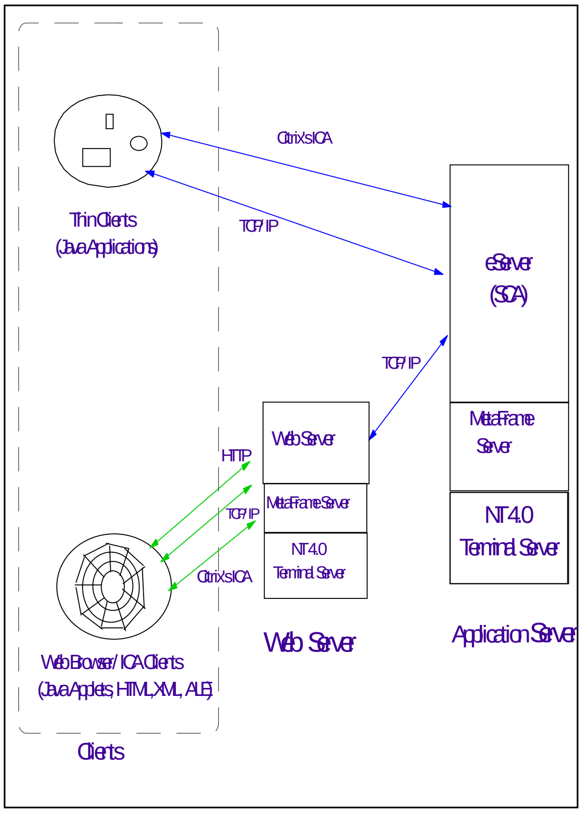


Figure 2: eSCA Architecture

protocol between the client and server. The Citrix ICA protocol separates application logic (in our case, the simulation engine) from the graphical user interface with minimum application development effort. The Citrix Metaframe server functions as the middleware. Metaframe also provides web-enabled application capability. This protocol is the backbone architecture of eSCA, and also happens to align with IBM's thin client network computing strategy.

As shown in Figure 2, the current architecture of eSCA provides two kinds of clients, eManager clients (Java application clients), and web-enabled clients. Java application clients provide a rich graphical user interface, interactive modeling, local host modeling, remote host modeling, seamless file transfer, batch experiments, and post-simulation data analysis. Web-enabled clients provide an infrastructure to access SCA from a browser. In addition, any Windows application can be invoked from our reusable architecture.

The following web servers are supported: Netscape Enterprise Web Server, Microsoft Internet Information Server. The Netscape Enterprise Web Server may or may not have its own Metaframe server.

The eSCA application server consists of a layer of Windows NT Terminal Server Edition (TSE), a Metaframe server layer, and an eServer layer. Windows NT Terminal Server Edition is the latest member of the Microsoft NT server family of products, delivering multi-user capabilities to the Windows NT 4 operating system through technology licensed from Citrix. Metaframe server is the second generation of the Citrix Winframe thin client server architecture. ICA is the distributed presentation services protocol, which separates the application logic from the graphical user interface. ICA delivers the SCA service such that the simulation engine executes on the server and the graphical user interface executes on the client.

To facilitate parallel-distributed simulation, we developed a cluster of eServers. eServer cluster is coordinated by an eMaster server. To accomplish this, each eServer is registered at the eMaster. The eMaster communicates client requests to the eServers, while functioning on the side as another eServer. Each eServer provides application specific protocols based on TCP/IP and ICA, including interactive modeling, file transfer, model catalogs, and E-mail notification.

## 4 eSCA SYSTEM FUNCTIONALITY

As mentioned above, eManger version of eSCA provides the workbench for modelers to conduct distributed supply chain simulation. The eManager version of eSCA provides the following functionality:

- Interactive modeling (remote and local)
- Model catalogs library
- Seamless model/file transfer
- Batch experiments
- Post-simulation data analysis
- Web-enabled SCA

### 4.1 Interactive Modeling

SCA has a rich animation /graphical capability for input and output. It is desirable for users to have an interactive modeling environment even it is server-based. The salient feature of distributed supply chain simulation is not only that the simulation engine resides on the server, but also to provide the users with rich graphical user interface for input and output as opposed to merely characters/command like input. Modern parallel simulation has gone a long way since von Neumann architecture (Fishwick 1995). Modern simulation always is synonymous with animation. The real challenge for distributed simulation will be how the graphical user interface at client side interacts with the simulation engine on the server side. Among others, the speed, the bandwidth, the synchronization, the load balance, and the parallel distribution (Fishwick 1995), are the main factors. The distributed simulation at its best must optimize all those factors. We believe that eSCA address those issue well. At the prototype stage, we believe eSCA is the first distributed simulation system of its kind in the industry both in architecture and in implementation.

Figure 3 shows the interactive modeling of eSCA through local model catalog and remote model catalog, whereby remote model catalogs reside in the available hosts (the cluster of server-based computers). The cluster server architecture in eSCA was designed with inter-server connection/ communication considerations. In this sense, the eServer architecture functions like parallel computing models. A typical scenario will be similar to the user case in section 2.1.

### 4.2 Model Catalog Library

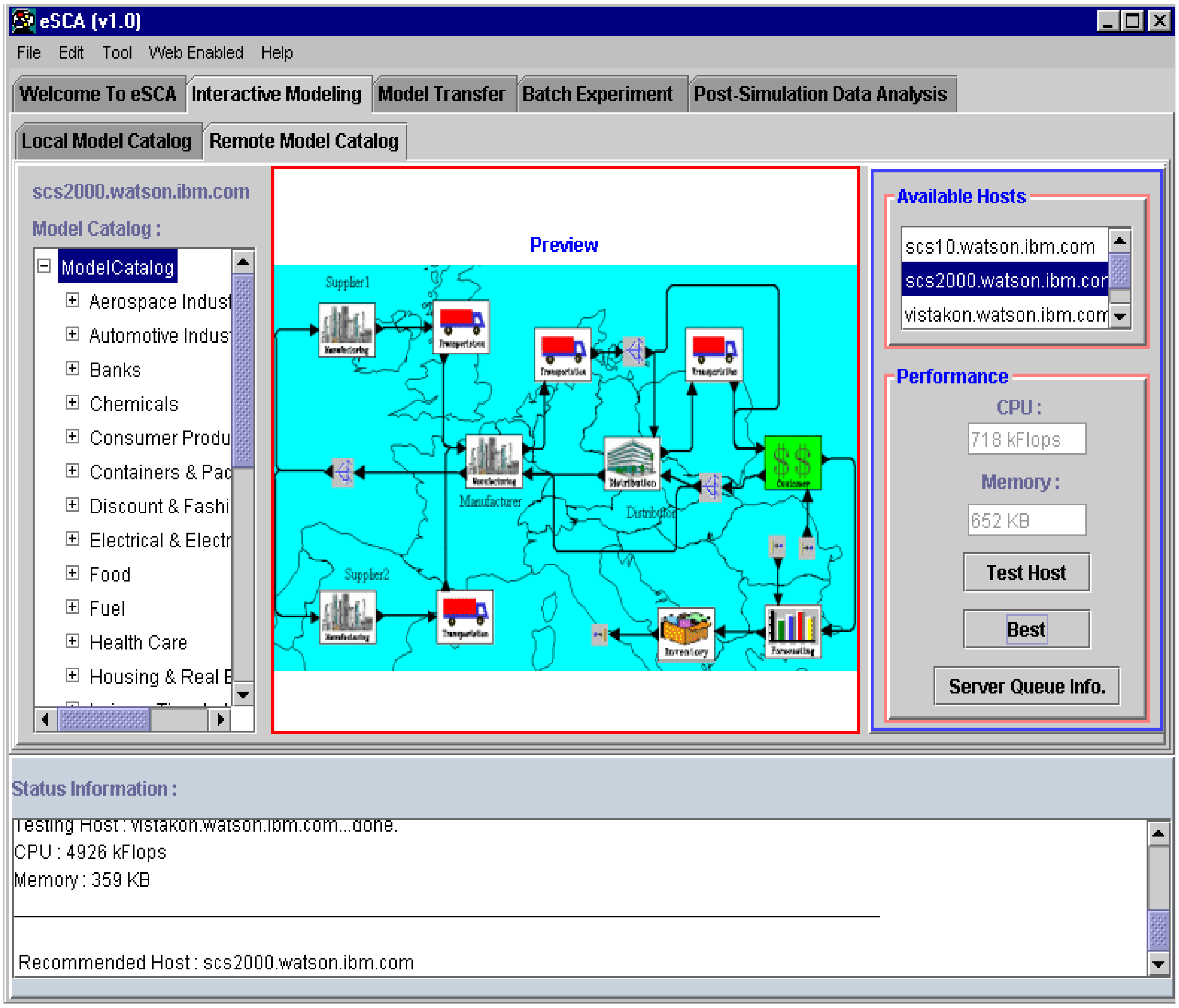
As knowledge management rapidly gained more and more attention, it is desirable for simulation community to have a central Model library, classified accordingly. One of the early motivations to develop eSCA is to provide a server-based central model catalog library. Figure 3 shows the interface of model catalogs library in the central server. The purpose of the model catalog is to provide modelers with reusable model template library, which classified according the standard industry segmentation. Modelers don’t have to build models from scratch. By using the model catalogs library, He/she can start his/her own model by scan the model Catalog library first, choose the one which is similar to the model he/she whishes to build, then edit, modify it with new set of input, finally run it. This would greatly increase the productivity. It also helps industry to make simulation easier.

### 4.3 Seamless Model/File Transfer

After extensive using of SCA, it is found that seamless model/file transfer is a necessary auxiliary for distributed supply chain simulation. eSCA provides a seamless two way model/file transfer functionality between local computer and remote hosts/servers. A typical scenario will be like this: a modeler starts to build his/her model from local computer. After en extensive edit, modify, and test for test purpose parameter, he/she will use seamless model/file transfer functionality to transfer the model build at local computer to remote host/server for further intensive simulation/execution. The functionality of seamless model/file transfer provides a vehicle to server-based computers to conduct distributed simulation. It is a customized, user-friendly, workbench-oriented FTP. The seamless model transfer function promotes the usability of server-based computing as well as distributed simulation.

### 4.4 Batch Experiment—“Human Interactive-off”

In the interaction with nature, with machine, with computer, there are times, human being wish to be “out of the loop”. We need to take a coffee break, to live in an



CPU : 718 kFlops

Memory : 652 KB

```
Testing Host : vistakon.watson.ibm.com....done.
CPU : 4926 kFlops
Memory : 359 KB

Recommended Host : scs2000.watson.ibm.com
```

Figure 3: Model Catalog Library in a Server-Based Environment

electricity-free simple life. Or just simply get away from the daily use of computer. The Batch experiment of eSCA just provides the function of "Human Interactive-off interface" to free the users from the interaction with the computer.

A typical scenario will be: Users built the models, using seamless mode/file transfer function to transfer models/files to the servers, then simply instruct servers to run the models/experiments without bothering to know the graphical user interface. The whole simulation are running behind the scene for a relatively long time, longer than the time users spent in front of graphical user interface at the client side.

### 4.5 Post-Simulation Data Analysis

A simulation would not complete without post-simulation data analysis. One of the advantages of using eSCA as a modeler's workbench is seamless integrate post-simulation data analysis tools into the workbench. By combining the model transfer and batch experiment functionality, post-simulation data analysis provides a scan, review, report and publish the results from the intensive computation and simulations.

The current version of eSCA integrates Microsoft Access for data browser and Microsoft Excel for scenario Analysis.

A typical scenario of the post-simulation is similar to the user case in section 2.4.

### 4.6 Web-Enabled SCA

It is desirable to have Web-enabled SCA functionality as IBM's e-business gains industry leadership. The eSCA provides a very rapid way to access SCA through the web. Basically, this function fulfills the promise of web-distributed supply chain simulation. It proves that it indeed is a reusable architecture for various applications.

## 5 SUMMARY

eSCA is a thin client/server-based, web-enabled architecture and front-end system for distributed supply chain simulation. eSCA extends the current capability of SCA through a computing network with server-based and web-enabled functionality. The eManager version of eSCA provides model catalogs, interactive modeling, seamless model/file transfer, batch experiments, and post-simulation data analysis. The web-enabled version of eSCA provides a rapid way for users to access the SCA through the web. The functionality of eSCA can be demonstrated in eManager client and Web Client (Web-Enabled). We believe that the thin client/server-based, web-enabled distributed simulation is the industry first and its architecture is reusable.

## REFERENCES

Bagchi, Sugato, Steve Buckley, Markus Ettl, and Grace Lin, "Experience Using the Supply Chain Simulator", *Proceeding of the 1998 Winter Simulation Conference*, Washington, DC, 1998.

Bass, Len, Paul Clements, and Rick Kazman, *Software Architecture in Practice*, Addison Wesley, 1998.

Buckley, Steve and Jeffrey Smith, "Supply Chain Simulation", *Georgia Tech Logistics Short Course*, Atlanta, Georgia, 1997.

Fishwick, Paul A., *Simulation Model Design and Execution: Building Digital Worlds*, Prentice Hall, 1995.

Harwood, Ted, *Windows NT Terminal Server and Citrix MetaFrame*, New Riders, 1998.

Jacobson, Ivar, Grady Booch, and James Rumbaugh, *The Unified Software Development Process*, Addison Wesley, 1998.

Lin, Grace, Markus Ettl, Steve Buckley, Sugato Bagchi, David D. Yao, Bret L. Naccarato and Rob Allan, Kerry Kim, Lisa Koenig, Extended Enterprise Supply Chain Management at IBM Personal Systems Group and Other Divisions, *Interfaces,* Vol,3, No. 1, 2000 (to appear).

## AUTHOR BIOGRAPHIES

**H. BOB CHEN** is a Software Architect at the IBM Thomas J. Watson Research Center in Yorktown Heights, NY. His current research interests include components, patterns and complexity. He has a Ph.D. degree in Computational Fluid Dynamics from Clarkson University.

**OLIVER BIMBER** is currently a Scientist at the Fraunhofer Institute for Computer Graphics, Division Rostock, Germany, and a Ph.D. Student at the Technical University of Darmstadt, Germany. He holds a B.Sc. in Commercial Computing from the Dundalk Institute of Technology, Ireland, and a Dipl.Inf (FH) in Scientific Computing from the University of Applied Science, Giessen, Germany. His research interests are Intelligent Human-Computer -Interfaces for Large Scale Virtual Environments.

**CHINTAMANI CHHATRE** is an employee of IBM Global Services, India and currently working at the IBM Thomas J. Watson Research Center in Yorktown Heights, NY. He received the Bachelor degree from Pune University in Computer Science. In addition to simulation, his interests include Web architecture and programming.

**ELIZABETH POOLE** is a Staff Software Engineer at the IBM Thomas J. Watson Research Center in Yorktown Heights, NY. She holds a B.A. in Computer Science and a BFA in Graphic Design, both from the University of Arizona. Her recent work has been in developing user interfaces for manufacturing applications, and her interests include Human-Computer Interaction and software systems integration.

**STEPHEN J. BUCKLEY** is a Research Staff Member at the IBM Thomas J. Watson Research Center in Yorktown Heights, NY. He is currently the manager of the Supply Chain Analysis department, which developed the Supply Chain Analyzer (SCA) and eSCA. He received the Ph.D. degree from MIT in Computer Science. In addition to simulation, his interests include algorithms, scheduling, and robotics.