A Portable and Collaborative Distributed Programming Environment

Chang-Hyun Jo
Department of Computer Science
California State University, Fullerton
Fullerton, CA 92834-6870
jo@ecs.fullerton.edu

Allen J. Arnold
Department of Computer Science
University of North Dakota
Grand Forks, ND 58202
U. S. A.

Abstract

Due to recent advances in network and computing facilities, use of the Internet is becoming popular and promising in various areas such as scientific collaboration, information distribution and business processing. The need for a collaborative programming environment on the Internet is a natural extension to provide a cyber workspace for distributed software engineering and distance education, both now and into the future.

We have previously introduced a distributed programming environment, and recently we have redesigned and implemented a portable distributed programming system that supports real-time multimedia communication on the Internet. This paper reports our experience adopting multimedia components on the distributed programming environment such as simultaneous multimedia outputs in the distributed systems and an efficient multimedia communication among the participants. Our work may bring a new genre of software engineering – distributed software engineering, which includes many different views from the existing techniques in configuration management and software life cycle.

Keywords

Distributed software engineering, distributed programming environment, portable and collaborative programming environment, remote software management

--
IMCSE 2003 – SERP'03
Las Vegas, Nevada, USA
June 23-26, 2003

1. Introduction

Due to recent advances in computing facilities and the development of the network in which the computing resources are connected and shared, the programming environment is migrating from a single, local environment to a distributed and global environment. Therefore, a programming environment that can smoothly support cooperative works on the network is necessary.

Computing using distributed objects provides flexible and scalable programming on the distributed and parallel programming environment. Many systems such as CORBA, DCOM, and Java support distributed object computing [12, 18, 19].

People work together on scientific research, scholarly work, and software development in the environment of the computer-supported cooperative work (CSCW) [8, 13]. To successfully adopt CSCW, the most important issue is to develop an environment in which the users can collaborate with one another easily and naturally.

The aim of this paper is to present our experience designing and implementing a portable and collaborative programming environment on a parallel and distributed programming environment such as the Internet. We have previously built a prototype of a collaborative programming environment named *Distributed Programming Environment (DPE)* [7]. Using this DPE, programmers and project managers (or teachers and students) can be located anywhere, and effectively collaborate once they connect to each other via the Internet. Recently we have redesigned the system and have extended it to support real-time communication using multimedia components such as simultaneous multimedia output on the multiple hosts and an efficient real-time multimedia communication among participants.

The DPE system provides various extensions to the existing system to support Computer Supported Collaborative Work (CSCW) and the system is very practical in real-world application. This paper introduces the architecture and functions of the portable DPE system supporting real-time multimedia communication, reports our implementation experience, and discusses various problems and our solutions in implementing the system.

Using DPE, programmers can submit their programs remotely. Testers and project managers can compile and run the programs at different sites. All of this can be done remotely in real-time. They can manipulate the same windows, where they can edit, compile, and execute the programs together through the Internet. At the same time they can communicate concurrently by text-based chatting, audio and video. This can greatly enhance the cyber-work-environment of the near future. It may also alleviate the current and growing shortage of computing expertise and reduce total software development costs by removing unnecessary meetings, travel and work space. The current prototype of the DPE system is implemented by using Java/CORBA in the Internet environment. Our work may bring a new genre of software engineering – distributed software engineering, which includes many different views from the existing techniques in configuration management and software life cycle.

In the next section, we will present the general architecture of the DPE system. In the Section 3, we will describe how the DPE system manipulates the distributed real-time multimedia components. Comparisons with other work and conclusions are provided in the Section 4.

2. The architecture of DPE

The Distributed Programming Environment (DPE) provides the following facilities and benefits:

- Computer supported cooperative work (CSCW) environment for distributed software engineering on the Internet
- Advanced cyber-work-environment to help solve the problem of the expert-shortage in computing
- Dynamic formation of team members with a progressive working environment
- Flexible and scalable programming in distributed and parallel programming environments
- Portable and extendible to support collaborative scientific research, scholarly work, and distance education
- Extendibility to support software engineering processes, methods, and tools for virtual software engineering environment.
- High performance integrated and distributed software engineering tools
- Reduction of total software development cost by removing unnecessary meetings, travels, and working places
- Easy access to distributed programming environments, smart user interfaces, machine independence, remote project management, remote access to distributed resources, mobility, and load balancing
- Generating a new software engineering paradigm - distributed software engineering

With the DPE, both managers and software engineers can do the following tasks:

- Programmers and managers can communicate and collaborate easily and naturally while being located anywhere in the world once they connect to each other via Internet.
- Increased productivity through more natural collaboration
- Supporting distributed synchronous/asynchronous working modes
- Distributed programming, distributed managing, mobile computing and security
- Servers/clients, managers/programmers, suppliers/users, and teachers/students collaborations
- Distributed software engineering management

2.1 The DPE architecture

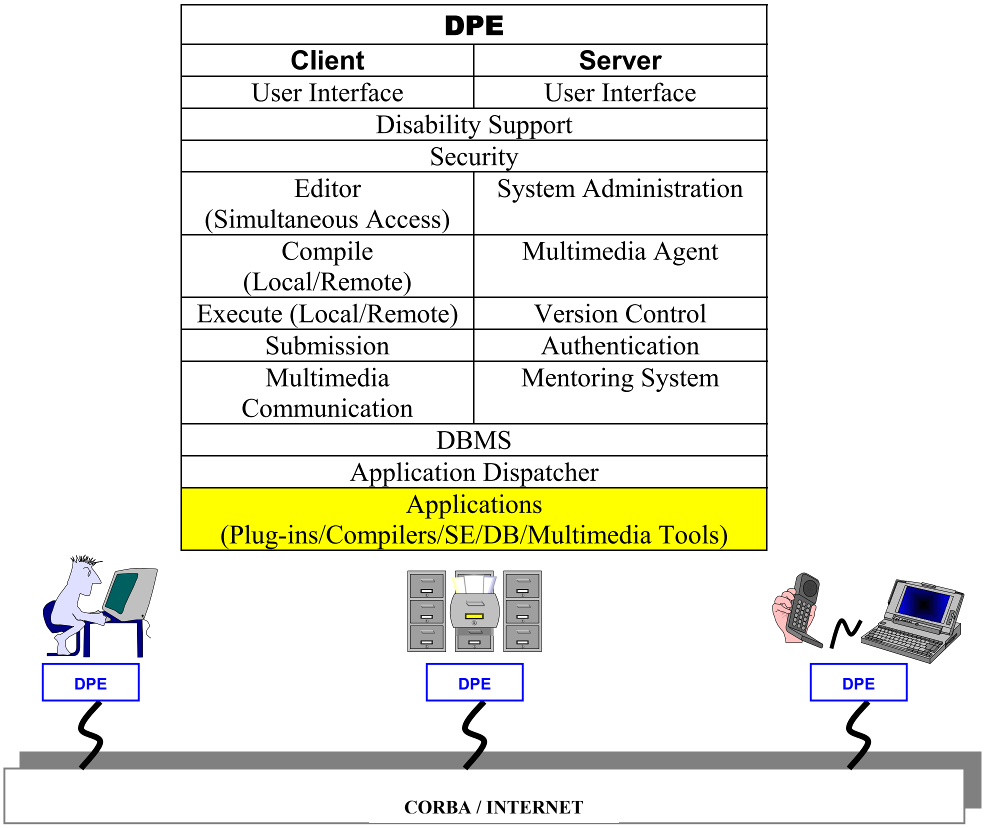
The DPE is an Internet-based software development system based upon a client/server model. Both DPE clients and servers are Java applications to avoid the limitation of web-based stateless systems. DPE utilizes CORBA IIOP for all non-multimedia communication and RTP for real-time audio and video [12, 18].

The DPE system maintains system security through a combination of an initial logon process and an ongoing identity validation process. The DPE itself is designed as a mobile system. The configuration of DPE is dynamic. A client can be connected to the network anytime and anywhere with functionality such as user interface, disability support, security, multi-access editing, simultaneous local/remote compilation, simultaneous local/remote execution, product submission, and communication through multimedia contents such as text-based chatting, audio and video support. The server component consists of several functions such as system administration, an agent to improve multimedia communication, version control of deliverables, dynamic authentication of participants, and a mentoring system to control information flow.

Figure 1 shows the architecture of the DPE system. DPE also includes the data management service to/from DB and several dispatchers of various kinds of applications such as plug-ins, compilers, software engineering tools, DB tools, and multimedia tools.

An example session of the DPE is also depicted in Figure 2. Several participants such as project manager, programmer and tester can simultaneously access the portable DPE system on the Internet and collaborate with each other to program, compile, execute, and test software projects.

An integrated development environment on distributed systems allows smooth communication between programmers located remotely, and may reduce the total software development cost and time by decreasing frequent meetings. A framework for distributed objects provides a higher level of both interoperability and transparency among the objects located in the distributed nodes. The DPE is a CSCW system that combines distributed computing and a collaborative software development environment.



DPE	
Client	**Server**
User Interface	User Interface
Disability Support	
Security	
Editor (Simultaneous Access)	System Administration
Compile (Local/Remote)	Multimedia Agent
Execute (Local/Remote)	Version Control
Submission	Authentication
Multimedia Communication	Mentoring System
DBMS	
Application Dispatcher	
Applications (Plug-ins/Compilers/SE/DB/Multimedia Tools)	

Figure 1. Architecture of Distributed Programming Environment

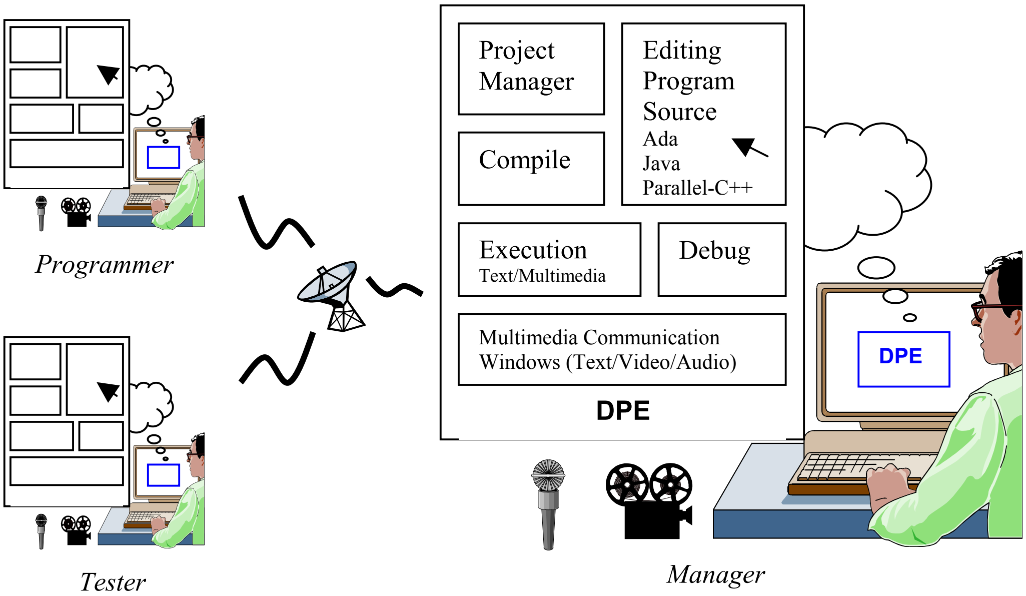
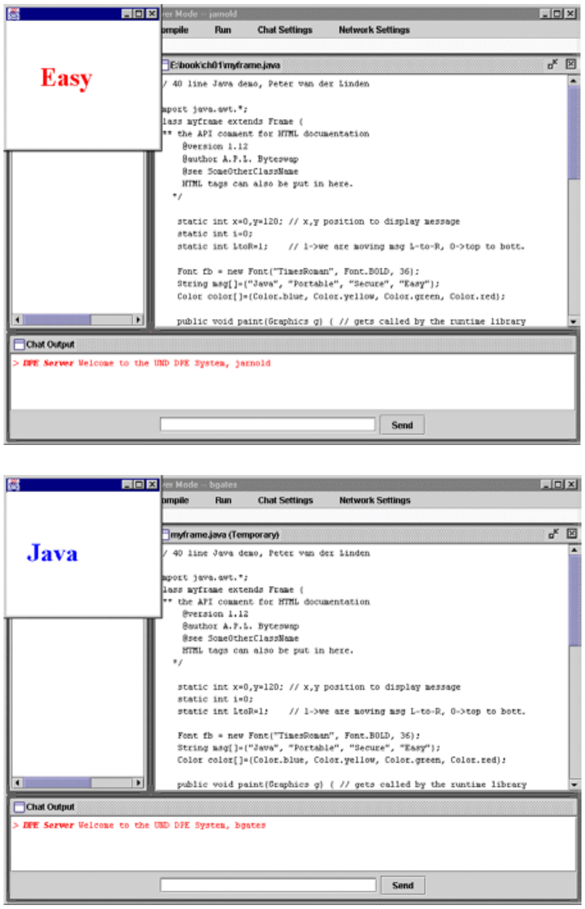


Figure 2. An example session of the DPE system

3. Distributed real-time multimedia components

To operate the DPE on the Internet properly, the interaction must be synchronized. First, it is necessary to synchronized real-time execution of the program in which output of the program is shown to both the programmers and the project manager. However, the output, including multimedia contents such as graphical output, should be handled as well. If the output at the programmer's site produces multimedia contents, the same output should be also shown at the manager's site. Such output should be compressed to reduce the bandwidth necessary for transmit, and the compressed output should be extracted and presented on the counterpart's screen within reasonable time.

To support better service on the DPE, the capability to transmit graphical output and multimedia components - motion pictures and audio contents, should be supported. For example, if a programmer compiles and executes his program on the remote site and the result includes graphical output, then the same output should be shown on the remote site of project manager in real-time. While they are looking at the same output, they may discuss the project, evaluate, and test programs on-line. Figure 3 shows the simultaneous multimedia output on two participants.



```
/ 40 line Java demo, Peter van der Linden

mport java.awt.*;
lass myframe extends Frame {
** the API comment for HTML documentation
    @version 1.12
    @author A.P.L. Byteswap
    @see SomeOtherClassName
    HTML tags can also be put in here.
  */

   static int x=0,y=120; // x,y position to display message
   static int i=0;
   static int LtoR=1;    // 1->we are moving msg L-to-R, 0->top to bott.

   Font fb = new Font("TimesRoman", Font.BOLD, 36);
   String msg[]={"Java", "Portable", "Secure", "Easy"};
   Color color[]={Color.blue, Color.yellow, Color.green, Color.red};

   public void paint(Graphics g) { // gets called by the runtime library
```

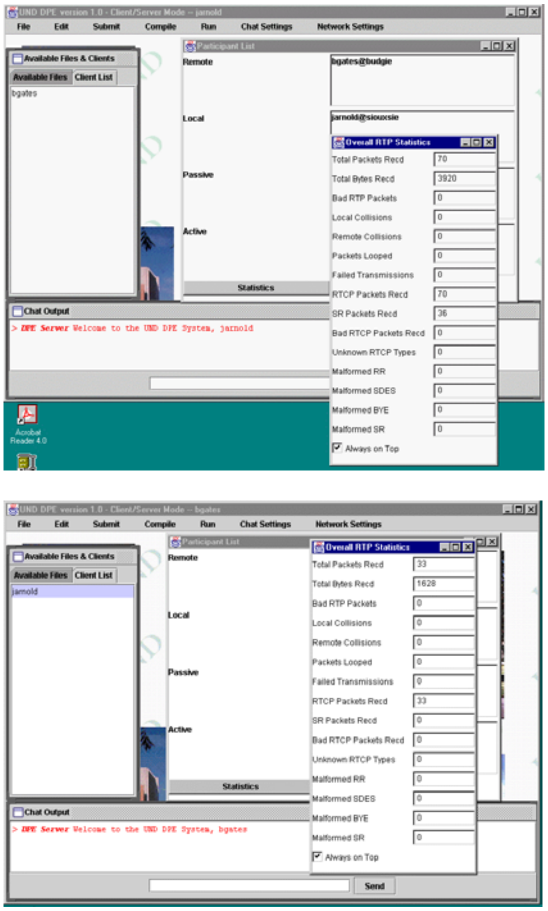
```
> DPE Server Welcome to the UND DPE System, jarnold
```

```
/ 40 line Java demo, Peter van der Linden

mport java.awt.*;
lass myframe extends Frame {
** the API comment for HTML documentation
    @version 1.12
    @author A.P.L. Byteswap
    @see SomeOtherClassName
    HTML tags can also be put in here.
  */

   static int x=0,y=120; // x,y position to display message
   static int i=0;
   static int LtoR=1;    // 1->we are moving msg L-to-R, 0->top to bott.

   Font fb = new Font("TimesRoman", Font.BOLD, 36);
   String msg[]={"Java", "Portable", "Secure", "Easy"};
   Color color[]={Color.blue, Color.yellow, Color.green, Color.red};

   public void paint(Graphics g) { // gets called by the runtime library
```

```
> DPE Server Welcome to the UND DPE System, bgates
```

Figure 3. Simultaneous multimedia output

To provide the most natural and productive service, communication among programmers and managers is essential. The DPE currently supports multimedia communication by using text-based chatting, audio and is in the process of implementing video. The DPE system is designed to support real-time audio and video communications. Various methods are currently under research at this time to optimize multimedia communication. Such new scheme for implementing the DPE system improves multimedia communication among participants. For example, if this DPE system will be used to support distance learning, DPE controls the information flow of audio/video communication among students and teachers very well. Teachers can easily control the information flow with the help from mentoring agents to divide sub-groups for discussion. Figure 4 shows that two clients are corresponding with each other using audio communication.



UND DPE version 1.0 - Client/Server Mode -- jarnold

bgates

bgates@budgie

jarnold@siouxsie

Overall RTP Statistics

Total Packets Recd 70
Total Bytes Recd 3920
Bad RTP Packets 0
Local Collisions 0
Remote Collisions 0
Packets Looped 0
Failed Transmissions 0
RTCP Packets Recd 70
SR Packets Recd 36
Bad RTCP Packets Recd 0
Unknown RTCP Types 0
Malformed RR 0
Malformed SDES 0
Malformed BYE 0
Malformed SR 0
☑ Always on Top

> DPE Server Welcome to the UND DPE System, jarnold

UND DPE version 1.0 - Client/Server Mode -- bgates

jarnold

Overall RTP Statistics

Total Packets Recd 33
Total Bytes Recd 1628
Bad RTP Packets 0
Local Collisions 0
Remote Collisions 0
Packets Looped 0
Failed Transmissions 0
RTCP Packets Recd 33
SR Packets Recd 0
Bad RTCP Packets Recd 0
Unknown RTCP Types 0
Malformed RR 0
Malformed SDES 0
Malformed BYE 0
Malformed SR 0
☑ Always on Top

> DPE Server Welcome to the UND DPE System, bgates

Figure 4. Real-time audio communication

The next generation of the DPE will not only include programming facilities, but also include the systems to support distributed software engineering environments. The processes, methods, and tools for distributed software engineering should be integrated in the DPE. The high performance Computer-Aided-Software-Engineering (CASE) tool can be combined into the DPE finally to support a full line of software engineering processes.

4. Comparison with other existing systems and conclusions

Tools for collaborative work are very useful for software engineering and science research. We have design and implemented a collaborative programming tool that supports programming in the distributed environment of the Internet and real-time multimedia communication, and simultaneous multimedia output among distributed systems. There are few comparable systems that currently exist in the commercial market and in research projects. The following is a comparison of our system with other works to show why our system has unique features, and to show where it is comparable with the work of others.

Lotus Notes [11] is definitely one of the pioneers in the area of collaboration and messaging software. Lotus Notes finds its root from PLATO Notes. Domino [11] provides interactive Web client access to data and applications on a Notes server. DPE is a kind of groupware to support distributed programming by software engineers on client/server systems. DPE includes facilities for collaborative work between programmers and managers, online discussions, software database, integrated programming environment, and tools for distributed software engineering, many of which the existing systems do not mention at all. The DPE supports communication, collaboration, and coordination among software engineers employed and/or engaged in the projects and distributed in the world, and an integrated application development environment on the Internet. The DPE is focusing on the future - virtual/cyber software factories that need distributed programming environments, while Notes and Domino focus on general business processes. However, the DPE may also be easily extended to support business processes.

There are many other related work, such as Single Display Groupware (SDG) [16], White Pine Software offers ClassPoint [20], WebCollab and DataCraft [3], The University of College

London (UCL)'s Real-time Transfer Protocol (RTP) [17], and some CSCW works [1, 2, 9, 10, 14, 15]. These works lack facilities to program, test, register and maintain software, and do not integrate facilities for distributed software engineering at all, while our system does. The collaborative work and tools are very useful for software engineering and science research. We have presented our experience to implement real-time multimedia communication facility into a distributed programming environment, which we built for use on the Internet.

The future work related to DPE includes audio user interface (AUI) and extension of multimedia communication using whiteboard and voice chatting. DPE is originally designed to support the audio user interface for the visually impaired persons. The research of audio user interface is separately undergoing and it has not yet adopted when this paper is written. Another extension to DPE such as whiteboard is also separately undergoing. Multilanguage support Ada and Parallel-C++ [5, 6] as well as Java [4] will be extended. The adoption of software engineering processes, methods, and tools into DPE is in progress too.

5. References

[1] CVU, California Virtual University, http://www.california.edu/campuses.html, (1999).

[2] Cornell, http://www.dl.cornell.edu/odl98/resources/cornell.resources.stm, (1999).

[3] IBM's Alphaworks, Webcollab, DataCraft, etc., http://alphaworks.ibm.com/, (1999).

[4] The Source for Java Technology, http://java.sun.com, (2003).

[5] Jo, Chang-Hyun and K. M. George. Language concepts using dynamic and distributed objects, Proc. of the ACM 1991 Computer Science Conference (CSC'91), ACM Press, (March 1991), 211-220.

[6] Jo, Chang-Hyun, et al. A Realization of Concurrent Object-Oriented Programming, Proc. of the ACM Symposium on Applied Computing (SAC'98), (Feb. 27 – Mar 1, 1998), 558-563.

[7] Jo, Chang-Hyun, et al. The Distributed Programming Environment on the Internet, ACM 1999 13th Annual Symposium on Applied Computing (ACM SAC'99), (Feb. 28 – March 2, 1999), 85-90.

[8] Kouzes, R. T., J. D. Myers, and W. A. Wulf, Collaboratories: Doing Science on the Internet, IEEE Computer, 29(8), (August 1996), 40-46.

[9] Lewandowski, S. M. Frameworks for Component-Based Client/Server Computing, ACM Computing Surveys, 30(1), (March 1998), 3-27.

[10] Lindwarm, D. and K. Norman, Student Evaluation of the Software in the AT&T Teaching Theater, CAR-TR-672, CS-TR-3069, ftp://ftp.cs.umd.edu/pub/hcl/Reports-Abstracts-Bibliography/93-06html/3069.html, (May 1993).

[11] Lotus Domino, http://www.lotus.com/, (1999).

[12] Orfali, R. and D. Harkey, Client/Server Programming with Java and CORBA, (2nd Ed.), John Wiley & Sons, Inc., (1998).

[13] Palmer, J. D. and N. A. Fields, Computer-Supported Cooperative Work: Guest Editors' Introduction, IEEE Computer, (May 1994), 15-16.

[14] Rose, A., W. Ding, G. Marchionini, J. Beale, Jr., and V. Nolet, Building an Electronic Learning Community: From Design to Implementation, Presented at the ACM SIGCHI'98 Conference on Human Factors in Computing Systems, http://www.learn.umd.edu/reports/chi98/chi98final.html, (1998).

[15] Shneiderman, B., E. Borkowski, M. Alavi, and K. Norman, Emergent Patterns of Teaching/Learning in Electronic Classrooms, Educational Technology Research and Development 46, 4, (1998), 23-42, also UMIACS-TR-98-21, CS-TR-3889, ftp://ftp.cs.umd.edu/pub/hcl/Reports-Abstracts-Bibliography/98-04html/98-04.html, (1998).

[16] Stewart, J., B. Bederson and A. Druin, Single Display Groupware: A Model for Co-present Collaboration, ftp://ftp.cs.umd.edu/pub/hcil/Reports-Abstracts-Bibliography/98-14html/98-14.html, (1998).

[17] University College London (UCL), Multimedia Integrated Conferencing for Europe, (1995).

[18] Visibroker for Java, VISIGENIC Reference Manual, V.3.2, "http://www.visigenic.com", (1998).

[19] Vogel, A. and K. Duddy, Java Programming with CORBA, John Wiley & Sons, Inc., (1997).

[20] White Pine Software, ClassPoint, "http://www.wpine.com", (1999).