wwHww : An Application Framework of Distributed Systems for Enduser-Initiative Development

Takeshi CHUSHO and Katsuya FUJIWARA
Department of Computer Science,
School of Science and Technology, Meiji University
Kawasaki, 214-8571, Japan
chusho@cs.meiji.ac.jp

Abstract

The number of endusers using the Internet increases on the inside and outside of offices. Enduser-initiative development of applications has become important for automation of their own tasks. As the solution by software reuse technology, this paper describes an Java-base application framework of the distributed systems such as the MOON (multi-organizational office network) systems for window work in banks, city offices, mail-order companies, etc. based on the philosophy : "All routine work both at office and at home should be carried out by computers." The application framework includes a common protocol for application forms and is composed of subframeworks corresponding to the three kinds of parts of client terminals, server-at-windows and the MOON servers. With respect to the problem of how to customize the application framework in the way of enduser-initiative development, two types of customization are given, namely, plug-in components and property definitions. Then domain experts can computerize their routine work by themselves. Our feasibility study confirms that the frozen spots and the hot spots in the framework account for 90 % and 10 % of the total Java source programs respectively. Furthermore, use of the application framework including a common communication protocol between clients and servers, brings high interoperability among distributed application systems.

Key words :

distributed system, Internet and intranets, application framework, object-oriented technology, enduser computing

1. Introduction

The number of endusers using the Internet increases on the inside and outside of offices. The spread of GII(global information infrastructure) accelerates this trend.

The word "enduser" has already implied people in the wider range than those who belong to enduser departments contrasted with information system support departments in user companies. Although many controversies about how to change work with computer systems are aroused [8, 15], this paper is based on the following philosophy:

"All routine work both at office and at home should be carried out by computers."

Our policy for this purpose is enduser-initiative development by using tools for implementing distributed information systems of the endusers, by the endusers, for the endusers because package software may not substitute for various work of various endusers.

As the solution by software reuse technology, this paper describes an application framework, wwHww (pron."who"), for enduser computing under distributed systems [2]. As a typical distributed information system, we direct our attention to an application system for windows or counters in banks, city offices, travel agents, mail-order companies, etc. Although some kind of window work such as mail-order business has already been put to practical use in current computer networks including the Internet, both friendliness of enduser interfaces for clients and automation of routine work for domain experts are still insufficient. In addition, the work to be automated may be limited to particular ones which make a profit over the development cost.

In the near future, the information society will require such new technologies that domain experts can automate their own work by themselves and that almost all clients can operate computers at home or at office without extra training or without the help of others. Furthermore, a common protocol between the windows and their clients must be developed for avoiding appearance of a number of incompatible interfaces corresponding to the explosive number of combination of the windows and their clients.

An application framework must be the solution for these problems. There are related studies in the research fields of

object-oriented technology for software reuse [7, 18, 22]. Essential concepts of object-oriented technology came out around 1970 and were expanded into programming methodologies through 1980's and into software analysis/design methodologies [5, 19] in 1990's. Object-oriented programming has already been used in practice into various software fields. In particular, distributed object management systems such as CORBA and DCOM become important as middleware for distributed object management with network transparency. As our policy, it is easy to extend the wwHww system in order to use these infrastructures as platforms. This is because the wwHww system was designed as a distributed cooperative system based on a message-driven model of object-oriented technology and was implemented as an Java-base application framework [12, 13, 21].

An application framework implies a reusable semi-complete application or a skeleton of an application that can be specialized to produce custom application, and that is represented by both a set of abstract classes and the way their instances interact [4, 14]. Recently for enterprise system construction, the Java-base framework [10] and business objects [1] have been directed attention to. One of the real issues is how to customize a framework for getting appropriate applications.

This problem is especially important in the way of enduser-initiative development. The wwHww system gives two types of customization, namely, plug-in components and property definitions. Then domain experts can computerize their routine work by themselves. Furthermore, use of the application framework including a common communication protocol between clients and servers, brings high interoperability among distributed application systems.

This paper presents the target of the wwHww system in Section 2, the requirements for the application framework in Section 3, the conceptual design in Section 4 and the software architecture in Section 5.

2. An Example of Application

Recently, business process re-engineering [9] has begun to attract notice since information technology may drastically improve efficiency and effectiveness of company activities. Furthermore, information technologies represented by the words "Internet" and "multimedia," will give the power of process re-engineering of human society.

Let's suppose that you move from some city to any city. How many windows in various organizations do you visit and how many application forms do you fill in there for an address change? For example, two city offices, a telephone company, an electric power company and a gas company for your residence, a post office for mail transfer, the transport bureau for your car life, banks, insurance companies and credit companies for your living, academic societies for mail etc. Furthermore the account section, the personnel section and the general affairs section in the office where you work.

Most of these windows require for us to visit there, get application forms, fill in them and hand them to domain experts at windows. If these processes can be executed by using our computer at home or at office, a lot of time will be saved.

As for domain experts at windows, they are asked about the same question of how to fill in the form again and again every day and repeat the same explanation and the same check of written applications. If these routine work can be processed by computers, a lot of cost will be reduced. Furthermore, domain experts may spend the saved time on service upgrade or some creative work.

Of course many kinds of window work have already been put to practical use in the Internet and intranets. For example, the WINGS(Web Interactive Network of Government Services) system of USPS [20] realizes one-stop shopping for integrated electronic government customer services. People can file a change of address with the U.S. Postal Service and forward that information to your choice of federal, state and local government sources. However, these systems must have been developed by professionals, not by endusers and are expensive.

Our goal is development and maintainance of applications by endusers. As a typical distributed information system, we direct our attention to application systems for window work.

3. Requirements for Application Framework

3.1. Application system architecture

The main purpose of the application framework, wwHww, is the easy construction of a multi-organizational office network (MOON) for window work. The MOON system is based on a client/server model and is partitioned into the following three parts as shown in Figure 1:

1. Client terminals to send written applications to windows, such as personal computers and workstations both at home and at office, portable computers for mobile computing outdoors, public telephones with terminals in town, etc.

2. Server-at-windows to receive written applications, such as windows in mail-order companies, city offices, travel agents, universities, academic societies, caterers, etc.

3. MOON servers to manage the network system.

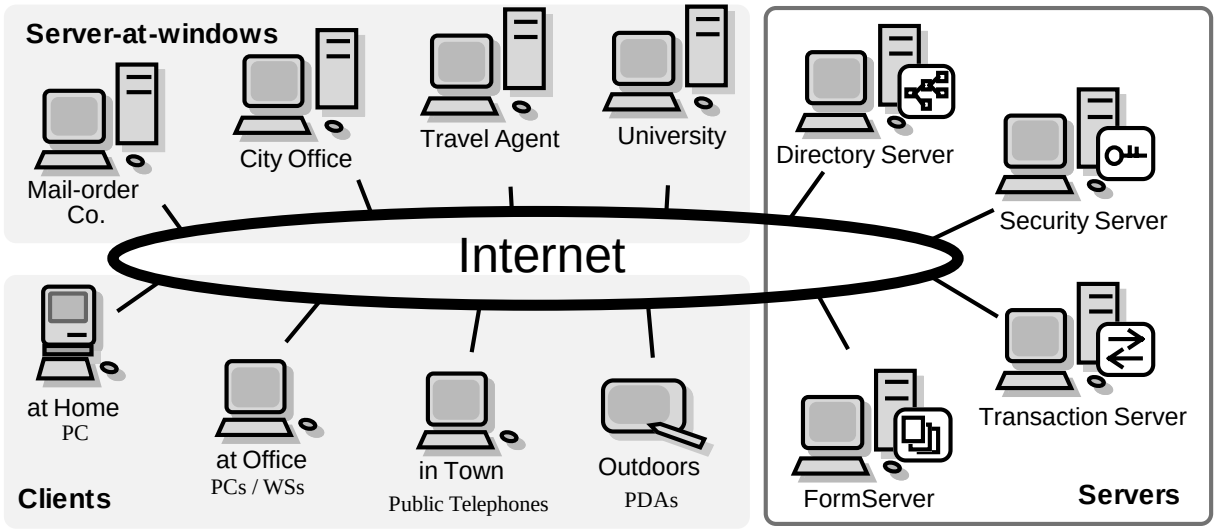

Figure 1. A MOON(multi-organizational office network) system.

The MOON servers imply the following four servers:

1. A directory server manages network addresses of server-at-windows to receive written applications as service directories of windows.
2. A form server manages various application forms for services at these windows, which forms are defined with help messages and selection menus by domain experts.
3. A transaction server stores written applications received by server-at-windows with the identification numbers, manages the states of the process and replies to inquiries about the states. It may be connected with a workflow system in the organization including the server-at-window.
4. A security server controls access rights to server-at-windows and the MOON servers, and manages authentication of clients.

3.2. Features of application systems

3.2.1. Automatic form processing

The first feature of the MOON system is electronic form processing with navigation for enduser operations by agents [16, 17] both in client terminals and in server-at-windows.

For enduser-initiative application development, the following requirements for the application framework of the MOON system are essential:

1. Clients can operate terminals and can teach the fixed operations to their agents by themselves. Then clients don't need to write such plain words as their names, addresses and phone numbers since their agents do them instead.
2. Domain experts can teach their expertise to their agents by themselves. By using the expertise, the agents lead clients into filling in the form and check the written form.

These facilities bring freedom from routines to both clients and domain experts.

3.2.2. A common protocol

The second feature of the MOON system is standardization of a common protocol for communication between client terminals and server-at-windows.

It must be very convenient if we can communicate our requests to a window via a computer network without visiting the window. However, we must learn various user interfaces as such windows increase. Furthermore, we must learn different operations of different types of terminals as we can use various types of terminals at various places for our requests. This inconvenience is caused by developing each application system individually. For example, although some commercial computer network service provider in Japan supports five mail-order book shops, their user interfaces are all different each other.

In order to avoid developing M*N kinds of user interfaces on application software for connecting M kinds of server-at-windows to N kinds of client terminals, we developed a common protocol between servers and clients, which reduces the number of user interfaces from M*N to M+N.

This common protocol brings the intelligent information retrieval for server-at-windows to clients as mentioned later.

4. Conceptual Design

4.1. The basic form

The common protocol for communication between server-at-windows and client terminals in the MOON system, is named the wwHww-protocol and includes the following message components of requests to windows:

>Who receives your request?

>What do you request to the window?

>How do you request it?

The name of wwHww is derived from 'who-what-how with WWW' and is pronounced as 'who' for convenience. The following component is added to these three components.

>Which is your request?

That is, the basic form of the wwHww-protocol is shown as follows:

(who, what, how, which)

4.2. Semantics of the protocol

The semantics of the protocol is based on a message passing concept of object-oriented technology. The four parameters of the protocol correspond to components of a message between objects as follows:

who : A message receiver

what : A method name

how : Parameters for a method invocation

which : A message number

In the MOON system, the who-parameter implies a window where a written application is sent to. The what-parameter implies the title of the application form. The how-parameter implies contents of the application form. The which-parameter implies a receipt number stamped on the received written application.

The states of values of these parameters affects semantics of the message. If a value of a message parameter is unknown, the message implies an inquiry about the parameter. This semantics is quite different from conventional object-oriented programming languages because it is illegal not to determine the message receiver, the method name and the actual parameters for conventional message sending. This extention in this paper, however, produced atractive effect. Examples are given in the next subsection.

4.3. Enduser interface

The actual enduser interface of the common protocol is different from the basic form which is the internal representation in the system. The enduser interface depends on the kind of the enduser's terminal. It may be an interface of filling in the form displayed on a screen. It may be the other interface of sequentially inputting answers corresponding to questions displayed in characters. Some of them may display icons or a menu for selection.

Examples of requests by using the common protocol are given in the basic form for convenience, where the following notation is used:

a, b, ... : Parameters with known values.

?a, ?b, ... : Inquiries about the parameters with known values, which request help messages.

x, y, ... : Parameters with unknown values.

?x, ?y, ... : Inquiries about the parameters themselves, which request all possibles for selection.

1. Examples of sending written applications:

   (a) (a, b, c, x)
   The written application, b, with the contents, c, is sent to the window, a. A message number will be assigned to the variable, x, by the window receiving this message.

   (b) (a, b, c, d)
   The written application, b, of the message number, d, is sent again to the window, a, for change of the contents, c.

   (c) (a, b, , ?d)
   The state in the process of the written application, b, of the message number, d, is inquired of the window, a.

   (d) (a, b, , -d)
   The written application, b, of the message number, d, which was already sent to the window, a, is canceled.

2. Examples of inquiries about application forms:

   (a) (a, b, ?x,)
   The application form, b, to be sent to the window, a, is displayed. How to fill in the form is navigated.

   (b) (a, ?x, ,)
   The title list of all application forms which the window, a, receives, is displayed.

   (c) (?x, ?y="k", ,)
   The list of titles of all application forms related to the keyword "k" is displayed with the names of windows receiving them. The system retrieves forms whose titles include the keyword or in which at least one of help messages includes the keyword.

(d) (?x, ?y, ,)
All windows and all titles of application forms which are received by those windows, are listed.

(e) (?a, , ,)
The explanation on the work of the window, a, is displayed.

(f) (a, ?b, ,)
The explanation on the application form, b, to be sent to the window, a, is displayed.

4.4. An example of operations

Let's consider a student who wants to get his transcript from a university and suppose that he does not know where and how he gets it in the university. Figure 2 shows operations to be done by using his personal computer as a client terminal of the MOON system at home. The operations and the basic form of a common protocol to be sent to the system at each step are described as follows:

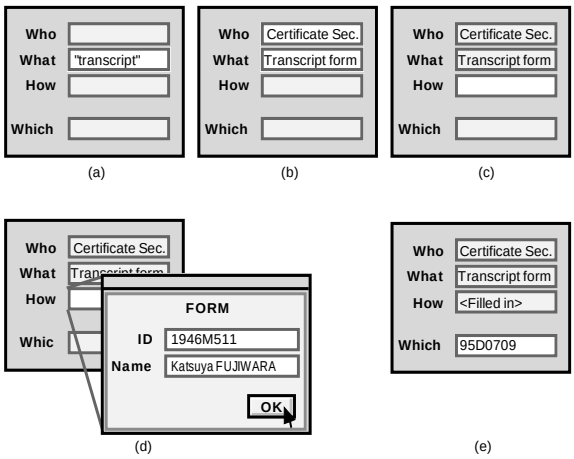

Figure 2. An example of operations of the MOON system at home.

(a) The keyword "transcript" is inputted to the what-column in the initial screen, and then the basic form of (?x, ?y="transcript", ,) is sent.

(b) The system displays "Certificate Sec." in the who-column and "Transcript form" in the what-column. By clicking the value area in the what-column, the basic form of ("Certificate Sec.", ? "Transcript form", ,) is sent.

(c) The system displays the help message on the transcript form. After clicking the value area of a blank in the how-column, the basic form of ("Certificate Sec.", "Transcript form", ?z,) is sent.

(d) The system displays the application form. After filling in the two blanks for the ID no. and the name, the basic form of ("Certificate Sec.", "Transcript form", ("ID no."="1946M511", "name"="Katsuya FUJIWARA"), w) is sent.

(e) The system displays the message number in the which-column, while the value is assigned to the variable, w. Then the system terminates by clicking the close button.

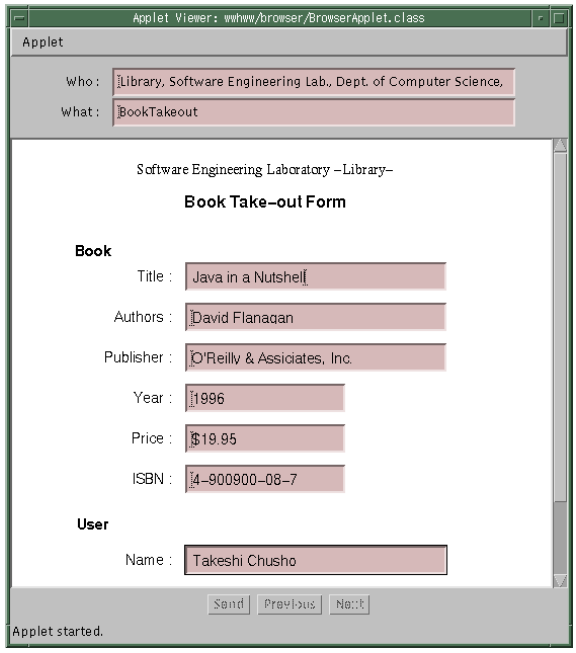

Figure 3. An example of the wwHww browser at a client terminal.

5. Application Framework

5.1. Software architecture

The first version of the application framework has been developed with an application of a library system, which is used in our laboratory [6]. Since there are no librarians in our laboratory, the following functions are convenient:

1. It is easy for us to know who borrows the book which we want to read because everyone fills in an electronic application form when taking out a book from our laboratory.

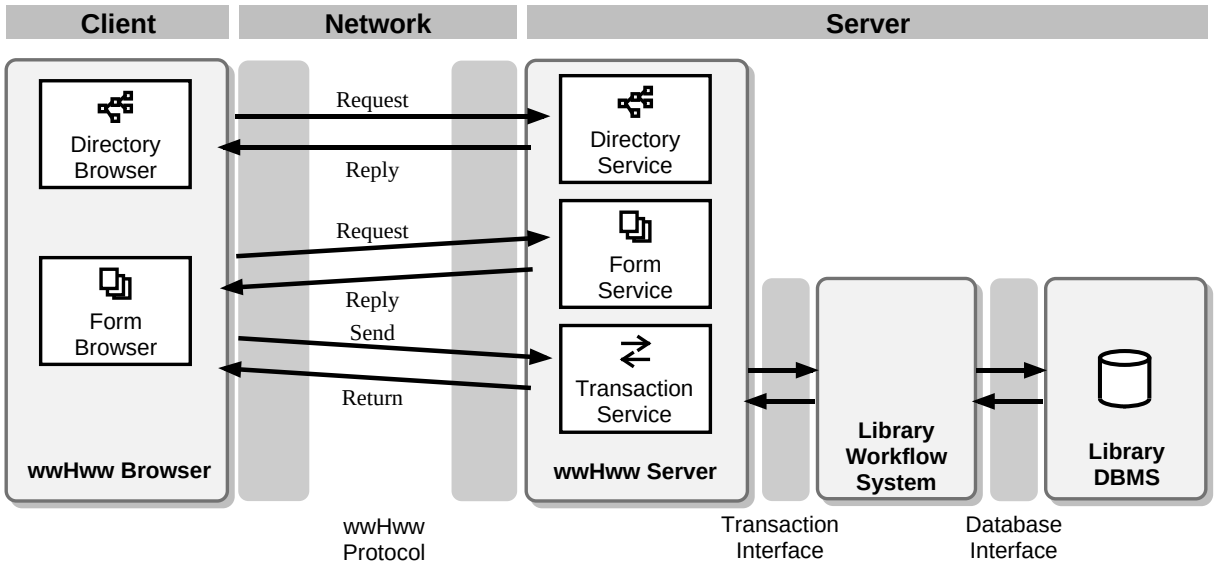

Figure 4. An application and the framework.

2. It is easy for us to know whether the book to be bought has been already registered or not because everyone fills in an electronic application form after he or she bought a book for our laboratory.

An example of an electronic application form is displayed by a wwHww browser as shown in Figure 3. This is a book take-out form. The head part of the form indicates the who-parameter of the wwHww protocol, that is, the name of the server-at-window, and the what-parameter of the wwHww protocol, that is, the name of the service to be requested. The body of the form implies the how-parameter of the wwHww protocol, that is, the application form itseslf requested.

The software architecture is based on a client/server model as shown in Figure 4. The wwHww browser of the client side is composed of two subsystems, that is, the form browser and the directory browser. The wwHww server of the server side is composed of three subsystems, that is, the directory server, the form server and the transaction server.

This system was implemented in Java and then runs on Solaris2, Windows95 or WindowsNT. There are two types of the wwHww browser, that is, a Java applet version as shown in Figure 3 and a Java application version. The wwHww protocol for communications between the wwHww browser and the wwHww server on the Internet, was inplemented by Java RMI. Although the workflow system is out of range of the application framework for MOON systems, It was implemented in Java and Oracle7.

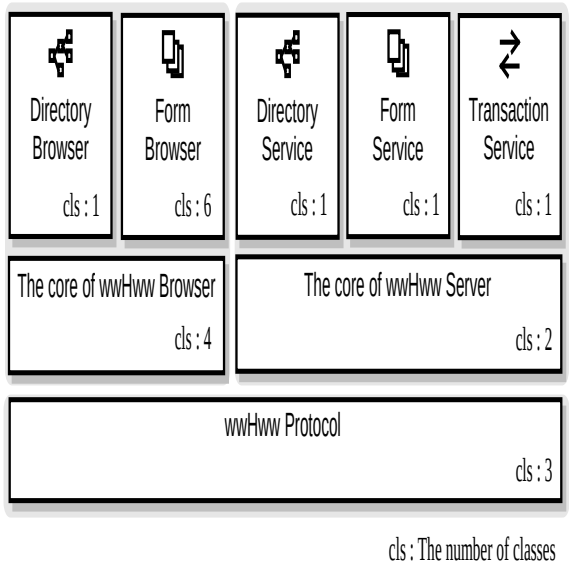

Figure 5. The wwHww application framework.

5.2. Classes for the framework

The wwHww application framework is composed of the eight subframeworks as shown in Figure 5, and supports the wwHww browser, the wwHww server and the wwHww protocol in Figure 4. It does not include the workflow system. The total number of classes for the application framework is 19 classes with 1,500 lines of Java source programs.

In addition, the number of classes for the workflow system is 7 classes with 600 lines of Java source programs.

The following two types of customization on hot spots are required for applying the framework to application development.

1. Use of plug-in components : This system provides a printer component and a DB component as interfaces to the workflow system, both of which support the typical processing of written forms. If the domain experts use these plug-in components, they need not any programs for transaction processing.

2. Definition of properties : This system provides property sheets for defining specification of application forms and help messages, which are dependent on applications. If the domain experts use these property sheets, they need not any programs for application forms.

5.3. An application building procedure

Consequently domain experts build applications by using the framework as follows:

1. Service definitions : Services to be done at the window, are defined.

2. Form definitions : Electronic forms for these services are defined while embedding navigation information into these forms.

3. Transaction processing definition : How to process written forms is defined. There are three typical processing methods of printing out, storing in a database or passing to a workflow system.

4. Registration : These definitions are registered into the corresponding servers.

For practical use, the application framework has been applied to a library system as mentioned before. First, services for registration, retrieval, lending and return of books in our laboratory were defined. Next, the formats of corresponding forms were defined. Then the part of the work flow system related to library database management was newly developed as a reuse component. Finally the system was installed into the server machine of our laboratory.

An example of a browser for system definitions by domain experts is shown in Figure 6. The left-hand part implies a hierarchical directory. The right-hand part implies definitions about the service for lending books.

The total of source programs amounts to 2,100 lines in Java. The part of window work corresponding to the MOON system, accounts for 70 % of the total. The frozen spots and the hot spots do for 90 % and 10 % of that respectively. In comparison with some report on trial of applying the early version of the San Francisco framework [11], which told that the frozen spots and the hot spots account for 40 % and 60 % respectively, our feasibility study implies that commonality of window work among various domains is high and that window work is suitable for the enduser-initiative development.

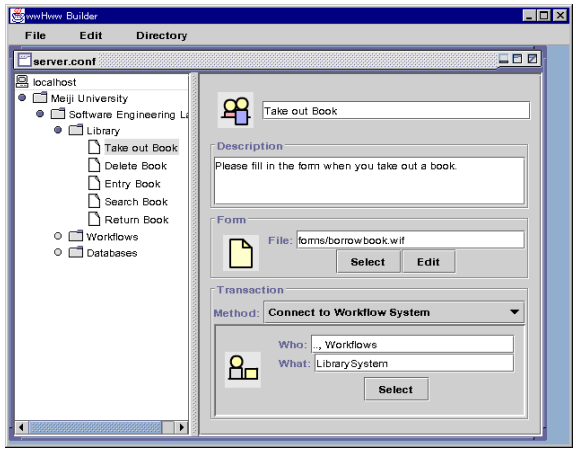

Figure 6. An example of the browser for system definitions by domain experts.

6. Conclusions

The application framework for distributed system was developed. A MOON (multi-organizational office network) system is easily developed based on this framework by domain experts themselves. In addition, the common protocol for communication between the M kinds of server-at-windows and the N kinds of client terminals reduces the number of application interfaces for the MOON systems from M*N to M+N.

Clients can get application forms without visiting windows, understand how to fill in the form and send them. Domain experts become free themselves from frequently asked questions at windows. These benefits were ascertained by developing the programs for feasibility study.

Further study is needed for improvement of user-friendliness for both domain experts and clients of windows by applying the framework to various applications in practical use. In addition, after development of an application for window work, domain experts will require development of the workflow system which is linked to the application for window work. Since we have already studied the application development environment for workflow systems [3], the both systems will be joined.

References

[1] Casanave, C., “Business-Object Architectures and Standards,” http://www.dataaccess.com/bodtf/BOPaper.htm.

[2] Chusho,T., “wwHww : An Object-Oriented Model for Enduser Computing in Distributed Office Systems,” (in Japanese), Information Processing Society of Japan, SIG on Software Engineering, vol.94, no.18, pp.33-40, Mar. 1994.

[3] Chusho,T., Matsumoto, M. and Konishi, Y., “M-base: Enduser-Initiative Application Development based on Message Flow and Componentware,” COMPSAC98, IEEE Computer Society, pp.112-120, Aug. 1998.

[4] Fayad, M. and Schmidt, D. C. (Ed.), “Object-Oriented Application Frameworks,” Comm. ACM, Vol. 39, No. 10, pp. 32-87, Oct. 1997.

[5] Fichman,R.G. and Kemerer,C.F., “Object-Oriented and Conventional Analysis and Design Methodologies,” IEEE Computer, vol.25, no.10, pp.22-39, Oct. 1992.

[6] Fujiwara, K. and Chusho,T., “Experiments in Application Framework for Distributed Systems on Multi-Organizational Office Network,” (in Japanese), Information Processing Society of Japan, SIG on Software Engineering, vol.97, no.74, pp.65-72, July 1997.

[7] Gamma, G., Helm, R., Johnson, R. and Vlissides, J., Design Patterns, Addison-Wesley, 1995.

[8] Grudin, J., “Computer-Supported Cooperative Work : History and Focus,” IEEE Computer, vol.27, no.5, pp.19-26, May 1994.

[9] Hammer, M. and Champy, J., Reengineering the Corporation, Harper Collins, 1993.

[10] IBM San Francisco, http://www.ibm.com/Java/Sanfrancisco/

[11] IBM Sharable Framework, (in Japanese), http://www.developer.ibm.com:8080/welcome/java/sfindex.html

[12] Islam, N., “Customizing System Software Using OO Framework,” IEEE Computer, vol.30, no.2, pp.69-78, Feb. 1997.

[13] Johnson, R. E., “Components, Frameworks, Patterns, ” Proc. Simposium on Software Reusability, pp.10-17, May 1997.

[14] Johnson, R. E., “Frameworks = (Components + Patterns),” Comm. ACM, Vol.40, No.10, pp.39-42, 1997.

[15] Kling, R., “Controversies About Computerization and the Organization of White Collar Work,” UCI Technical Report (Draft 7a) , 1995.

[16] Maes, P., “Agents That Reduce Work and Information Overload,” Comm. ACM, vol.37, no.7, pp.30-40, Jul. 1994.

[17] Malone, T., Lai, K. and Fry, C., “Experiments with Oval : a Radically Tailorable Tool for Cooperative Work,” CSCW92, pp.289-297, 1992.

[18] Mellor, S. J. and Johnson, R., “Why Explore Object Methods, Patterns, and Architectures ?,” IEEE Software, Vol. 14, No. 1, pp.27-30, 1997.

[19] Monarchi,D.E. and Puhr,G.I., “A Research Typology for Object-Oriented Analysis and Design,” Comm. ACM, vol.35, no.9, pp.35-47, Sep. 1992.

[20] MoversNet, http://www.usps.gov/moversnet/

[21] Sparks, S., Benner, K. and Faris, C., “Managing Object-Oriented Framework Reuse,” IEEE Computer, vol.29, no.9, pp.52-61, Sep. 1996.

[22] Tepfenhart, W. M. and Cusick, J. J.,”A Unified Object Topology ,” IEEE Software, Vol. 14, No. 1, pp.31-35, Jan. 1997.