A Middleware for Problem Solving Environment on Heterogeneous Computers Using Web-based Agents Cooperation: WAPSE

Yoshikazu Hayase, Shigeo Kawata
1 Toyama National College of Technology, 1-2 Ebie-Neriya, Imizu, Toyama 933-0293, Japan, hayase@nc-toyama.ac.jp
2 Utsunomiya University, 7-1-2 Yohtoh, Utsunomiya, Tochigi 321-8585, Japan, kwt@cc.utsunomiya-u.ac.jp

Abstract
A PSE (Problem Solving Environment), called as WAPSE, is proposed and constructed based on a cooperation of Web-based agents, which are distributed on one or more heterogeneous computers. WAPSE is a middleware for PSE using cooperation of a simple Web-based agents' group distributed on heterogeneous Web servers and clients. WAPSE's agents cooperate autonomously in a group and solve a problem. A user interface as a part of Web-based PSE is built easily by WAPSE on a Web browser for an easy use of PSE. A simple process combining multiple modules is already popular, because Web service technologies have been progressed. However, it is difficult to build a Web-based agents' system in primitive Web environments, because the agents must be able to behave autonomously and communicate with each other. It is troublesome to deploy a large numbers of Web-based agents to distributed computers at once. The debugging of a Web-based agent and change of the processing computers are commonly problems at a development stage. Therefore, we developed WAPSE so that a developer builds Web-based agents more easily. PSE developers can build Web-based agents without difficulties on WAPSE, because WAPSE provides the basic functions such as communication and the file lock. Though the WAPSE's architecture is simple, it realizes a cooperation among distributed agents in asynchronous. One simple Web-based agent's behavior is predictable. However, its behavior could become complex even in cooperation of two simple Web-based agents. The cooperation among many simple Web-based agents provides complicated behavior. Furthermore, WAPSE runs also on several mobile terminals as processing computers though a Web browser, on which AJAX is available.

Keywords: Web-based System, Distributed Processing, Agent, Agent based scientific computation, PSE (Problem Solving Environment)

1. Introduction
A purpose of PSE (Problem Solving Environment) research is a provision of the environment where users can concentrate on a problem solving and need not have the specialized knowledge of the underling hardware or software [1-5]. A part of developed applications, which support users, have capability equivalent to PSEs. However, most of these applications require specific skills to use of users. On the other hand, almost everyone can easily use a Web browser without assistances, because the Internet became popular and important. Then, the application developers, who are not the specialists of computer and network, desire to have PSEs, for the easy development of applications. There are various PSEs for a specific problem solving in the different composition including a research institute and the university, etc. In some problems, these PSEs customized slightly on heterogeneous servers over the different organizations can cooperate in problem solving by using UNICORE [6], GRID [7], NAREGI PSE [8, 9], etc. However, since many organizations use firewalls and different policies, a task for the communication among the organizations is complicated. On the other hand, a program using the Web technology can communicate among servers relatively easily. In addition, since the Web service technologies have been progressed, a simple cooperative process with some modules is possible in primitive Web environments. Various Web servers connected to the Internet can share each resource. Therefore, the architecture using the Web service technology would be appropriate for some PSEs. Moreover, a user interface that is constructed based on a Web browser is a promising candidate for PSE's user interface. A user interface as a part of Web-based PSEs is built based on a Web browser. The main system of Web-based PSEs except a user interface is unseen to end users.
PSE users access Web-based simple PSEs using a Web browser to solve problems (see Fig. 1). Although the computer literacy skill of PSE users still varies, almost everyone can use a Web browser without assistances. A Web-based PSE system is built on a popular Web server environment such as Apache and PHP scripts, etc. A Web-based PSE developer deploys Web modules for a PSE system to distributed processing computers, that are Web servers. These Web modules cooperate to solve a problem according to a design by a developer. We define a Web-based agent as a Web module that can behave asynchronous autonomously [10]. It is difficult to build a Web-based agents' system in primitive Web environments, because the agents must be able to behave autonomously and communicate with each other. Furthermore, it is troublesome to deploy a large numbers of Web-based agents to distributed computers at once. The debugging of a Web-based agent and change of the processing computers are commonly problems at a development stage. Therefore, we developed WAPSE [11-14] so that a developer builds Web-based agents more easily. WAPSE is a middleware for PSE using cooperation in asynchronous of a simple Web-based agents' group handled distributed on heterogeneous Web servers and Web clients. Web-based PSE developers can build Web-based agents without difficulties on WAPSE, because WAPSE provides the basic functions such as communication and the file lock. PSE developer can perform the maintenance of agents at once easily, because WAPSE has the mechanism that deploys Web-based agents to distributed computers. On the other hand, Web server process response is unstable on a network, because the response time is influenced by a load of servers, network traffics, etc. A response of a Web-based PSE, which cooperates with Web-based agents according to a static sequential workflow, is a sum of Web-based agents' uncertainty. A response of a Web-based PSE constructed with many Web-based agents becomes more complicated. Though the WAPSE’s architecture is simple, it realizes a cooperation among distributed agents in asynchronous. One simple Web-based agent's behavior is predictable. However, its behavior could become complicated even in cooperation of two simple Web-based agents. The cooperation among many simple Web-based agents provides complicated behavior. This paper is structured as follows: in Section 2 we discuss the cooperation of the Web modules as the Web-based agents. In Section 3 we describe WAPSE for Web-based PSE and show some application examples solving problems. Finally, Section 4 concludes.

2. Cooperation of Web modules as PSE

Simple process in cooperation of some Web modules is already possible, because Web service technologies have been progressed. For example, a Web module is scripts which are described by PHP or Perl, etc. There are two categories roughly of sequential cooperation and asynchronous cooperation for distributed Web modules.
2.1. Sequential cooperation

Some Web modules which constitute Web-based PSE operate according to workflow sequentially among distributed Web servers. Some sequential cooperation workflow of Web modules is expressed in Eq. (1) and can easily predict a result.

\[
\begin{align*}
\text{Module}_1 & \Rightarrow \text{Module}_2 \Rightarrow \text{Module}_3 \Rightarrow \cdots \Rightarrow \text{Module}_{n-1} \Rightarrow \text{Module}_n \\
\text{result}_n &= \text{Module}_n(\text{Module}_{n-1}(\text{Module}_{n-2}(\cdots (\text{Module}_1()))))
\end{align*}
\]  

Eq. (1)

In this respect, each Web module waits for processing before one to be finished. Figure 2 is an example of sequential cooperation of multiple Web modules. BABEL [15, 16] is a PSE developed using Web modules sequential cooperation as a Web technology dictionary of multilingual engineering terms for foreign students in Japan. BABEL is a term-based translation support system by sequential cooperation between two Web modules that are a multilingual dictionary module and a hypertext formation module. Although the BABEL is constructed with these two Web modules, users can use the BABEL without being conscious of the cooperation of modules.

Figure 2. A sequential cooperation of multiple distributed Web modules constitutes a Web-based PSE. The server-\(\alpha\) is a simple PSE which consists of module-A and module-B, and each module cooperates. Although the module-A is required to constitute PSE, the server-\(\gamma\) has only the module-D. Therefore, a PSE which consists of module-D cooperates with the module-A which is on the server-\(\alpha\) or the server-\(\beta\). For example, a user uses BABEL using Web browser as user interface on the user's own computer. Each Web module can cooperate with the local database system such as MySQL, too.

Eq. (2) shows that two modules join in one module.

\[
\begin{align*}
\text{Module}_{k-2} & \Rightarrow \text{Module}_{k-1} \Rightarrow \text{Module}_k \\
\text{result}_k &= \text{Module}_k(\text{Module}_{k-1}(), \text{Module}_{k-2}())
\end{align*}
\]  

Eq. (2)

Eq. (3) shows an expression, in which one Web module is connected to plural modules.

\[
\begin{align*}
\text{Module}_k & \Rightarrow \{\text{Module}_{k+1} \} \\
\text{result}_{k+1} &= \text{Module}_{k+1}(\text{Module}_k()) \\
\text{result}_{k+2} &= \text{Module}_{k+2}(\text{Module}_k())
\end{align*}
\]  

Eq. (3)
Therefore, the slowest Web module becomes a bottleneck in a Web-Based PSE system using a sequential cooperation. In addition, we know that the module processing time is not constant all the time.

### 2.2. Asynchronous cooperation

Many distributed simple Web modules can cooperate also asynchronously. Behavior of one simple Web module is predictable, because a simple Web module executes only simple processing. However, each behavior of asynchronous cooperation of many Web modules becomes less simple. All modules behave independently. We define a Web-based agent as a Web module that behaves asynchronously and autonomously. The asynchronous cooperation in many distributed Web-based agents is expressed in Eq. (4). Each \( \tau \) as arbitrary time at the state of the Web-based agent is different values with each other in the equation.

\[
\text{Agent}_i(\tau) = \text{strategy}(\text{Agent}(\tau))
\]

\[
\text{Agent}(\tau) = \{ \text{Agent}_i(\tau) \mid \tau \in [t - \Delta t, t] \}
\]  

(4)

The state of one Web-based agent is decided by the agent's own rule depending on the state of other agents. However, a Web-based agent cannot obtain the information about all modules at the same time, and the states of other Web-based agents are always updated successively. In addition, by some kind of defects, the information from a certain Web-based agent may not be provided. Therefore, the slowest Web-based agent becomes the bottleneck in a Web-based PSE system using an asynchronous cooperation of many distributed simple Web modules.

![Figure 3. An asynchronous cooperation of multiple distributed Web-based agents constitutes a Web-based PSE. Each Web-based agent is linked with other Web-based agent(s). Each Web-based agent also cooperates autonomously with related other Web-based agents in asynchronous.](image)

Figure 3 is an example of Web-based PSE by WAPSE for a four-coloring problem and shows one of the solutions. A Web-based PSE using an asynchronous cooperation of Web-based agents mentioned above can solve a four-coloring problem. Each Web-based agent looks at the colors of adjacent Web-based agents and chooses their color. Each Web-based agent of A1-A6 has the same program and strategy. The Web-based agents perform asynchronous cooperation even if two or more Web-based agents are deployed to one computer if it has a multitasking OS like Linux, etc. The asynchronous cooperation in the distributed 6 Web-based independent agents is expressed in Eq. (5).
\[ \text{Agent}_{A1}(t) = \text{strategy}(\text{Agent}_{A2}(\tau), \text{Agent}_{A4}(\tau)) \]
\[ \text{Agent}_{A2}(t) = \text{strategy}(\text{Agent}_{A1}(\tau), \text{Agent}_{A6}(\tau)) \]
\[ \text{Agent}_{A3}(t) = \text{strategy}(\text{Agent}_{A4}(\tau)) \]
\[ \text{Agent}_{A4}(t) = \text{strategy}(\text{Agent}_{A1}(\tau), \text{Agent}_{A3}(\tau), \text{Agent}_{A5}(\tau), \text{Agent}_{A6}(\tau)) \]
\[ \text{Agent}_{A5}(t) = \text{strategy}(\text{Agent}_{A4}(\tau), \text{Agent}_{A6}(\tau)) \]
\[ \text{Agent}_{A6}(t) = \text{strategy}(\text{Agent}_{A2}(\tau), \text{Agent}_{A5}(\tau), \text{Agent}_{A4}(\tau)) \]

3. WAPSE (Web-based modules as Agents for PSE)

In nature, many simple creatures or cells cooperate asynchronously with each other to solve their problems autonomously. A Web module is a Web-based agent if each Web module behaves asynchronously autonomously in a group to solve a problem.

3.1. Web-based agent using Web module and XML

Each Web-based agent should be able to cooperate even if all Web-based agents behaved asynchronously autonomously. The Web-based agent in WAPSE comprises the following components.


- Properties: Each Web-based agent has the Web-based agent's own properties. A Web-based agent's own id is necessary to distinguish each Web-based agent. A Web-based agent's own data is depending on a problem. A Web-based agent knows information about the relationship to other Web-based agents. Each Web-based agent has the strategy for solutions to each problem.

Properties for a Web-based agent are described using XML in WAPSE. A Web-based agent's behavior is described by PHP scripts and AJAX scripts.

3.2. WAPSE as a Middleware for PSE

WAPSE is a middleware for PSE using cooperation of a simple Web-based agents' group distributed on heterogeneous Web servers and clients (see Fig. 4).

![Figure 4](image-url)

Figure 4. The architecture of WAPSE for heterogeneous Web servers and clients. WAPSE provides basic functions such as the communication for Web-based agents. Each Web-based agent has common scripts to behave depending on a problem. A Web-based PSE by WAPSE is constructed in a group of Web-based agents where the properties are described by a XML file for each Web-based agent.
Scientific problems would be solved by asynchronous cooperation among some distributed Web-based agents. However, it is difficult to build a Web-based agent in primitive Web environments. It must have communication among a large numbers of Web-based agents. It is also troublesome to deploy a large numbers of all Web-based agents to each computer at once. The debugging and change of the processing computers are common at a development stage. Therefore, we developed WAPSE so that developers can describe Web-based agents easily. We describe WAPSE by PHP scripts, that are the super class of Web-based agent. WAPSE provides the basic functions such as communication and the file lock. On the other hand, recently various mobile terminals are being used. WAPSE accommodates also mobile terminals as processing computers. The handling of calculation or visualization at mobile terminals is carried out using AJAX and HTML5.

3.3. Computer network constitution for WAPSE

Web-based PSE applications on WAPSE consist of XML files, PHP script and AJAX script for Web-based agents. These Web-based agents are deployed to each processing machine and solve a problem by asynchronous cooperation. A processing machine is a Web server computer or a terminal computer with a Web browser that is available to AJAX. There is a management server for management of deployment and the visualization. In addition, there is a sub management server for dynamic deployment to terminal computers (see Fig. 5).

3.4. Deployment and processing mechanism

WAPSE has the mechanism that deploys Web-based agents to distributed computers. Figure 6 shows two screen shot examples of the Web user interface for the management server system on WAPSE. A Web-based PSE developer can easily build the user interface for the management system using the Web technology such as AJAX or HTML5.

Figure 5. Conceptual image of multiple computers on WAPSE

Figure 6. Example of screen shots of the Web user interfaces for the management server system on WAPSE. The user chooses some servers for deployment of agents with a server list. When a user clicks the button of Deploy, results of deployment of agents are displayed.
First a developer of Web-based PSE by WAPSE installs a deployment script in the processing machine, which is a Web server. Then, a Web-based PSE developer executes the deployment of a Web-based agent at a management server from a Web browser or a command line. In addition, a Web-based agent is assigned automatically, when Web-based PSE users access a sub management server from each Web browser to use a Web-based PSE application. So, the Web-based PSE developers can easily perform the deployment of many Web-based agents by WAPSE (see Fig. 7). The Web-based agents deployed begin to work depending on the order from a Web-based PSE user.

**Figure 7.** Web-based agents are deployed to distributed heterogeneous processing computers and to Web-based agents’ pool on sub management computers. The Web-based agent behaves autonomously during its life time.

### 3.5. Web-based agents’ response speed

Each Web server to affect a Web-module has the different response speed by the specification of the Web server. A Web server’s processing response is unstable, because Web server programs are processing on TCP/IP [17, 18]. In addition, load averages of a Web server always change, because the Web server has many processes for various services. WAPSE is influenced by the response delay. We examined the response speed between Web-based agents on WAPSE with two server computers of the same specifications (see Table 1).

**Table 1. Measurement Computers’ Specification**

<table>
<thead>
<tr>
<th>Server ID</th>
<th>Specification Items</th>
<th>CPU</th>
<th>Memory</th>
<th>NIC</th>
<th>OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>X</td>
<td>AMD Athlon™ 64 3500+ 2.2GHz</td>
<td>8GB</td>
<td>Gigabit</td>
<td>Ubuntu Server 12.04.1 LTS</td>
<td></td>
</tr>
<tr>
<td>Y</td>
<td>AMD Athlon™ 64 3500+ 2.2GHz</td>
<td>8GB</td>
<td>Gigabit</td>
<td>Ubuntu Server 12.04.1 LTS</td>
<td></td>
</tr>
</tbody>
</table>
Figure 8. Computer network environment for demonstrations of WAPSE. These computers are connected to the LAN of Toyama National College of Technology.

Figure 8 shows the computer network environment that we tested. On this test, each Web server is examined in the same software environment such as the OS and the Web server application of the computer. A Web-based agent deployed to each server tries an acquisition of the other Web-based agent's data. A response speed is the time from the start to the end for this trial. These results in Fig. 9 do not include the influence by the DNS server, because we described an IP address directly in a Web-based agent for measurement. Most of reply speeds from each agent are around 0.06[s]. However, there is sometimes a big fluctuation, and agents' reply speed patterns are different. Therefore, the fluctuation and the delay affect the processing speed of the distributed system. This fluctuation is a feature of WAPSE.

Figure 9. Response speed (End time – Start time). The data of upper graphs are of the first two minutes of two Web-based agents. The lower graph is the zooming data from 40 to 50 seconds. The size of the text data, used for the measurement of the response speed, is 370 bytes.

Figure 10 shows the specification of Web-based agent is composed of an active part and a passive part. The Web-based agent behaves autonomously during its life time. The active part of a Web-based agent functions as an application on a processing machine. We used a Web user interface of WAPSE management server system for the deployment of agents and activation of agents.
3.6. Examples of Web-based agent on WAPSE

The first example is a sort problem, which shows that each Web-based agent can exchange correctly the data among the Web-based agents in cooperation asynchronously. We prepare a set of Web-based agents that have the element of each target data for the sorting in Eq. (6).

\[
\begin{align*}
\text{Agent} &= \{\text{agent}_i \mid 1 \leq i \leq n\} \\
\text{Data} &= \{d_i \mid 1 \leq i \leq n\} \\
\text{Set} &= \{\text{agent}_i.d \mid 1 \leq i \leq n, \text{agent}_i.d = d_i\}
\end{align*}
\]

Figure 11 shows an example, which describes properties of the Web-based agent 0010 in the XML file. In this case, the Web-based agent 0011 is next to the Web-based agent 0010. Parts of the properties for the Web-based agent 0011 are described in the XML file of Web-based agent 0010 as the known properties.

A Web-based agent knows only a Web-based agent, which is neighboring by a value of properties for the 'link' of XML. The total data are sorted correctly by each Web-based agent processing of the following algorithm in Eq. (7).

\[
\text{exchange}(\text{agent}_{\text{own}.d}, \text{agent}_{\text{next}.d}) \text{ if } \text{agent}_{\text{own}.d} < \text{agent}_{\text{next}.d}
\]
Figure 12 shows a specification example of a Web-based agent to implement an algorithm of Eq. (7) on WAPSE. Each agent cooperates autonomously with a neighboring agent in asynchronous. The value is exchanged in many pairs of agents in this specific example.

![Figure 12. Example of a specification diagram for Web-based sort agent on WAPSE.](image)

The second example is a four-coloring problem (see Fig. 13). The information about the relationship to other Web-based agents and a strategy of each Web-based agent are described in each XML file. Each Web-based agent updates the Web-based agent's own data decided by the Web-based agent's own strategy. The result may be different, even if the strategy of each Web-based agent is the same.

![Figure 13. Example of four-coloring problem with 29 areas and one of the solutions by WAPSE.](image)

Figure 14 shows a specification example of a Web-based agent implemented on WAPSE. Each agent looks at the color of an adjacent agent and chooses its own color.

![Figure 14. Example of a specification diagram for the four-coloring Web-based agent on WAPSE.](image)

The final example is the same four-coloring problem expanded to mobile terminals. A part of Web-based agent acts on terminal computer with Web browser, that is available to AJAX. However, JavaScript cannot read or write files on the client computers by client side programs. In addition, AJAX cannot communicate mutually and directly. A Web-based agent operates the Web-based agent's own XML file in the sub management server by cooperation programs of AJAX and PHP. Therefore, the specifications diagram for the four-coloring changes as Fig. 15.
We demonstrated WAPSE using terminal computers. There are several heterogeneous computers such as desktops, notebooks, ultrabooks, some Tablets, PDAs and smart phones (see Fig. 16). OS of these computers are Windows, iOS and Android. In this case, we used a notebook for the dynamic visualization of the transformation and used the other computers for handling of the four-coloring problem.

4. Conclusions

We developed WAPSE so that a developer can more easily describe Web-based agents. WAPSE is a PSE for a Web-based PSE, that is constructed based on asynchronous cooperation of Web-based agents using Web modules, distributed on one or more heterogeneous servers. It is difficult to build a Web-based agent in primitive Web environments. It must be communicated among a large numbers of agents. It is also troublesome to deploy a large numbers of Web-based agents to all the distributed computers at once. The debugging of Web-based agents and changes of the processing computers are common at a development stage. We can simply build PSE using cooperation of Web-based agents by WAPSE. The deployment mechanism of WAPSE is available in Web-based PSEs that operate according to workflow sequentially among distributed Web servers. Accordingly, WAPSE helps users to concentrate on problem solving. A distributed Web system is constructed in heterogeneous computers and network environments, that have performance fluctuations influencing the processing speed. Although Web-based PSE developers must improve the response delay of Web modules by the customization of the Web module for a high speed processing according to a static sequential workflow, the Web-based PSE is effective in a problem domain that many modules as agents cooperate autonomously in a group to solve problems. WAPSE may open a helpful problem solving environment in this new domain.
5. Acknowledgments

The authors would like to acknowledge PSE (Problem Solving Environment) Research group Japan for valuable discussions. The work was partly supported by JSPS and MEXT.

6. References