Development of Session Management Mechanism for Cellular Phone with WWW Connection

Satoru Uehara[†,‡], Osamu Mizuno[‡] and Tohru Kikuno[‡]
[†] NTT Data Corporation, Japan
[‡] Department of Informatics and Mathematical Science,
Graduate School of Engineering Science, Osaka University, Japan
{**s-uehara, o-mizuno, kikuno**}**@ics.es.osaka-u.ac.jp**

Abstract

The needs for accessibility of cellular phones to Web based client-server systems are increasing. For this access, efficient management of the context data must be implemented on the cellular phone. However, physical restrictions of the cellular phone make it difficult to manage context data by applying the conventional methods. Additionally, the conventional methods do not show sufficient performance to tolerate heavy accesses from cellular phones.

We therefore design a new session management mechanism that enables cellular phone to use Web client-server system efficiently. To do so, we decide to utilize key concepts in the CDS mechanism, which was previously developed by us in order to implement session management on the Web. By utilizing the CDS mechanism, we develop a new session management mechanism that connects cellular phone and Web systems. Then, we can actually implement the session management with high safety and reliability. The result of a benchmark evaluation shows that proposed mechanism has higher performance than the conventional mechanism.

1 Introduction

Recently, the Web is becoming the main base system on which client-server systems are developed. Many Web servers have been provided, and functions and performance have evolved.

On the other hand, mobile devices such as the cellular phones that can access the Internet have been developed (the Web cellular phone), and these mobile devices are becoming widespread explosively [3]. These mobile devices have Web access capability, and Web browser software is on them. The Web browsers on mobile devices can show hyper text including graphical images, and can send the user's input to a Web server. Because of widespread use of mobile devices with Web access, Web sites that these devices can use have also been increasing.

The development of the Web-based client-server systems has several problems that have not occurred in conventional client-server systems. One problem concerns data shared between user-sessions. The HTTP protocol is 'sessionless', so the connection between a client and a server is disconnected for each communication. When a Web browser accesses a Web server and then successively accesses the server again, the information of the previous access is not maintained. The data shared among user-sessions must therefore be maintained by another method. Hereafter, we call such shared data "context data" and we call the management of several user sessions by keeping the context data "session management".

Several widely used methods for session management have been proposed [2] and developed. However, for Web browsers on workstations and mobile devices, these methods are not sufficient from the viewpoints of security, reliability, and performance.

In order to realize the session management for the Web cellular phones, we will propose a new mobile context data storing (Mobile CDS) mechanism. We design the Mobile CDS mechanism to realize session management with high reliability, high security, and high performance, and it is free of the physical restrictions of the Web cellular phones.

2 Requirements for Web Cellular Phones

We analyze the requirements of session management for Web cellular phones' Web browsers. Web cellular phones are becoming widespread explosively, and an effective session management mechanism is necessary.

The requirements of session management for mobile devices are summarized as follows:

1) Independence from specifications

The specifications of the Web cellular phones are different and depend on the service provider. Web sites should support such protocol specifications, content languages, and so on [3]. Session management for Web cellular phones should be applicable to the various specifications.

2) Unrestricted size of context data

Web sites that can be used by the Web cellular phones are increasing. If the system has the same functions as conventional Web systems, the amount of context data will increase. So it is desirable that there is not restriction on the size of the context data.

3) Invisibility

Since the context data may include data that should be kept secret from users, the context data must not be visible to users. Thus the session management mechanism for Web cellular phones must have invisibility.

4) High reliability

If a trouble occurs at the Web server and context data is lost, the Web program must abort. However, if the context data is saved, it is possible to recover the Web program. So the session management mechanism for Web cellular phones must have high reliability.

5) High performance

When Web sites are built for consumers, high access performance is needed to meet the widespread use of mobile devices, especially Web cellular phones. So the session management mechanism for Web cellular phones must have high performance.

3 Mobile CDS Mechanism

In this section, we propose the new context data storing mechanism for the Web cellular phones (hereafter, the Mobile CDS).

3.1 CDS mechanism

We have already proposed the context storing mechanism for the workstations (hereafter, the CDS mechanism) [4, 5]. The CDS mechanism realizes document-view architecture on the Web system. And it has advantages of high reliability and high security.

In order to execute the CDS mechanism, the invocation command of the CDS component must be described in a HTML document. Then the CDS component is loaded into the Web browser's process. At the same time, the CDS server is also invoked in a different process (from the Web browser's process) and it starts managing the context data.

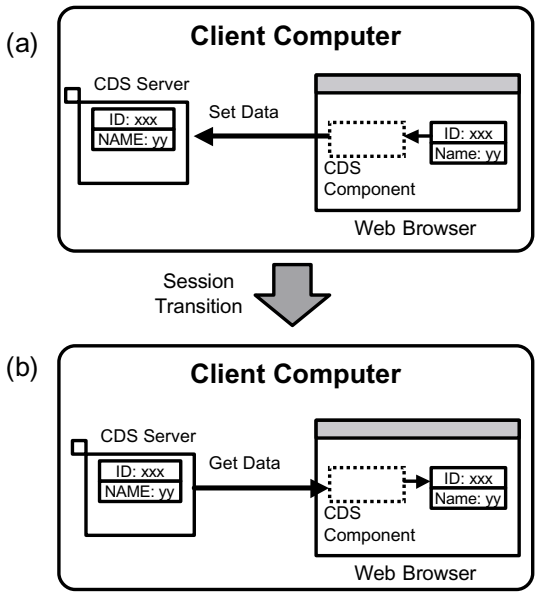

Figure 1. Basic behavior of CDS

The following shows an example to maintain a user's ID and a user's name during successive sessions: First, assume that user inputs ID and name. Then, the CDS component and the CDS server are invoked at the same time. The CDS component calls the method to create the keys, such as "ID" and "NAME", and to set the value for each key (for example, ID=xxx and Name=yyy as shown in Figure 1(a)). Next, assume that the current session transits to the next session (as shown in Figure 1(b)). Just after the transition, the context data such as "ID" and "Name" are not maintained on the Web browser. Then, the CDS component calls the method of the CDS server to get these context data.

3.2 Structure of Mobile CDS

The Mobile CDS is a software component that uses the CDS and runs on the Web server computer. The Mobile CDS is used by Web programs that also run on the Web server computer.

Figure 2 shows the state of the Web server computer when the Web server and the Mobile CDS are running in main memory. The Mobile CDS maintains plural context data and receives method calls from Web programs. Methods calls are performed using inter-objects communication. The Mobile CDS is implemented as an executable form (EXE) that runs independently from the Web server process. Even if the Web server process fails, the context data is not lost. This makes it possible to recover the previous session.

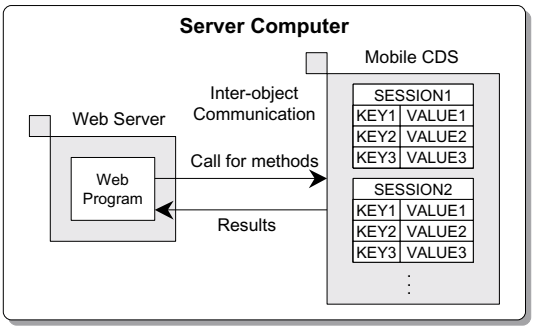

Figure 2. Structure of Mobile CDS

3.3 Basic behavior of Mobile CDS

Figure 3 shows the basic behavior of the Mobile CDS. In the figure, a Web cellular phones are using a Web program, and the unique session ID "0001" has been assigned to the session in advance.

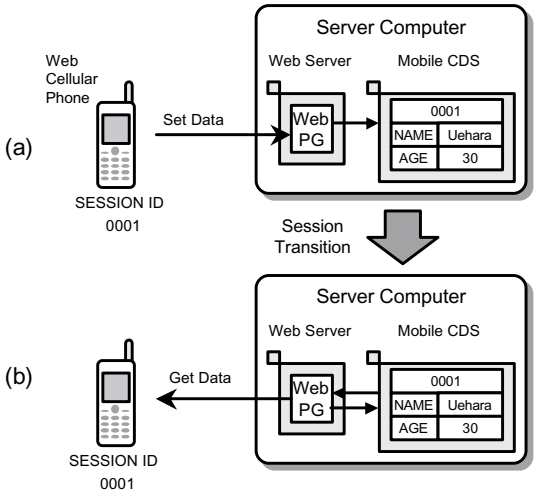

Figure 3. Basic behavior of Mobile CDS

In the example shown in Figure 3 a user's name and age are maintained during successive sessions. Session management is performed as follows:

1) First, the user inputs their name and age in the form, and submits the form.

2) The Web program invokes a method of the Mobile CDS, and allocates space for context data with the session ID "0001". The Mobile CDS creates the keys, "NAME" and "AGE", and stores their values.

3) The user moves to the next page. Just after the transition, the context data such as "NAME" and "AGE" are not maintained in the Web server. The Web program therefore calls an Mobile CDS method to get the context data for the session ID "0001".

4) Finally, the context data is shown to the user.

3.4 Implementation of Mobile CDS

We implemented the Mobile CDS using Microsoft Visual C++. The Mobile CDS can be used in server-side programs. The following 8 major functions are manipulated in the server-side programs: 1) `PrepareContextDataSet` method, 2) `Remove` method, 3) `RemoveAll` method, 4) `SetData` method, 5) `GetData` method, 6) `GetCount` method, 7) `GetName` method, 8) `SetTimeout` method. Context data may be newly created or removed by applying methods 1), 2) and 3). The content of the context data can be set or obtained by the methods 4), 5), 6), and 7). Finally, the valid term of the context data is set by the method 8).

4 Application to the Real System

4.1 Performance comparison

We implemented the Mobile CDS and have applied it to a real system. The target system is based on the Web, and receives a reservation for use of public facilities from the Web cellular phones.

We evaluate the performance of Mobile CDS mechanism. We performed benchmark tests using the Microsoft Web Capacity Analysis Tool (WCAT) that is software for a benchmark test of Web system. This software can simulate the Web accesses of any number of Web clients by running multiple threads. The response from the Web server is recorded in real time, so the average response time etc. may be found.

We compared the average response times of the Mobile CDS method with the database method. (Note that we do not compare with cookie methods since most of cellular phones do not have any cookie mechanisms.) First, we implemented session management using the Mobile CDS and then using a database. We used Microsoft SQL Server 7.0 as a database.

Next, we measured the performance of each Web server quantitatively, while maintaining multiple accesses to the Web servers. The Web programs accessed 20 context data items per context data set access, because we considered this real use. The accesses consisted of references to the context data and updates to the data, in the ratio of 8:2. We varied the number of access at same time, selecting 25, 50, 75, or 100.

Table 1 and Table 2 show the results of the benchmark tests. Table 1 indicates the average response time (in seconds) for each case, and Table 2 indicates the total number of connections that were processed in an hour. From these

Table 1. Result of benchmark tests(average response time in seconds)

	Access at same time(times)			
	25	50	75	100
Mobile CDS	0.64	1.40	2.14	2.68
Database	18.25	36.68	54.85	73.57

Table 2. Result of benchmark tests(total number of connections in an hour)

	Access at same time(times)			
	25	50	75	100
Mobile CDS	468,912	427,056	419,988	446,556
Database	16,104	15,744	15,528	15,072

results, the system with session management using the Mobile CDS showed dramatically higher performance than the database system. Thus, we consider the Mobile CDS mechanism's high performance verified.

4.2 Evaluation of other requirements

In this subsection, we evaluate the advantages of the Mobile CDS system and compare them with the other requirements explained in Section 2.

1) Independence from specifications

   The Mobile CDS only requires Web cellular phones to support URL encoding method. URL encoding is supported by almost all mobile devices so the Mobile CDS mechanism can be applied to mobile devices of various specifications.

2) Unrestricted size of context data

   The Mobile CDS maintains the context data in the main memory of the server computer. Thus the size of context data doesn't depend on the physical restrictions of Web cellular phones. We can say there is no restriction on the size of data in the Mobile CDS.

3) Invisibility

   The context data is stored in the Mobile CDS which runs in the main memory of the server computer. We performed tests and checked that it is quite difficult to access the context data. So it can be said that the Mobile CDS mechanism has the advantage of invisibility.

4) High reliability

   We performed reliability tests. We stopped the Web server serving during the operation of a Web system that was using the Mobile CDS mechanism. However, the Mobile CDS mechanism was unaffected by the problems of the Web server, and continued to maintain the context data. We were able to recover the state before the Web server was stopped, after the Web server resumed serving.

According to the discussion above, we can say that the Mobile CDS satisfies the requirements of session management for the application of mobile device's Web browsers.

5 Conclusion

In this paper, we designed a new mechanism for Web cellular phones to maintain the context data in Web systems and implemented it. The proposed system showed high performance and reliability without restriction of Web cellular phones. We also applied the proposed mechanism to the actual system, and verified the advantage of proposed mechanism by benchmark tests. Future research includes the implementation of the Mobile CDS mechanism on the platform independent from Microsoft's one.

References

[1] D. Kristol and L. Montuli, "HTTP State Management Mechanism," RFC 2109, Network Working Group, 1997.

[2] Netscape Inc., HTTP Cookies, *http://home.netscape.com/newsref/std/cookie_spec.html*

[3] J. Krikke, "Graphics Applications over the Wireless Web: Japan Sets the Pace," *IEEE Computer Graphics and Applications*, vol.21, no.3, pp.9–15, 2001.

[4] S. Uehara, O. Mizuno, and T. Kikuno, "An implementation of electronic shopping cart on the Web system using component-object technology," *Proc. of 6th Workshop on Object-oriented Real-Time Dependable Systems (WORDS2001)*, pp.85–92, 2001.

[5] S. Uehara, O. Mizuno, and T. Kikuno, "Design of New Mechanism for Context Data Storing on Web System and Its Implementation using Component-Object Technology," *Trans. of IEICE* (In Japanese), vol.J84-D-I, no.6, pp.713–722, 2001.

[6] S. Uehara, "Design and evaluation of new session management method for WWW application systems," Ph.D. dissertation, Osaka University, 2001.