Service adaptation method using hierarchical service state for seamless service environment

Norihiro KAWASAKI, Hiroyuki KASAI, Yoko KIKUTA, and Kenichi YAMAZAKI
Network Laboratories, NTT DoCoMo, Inc.

1. Introduction

In the ubiquitous computing environment, there will be an enormous number of terminals around us, and all of them will be connected to networks. In this situation, services should be provided seamlessly. That is to say, any service should be transferable from a terminal to another terminal without stopping the service. We call this the seamless service environment (SSE). A typical example of seamless service is shown in figure 1. In this example, she is enjoying a streaming service on her desktop PC (the old terminal). If she goes out, the streaming service can be transferred to her cellular phone (the new terminal) and she can continue enjoying it in SSE.

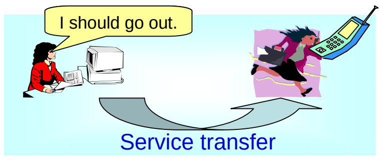

Figure 1: An example of seamless service.

We have already proposed the "service transfer technology," a core part of SSE [1]. It enables transfer of a service between different terminals and basically follows the following sequences; 1) SSE server gets the current "service state," a set of information about the service to be transferred, from the old terminal, 2) modifies the service state in a certain manner, and 3) sends the modified service state to the new terminal. As a result of the service transfer, the new terminal receiving the modified service state can start a new session and hence the service is transferred from the old terminal to the new one. Figure 2 shows the general architecture of SSE. In this architecture, the service transfer technology is implemented in the SSE server. The main concept of this architecture is that the SSE server knows all information such as terminal profiles, and user profiles, and signals sequence for service transfer, so that the SSE clients can be thin.

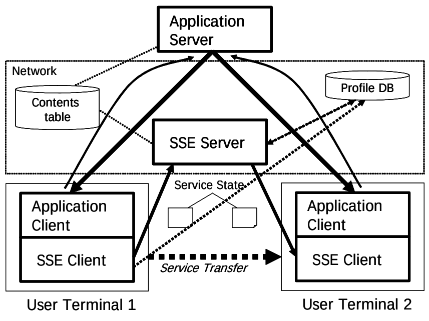

Figure 2: Service transfer scheme.

The requirements for service transfer are given as follows:

Req.1: Continue service in its entirety

Req.2: Continue service in different environments

Req.1 involves detailed information about the service on the previous terminal (Guarantee of service identity). Req.2 ensures support in different environments (Realization of service adaptation).

We have already proposed the service transfer method of the service state description. The service state consists of application service information, layout information, and environment information [2]. In figure 1, Application service information includes the contents URL, elapsed time of the contents, and the name and version of the used application. Layout information includes the size and position of windows in desktop environment. Environment information includes a terminal specification, and network I/F and bandwidth. Thus, elements and representation of the service state were mainly considered [2]. However a concrete method for service adaptation was not considered sufficiently for Req.2.

Service adaptation is realized by changing each element of a service. In figure 1, if a service is transferred to the new terminal that does not have the application used in the old terminal, an application and contents of the new terminal should be changed. As a result, the service is adapted in order to fit for the new terminal. In this paper, we propose a service adaptation method using the service state in order to meet Req.2.

2. Service adaptation method using hierarchical service state

Our basic idea for Req.2 is that by designing the service state structure deliberately, "changing a service state" problem becomes the matter of sub-structure replacement. In this section, we describe how we design the service state and how the service state structure is converted.

2.1 Hierarchical service state

In order to convert a service state simply, we adopt the tree structure for the service state, as shown in figure 3. The tree structure enables a set of elements which have the same attribute to be represented together. It enables the conversion of a service state to be regarded as the sub-tree replacement of a service state, as shown in figure 4. Here, we call a label assigned from root to leaf "level."

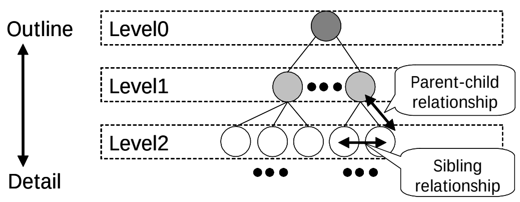

Figure 3: Hierarchical service state.

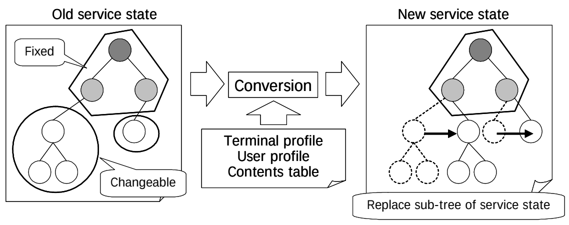

Figure 4: Conversion of service state.

2.2 Conversion of service state

In this section, we explain the conversion of the hierarchical service state. The conversion consists of three steps as follows.

[Step1] Decision of service state candidates

First, a system or user decides which element of the service state is fixed or which element is changeable. All of the service state candidates, which are replaced with candidates for changeable elements, are decided using terminal profile, use profile, and contents table.

[Step2] Evaluation of service state candidates

All of the service state candidates decided in step 1 are evaluated, in order to search the service state suitable for the new environment. The most suitable service state among all service state candidates is selected.

[Step3] Conversion of service state

The old service state is converted into the service state decided in step 2.

3. Implementation

3.1 Hierarchical service state

Table1: Hierarchy of service state elements.

Level0	***Level1***	***Level2***	***Level3***
State	Service Continuity	Application Service	Content
			Application
			Option
		Layout	Window
	Service Adaptation	Environment	CPU usage
			Memory usage
			Network

Level3	***Level4***	***Level5***	***Level6***	***Level7***
Content	Genre	Title	Reference	Pointer
				Format
			Value	
Application	Name	Version		

Table 1 shows the hierarchy of service state elements.

Service state descriptions are written in XML. This language can represent a hierarchical structure in a straightforward manner [2]. Figure 5 shows an example of the service state description. The indentation depth of the description is based on the level of tree structure.

```
<state>
<applicationService type="Streaming">
  <content>
     <genre>drama
        <title>Movie
           <reference>http://***.net/movie.ram
              <pointer>10:12</pointer>
              <format>real media</format>
           </reference>
        <title>
     </genre>
     …
  </content>
  <application>
     <name>
     Real Player<version>7</version>
     </name>
  </application>
  <option>
    …
  </option>
</applicationService>
…
</state>
```

Figure 5: Service state description.

3.2 Conversion of service state

Figure 6 shows an example of service adaptation of an application and contents. Figure 4 shows the conversion of a service state. In this example, the old service state is a state of watching a contents "Movie" via application "RealPlayer®" in a Desktop PC. The service state is converted in order to suit the new terminal PDA environment.

[Step 1] Decision of service state candidates

First, fixed elements and changeable elements are decided by a user or a system. Here, elements from root level to content title "Movie" are fixed, but a content reference and content elapsed time are changeable. An application "RealPlayer®" is also changeable.

[Step 2] Evaluation of service state candidates

Service state candidates are evaluated by terminal profile, user profile, contents table, and the service state. Here, an available application in the new terminal PDA is "Microsoft® Windows Media™ Player," the display size is "3.5inch," and the display resolution is "240x320."

[Step 3] Conversion of service state description

The service state description is converted using the standardized technologies such as XSLT/DOM. Here, contents are alerted according to the bandwidth, the display size, and the display resolution. An application "Microsoft® Windows Media™ Player" is selected in the new terminal.

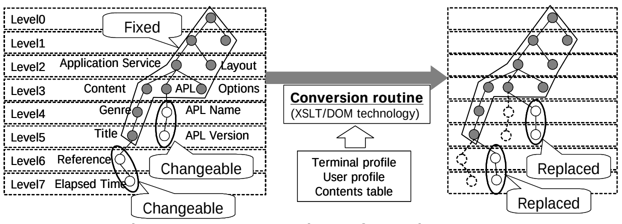

Figure 6: Conversion of service state
(e.g. selection of application and content).

4. Conclusions

We have proposed the service adaptation method using the hierarchical service state. In our proposed method, the service state can be represented in detail (Guarantee of service identity), and can be changed in order to suit the new environment (Realization of service adaptation).

Reference

[1]Y. Kikuta, et al., "System architecture of the seamless service environment platform," IEICE General Conference, Mar., 2003.
[2]N. Kawasaki, et al., "Service state description for seamless service environment," FIT2002, M-43.