Migratory Adaptive User Interfaces

Takuto Yanagida and Hidetoshi Nonaka
Graduate School of Information Science and Technology
Hokkaido University, Sapporo, 060-0814, Japan
Tel: +81-11-706-6861, Fax: +81-11-706-6862
E-mail: {takty, nonaka}@main.ist.hokudai.ac.jp

***Abstract*—We propose a new solution named interface client/logic server (ICLS), targeting dialog-based interactive services, supporting user interface (UI) migration, and offering adaptive UIs for devices and services. Constant improvements of technology have brought a large variety of platforms, and that has made users' new demands about the services. The first is that the users would like to use services through different devices and modalities depending on their use contexts. The second is that the users would sometimes like to change devices and take their tasks from one to another, which is called UI migration. Our architecture ICLS is designed based on client/server model. In ICLS, we use XML documents written in abstract interaction description language (AIDL) as logical descriptions of UIs, and introduce one of the semantic web technologies adding the function of expressing meanings of interactions.**

I. Introduction

Constant improvements of technology have brought a large variety of platforms (such as mobile phones, PDAs, and music players including desktop PCs) used for interactive services, and that has made users' new demands about the services. The first is that the users would like to use services through different devices and modalities depending on their use contexts. The second is that the users would sometimes like to change devices and take their tasks from one to another, which is called user interface (UI) migration [1]. However, conventional ways of associating devices and services doesn't meet the users' demands because of costs and inadequate separation between services and devices.

We propose a new solution named *interface client/logic server* (ICLS), targeting dialog-based interactive services, supporting UI migration, and offering adaptive UIs for devices and services. Our architecture aims at services like web applications, in which some input facilities are used on some dialogs or pages updated as state transitions. ICLS allows service developers to declare concrete semantics of interactions on services in logical descriptions with a machine-readable way, and realizes the richness of generated UIs on devices. ICLS has a mechanism of attaching devices to existing session for UI migration, and when the users do not disconnect the previous device after attaching new device, they can utilize the two devices simultaneously.

II. Architecture

Our architecture ICLS is designed based on client/server model. The term *interface clients* stands for various devices and platforms in which client applications are implemented, and the term *logic servers* stands for various services in which server applications are implemented. Once devices (clients) or services (servers) are developed in accordance with the specification of ICLS, no revisions are required when new service or device is introduced.

Figure 1 shows an example of the flow of a session between a client and a server for a *reminder service*, which manages items on users' schedules. A session of a service starts by a request from a client to a server (1). After that, the client receives a logical description from the server (2). The client constructs DOM tree, and generate a UI based on the tree (3). Logical descriptions are written in the language, *abstract interaction description language* (AIDL), which is an XML application we have developed. After the UI generated on the client, the user can check his schedule item list with the device like a portable music player with a small screen and a wheel control (4). The user notices that he has to enter a new item into the list, and he operates the small wheel control on the player to do it. The client changes the DOM tree corresponding to the user's UI operations (5). The client sends this DOM changes as messages to the server (6). The server performs the service according to the message from the client (7).

During a user is accessing a service through one client with a session, the user can also try to access the same service through another client with the existing session. Here, the DOM trees of these clients are synchronized and updated simultaneously, and then, a migration is performed when one of the clients except for the first one is disconnected. When a client establishes a connection in a session, the server issues a unique session ID to the client. Using this ID, the user can access the same session. Figure 2 shows an example of the flow of a migration process between two clients: client 1, and client 2. A new client (client 2) obtains a session ID from

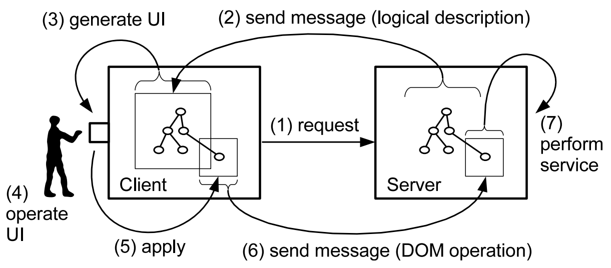

Fig. 1. Flow of communication process between a interface client and a logic server.

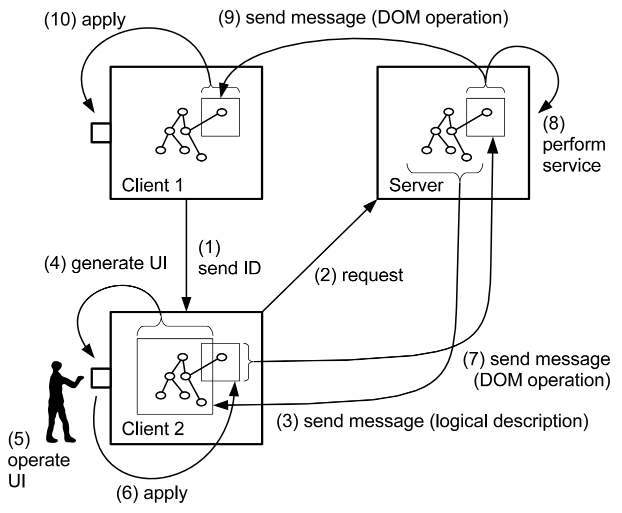

Fig. 2. Flow of migration process from the client 1 to the client 2.

the existing client (client 1) (1). A simultaneous session of a service starts by a request with an ID from a client to a server (2). After that, the client receives a logical description used in the session from the server (3). The new client executes the same UI generation process and sends a user's operation to the server (4, 5, 6, 7, 8). After performing the service, the server broadcasts the received message to another client (9). The client 1 receives the message, and applies it to its DOM tree and its UI (10). In the example, the user can access the server with the same session through a PC, and continue to type in the content of the item, without restart of the service or reentering the time data.

III. Logical description language

In ICLS, we use XML documents written in AIDL as logical descriptions of UIs, and the documents describe interaction structures, presentations, and task models in services. AIDL is an application of XML, and supports the clear separation of interface clients and logic servers on ICLS, because it is based on web standards XML and has no device- or modality-specific contents. In AIDL, arbitrary UI structures and their current state (meaning the history of users' operations) are described as *selection acts*, which represent essential function of UI elements in common among devices. Selection acts consist of three elements: a *type* (a set of choices), a *meaning* (a purpose in a service), and a *state* (a current state). Selection acts are grouped with other selection acts and groups, and compose an *interaction tree*. A description of AIDL can be seen as a tree where selection acts, choices, current states, and groups are expressed as XML nodes.

We introduce one of semantic web technologies, resource description framework (RDF) for adding a function of expressing meanings of interactions on certain services in logical descriptions. In ICLS, RDF classes are used for the meaning expressions, and the hierarchies of these classes are used for the inference of meanings. Interface clients can infer meanings based on the hierarchy of RDF classes applied to the general-specific relationship of meanings, and they can address more meanings than ones actually implemented. Meanings in AIDL descriptions are exploited to specify the purposes of selection acts and to relate them to client-specific UI elements.

IV. Implementations

We implemented ICLS as a framework, which is a class library written in Java language (JDK 1.6) with a semantic web library Jena [2]. In order to verify the feasibility of the ICLS specification and the stability of the communication protocol, we developed three interface clients and two logic servers. Three clients are a GUI client, a simulator of mobile device, and a simulator of voice UI. Two servers are a reminder service and a remote controller of a virtual appliance.

V. Related work

As a general solution, a broad range of research was proposed, and almost all of them employ the approach of model-based UI design which commonly utilize logical descriptions.

Web migratory interface system was proposed in [3], which targets arbitrary web applications and performs a reverse engineering of existing web pages in order to obtain their logical information. This approach has a benefit for specific web applications, but it does not suit our purpose.

Ubiquitous interactor (UBI) [4] addresses service specific domains with *customization forms* in logical descriptions for developing services without dependence on devices, but it does not consider migration. Since the customization forms have no portability among different devices, developers have to customize their descriptions for each device.

Personal universal controller (PUC) [5], [6] was proposed for remote controlling various appliances with only PDAs, but there is no consideration of migration there. PUC uses smart templates for generating conventional presentations on some service domains, but the difficulty of defining the smart templates is not mentioned.

VI. Conclusion and future work

We presented a new solution, interface client/logic server architecture, which has some limitations but offers simultaneous interfaces, and we showed the new application of outcome on the other research field. As our future work, we have to remove some limitations imposed on our solution, to consider exploiting the outcomes of other work, and to evaluation it.

References

[1] R. Bandelloni and F. Paternò, "Flexible interface migration," in *IUI 04*, 2004, pp. 148–155.
[2] Hewlett-Packard Development Company, L.P., "Jena," available at `http://jena.sourceforge.net/`.
[3] R. Bandelloni, G. Mori, and F. Paternò, "Dynamic generation of migratory interfaces," in *Proceedings Mobile HCI 2005*, 2005.
[4] S. Nylander, M. Bylund, and A. Waern, "The ubiquitous interactor–device independent access to mobile services," in *CADUI'2004*, 2004, pp. 274–287.
[5] J. Nichols, B. A. Myers, M. Higgins, J. Hughes, T. K. Harris, R. Rosenfeld, and M. Pignol, "Generating remote control interfaces for complex appliances," in *UIST 2002*, 2002, pp. 161–170.
[6] J. Nichols, B. A. Myers, and K. Litwack, "Improving automatic interface generation with smart templates," in *IUI 04*, 2004, pp. 286–288.