Customized Interface Generation Model Based on Knowledge and Template for Web Service

Rui Zhou, Jinghan Wang, Guowei Wang, and Jing Li
School of Computer Science and Technology, University of Science and Technology of China, Hefei 230026, China
Email: {rayzhou, heiswjh, weiking}@mail.ustc.edu.cn, lj@ustc.edu.cn

Abstract—With the development of Service-Oriented Architecture, more and more researches have provided automatic and semi-automatic approaches to end-user. Users can construct their own applications with web services. However, it is hard for most end-users to customize the interfaces of applications with current service composition methods. To address this issue, an interface generation model was proposed to provide customized user interface and interaction workflow. In this model, knowledge was involved to instruct the workflow of interaction. Templates were adopted to describe the user interface. Some significant points, such as user definition, data profile, user interaction workflow, interface description, were discussed in detail. A prototype system was implemented. Some demos have been shown to verify the customized interface generation model. With this model, end-users can define the interfaces and interaction workflows of web services with rules and templates. It supplies the gap of user interface in service composition. Compared with the current interface generation in service composition, the proposed model is more flexible and more effective for end-users.

Index Terms—Interface Generation; Service Composition; Knowledge-Based; End-User Development

I. INTRODUCTION

In recent years, with the development of Service-Oriented Architecture (SOA)[1], more and more researches have proposed automatic or semi-automatic approaches to end users to construct their own applications [2] [3] [4]. In these researches, the end-users are able to take the advantage of the SOA to design a variety of applications. However, unlike the enterprise SOA, the most of end-user defined applications with automatic service composition methods lack suited user interfaces and interaction workflows. In applications for end-user, user interactions are important [5]. Without interaction, the information from different services cannot be displayed for users in appropriate way. It will cause that the service composition systems become unacceptable for users. Hitherto, there are approaches and systems that can be chosen to make service composition by end-user, but few of them can provide customizable user interaction model. In resolve this gap, the user interface and interaction workflow in web services composition is explored.

The models of user interaction for services composition have been extensively studied. Some researchers tried to generate interfaces from web service descriptions. For example, an annotation tool based on WSDL was involved to enhance the user interface generation process for services [6]. The developers or users should add annotations to define UI features. A match algorithm based on descriptions has been proposed to support the discovery of UI components and the provision of suggestions that can help to develop user interfaces for service-based applications [7]. And the match relies on descriptions about functions and components of UI. Those suggestions can be used to generate an interface. However, this approach still needs manual adjustments in most time. An adaptive user interface generation framework for web services was proposed [8]. The framework involved the WSDL and user’s profile to generate a suitable interface. The developers of web services were required to add an extra description.

Some researches were focused on UI composition in application composition. An implemented approach was proposed to reuse user interfaces while composing services [9]. This approach relied on abstracting these applications to be composed and these methods of web services. Then, it achieved a composition at the abstract level to regenerate a concrete user interface in a target language. An application composition driven by UI composition was proposed [10]. It was based on existing application’s UIs and their semantic descriptions. Generally, these methods were only suitable to those services with their own user interfaces.

In other service composition systems, there are researches on user interaction model. For example, in [11], the service-oriented user interface modeling and composition has discussed. This model suits a system of interface composition. Another interface composition approach in presentation layer has been shown in [12]. In [13], a method which includes interface define is provided for end user by mash up in web page. A tool for composing user interface in mobile for end user has been introduced in [14]. An automatic web-based user interface model is discussed for SOA-based system in [15]. However, in the systems of knowledge based service composition, user defined interaction and interface method is not sufficient.

In our previous research, a knowledge-based development approach for end-user in cloud computing, called Cloud Brain, have been proposed in [16]. Its main idea is that the user’s knowledge can be stored in cloud in
the form of rules. With the innumerable computing and storage resources in cloud, these rules can accumulate and reason, which can help the users to think and provide functions for them. As a user-oriented system, it is significant to find a perfect way to provide information and actions from system for user in a suitable manner. In the Cloud Brain system, the information and actions are produced by web services. These services are called with rules. Therefore, they cannot send the information and actions to users directly. Then, the client program in user's device hardly faces to various data from different web services. Furthermore, users have some obstacles to use these actions and information for further operations.

In this paper, an end-user customized user interface generation model is proposed in the Cloud Brain system. The UI template is involved to describe the interface. With this model, some key points have been considered, and a prototype system has been developed. End-users can make use of this interface generation model to define customized interface and interaction workflow in the procedure of service composition. Compared with the current interface generation in service composition, the proposed model is more flexible and more effective for end-users.

The paper is organized as follow: The user interface generation model and some key points are discussed in Section 2 in detail. Then the development of prototype system and demo are demonstrated in Section 3 and Section 4. In final, we conclude the approach and expect the future work.

II. USER INTERFACE GENERATION MODEL

A. Knowledge-Based Interface Generation

The user interaction model in this paper is combined with the knowledge-based service composition method in Cloud Brain [16]. In the service composition model, users select services and define their execution process by defining the rules. These rules are the representation of users' knowledge. Every rule includes two parts. One is Left Hand Side (LHS), which describes the conditions of the rule. The other part is Right Hand Side (RHS), which describe the actions. These two sections make up the “IF-THEN” statement. If the condition in LHS is matched by some facts which indicate the user's context, the rule will be fired, and the actions in RHS will be executed. For services composition, the output of services can be matched with the conditions as a fact. The matched fact will be send to these services as parameter. In this way, rules connect different web services to compose applications for users. These applications are not integrated software running on client device or server. So, there is a problem that how the users can interact with the applications. Therefore, a user interface generation model was proposed to solve the problem. The model of interaction can blend in the Cloud Brain user development system well.

As shown in Fig. 1, the user interface generation model has two stages. The first stage is user interaction definition, in which users define the interaction process. The other stage is user interaction action, in which users interact with the system and web services.

The following components will be concerned in the user interface generation model of Cloud Brain: end-device, knowledge editor, UI template editor, UI service, template repository, fact base, rule engine and some web services. In Cloud Brain, these components exist as different roles. The end-devices are the user access devices, such as mobile phone, notebook and tablet PC. It collects user's input and operation. The knowledge editor is a tool for end-user to edit their own rules. In the UI template editor, end-user can define the user interface with an easy way. The UI service is a main component to response user interaction specially. The template repository stores the UI templates. Fact base manages the facts from different devices and sources. The rule engine is the core component of Cloud Brain. It maintains the rules, matches the facts, and executes the actions. Web services existing in internet provide all kinds of functions for user. These components will be described in detail in later sections.

In the proposed user interface generation model, the users need to define their interactions before the interaction executing. So, there are two stages in the model as follow.

The first stage is user interface definition stage. In this stage, user can define the interaction workflow by rules. In these rules, the facts, which will be displayed to user, are referred. At this time, users can choose a UI template to display these facts. The template comes from template repository or template editor. User can design the display form of the facts and events that can be triggered in the interaction.

The other stage is user interaction execution stage. The process of this stage is follow:

1. A fact is produced by web service. The fact will match the conditions of rules in the rule engine, if the facts need to output to user.
2. When some rules’ conditions match the fact successfully, the right hand side of the interaction rule will be executed. The UI service will be invoked. The template referred in the rule and the matched fact will be used as parameters of the service.
3. The UI service will merge the fact and template together into an interaction interface. The interface will be pushed and displayed in user's device.

4. User can operate in the interface. It will produce some operation events. These events will be inserted into fact set in the form of facts. So, they can take part in later procedures of reasoning and interaction.

**B. Interaction Rule**

In this interface generation model, the interaction rule is a form of regular rule in the Cloud brain system. The structure of interaction rule is same as other rules. Therefore, they can take the reasoning with each other. The interaction rule also has the left hand side (LHS) and the right hand side (RHS). Conditions in the LHS will be mainly used to check the facts which need to be displayed or interacted. These facts come from the output of web services usually. For example, the weather fact is from the weather web service. In the RHS of interaction rule, the UI service is called with the parameters. The parameters are the UI template ID and the matched facts normally.

**C. User Interface Description**

In order that users can define their interfaces, an UI template is involved into this model to describe the user interface.

In the UI template, the framework of interface, the form of information display, and the operation event are defined. The UI template is described in XML. It includes two parts: display and operation. The high level tags `<show>` and `<operation>` respectively present these two parts. In the display part, the information from web service is presented in the form of HTML. In the HTML, there are some special tags to indicate the location of the facts contents. These tags are in exceptional type of HTML, in order to distinguish it from the tags in HTML. In the `<operation>` section, the candidate operations are described by the components `<event>`. These events can be triggered by users' operation in the interface. In every `<event>` tag, type attribute and the content of event are included. The type attribute indicates the type of the event, such as button click, form submitting, and text input.

**D. User Interface Service**

In order to display these interfaces to users, a user interface service is provided in our system. This service is designed to generate interfaces and send them to users. It is implemented in our system, but it can be involved as a web service. Users can define interaction rules to invoke the UI service. There are two input parameters in the invocation. They are fact and template. The fact can be transmitted by the rule from the fact database, when the fact matches the rule. The template will be selected from UI template repository.

In the UI service, a web servlet was designed to generate HTML web page to users. When the UI service is called, it will insert fact data into the HTML segment in the template. The special tags in this segment can indicate this operation. In addition, the events in the template will be connected this fact, in order to make the events correlation with fact. These segments in the template will be converted to HTML document. The web servlet can send the web page to user's browser.

**E. Data Profile**

For the user interaction, some input data and output data are maintained in the system. In order to integrate them into the cloud brain system, the interaction data can be consider as some kinds of facts, which is the basic data represent form in the cloud brain model. As mentioned in the cloud brain, the facts are managed by ontology. In the facts ontology, some types named “input” and “output” exists to indicate these user interaction facts. These facts mainly consist of user request event and users UI display facts, and there are many kinds of templates stored in template repository. An editor tool was designed to users to define new template. When users define their interfaces, they can choose a template from the repository or edit a new template by the editor. The templates can be involved in the interaction rules with their ID.

The Fig. 2 is an empty user interface template XML file. It shows the structure of template description.

**F. The Lifetime of the UI Data**

The lifetime is the remaining time of data in the system. In this user interaction model, the lifetime of the data is the period when the input and output data stay in the working memory of rule engine. If some data still exist in the system after the correlative rules are triggered, in some situations, they will make some incorrect operations by fire new rules. On the contrary, if some data are retracted from memory before being invoked completely, the functions and the data of the interaction will be damaged. For the input and output facts, the impacts of their lifetime to interaction procedure are different. If the input event facts persist in the system after their function complete, they will match some rules again incorrectly when some new relevant facts or rules are inserted into
the system. It may cause reduplicate response to user's one operation. On the contrary, if the input facts are removed earlier, it can cause some required services have no response. Because one operation event can match and trigger several rules and services in this model, the fact removed in the former rules can causes the mismatch in the posterior rules. For the output data and facts, if these facts remain in system after send to users the output notices to users can be repeated. For these output facts which will be invoked by another service, if they disappear after output, the subsequent services will hardly get these data. Consequently, the lifetime of every interaction fact is significant to the correct interaction.

In order to resolve the problem of interaction facts lifetime, two methods have been purposed to control the lifetime in these model. They are automatic extinction by system and deletion in rules by user definition. The first method, fact extinction by system, is automatic, according to the definition of fact. As mentioned above, the facts are defined and managed with ontology semantic descriptions. In the descriptions, user or service provider can define the lifetime of these facts. The system will control the extinction of facts according to the definitions. In the other method, users can add the fact retraction operation in rules. These rules will dispose facts when they are triggered. The retract operation can either be added in the RHS of interaction rules or be involved in a separate rule with special conditions. When several rules are triggered by a set of facts simultaneously, the different action sequences will influence the result of interaction. It will make mistake or function deficiency, when the retract fact operations exist in rules especially. In this situation, users can define priority for rules, in order to control the lifetime of facts and avoid that the retract operation affects other operations. When some rules have same priority or have no priority, the retraction operation in them also do not affect other actions in meantime.

G. User Interaction Workflow Specification

UI workflow specification matching is as important as interface for users and services. Even if the rule-based matching results for the visual template profile and the data profile are identical, it is not sufficient to establish that the user interaction is the one that is actually needed. The order and timing of data must also match, because they can also influence the result of interaction. For example an authentication UI may just simply send the user's ID and password by interaction event facts.

Various workflow specification languages have been proposed to model the application behavior including BPEL-WS [17], and PSML-S [18]. However, matching based on the workflow is difficult, and thus a simplified workflow model is required. The workflow profile needs to catch the essential inter-operation relation of the user interaction and the other services from the application.

Each type of user interaction classifications in the cloud brain system has a corresponding workflow. Hybrid interactions have more complicated workflows that are composed from the following basic interaction workflow. In this system, there are several user interaction workflow models mainly.

Data collection user interaction workflow: This type of user interaction workflow often has some sequential step to collect input events and data from users and send them to web services. The steps are shown in Fig. 3.

The first step of collection workflow is that user inputs data or operation by local user interface in user's device. The local user interface is a part of application in mobile device. User can edit information in a special form with the interface. The information may express user's situation, emotion, requirement, and other input. Then the input information will be sent to cloud brain system and be inserted to fact. As mentioned before, the collected data are expressed in form of fact. They can be treated as a part of context or situation. The fact can match rules in rule engine. Then some web services may be called and return the result or response.

Data presentation user interaction workflow: This type of UI often has an interactive workflow. An interaction initiates by a web service result, displays it to user and waits the response of user. The workflow diagram is shown in Fig. 4. It often has a sequential workflow. When some web services return results, which are required to present to user, the workflow starts. The data in the
results will match the interaction rules and the matched data will be sent to user's device with UI service and template. Then user can see the information and operate with it as required. After user operation, the workflow becomes as data collection.

Monitor user interaction workflow: This kind of UI workflow is a periodical interaction workflow. In this workflow, the period is considered in the definition of rules. The periodic rules will fired with some condition in cycle. Then some web service will be called and return result to system. User can acquire prompt on time. User can define periodic rules to request some services like weather, social network, and schedule at regular intervals. For example, a weather query rule can be defined to call the weather service every morning. The Fig. 5 shows the process of monitor user interaction workflow.

III. SYSTEM IMPLEMENTATION

A. Prototype System Architecture

In order to verify this user interaction model, a prototype has been implemented in our cloud brain architecture. Fig. 6 shows the whole implementation of cloud brain. In this figure, elements in the red rectangle are specialized for user interaction. In the implementation of cloud brain, the rule engine is based on Drools [19]. The fact base is a fact management system with Jena [20] and MySQL.

B. UI Service

As mentioned before, the UI service consists of a web service access and a web servlet server. For implementation, a message queue and a push server are involved to transmit data. Fig. 7 shows the UI service and mobile device side of the interaction prototype. The web service provides an interface based on SOAP protocol. These rules can invoke the service with the protocol directly. The fact and UI template are sent to the UI service as parameters. In the service, template parser is involved to analyze the interface XML description. The result will be used to construct a HTML page in HTML generator. A message queue based on RabbitMQ [21] maintains and transmits these HTML documents. For notice in user's device, a push server can push a notice to user's device when a new HTML document is generated into the message queue. In this prototype, the push server is designed to respond to Android application polling. In this notice, a URL for the UI interface is attached. When a user accesses the URL by a browser in his device, a request will be sent to servlet server. The servlet server in UI service can extract the HTML document from the message queue, and return the web page to user's device as response. When the user operates in the web page, servlet server will respond to this operation and insert an event fact into the system.

C. Knowledge Editor

A knowledge editor has been developed for end-user to compose their rules. The tool is also based on Android mobile system. Users can view, edit and create rules in a graphic user interface. For the purpose of creating a rule,
the interface can be separated into two parts: condition and action. To create a condition, a user can choose the condition class and the property from the drop-down box. Then the condition value can be given by the text editor. A user can add more conditions by the “add condition” button. Likewise, a user can define actions by the action part of the interface. Finally, the rule will be inserted into the knowledge base after the “insert rule” button is pressed. Fig. 8 shows the interface of the knowledge editor.

Figure 8. The interface of rule editor

Likewise, users are not required to create all the rules by themselves. Some rule libraries are available to users. In these libraries, the common senses of interaction and fact reasoning are defined. Users can import them to their own knowledge base.

D. Template Editor

Besides the knowledge editor, a template editor was also implemented for users to define the interface template. The Fig. 9 shows the structure of the template editor. The template editor contains the following elements:

Type list of web service output: users can choose the type of outputs which the new interface template suits for. These outputs will map to these tags in HTML.

Display HTML generation: it is a simple HTML editor with graphic user interface like Frontpage. Users can editor a HTML page to display the web service output.

Operation widget library: it contains a set of UI widgets, such as button, check box, drop list, and so on. Users can choose the suited widget into the template. In the generated interface, these widgets will produce UI events when a user operates in them.

Event definition: it defines the operation events by the chosen widgets

XML generation: finally, both the display HTML and event definition will be packed into a template xml file.

IV. DEMO

In order to demonstrate the process of the user interaction model and prototype, a demo has been designed with the prototype system. In this demo, some interaction rules have been edited in the rule engine. For example, an interaction rule named “Weather Fact Display” is shown in Fig. 10. This rule can be run by the rule engine in the prototype.

If the rule is triggered, a data presentation workflow will start. When the fact, whose type is “Weather Fact”, is produced in the fact base by a weather web service, it will make the rule start to execute. Then the UI service will be called with the interaction template “template.xml”. Finally, the weather fact will be retracted from fact base, in order to avoid wrong repetitive execution.

Some other rules were also considered in the demo. For example, the “MapDisplay” rule defines the interface of the map in user’s device, and the “ScheduleDisplay” rule defines the calendar notice interface for user. Other rules also can be added in to the system with the knowledge editor.

Figure 10. Interaction demo rule

Figure 11. Device display of schedule service and map service
In the UI service, a user interface page based on HTML is generated, when interaction rules are triggered. After that, this page will be pushed to user’s smart phone. The user can hit the notice in Android device to view the interface in a browser.

Fig. 11 is outputs of the schedule service and the map service. In the schedule service interface, the check box “map” and “weather” are invoked to produce UI operation events. Users can choose these items or not. For example, if a user chooses the “map” and submit, a map requiring event will be inserted into fact database. Then, the system will execute rules to request map service. The output of map service will trigger the “MapDisplay” rule, and show the map to user like the right one of Fig. 11.

Fig. 12 is the smart phone screenshot of the weather display interface. For the weather service in this demo, a Chinese web service is invoked. So, the weather information is displayed in Chinese.

The demo can verify the correction of the presentation workflow in the proposed model and prototype. Some other rules like “Interface event for weather” and “Interface event for map” are constructed to complete other workflow. These results will be no longer described because of space cause. All of these basic workflows can be supported in the prototype system.

Through the prototype system and these demos, users can custom the interfaces of services. Compared with other methods, the knowledge-based approach is more flexible. End-users who have no programming ability can define the interface and interaction workflow by creating rules and UI templates.

V. CONCLUSION AND FUTURE WORK

In this paper, we propose an end-user customized user interface generation model, which is suitable for the service composition system based knowledge. Users can define the interaction workflow and interface depending on their knowledge and templates. To realize the model, the architecture and approach have been described. Furthermore, special details of related rules, interface description, user interface service and the data profile have been discussed to make the proposed model more concrete and comprehensive. Finally, a prototype and a demo have been illustrated to verify this model.

For the future work, we will consider more types of facts into our interface model. For example, we will invoke user’s context and the sensors from device and web of things. Then, the context of the device will also be considered to using to generate more suited interface for different device hardware.

ACKNOWLEDGEMENT

The work is supported by the National Key Technology R&D Program under Grant No.2012BAH17B03, USTC-Lenovo Joint Laboratory for Cloud Computing, Supercomputing Center of USTC, and the USTC Innovation Foundation of Graduate Student.

REFERENCE


Rui Zhou is a PhD student in Computer Science and Technology at the University of Science and Technology of China. He received his bachelor degree in Xidian University in 2009. His research interests include Cloud computing, service-oriented computing, mobile computing and context-aware. He is author of some research papers published at conference proceedings.

Jinhuan Wang is a master student in Computer Science and Technology at the University of Science and Technology of China. He received his bachelor degree in Hefei University of Technology in 2011. His research interests include Cloud computing, rule-based computing, mobile computing and distribute computing.

Guowei Wang is a PhD student in Computer Science and Technology at the University of Science and Technology of China. He received his bachelor degree in the University of Science and Technology of China in 2012. His research interests include Cloud computing, rule-based computing, mobile computing and distribute computing.

Jing Li received his B.E. in Computer Science from University of Science and Technology of China (USTC) in 1987, and Ph.D. in Computer Science from USTC in 1993. Now he is a Professor in the School of Computer Science and Technology at USTC. His research interests include Distributed Systems, Cloud Computing and Mobile Computing. He is author of a great deal of research studies published at national and international journals, conference proceedings as well as book chapters.