Event-driven Coordination Rule of Web Services enabled Devices in Uniquitous environments

Kangchan Lee[*], Wonsuk Lee[*], Jonghong Jeon[*], Seungyun Lee[*], Jonghun Park[**]
[*]ETRI(Electronics and Telecommunications Research Institute)/[**]Seoul National University

Ⅰ. INTRODUCTION

Current WWW, which is based on the HTML, is operated by browsing Internet information in user side and executing the program in server side. But Web Services, which is distributed component model, can access/use/reuse a remote web object by XML based standard protocol. Web Services is usually used to extending the functions of business domain applications. According to the characteristics of Web Services, it could support the diversity of terminal, network and user environment for ubiquitous computing environment.

Ubiquitous computing network comprises a variety of distributed service devices. Today Web Services technology enables the heterogeneous devices to provide their own services and interact with each other via well-defined Internet protocol. Nevertheless, service devices in ubiquitous environment require more event-driven, autonomous interactions beyond the rather passive service-oriented architecture of the present time. So, in this paper, the technologies of Web Services for ubiquitous environment is named as Ubiquitous Web Services(UWS).

UWS shall support not only simple wired computing environment but also various terminals and computing environment (e.g. home network, Telematics, Broadband Convergence Network, Mobile, Broadcasting, etc.). It will be continuously enhanced and converged in multiple domains. Also, Ubiquitous Web Services technologies that support platform independent application integration cope with limitation of legacy middleware technologies based on distributed object. Therefore it could be software infrastructure for ubiquitous computing with other Web technologies.

In this paper, we propose the development of ECA (Event-Condition-Action) rule description language for the ubiquitous environments in an attempt to support capability for autonomous interactions among service-oriented devices in ubiquitous computing network. The rules are embedded in distributed devices which invoke appropriate services in the network if the rules are triggered by some internal or external events. The presented ECA-based device coordination approach is expected to facilitate seamless interoperation among the Web Services-enabled devices in the emerging ubiquitous computing environment.

Ⅱ. ECA RULE DESCRIPTION LANGUAGE

Ubiquitous computing environments are becoming more heterogeneous and service rich domains[1]. Devices with particular services are interconnected each other via various types of net-works. While web services technology has become a de facto standard for business application integration[2][3], it is also rapidly emerging as an effective means for achieving inter-operability in ubiquitous computing environments[4][5].

Also, in the current Web Services, the characteristics are described in WSDL, a services developer shall know the location of the WSDL, or find the appropriate WSDL in the service registry such as UDDI. It means that user can not know the additional and dynamic information like service policy, capabilities, state information, and etc. In addition, in the ubiquitous computing environment, a service shall add or remove momentarily. So, there is need to develop the specification for (i)dynamic service discovery, (ii)dynamic service binding, and (iii)dynamic service composition for the ubiquitous environments.

An ECA (Event-Condition-Action) rule description language is introduced for event-driven coordination of Web Services-enabled devices. The ECA rule is originally devised to provide traditional database systems with the capability of event-driven,

instantaneous response. Due to its advantages of distinct and comprehensible rule description, it has been spread into a variety of domains and applications, including expert systems[6], agent collaboration[7], policy-based control and management[8][9], and middleware[10][11].

Even in the field of distributed computing, event-based rule management has been actively applied to coordinate and control distributed systems. A representative work is the policy-based control and management presented in [8][9]. They proposed the policy description languages (PDL) for a centralized server to control a distributed system. In particular, Shankar[12] defined ECAP with post-conditions of devices, and addressed weak points of previous ECA rules. Nevertheless, the centralized rule execution and control has made it difficult to apply such a result to ubiquitous computing environment, mainly due to the issues of communication overhead among devices, service encapsulation, and private information management.

Furthermore, there are several previous research results on event-based service computing. SCXML (State Chart XML)[13] is an XML-based language that can define a control mechanism for distributed finite state machines by specifying their state transitions via external events. Yet, it focuses on the description of an individual device instead of a system of devices. On the other hand, WS-Eventing[14] defines a framework of effective event exchanges between Web Services. Especially, the event messages are delivered by use of a publish/subscribe mechanism, and the event content in a message can be described without restrictions for a specific application. Accordingly, this proposal can be used as a protocol to implement event-driven architectures based on Web Services, and it is adopted as a base protocol for our research.

This position paper proposes an Event-Condition-Action rule description language for Web Services (named WS-ECA) in order to support reactive interactions among service-oriented, heterogeneous devices in ubiquitous computing environment. The proposed language bases on the ECA rules which are embedded in distributed service devices and then are triggered by events from internal and external services. When the triggered rule satisfies some condition, it will be activated to invoke appropriate services.

Ⅲ. ECA RULE BASED MANGEMENT IN UBIQUITOUS SERVICE COMPUTING

ECA rules enable the Web Services devices to activate each other via event based interaction. The ECA rules have been designed so that they can satisfy the following requirements that are necessary for effective coordination of systems of ubiquitous service devices.

- Conditional response and event filtering: The rules can be activated by appointed events and designated conditions according to user preference.
- Event passing: Events satisfied by specific conditions can be forwarded to another specific device, broadcast to multiple devices, or multicast to predefined devices.
- Temporal reaction: For the same event type, different actions can be performed according to their occurrence times.
- Logical expression in rules: Rule schema can support logical expression in the rule definition such as conjunction, disjunctions, and negation.
- Rule chaining: Complex rules can be decomposed and expressed by several simple rules.

In the ubiquitous computing environments considered in this paper, service devices are assumed to be interacting via the events registered through a publish/subscribe mechanism. Specifically, the events, conditions, and actions that constitute the ECA rules are defined as follows: Events are notification messages from internal or external service device. Conditions are Boolean expression that must be satisfied in a device for some actions to occur. Finally, actions are the instructions that provide active functionality for service devices, which includes service invocation and event generation. Figure 1 shows the components of the proposed WS-ECA framework. An WS-ECA rule has three basic components, namely event, condition, and action. In addition, the schema of the rule description document shall define variables to facilitate condition evaluation based on event messages and system states.

Figure 1 shows the components of the proposed WS-ECA framework. An WS-ECA rule has three basic components, namely event, condition, and action. In addition, the schema of the rule description document defines variables to facilitate condition evaluation based on event messages and system states.

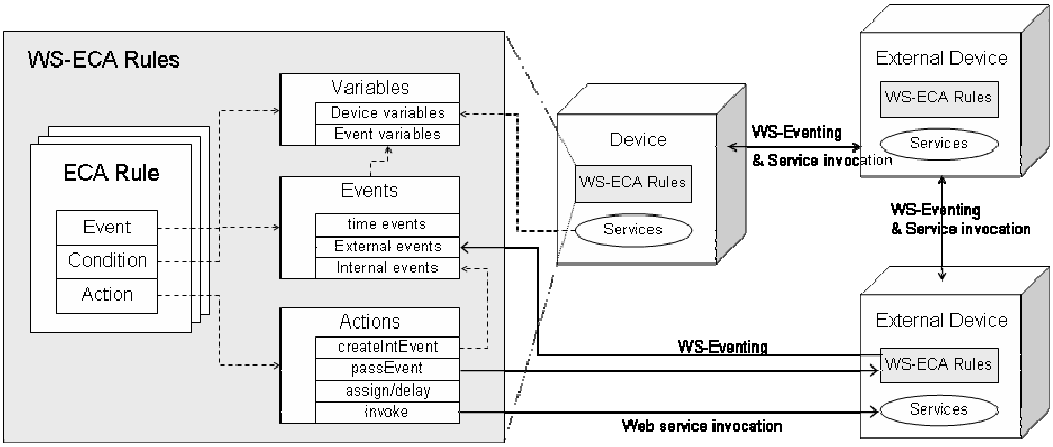

Figure 1. The component of WS-ECA Framework

3.1 Event

An event is the incident that triggers a rule. The events that can trigger a rule are called primitive events, and they include the following four types:

- **time events**: The event is generated by a timer at some specific point of time. Time events have three types of events: *absolute*, *periodic*, and *relative*. The event of *absolute* type is generated once at some time point, the event of *periodic* type occurs periodically, and finally the event of *relative* type is specified relative to some specific event by use of 'before' or 'after' operator.
- **internal events**: The event is generated by the internal system components including the rule engine and the device. It can be used to recognize the state change of a device or to trigger other rules.
- **external events**: The event is generated from a publishing device and is transmitted to subscriber devices through WS-Eventing.
- **service events**: The event can be one of two types: *before* and *after*. Specifically, the *before* (*after*, respectively) type is generated *before* (*after*, respectively) the specific service of a device starts (finishes, respectively).

More than one of these four primitive events may constitute com-posite events with the following logical operators.

- **disjunction (e*1*|e*2*|..|e*n*)**: The composite event of type "*OR*" has more than one sub-event. One or more of the sub-events must occur within its specific time interval.
- **conjunction (e*1*&e*2*&...&e*n*)**: The composite event of type "*AND*" has more than one sub-event. All of the sub-events must occur one or more times within some specific time interval.
- **sequence (e*1*,e*2*,...,e*n*)**: The composite event of type "*SEQ*" has more than one sub-event. All of the sub-events must ever occur sequentially within some specific time interval.
- **negation (~e)**: The composite event of type "*NOT*" has only one sub-event. The sub-event must not occur within its specified time interval.

3.2 Condition

The condition part of an ECA rule is a Boolean statement that must be satisfied to in order to activate a rule. In WS-ECA, the condition statement is described in terms of an XPath expression. The expression in the condition can use the values excerpted from the event part of a rule through the use of extension functions of XPath's built-in functions as shown in Table 1, or the variables defined in a rule document.

Variables can be aliases for specific elements of an event in the rule (called event variables) or user-defined states of a system (called device variables). Two types of variables are used to ex-press the conditions conveniently or to assign input data for service invocation in the actions.

3.3 Action

The action part of a rule is the instruction that must be executed by an internal or external device when a triggered rule is activated. In the proposed framework, the allowed types of actions can be one of the following:

- **invokeService (service)** : The action invokes an internal or external service.
- **createExtEvent (event)** : The action creates an external event and publishes it to the subscribing devices.
- **createIntEvent (event)** : The action creates an internal event and triggers other rules of the same device.
- **timeDelay (time)** : The action is used in combination with other actions for the purpose of delaying them for some specific time duration.

The action parts of an ECA rule may consist of above primitive actions or their composite actions. A composite action is composed of more than one primitive or composite action with two basic operators: conjunctive and disjunctive operators. We do not consider sequential actions since they can be defined by use of a series of ECA rules. Figure 5 shows the proposed schema for the action element.

- **conjunction (a*1*&a*2*&...&a*n*)**: The action requests to execute all of its sub-actions. It contains a transaction attribute that indicates whether or not the rule engine should guarantee atomicity of all sub-actions. The attribute is a Boolean variable of which the default value is 'false'.
- **disjunction (a*1*|a*2*|...|a*n*)**: The action requests to execute one of its sub-actions. It contains a sequence attribute that indicates whether the rule engine should execute the sub-actions in order. The attribute is of Boolean type and its default value is 'true'.

Ⅳ. CONCLUSION AND FUTURE WORK

This paper describes an event-based rule description language for the purpose of effective coordination of Web Services-enabled devices in ubiquitous computing environment. The WS-ECA rules enable the service devices to interact with each other via WS-Eventing. WS-ECA can be a stateless and light-weight service description language that can support instantaneous activation upon a WS-Eventing message. It has advantages that users can describe required interaction among the devices in ubiquitous computing environment where multiple devices exchange their events based on a publish/subscribe mechanism.

The presented framework on event-driven coordination of distributed Web Services-enabled devices is expected to contribute to the efficient implementation of emerging ubiquitous service-based systems, and we believe that related standardization activities are required within W3C for the ultimate Ubiquitous Web.

REFERENCES

[1] A. Friday, N. Davies, N. Wallbank, E. Catterall, and S. Pink. Supporting Service Discovery, Querying and Interaction in Ubiquitous Computing Environments. Wireless Networks 10:631–641, 2004.

[2] Y. Huang, H. Garcia-Molina. Publish/Subscribe in a Mobile Environment. Wireless Networks 10:643–652, 2004.

[3] S. Vinoski. Integration with Web Services. IEEE internet computing, 7(6): 75-77, 2003.

[4] A. Carter and M. Vukovic. A Framework For Ubiquitous Web Service Discovery. In Proc. of the 6th UbiComp, 2004.

[5] A. Sashima, N. Izumi, and K. Kurumatani. Location-Mediated Coordination of Web Services in Ubiquitous Computing, in Proc. of IEEE Int'l Conf. Web Services (ICWS'04), pages 109-114, 2004.

[6] N. Bassiliades, and I. Vlahavas. DEVICE: Compiling production rules into event-driven rules using complex events. Information and Software Technology, 39:331-342, 1997.

[7] K. Liu, L. Sun, A. Dix, and M. Narasipuram. Norm Based Agency for Designing Collaborative Systems. Information Systems Journal, 11(3): 229-247, 2001.

[8] S. Calo, and M. Sloman, Policy-Based Management of Net-works and Services. Journal of Network and Systems Management. 11(3):249-252, 2003.

[9] J. Lobo, R. Bhatia, and S. Nagvi. A Policy Description Language. In Proc. of National Conference of the American Association for Artificial Intelligence, Orlando, FL, 1999.

[10] M. Cilia and A. Buchmann. An Active Functionality Service for E-Business Applications. ACM SIGMOD Record, 31(1): 24-30, 2002.

[11] K. Dube, B. Wu, and J. Grimson. Framework and Architecture for the Management of Event-Condition-Action (ECA) Rule-Based Clinical Protocols. In Proc. of the 15th IEEE Symp. on Comp.-Based Med. Sys., pages 288-294, 2002.
[12] C.S. Shankar, A. Ranganathan, and R. Campbell. An ECA-P Policy-based Framework for Managing Ubiquitous Computing Environments. In Proc. of the 2nd Int'l Conf. on Mobile and Ubiq. Sys., 2005.
[13] R.J. Auburn, J. Barnett, M. Bodell, and T.V. Raman. State Chart XML (SCXML): State Machine Notation for Control Abstraction 1.0. W3C Working Draft, 2005.
[14] D. Bank et al. Web Services Eventing. 2004. http://ftpna2.bea.com/pub/downloads/WS-Eventing.pdf