A Workflow Language based on Structural Context Model for Ubiquitous Computing

Joohyun Han, Yongyun Cho, and Jaeyoung Choi

School of Computing, Soongsil University,
1-1 Sangdo-dong, Dongjak-gu, Seoul 156–743, Korea,
jhhan@ss.ssu.ac.kr, yycho@ss.ssu.ac.kr, choi@ssu.ac.kr

Abstract. Workflows used in business processes and distributed computing environments have supported service automation by connecting many tasks with rules and/or orderings. To adapt these workflows to ubiquitous computing, we must specify the context information on their transition conditions. In this paper, we propose uWDL, Ubiquitous Workflow Description Language, to specify context information on the transition constraints of a workflow in order to support adaptive services, and we designed a structural context model to express the context information in uWDL. Furthermore, uWDL is designed based on Web services, which are standardized and independent of various heterogeneous platforms, protocols, and languages. In order to verify the effectiveness of uWDL, we designed and implemented a scenario described with uWDL, and demonstrated that the uWDL system provides users with autonomic services in ubiquitous computing environments.

1 Introduction

Ubiquitous is a Latin word which means that the computing environments are absorbed into the physical world and integrated in everyday life [1]. In such ubiquitous environments, more sensibility is required than in traditional computing environments. Deriving context information such as location, status and actions of users, devices, and network by sensing physical environments is necessary to provide users with context-aware services based on this context information.

The workflow model in [2] supports service automation through a sequence of rules in processing tasks. It has been successively applied to traditional computing environments such as business processes and distributed computing in order to perform service composition, flow management, parallel execution, and time-driven services. It is also the task of the workflow to provide a service to the right person or the right application at the right time so that the service for a specific task can be carried out [3]. Users of ubiquitous computing environments may want to receive services in an appropriate form, at the appropriate time, and without user intervention, on dynamically changing environments [4, 5]. In order to support automatic service in these environments, we need to adapt workflows to ubiquitous computing. However, workflows must satisfy at least the following two requirements. First, a context-aware state transition is

required to provide users with appropriate services according to a user's current situation. Second, a platform and language-independent standard service interface needs to integrate, manage, and execute ubiquitous applications which consist of heterogeneous protocols on various platforms [6].

In this paper, we propose uWDL (Ubiquitous Workflow Description Language) which specifies the context information on the transition conditions of workflow services to provide users with an adaptive service for a user's current situation. It specifies context information as the perspective of rule-based reasoning which can effectively infer his/her current situation in a simple and flexible way. We also designed a structural context model to express the context information in uWDL. Furthermore, uWDL is designed based on Web services interfaces, which are not only already specified and widely used but independent of various platforms, protocols, and languages. By interpreting a scenario described with uWDL and executing the scenario, the uWDL system can effectively provide users with context-aware and autonomic services.

2 Related Work

UBL(Universal Business Language) [7], CC(Core Component), and BIE(Business Information Entity) [8] provide a formalized document style based on the ebXML [9] framework to companies for systematically supporting interoperability of the documents across the companies. These components are used in e-Business for designing the CC with UML(Unified Modeling Langauge), adapting the business context to BIE and translating the BIE into various form such as XML schema and DTD for machine readability. However, because they are designed for the purpose of e-Business, it is difficult to efficiently specify context information dynamically obtained from the sensor network in ubiquitous computing environments. uWDL can specify context information aggregated from the various sources in ubiquitous computing. As it is a workflow language, it can easily represent a flow of adequate services and select the service according to the user's situation. Furthermore, because uWDL is designed to support Web services which are platform and language-independent standard service interfaces, it can express dependency and parallel execution among the services in the heterogeneous ubiquitous computing environments.

BPEL4WS [10], WSFL [11], and XLANG [12] are Web service-based workflow languages for business processes and distributed computing environments. They support service transition, and use XML-typed messages defined in other services using XPath. However, context information is a complex data set that includes data types, values, and relations among the data types. XPath cannot sufficiently describe such diverse context information as this because it can use only condition and relation operators to decide transition conditions. uWDL uses a context triplet - subject, verb, and object - in order to express high-level context information as transition conditions which existing workflow languages are unable to support.

3 A Structural Context Model

A context in ubiquitous computing environments can indicate any information which is used to characterize the situation of an entity [13]. A context-aware application uses context information and performs context-appropriate operations [5, 13]. In ubiquitous environments, all services head for context-aware services to provide services appropriate for the user's situation. In order to provide a context-aware workflow service in ubiquitous environments, an appropriate service is selected and executed based on the context information. Workflow services must be executed conditionally, concurrently, or repeatedly according to the conditions specified in the workflow. It is also required that the workflow specifies the constraints that influence the execution order of services such as start time, end time, deadline, and conditions.

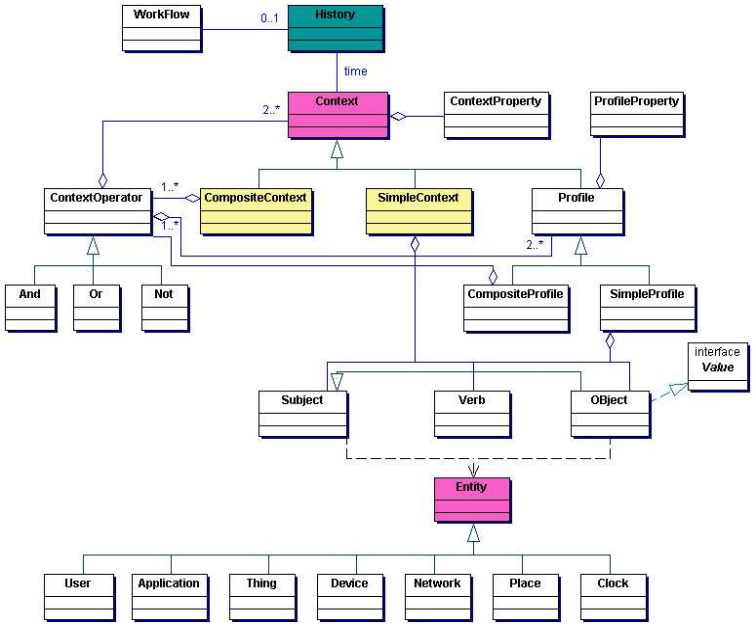

Fig. 1. The structural context model

Figure 1 shows the class diagram of the structural context model. The structural context model expresses ubiquitous context information from the viewpoint of knowledge structure. Because it has an information structure to express complex context information, it is possible to describe contexts in order to define the constraints of the state transition in uWDL. The following subsections explain some components which are context, entity, context triplet, simple context, and composite context shown in Figure 1 to specify the context information on the transition condition of services in uWDL scenario documents.

3.1 Context and Entity

The context used in ubiquitous computing is very different and varies according to domains. Generally, contexts are classified by categories such as time, location, identity, and activity. Any information obtained from ubiquitous middleware is objectified with an entity. An entity represents a person, a place or an object relevant to the interaction between a human and the corresponding virtual computer world. A context means information used to characterize the situation of entities. An entity categorized into a specific domain has the type of the domain and the value of its type.

In ubiquitous computing environments, the context information is expressed as entities and the entities are decided by the domains. For example, every entity, such as a human, application, goods, device, network, location, and system timer, constitutes all environments in human life [13]. Context information, such as the location and behavior of a man, device status information, an object in a nearby location, schedule information, an application running on a computing device, the network bandwidth, temperature, brightness, and volumn of noise, is obtained by these entities. Such context information is obtained by sensing the physical environment's entity or profile information. Profile information is the explicit information of entities. Context information obtained by sensing the environment varies dynamically over time, but profile information is fixed or varies slowly [14].

3.2 Context Triplet

In Figure 1, a context can be composed of several independent entities obtained from a sensor network. Context has a sequence of subject, verb, and object according to the meaning of the entities. This sequence is called a context triplet. The context triplet is similar to the concept of RDF(Resource Description Framework) [15]. RDF is a language to describe a resource's meta-data and expresses a resource as a triplet of {subject, predicate, object}. Because RDF can efficiently describe a knowledge structure, most ontology languages are based on RDF.

A context can be described through a context triplet associated with a particular situation. For example, let's suppose a situation, "John studies in room 301". We can divide the situation into a context triplet which is a {(PersonType, John), (ActivityType, study), (RoomType, 301)}. Each of the entities corresponds to subject, verb, and object in the context triplet according to their meanings, and in some cases a context would be composed of more than one context triplet for describing the specific situation. Finally, the context expressed by the context triplet is compared with the entities obtained from a sensor network for service transition in uWDL.

3.3 Simple Context and Composite Context

Context information varies from a low level context describing facts in physical environments to a high level context obtained by artificial intelligence. In Figure

1, context and profile information is described using a context triplet to express situation information. This context triplet is called simple context information. Some context information cannot be expressed only in simple context information. Such context information is called composite context, which combines a set of simple context information using the 'and', 'or', and 'not' operators.

4 A Ubiquitous Workflow Description Language

Although current workflow languages such as BPEL4WS, WSFL, and XLANG can specify the data flow among services based on Web services, these workflow languages do not support the ability to select services using context, profile, or event information in ubiquitous computing environments. Therefore, it is difficult to express relationships among the services using traditional workflow languages in ubiquitous environments.

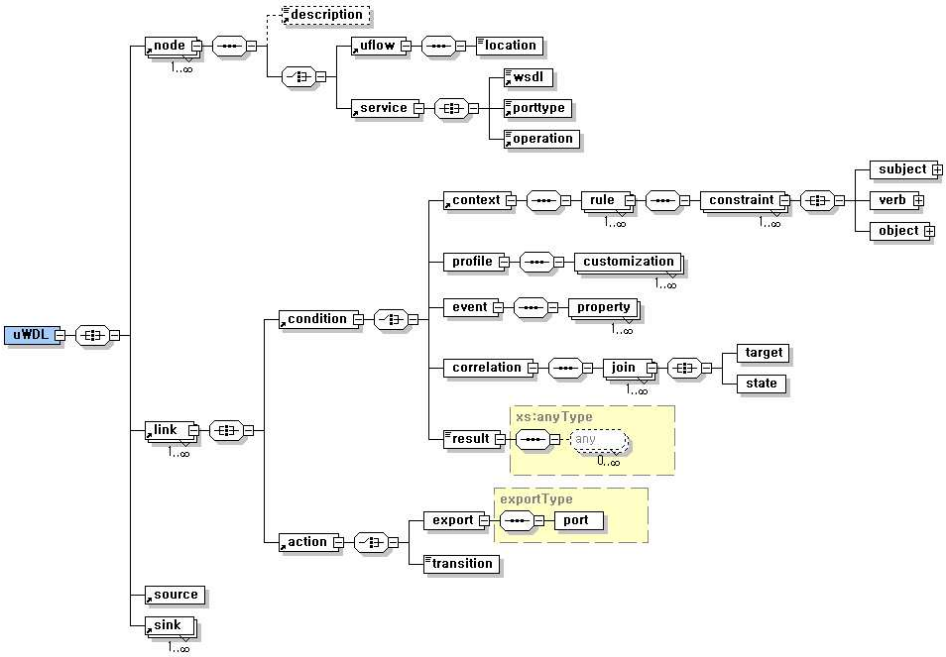

Fig. 2. uWDL schema

uWDL (Ubiquitous Workflow Description Language) is a Web service-based workflow language that describes service flows and provides the functionalities to select an appropriate service based on high-level contexts, profiles, and events information, which are obtained from various sources and structured by Ontology [16]. To provide these functionalities, uWDL specifies the context and profile

information as a triplet of {subject, verb, object} based on a structural context model for rule-based reasoning which can effectively represent the situation in a simple and flexible way. Figure 2 shows the schema structure of uWDL.

4.1 <node> element

The <node> element points to an operation that provides a functionality of Web services in ubiquitous environments. Web services use WSDL (Web Services Definition Language) to describe the port types and operations of specific Web services. Therefore, uWDL uses the <service> element and subelements of the <service> - <wsdl>, <porttype>, and <operation> - to describe the service location, type, and operation of a specific Web service, and the <uflow> element directs a reference to another uWDL document. Table 1 shows the EBNF(Extended Backus Naur Form) notation of the <node> element.

Table 1. EBNF Notation of the <node> Element

Node	::= Description?, (Uflow | Service)
Uflow	::= Location
Service	::= Wsdl, Porttype, Operation

4.2 <link> element

The <link> element is the most important part of uWDL. It specifies context, profile, and event aggregated from ubiquitous environments and defines the flow of services. The <link> element is composed of <condition> and <action> elements. The <condition> element uses <context>, <profile>, and <event> subelements to specify the context, profile, and event status of a specific node, respectively. If the calculated value of the status satisfies a given condition, the action described in the <action> element is performed. The <action> element consists of <export> and <transition> elements, where <export> has a control link and a data link according to its attribute, and <transition> specifies the state change of the current node.

The <condition> element is responsible for selecting the appropriate service based on the context, profile, and event information. The important element is <context>, which contains <constraint> to specify context information standardized by Ontology. The <constraint> element has the subelements of <subject>, <verb>, and <object>, and is designed based on the context triplet of the structural context model in Section 3. The information of subject, verb, and object represents abstract information such as location, computing device,

user activity, and the social situation of the diverse domain in ubiquitous computing environments. The <constraint> element expresses a context based on the relationship of the subject and the object, which are instances of the entities. The <subject>, <verb>, and <object> elements also have an attribute of 'type'. The type attribute represents a property of an entity in a domain. The composite attribute of the <constraint> element has an attribute value of 'and', 'or', and 'not'. By using these values of the attribute, it is possible to express the relationship among the simple contexts and describe a high-level complex context. The <rule> element means a set of the <constraint> elements, and represents the high-level expression to infer a social situation. Table 2 shows the EBNF(Extended Backus Naur Form) notation of the <link> element.

Table 2. EBNF Notation of the <link> Element

Link	::= Condition, Action
Condition	::= Context | Profile | Event | Correlation | Result
Context	::= Rule+
Rule	::= Constraint+
Constraint	::= Subject, Verb, Object
Profile	::= Customization+
Event	::= Property+
Correlation	::= Join+
Join	::= Target, State
Action	::= Export | Transition
Export	::= Port

5 The Architecture for Handling Contexts in uWDL

A uWDL document designed for a specific scenario must be translated and executed to provide an adaptive service for a user's situation. To achieve this purpose, we need a process to manipulate contexts aggregated from a sensor network. Figure 3 shows the architecture for handling the contexts expressed in uWDL. The uWDL parser parses an uWDL scenario document and produces a DIAST (Document Instance Abstract Syntax Tree) [17, 18] as a result. A DIAST represents the syntax of a scenario document. A DIAST is used to compare contexts expressed in a scenario with entities aggregated from a sensor network to verify their coincidence. A context consists of a triplet of {subject, verb, and object} in sequence. A context is described with one or more constraint elements, and each constraint is represented by a triplet. These triplets are used as nodes to construct the DIAST's subtree. In Figure 3, the partial subtree in dotted lines indicates a subtree that makes up context constraints in the scenario.

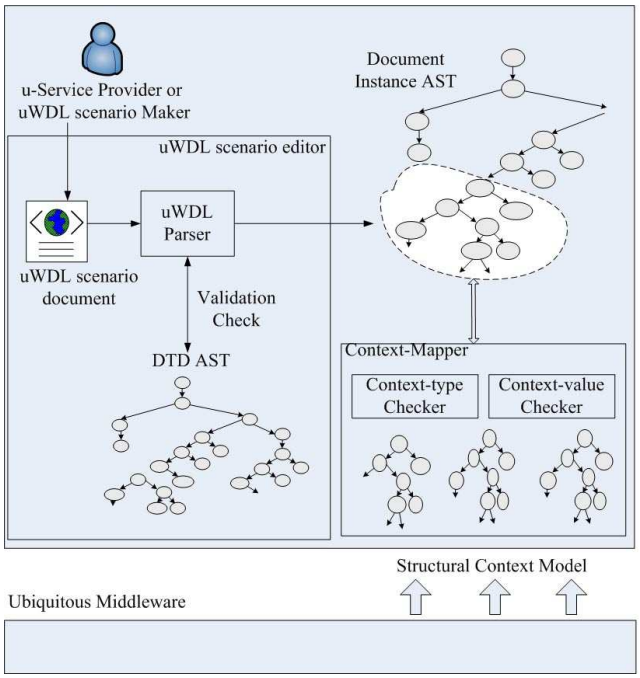

Fig. 3. The architecture for handling the context in uWDL

The context mapper extracts types and values from objectified entities aggregated from the sensor network, and composes a subtree which consists of subject, verb, and object information. The context mapper then compares the type and the value of an entity with those of the constraint elements in the DIAST's subtree. If the type in the entity matches with its counterpart in the constraint element, the context mapper regards it as a correct subelement of the constraint element. If each entity has the same type, it may be ambiguous to decide a context's constraint according to its entity type only. The problem can be resolved by comparing the value of the objectified entity with that of the constraint element in the DIAST's subtree.

6 Experiments

In this section, we show a process to decide the state transition according to the context information. The purpose is "Implementing a service which prepares an office meeting automatically according to a user's schedule." The scenario designed by using the uWDL scenario editor is as follows: "John records an appointment on his notebook computer that there is a presentation in Room 313 at 10:00 AM. John moved to Room 313 to participate in the meeting at 9:40 AM. There is a RFID sensor above room 313's door, and John's basic context

information (such as name and notebook's IP address) is transmitted to a server. If the conditions, such as user location, situation, and current time, are satisfied, then the server automatically downloads his presentation file and executes a presentation program."

For testing, we developed a uWDL scenario editor with which we created a scenario of an office meeting. Figure 4 shows the uWDL scenario editor. The uWDL scenario editor is composed of available services, constraint information, and so forth. The available services show a list of the Web services available in the current environment. The structure window shows the structure of the DIAST's subtree for the constraint element highlighted in the editing window.

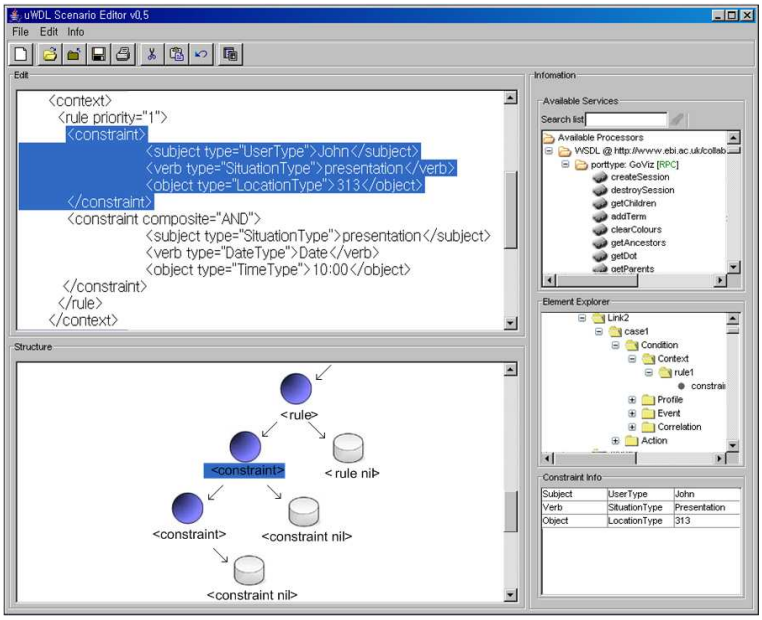

Fig. 4. uWDL scenario editor

Figure 5 shows a uWDL instance document created with the uWDL scenario editor for the above scenario. The constraint information presents a list of the entity types aggregated from a sensor network within the current environment, and a list of the entity values registered in each entity type. If the context mapper receives entities objectified as (SituationType, presentation), (UserType, Michael), (UserType, John), and (LocationType, 313) from the sensor network during processing of the scenario, it compares the entities' types and values with those of the constraint element highlighted in the DIAST's subtree of Figure 4. At the moment, because the entity (UserType, Michael) is not suitable for any one of the subtree's elements, it is removed. As a result, the entity (UserType, John) is selected as a context for the DIAST's subtree.

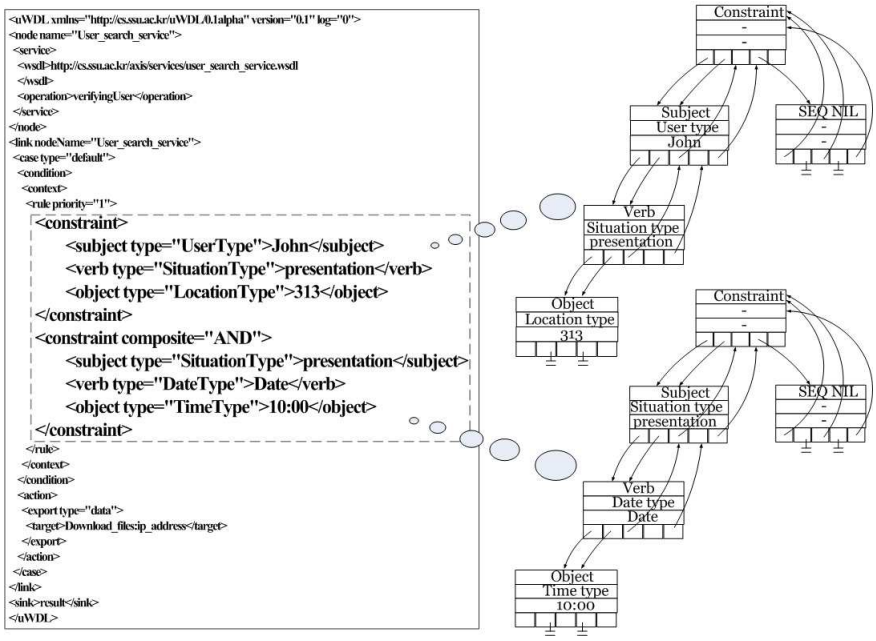

Fig. 5. The office meeting scenario and the DIAST's subtree produced by the uWDL parser for the scenario

7 Conclusion

In this paper, we proposed uWDL (Ubiquitous Workflow Description Language) which can describe service flows for a ubiquitous computing environment, and we designed the structural context model to express the context information in uWDL. Because uWDL is based on Web services, it is able to integrate, manage, and execute various heterogeneous services in ubiquitous environments. In a ubiquitous computing environment, a service engine needs a method to recognize a user's context and situation information to decide the state transitions of a service flow. As explained in Section 2, current workflow languages are for business processes and distributed computational workflows, and they are not suitable for describing the context and situation information needed for service transitions in a service flow of ubiquitous computing environments.

We designed uWDL so that it can specify the context information on transition constraints of a service workflow in ubiquitous computing environments. As a result, users are provided with an appropriate service according to the user's context information. We implemented an uWDL scenario editor and developed a sample scenario described with the uWDL, and we demonstrated that the uWDL system provides users with autonomic services in ubiquitous computing environments. In the near future, we will expand the uWDL schema to express more detailed situations by assigning semantic information to Web services.

Acknowledgements

This research is supported by the Ubiquitous Autonomic Computing and Network Project, the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea.

References

1. M., Weiser: The Computer for the 21st Century. Sci. Amer. (1991)
2. D., Hollingsworth: The Workflow Reference Model. Technical Report. TC00–1003. Workflow Management Coalition (1994)
3. Wil, van, der, Aalst, Kees, van, Hee: Workflow Management, Models, Methods, and Systems. The MIT Press. pp.147 (2002)
4. D., Saha, A., Mukherjee: Pervasive Computing: A Paradigm for the 21st Century. IEEE Computer. IEEE Computer Society Press (2003) 25–31
5. Guanling, Chen, David, Kotz: A Survey of Context-Aware Mobile Computing Research. Technical Report. TR200381. Dartmouth College (2000)
6. Mack, Hendricks, Ben, Galbraith, Romin, Irani, et al.: Professional Java Web Services. WROX Press. (2002) 1–16
7. Bil Meadows, Lisa Seaburg, "Universal Business Language 1.0", OASIS Committee Draft, Sep. 2004.
8. "Core Components Technical Specification V2.01", UN/CEFACT Technical Specification, Nov. 2003.
9. Anders Grangard, Brian Eisenberg, Duane Nickull, "ebXML Technical Architecture Specification v1.0.4", OASIS, Feb. 2001.
10. Tony, Andrews, Francisco, Curbera, et al.: Business Process Execution Language for Web Services. BEA Systems. Microsoft Corp. IBM Corp., Version 1.1 (2003)
11. Frank, Leymann: Web Services Flow Language (WSFL 1.0). IBM (2001)
12. Satish, Thatte: XLANG Web Services for Business Process Design. Microsoft Corp. (2001)
13. Anind, k., Dey: Understanding and Using Context, Personal and Ubiquitous Computing. Vol 5. Issue 1. (2001)
14. Karen Henricksen, Jadwiga Indulska, Andry Rakotonirainy: Modeling Context Information in Pervasive Computing Systems, Pervasive 2002, LNCS 2412 pp.167–180 (2002)
15. W3C: RDF/XML Syntax Specification, W3C Recommendation (2004)
16. Deborah, L., McGuinness, Frank, van, Harmelen, (eds.): OWL Web Ontology Language Overview. W3C Recommendation (2004)
17. Aho, A., V., Sethi, R., Ullman, J., D.: Compilers: Principles, Techniques and Tools. Addison-Wesley (1986)
18. Bates, J., Lavie, A.: Recognizing Substring of LR(K) Languages in Linear Time. ACM TOPLAS. Vol.16. No.3. pp.1051–1077 (1994)