

# Web Service Offerings Language (WSOL) Support for Context Management of Mobile/Embedded XML Web Services

Vladimir Tosic<sup>1\*</sup>, Hanan Lutfiyya<sup>2</sup>, Yazhe Tang<sup>2</sup>

<sup>1</sup> Department of Software Engineering, Lakehead University, Canada

<sup>2</sup> Department of Computer Science, The University of Western Ontario, Canada  
vladat@computer.org, hanan@csd.uwo.ca, tyazhe@csd.uwo.ca

## Abstract

*Specification of monitored context properties and their influence on behavior of Web services and management activities is a prerequisite for context-sensitive operation, which is a characteristic management issue for mobile/embedded Web services. Our Web Service Offerings Language (WSOL) 1.1 provided formal specification of classes of service, different types of constraints, and management statements for Web services. In this paper, we present new WSOL version 1.4 that enables specification of monitored context properties, context monitoring schedules, context management parties, and context exchange mechanisms. It also enables descriptions of how context influences behavior of Web services, monitoring of Web services, and dynamic adaptation of Web service compositions. We verified feasibility of these improvements using syntax checks of the WSOL 1.4 grammar and modifications of prototype WSOL-related tools. We validated them on emulated case studies. An important novelty of this work is using context information in contract-based management of Web services.*

## 1. Introduction and motivation

**Mobile/embedded XML (Extensible Markup Language) Web services** support ad hoc integration of diverse software running in mobile and/or embedded environments with other software running on the Internet, through the use of XML-based technologies such as SOAP and the Web Services Description Language (WSDL). Example application areas are mobile business, fleet management (e.g., truck tracking), and disaster relief. We will use the truck tracking example to illustrate important concepts. There is a frequent need to track movement and condition of

\* This work was done while Vladimir Tosic was at the University of Western Ontario as a post-doctoral fellow funded by the Natural Sciences and Engineering Research Council of Canada (NSERC).

trucks, particularly those transporting precious and/or hazardous goods. A mobile/embedded Web service may execute within a truck to provide information such as: geographic location, speed, quantity of gas left, air pressure in tires, and engine temperature. There are several possible clients of this Web service: the transportation company, companies whose goods are transported, police, and road assistance service.

**Management of Web services** is business-critical for many uses of Web services, because it ensures regular operation and handles run-time problems [1, 2, 3]. Management activities can be classified into monitoring and control. Monitoring includes measurement or calculation of quality of service (QoS) metrics, evaluation of requirements and guarantees, calculation of prices, and accounting. Control to meet guarantees and adapt to changes includes re-configuration of the Web service, re-negotiation of contracts used between Web services, and re-composition of Web services. Formal and precise specification of management information is necessary for successful management activities.

Mobile/embedded Web services often execute in environments with limited resources, e.g., scarce run-time memory, relatively low processing power, limited battery lifetime, and slow wireless links. **Management of mobile/embedded Web services** has to deal with these specifics. Further, the following management issues are characteristic to mobile/embedded Web services: (1) Context-sensitive operation; (2) Relatively frequent disturbances and changes of communication-level QoS; and (3) Possibility of relatively frequent disconnection during execution.

In the mobile computing literature, there are several different definitions of the term 'context'. A popular definition states "Context is any information that can be used to characterize the situation of an entity." [4] Unfortunately, management activities require precision that is missing from the cited definition. Any management information (e.g., values of QoS metrics) can be used to characterize situation of

an entity, so it can be viewed as context. However, mobility brings new challenges to management systems, the challenges for which the management information maintained for non-mobile systems is not enough. Therefore, we adopted a stricter definition: **context** of an application refers to information about *external* run-time circumstances that determine execution. A **context property** is an attribute of context. For a Web service, context does *not* include a description of its implementation, input values provided by a client, changes to its state caused by execution, or values of QoS metrics. For example, current time and country of location are context properties for a mobile banking Web service, but account balance is not (it is part of Web service's state). Similarly, the response time of code that implements a Web service is not context (it is determined by internal processes), but a network delay time for Web service messages is context (it is external and not fully controlled).

As a truck with a truck-tracking Web service passes an international border, there might be a need to change what information is provided and/or with what granularity. For example, there is a change in measurement units at the USA/Canada border. Thus, the country of location influences behavior of the truck-tracking Web service. Further, the truck-tracking Web service might provide different QoS (e.g., response time, availability) guarantees and/or charge different prices in different countries, which also influences monitoring activities. Changes in context may require reconfiguration of the Web service (e.g., to provide new operations required by law in the new country), changes in contracts (e.g., to accommodate slower wireless infrastructure clients might be offered contracts with lower QoS at lower price), and Web service re-composition (e.g., to send information to clients in the new county).

To enable context-sensitive behavior of a mobile/embedded Web service, a management system should support publishing, monitoring, storing, processing, analyzing, communicating, and updating context information. It should also enable using context for monitoring and control activities. It is also important to enable using context information for discovery and selection of Web services and their classes of service. A prerequisite for these activities is appropriate **specification of context management information**.

The existing solutions for management of Web services in non-mobile/non-embedded environments and the current systems for hosting mobile/embedded Web services do not address these issues. Our Web Service Offerings Language (WSOL) 1.1 and management Web Service Offerings Infrastructure

(WSOI) 2.0 better accommodate frequent change and incur less memory and communication overhead than related management works [2, 3, 5]. Therefore, we decided to extend them for management of mobile/embedded Web Services. The results are WSOL 1.4 and WSOI 3.0. In this paper, we present the **WSOL 1.4 support for context management**. We mention some of the WSOI 3.0 improvements supporting the presented WSOL 1.4 solutions, but they will be discussed in detail in another publication.

The remainder of the paper is organized as follows. Section 2 discusses related work. Section 3 gives a brief overview of WSOL 1.1 and its partial support for specification of context management information. Section 4 describes in detail WSOL 1.4 extensions supporting context management. The last section summarizes conclusions and future work items.

## 2. Related Work

Many recent works are related to context modeling, processing, and/or management, e.g., [6, 7, 9, 4, 10]. They explored the concept of context in various circumstances and from various viewpoints. However, work exploring context specification and management for mobile/embedded Web services from the viewpoint of a management system for various management activities and various types of Web services (including non-mobile/non-embedded) are missing. Several specialized formats for specification of context were developed [7], but they are different from the developed models for specification of other management information (e.g., QoS). In particular, context models and representations developed by the Semantic Web community (e.g., [8]) are not directly useful for management because they do not describe in detail management activities to be performed and are not accompanied by appropriate management infrastructures. On the other hand, there are some similarities in both semantics and syntax of context properties and other management information. For example, both are measured/calculated, aggregated, exchanged, processed, stored, and accounted in a similar manner. Further, context properties often have numerical value with associated measurement unit, a common format for other management information. Using different formats to describe similar information can lead to problems in interoperability and increases run-time overhead, which is an important issue in mobile/embedded environments. A format specifying context properties through extensions of existing management information formats for Web services would be beneficial, but was not developed.

The major related works on comprehensive description of Web services are the Web Service Level Agreement Language (WSLA) [1] and the Web Ontology Language – Services (OWL-S) [11]. These and other relevant languages do not contain context-related concepts. A common issue with using them for mobile/embedded Web services and extending them for context specification is their run-time overhead, which is greater than for WSOL [5, 2, 3].

Current commercial tools for hosting mobile or embedded Web services, such as the Web Services Toolkit for Mobile Devices (WSTKMD) also do not address context specification and management.

### 3. WSOL 1.1

**WSOL 1.1** [5, 2, 3] is a language for formal specification of classes of service, different types of constraints, and management statements for Web services. It is compatible with the existing Web service standards and adds support for management of Web services and their compositions, as well as for selection of Web services and their classes of service.

A **service offering** in WSOL is a formal representation of one class of service for a Web service. One Web service can be associated with multiple service offerings. A service offering can contain various constraints (functional, QoS, access rights), management statements (e.g., prices and monetary penalties), and reusability constructs. Every **constraint** formally states some condition to be evaluated, represented as a Boolean expression. This expression can contain other expressions (Boolean, arithmetic, arithmetic with measurement units, string, time/duration), references to monitored QoS metrics, and external operation calls. For example, QoS constraints check whether monitored QoS metrics are within specified limits. To achieve reusability of definitions of QoS metrics, they are outsourced into external ontologies, while WSOL files specify which QoS metrics are monitored, when, and by which management party (provider, client, an independent/third party). A WSOL **statement** is any construct, other than a constraint, that contains important management information. A **service offerings dynamic relationship (SODR)** describes what service offering could be an appropriate replacement if some constraints from the used service offering were not met. Since such relationships can change during run-time, SODRs are specified outside service offerings. Figure 1 shows example WSOL constructs – one service offering (“S01-Can”) containing one constraint (“countryIs-Canada”), and one SODR (“SODR1-CanToUSA”).

Their explanations are given in the next section. The figure uses WSOL 1.4 syntax (WSOL 1.4 extensions are bolded), but WSOL 1.1 examples are similar.

**WSOL parser called Premier** was designed and prototyped to demonstrate correctness of WSOL syntax. In addition, we developed corresponding management infrastructure **WSOI 2.0** [2, 3]. It enables monitoring of WSOL service offerings and run-time adaptation of Web service compositions using manipulation of service offerings. The WSOI management information model stores run-time values (e.g., of QoS metrics) from monitoring activities. They are not specified in WSOL files due to their run-time changes. The implemented dynamic adaptation algorithms and protocols address selection, switching, deactivation, reactivation, deletion, creation of service offerings. A WSOI 2.0 prototype demonstrated feasibility and usefulness of WSOL and WSOI.

### 4. WSOL 1.4 Extensions

Improvements introduced into WSOL 1.4 can be categorized into three broad groups: 1) support for context-sensitivity, 2) support for handling disconnection, and 3) other modifications and extensions. We concentrate here only on the first group, while the other two will be discussed in forthcoming publications. In particular, using context information in Web service management systems this requires **specification of context management information** that addresses the following two sets of questions:

- a) Which context properties are monitored, when, and by which management parties? When and how are values of context properties exchanged between management parties?
- b) How do context properties influence behavior of Web services (particularly providers), monitoring of Web services, and dynamic adaptation of Web service compositions?

#### 4.1. Describing context and its monitoring

WSOL 1.1 and WSOI 2.0 support context specification and management only to a limited extent, insufficient for description of various context properties. WSOL 1.4 specification of context properties is analogous to the specification of QoS metrics, in order to leverage similarities between QoS metrics and context properties and the WSOL 1.1 and WSOI 2.0 support for QoS management of Web services. We could have describe context properties (e.g., country of location) as WSOL 1.1 QoS metrics and then limit them in various constraints and service offerings dy-

```

<wsol:serviceOffering name="SO1-Can" service="tt:truckTracking"
accountingParty="http://wsolaccountant.com"
evaluatingParty="http://wsolmonitor.com">
...
<wsol:constraint
  xsi:type="contextConstraintSchema:contextConstraint"
  name="countryIsCanada" contextSensitive="true">
  <expression:booleanExpression name="inCanada">
  <expression:stringExpression>
  <expression:contextProperty
    propertyType="contextOntology:Country"
    dataType="xsd:string" propertyUnit="WSOL-NOUNIT"
    service="tt:truckTracking" monitoredBy="WSOL-PROV">
  <expression:usedContextProperty propertyType=
    "contextOntology:ProvinceOrState"
    dataType="xsd:string" propertyUnit="WSOL-NOUNIT"
    service="tt:truckTracking" monitoredBy="WSOL-PROV"/>
  </expression:contextProperty>
  </expression:stringExpression>
  <expression:arithmeticComparator type="==">
  <expression:stringConstant value="Canada">
  </expression:booleanExpression>
</wsol:constraint>
...
</wsol:serviceOffering>
...
<sodr:SerOffDynRel sodrName="SODR1-CanToUSA">
  <sodr:currentSO name="SO1-Can" />
  <sodr:unsatisfiedConstraints>
  <sodr:unsatisfiedConstraintRef name="countryIsCanada"/>
  </sodr:unsatisfiedConstraints>
  <sodr:replacementSO name="SO2-USA"/>
</sodr:SerOffDynRel>

```

**Figure 1. Examples of WSOL 1.4 constructs**

dynamic relationships. However, a Web service is responsible for values of QoS metrics (even when it cannot fully control network transfer), while this is not the case with context properties. This semantic difference can affect specification and calculation of prices and monetary penalties. Another difference is that constraints on context properties (contrary to QoS constraints) are usually evaluated after changes in context. Due to these and several other differences, we chose a somewhat different approach, described through answers to several important questions.

**1. Which context property is monitored?** To enable specification of answers to this question, we added to WSOL 1.4 new constructs (<contextProperty> and <usedContextProperty>) analogous to existing constructs specifying which QoS metrics are monitored. These constructs enable that WSOL service offerings can differ in what context properties are monitored. Analogously to QoS metrics, the actual definitions of context properties are outsourced to external ontologies, so that they are reusable. Similarly, run-time values of context properties are not described in WSOL files, but in the WSOI 3.0 management information model. Figure 1 contains one example <contextProperty> element. It specifies that

context property “Country” with data type “string” and no measurement unit is monitored. It has one sub-element <usedContextProperty> that specifies that context property “ProvinceOrState” with data type “string” and no measurement unit is monitored and used for calculation of “Country”. These context properties are defined in ontology “contextOntology”.

**2. For which party is the context property monitored?** To enable descriptions whether context properties are monitored for Web services, clients, or management third parties, we added new attributes (‘service’ and ‘monitoredFor’) of the WSOL constructs describing context properties or QoS metrics.

**3. Which party monitors the context property?** Since the monitoring party need not be the same as the party for which monitoring is performed, there is an additional attribute (‘monitoredBy’), analogous to an attribute used for describing QoS monitoring.

**4. When is the context property monitored?** Similarly to QoS metrics, context properties can be monitored before/after operation execution, before constraint evaluation, or periodically. WSOL 1.4 improves specification of schedules (attribute ‘monitoringSchedule’ and element <schedule>) for periodic monitoring of QoS metrics and context properties.

**5. When are monitored values of the context property exchanged between management parties?** It is possible that one management party monitors a context property, but that another management party evaluates the constraint containing this context property. In such situations, the former party has to send monitored values to the latter party. WSOL 1.1 implied that exchange is performed immediately after monitoring and before the enclosing constraint is evaluated. A new WSOL 1.4 attribute (‘exchangeSchedule’) describes more complex situations.

**6. How are monitored values of the context property exchanged?** In our work, monitored values of context properties and QoS metrics can be exchanged using SOAP headers, special push or pull operations built into WSOI, or operations defined through WSOL external operation calls. WSOL 1.4 improves specification of whether the source pushes the information or the recipient pulls the information (this is determined with Boolean attribute ‘pushed’).

## 4.2. Describing influence of context on Web Services and management activities

In different contexts, a Web service can require different inputs or initial conditions, provide different results, provide different QoS, bill different prices and monetary penalties, use different management

third parties, perform different manipulation of service offerings, or perform different re-configuration of Web service. WSOL 1.4 models this with several different extensions discussed through answers to the following questions.

1. **How do context properties influence behavior of Web services?** WSOL describes behavior of Web services through different constraints. Hence, influence of context properties is specified using constraints. Three sub-topics are addressed in WSOL 1.4.

a) It is possible to use **expressions that limit context properties** in any constraint or statement. In this way, values of constraints and statements can differ in different contexts. For example, a post-condition of the truck-tracking Web service could state that an operation returns speed of the truck in kilometers/hour in Canada or in miles/hour in the USA. (The addition of the element <contextProperty> enabled this.)

b) It is possible to **check whether a context property (or a QoS metric) is monitored without limiting its value**. For example, a provider Web service may monitor its geographic location and modify its operation according to the location, but not all WSOL constructs influenced by location have to limit its values and describe to consumers all details of location-sensitivity. (For this purpose, we added Boolean unary prefix operator <isMonitored>.)

c) In some situations, it is useful to **specify that within a particular WSOL service offering a context property must have particular values**. If run-time monitored values are not within these limits, then appropriate actions, such as switching between service offerings, can be undertaken to adapt to the new context. For example, a service offering of the truck-tracking Web service can specify that country of location must be Canada (e.g., because it relies on wireless communication provided only in Canada). Then, when a truck with such Web services enters USA, this constraint is no longer satisfied and dynamic adaptation can be initiated automatically. To model such restrictions, WSOL 1.4 adds built-in schemas for two new types of constraints: **context constraints** and **periodic context constraints**. WSOL service offerings can differ in context constraints and/or periodic context constraints. By default, a context constraint is evaluated on change of context, while periodic context constraint is evaluated both periodically and on change of context. Figure 1 shows an example context constraint “countryIsCanada” that limits that the truck-tracking Web service must be in Canada for this service offering to be satisfied.

2. **How do context properties influence monitoring of Web services?** Since different constraints

require at least slightly different monitoring, the WSOL 1.4 extensions described for the previous question are also relevant for monitoring activities. Further, WSOL 1.4 introduces three other extensions.

a) Specification of **which constraints and statements are used in particular context** is needed. For example, the truck-tracking Web service might provide a QoS guarantee in urban Toronto, but not in sparsely populated Northern Ontario. For this purpose, we added new optional element (<relevantIf>) to constraints and statements. It contains one Boolean expression that is evaluated before the main expression in the constraint/statement to determine whether the constraint/statement is relevant in some context. If this expression is not satisfied, the constraint is not evaluated, so the overall value of the constraint is neither “true” nor “false”.

b) It might be important to specify **whether a constraint (of any type) should be re-evaluated after a change of context or not**. For example, after a truck with the truck-tracking Web service crosses an international border, it is useful to immediately re-evaluate the context constraint limiting its geographic location, so that appropriate actions (e.g., switching of service offerings) can be started. For this purpose, we added new optional Boolean attribute (‘contextSensitive’) to all constraints. If its value is “true”, the constraint is re-evaluated when the context changes. In Figure 1, the value of ‘contextSensitive’ for the contextConstraint “countryIsCanada” is “true”.

c) Different contexts can require **use of different management third parties** for monitoring, evaluation, and accounting activities. When a truck with a truck-tracking Web service moves from Canada to the USA, management third parties used in Canada can become inappropriate (e.g., too distant). This is a special case of adaptation to changes in context by using manipulation of service offerings, discussed next.

3. **How do context properties influence dynamic adaptation of Web service compositions?** WSOL 1.4 supports WSOI 2.0 algorithms and protocols for **manipulation of service offerings** by allowing use of context constraints, periodic context constraints, and context properties in service offerings dynamic relationships (SODRs). In Figure 1, the SODR “SODR1-FromCanToUSA” specifies that if a client is using the service offering “SO1-Can” and its constraint “countryIsCanada” is not satisfied, the appropriate replacement service offering is “SO2-USA”. Also, while WSOL 1.1 required that both service offerings in a SODR be specified for same Web service, WSOL 1.4 allows that they are specified for different Web services to support Web service re-composition.

### 4.3. Verification and validation

To **verify feasibility and implementability** of WSOL 1.4 improvements presented throughout this section, we defined or modified relevant WSOL schemas and checked their syntax correctness. In addition, we thoroughly studied required modifications of corresponding WSOL tools – the Premier WSOL parser and the WSOI management infrastructure. The presented improvements of the WSOL grammar required minor changes of the Premier WSOL parser, mainly additional semantic checks. The prototype of the Premier WSOL parser already supports a number of WSOL concepts for management of mobile/embedded Web services, while further improvements will be added in the near future. The most important modification required for WSOI was extension of the WSOI management information model to store, process, and communicate context information. The implementation of a WSOI 3.0 prototype is in progress. Due to the similarities in modeling of QoS metrics and context properties, a lot of WSOI 2.0 code was reused in the WSOI 3.0 support for context.

To **validate usefulness** of the proposed WSOL 1.4 improvements, we explored two case studies (one with financial Web services, the other with the truck-tracking Web service) for which we developed example WSOL files and checked their syntax and semantic correctness. We use the same case studies for verification and validation of the WSOI 3.0 prototype.

### 5. Conclusions and future work

Context-sensitive operation is one of the characteristic management issues for mobile/embedded Web services. Specification of context information is a prerequisite for supporting context management. In spite of many recent related works, context specification for a management system performing various management activities and used by various types of Web services was not researched previously.

WSOL 1.4 provides support for context sensitivity within a language for contract-based management of Web services. It enables specification of monitored context properties, monitoring schedules, involved management parties, and context exchange mechanisms. It also allows descriptions of how context influences behavior of Web services, their monitoring, and dynamic adaptation of their compositions. We also built into WSOL 1.4 support for two new types of constraint – context constraints and periodic context constraints. We verified syntax correctness of this new grammar and are implementing correspond-

ing modifications of the Premier WSOL parser and new WSOI 3.0 to better verify feasibility and implementability. We validated usefulness of these improvements on emulated case studies.

The main item for our future work is completing the prototype implementation of WSOI 3.0. It will contain improvements corresponding to all new WSOL 1.4 features. Our longer-term objective is a powerful management system for Web services executing in diverse environments, both non-mobile/non-embedded and mobile/embedded. WSOL 1.4 and WSOI 3.0 are an important step towards this goal.

### References

- [1] Keller, A., Ludwig, H. The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services. *Journal of Network and Systems Management*, Plenum Publishing, Vol. 11, No 1 (Mar. 2003), pp. 57-81.
- [2] Tomic, V., Pagurek, B., Patel, B. Esfandiari, B., Ma, W. Management Applications of the Web Service Offerings Language (WSOL). *Information Systems*, Elsevier, Vol. 30, No. 7, pp. 564-586.
- [3] Tomic, V. *Service Offerings for XML Web Services and their Management Applications*. Ph.D. Dissertation, Carleton University, Ottawa, Canada, August 2004.
- [4] Day, A.K. Understanding and Using Context. *Personal and Ubiquitous Computing Journal*, Springer-Verlag, Vol. 5, No. 1 (2001), pp. 4-7.
- [5] Tomic, V., Patel, K., Pagurek, B. WSOL – A Language for the Formal Specification of Classes of Service for Web Services. In *Proc. of ICWS'03* (Las Vegas, USA, June 2003), CSREA Press, pp. 375-381.
- [6] Indulska, J, De Roure, D. *Proc. of the First International Workshop on Advanced Context Modeling, Reasoning, and Management* at UbiComp 2004 (Sep. 2004, Nottingham, UK). On-line at: <http://pace.dstc.edu.au/ContextWorkshop2004Program.html>
- [7] Strang, T., Linnhoff-Popien, C. A Context Modeling Survey. In *Proc. of the First International Workshop on Advanced Context Modeling, Reasoning, and Management* at UbiComp 2004 (Sep. 2004, Nottingham, UK).
- [8] Forstadius, J., Lassila, O., Seppanen, T. RDF-Based Model for Context-Aware Reasoning in Rich Service Environment. In *Proc. of the 2<sup>nd</sup> Workshop on Context Modeling and Reasoning (CoMoRea)* at PerCom'05 (Hawaii, USA, March 2005), IEEE, pp. 15-19.
- [9] Benatallah, B., Maamar, Z. (eds.) Special Issue on m-Services. *IEEE Trans. On Systems, Man and Cybernetics, Part A*, IEEE, Vol. 33, No. 6 (Nov. 2003), pp. 665-757.
- [10] Maamar, Z., Sheng, Q.Z., Benatallah, B. On Composite Web services Provisioning in an Environment of Fixed and Mobile Computing Resources. *Information Technology and Management*, Kluwer Academic Publishers, Vol. 5, No. 3-4 (July-Oct. 2004), pp. 251-270.
- [11] The OWL Services Coalition. *OWL-S: Semantic Markup for Web Services. Ver. 1.0*. WWW page (Nov. 2003): <http://www.daml.org/services/owl-s/1.0/owl-s.html>