On Requirements for Ontologies in Management of Web Services

Vladimir Tosic, Babak Esfandiari, Bernard Pagurek, and Kruti Patel

Network Management and Artificial Intelligence Laboratory
Department of Systems and Computer Engineering
Carleton University, Ottawa, Ontario, Canada
{vladimir,babak,bernie,kpatel}@sce.carleton.ca

Abstract. Formal specification of various constraints, including quality of service (QoS) and price, is very important for successful dynamic (i.e., runtime) composition of Web Services. For specification of these constraints, it is important to formally define QoS metrics, measurement units, and currency units used. Ontologies provide a mechanism for such formal definition. In this paper we discuss some requirements for ontologies that can be used in representing QoS constraints and in management of Web Services, with special focus on QoS metrics, measurement units, and currency units. Particularly, we emphasize the need for the formal representation of dependencies and relationships between QoS metrics, even when such information seems redundant. Our study of existing ontologies showed that they need further work to satisfy our requirements. We also state the need for independent, third-party Web Services for ontological translations between different QoS metrics, measurement units, and currencies.

1 Introduction

Our research group has extensive experience in applying advanced technologies for managing computer and communication networks, distributed systems, and services. For example, we have investigated the use of expert systems, neural networks, and mobile agent technologies to enhance control and management of a network. Several of our recent projects are related to the management of e-services (electronic services; a.k.a. Internet service components) and their compositions. The most recent of these projects are oriented towards Web Service technologies.

One aspect of these projects is formal and unambiguous specification of various constraints for Web Services. These constraints form electronic contracts between composed e-services and are the basis for monitoring and management activities. Consequently, the semantics of various constraints have to be defined formally and unambiguously, using appropriate ontologies. In this position paper, we discuss some requirements for such ontologies.

The paper is structured as follows. First, we briefly summarize our current research related to the management of Web Services and their compositions. Then, we mention some other recent works on formal specification of constraints for Web Services.

Next, we discuss our requirements for ontologies of QoS (Quality of Service) metrics, measurement units, and currency units. Then, we briefly review how some existing ontologies are related to our requirements. At the end, we summarize conclusions and challenges for future work.

2 Our Current Research

We believe that as the number of Web Services on the market that offer similar functionality increases, the offered QoS and price/performance ratio, as well as adaptability, will become the main competitive advantages. One of the conclusions of our past project on dynamic service composition [1, 2] was that comprehensive formal specification of e-services supports selecting appropriate e-services in the process of dynamic service composition and that it can help reduce unexpected interactions between the composed Web Services.

It can be useful to enable a Web Service to offer several different classes of service to consumers. Here, by a class of service we mean a discrete variation of the service and QoS provided by a Web Service. Classes of service can differ in usage privileges, service priorities, response times guaranteed to consumers, verbosity of response information, etc. They may imply different utilization of the underlying hardware and software resources and, consequently, have different prices. For example, a Web Service for buying stock can offer several classes of service that differ in the guaranteed maximum response time and, correspondingly, in price. With multiple classes of service, consumers get additional flexibility to better choose service and QoS that they will receive and pay for and minimize the price/performance ratio and/or the total cost of received services. Classes of service of a Web Service can differ in many various aspects, so a formal definition of a class of service is determined by a combination of formal definition of various constraints. We refer to such a formal representation of one class of service as a service offering.

For specification of service offerings for Web Services, we develop a comprehensive XML-based notation called WSOL (Web Service Offerings Language). WSOL is fully compatible with WSDL (Web Services Description Language). More information about WSOL and the status of its development, as well as appropriate examples, can be found in [3], while here we will give only a very brief summary. The syntax of WSOL is defined using XML Schema. Service offerings in WSOL are specified separately from the WSDL description of the Web Service. WSOL service offerings of one Web Service relate to the same characteristics described in the corresponding WSDL file, but differ in constraints that define classes of service. WSOL currently enables formal specification of functional constraints (pre- and post-conditions, and invariants), QoS (a.k.a., non-functional) constraints, simple access rights, price (i.e., cost), entities (the Web Service, the consumer, or some trusted third party) responsible for monitoring particular constraints in the service offering, and relationships between service offerings. In the future, we will extend WSOL with formal specification of some other constraints. In our work, specifications of different constraints are separated into multiple distinct dimensions to achieve greater flexibility and reusability of specifications. However, we integrate them into service offerings for easier choice by consumers (which are in our work other Web Services, not human end users). Note that some constraint dimensions are mutually orthogonal and independent, but some (like response time and throughput of a particular number of invocations) are mutually dependent. Consequently, we find that appropriate separation and integration of constraint dimensions is a very important issue that WSOL currently only partially addresses. Related to this issue, we also have to improve specification of relationships between constraint dimensions and between service offerings.

Composing complex information systems from Web Services, especially during run-time, can significantly increase system agility, flexibility, and adaptability. However, to further increase these qualities, such compositions have to be managed and adapted to various changes, particularly to those changes that cannot be accommodated on lower system levels like communication software, operating system, etc. This management and adaptation should occur while the information system is running, with minimal disruption to its operation and with minimal human involvement. In other words, it should be dynamic and autonomous. We want to achieve management by dynamic adaptation of compositions of Web Services without finding alternative Web Services. In other words, our dynamic adaptation should not break existing Web Service compositions, but only adjust them to new circumstances. This goal differentiates our work from the past work on adaptable software, like the architecture-based approaches based on finding alternative components and rebinding [4]. To achieve this goal we are researching dynamic adaptation capabilities based on manipulation of service offerings. Our dynamic adaptation capabilities include switching between service offerings, deactivation/reactivation of existing service offerings, and creation of new appropriate service offerings. We are also developing a corresponding infrastructure, called DAMSC (Dynamically Adaptable and Manageable Service Compositions). Among other issues, DAMSC will enable various manipulations of WSOL descriptions of Web Services. More information about our dynamic adaptation capabilities and the DAMSC infrastructure can be found in [5].

Let us here illustrate with an e-business example our work on service offerings and the corresponding dynamic adaptation capabilities. This example extends the example given in [3]. In some B2B (Business-to-Business) systems, several financial market analysis Web Services can consume the services of one or several stock market notification Web Services. In turn, the financial analysis Web Services can be used by other Web Services, like those providing decision support. Let us assume that these Web Services are provided by different vendors. The stock market notification Web Services can offer multiple service offerings, differing in the verbosity of provided information, in the rate of notification, in the priority of notification of significant market disturbances, in the guaranteed response time, etc. These service offerings would have correspondingly different prices. Using the appropriate stock notification Web Service and its appropriate service offering could help a financial analysis Web Service to provide its consumers appropriate service and QoS at competitive prices, as well as to maximize the monetary gain for its vendor. Further, the adaptability of the relationship between a stock market notification Web Service and a financial analysis Web Service might be a very valuable feature in a turbulent stock market.

For example, depending on the analysis of the current situation, the financial analysis Web Service could want to dynamically switch between different service offerings of the same stock market notification Web Service. Also, if a consumer of the financial analysis component wants to adjust the service it gets, this adjustment might require dynamic adaptation of the relationship between the financial analysis and stock notification Web Services. If for some reason (e.g., mobility) there are some temporary disturbances of the communication between these two Web Services, then the financial analysis component might have to temporarily deactivate its service offerings it can no longer support. These service offerings can eventually be reactivated after another change of circumstances. The main issue in such a scenario is what to do with the Web Services using the deactivated service offering. We have developed support for handling such cases, described in more detail in [5]. Dynamic evolution (i.e., versioning) of the stock market notification Web Service can result in the need to dynamically create new service offerings for this Web Service, but also for the financial analysis Web Service.

3 Other Works on Formal Specification of Constraints for Web Services

WSOL is not the only ongoing work on formal specification of various constraints for Web Services. However, its main distinguishing characteristics are the explicit concept of multiple classes of service associated with the same Web Service functionality and the support for manipulations of such classes of service.

IBM has been working ([6]) on WSEL (Web Services Endpoint Language). One of the goals of WSEL is to enable specification of some constraints, including QoS, for Web Services described with WSDL. To date, there is no detailed publication on WSEL.

The goal of the DAML-S project [7] -a part of the DARPA Agent Markup Language initiative [8] —is to semantically describe Web Services, which includes specification of functional and some QoS constraints. It is an effort independent of the W3C (World Wide Web Consortium) set of standards for Web Services. In other words, DAML-S is, contrary to WSOL, independent of WSDL. In DAML-S, functional and QoS constraints are defined in several properties of the *ServiceProfile* class (*precondition* and *effect* for functional constraints; *qualityGuarantee*, *degreeOfQuality*, and *qualityRating* for QoS constraints; *domainResource* for dependencies on the execution environment). However, at this time, these properties are only placeholders for constraints, because one can use any kind of DAML object for them. To be more precise, values of the *precondition* and *effect* properties are instances of the *ConditionDescription* class, but the *statement* property of the latter class can have any DAML object as a value. In our opinion, this is not an appropriate approach to formal specification of constraints because it does not support easy and unambiguous constraint evaluation and automatic generation of constraint-checking code very well. It is expected that the DAML rule language, when it is developed, will be used for formal specification of values of the *statement* property in the *ConditionDescription* class. In addition, *qualityGuarantee*, *degreeOfQuality*, *qualityRating*, and *domainResource* are defined vaguely, without any appropriate formalism. Consequently, we believe that, although DAML-S has potential for powerful semantic description of Web Services and offers significantly more powerful specification than WSDL, the combination of WSDL and WSOL currently provides better formal specification of functional and QoS constraints, as well as of price. Note also that DAML-S enables a service to provide multiple service profiles, each describing functionality and various constraints. However, contrary to WSOL, DAML-S does not explicitly define the concept of classes of service that relate to the same functionality. Further, it does not address complex relationships between various constraint dimensions and between classes of service. Consequently, we find that WSOL has advantage in systems using manipulation of classes of service.

Another approach to formal XML-based specification of QoS constraints for Web Services is described in [9]. In this paper, an XML-based language for formal definition of SLAs (Service Level Agreements) for e-services is presented. The authors first describe requirements for such a language and then discuss how their language enables formal specification of contract parties, SLA parameters, and obligations (service level and action guarantees) of the contract parties. The formal definition of SLA parameters includes information about which service elements (e.g., particular operations) these SLA parameters relate to, and how they are measured or computed. One of the goals of this work is to achieve applicability to various types of e-services, not only to Web Services. Consequently, this work is independent from the set of Web Service standards defined by W3C. However, a very important application area of this work is formal specification of SLAs between Web Services. Contrary to WSOL, this work addresses only formal specification of QoS constraints.

Most likely there are also some other ongoing efforts aimed at the formal specification of various constraints for Web Services. A comprehensive comparison of WSOL with all such approaches will be published elsewhere. In this paper, however, we want to point out that no matter which of these approaches to formal specification of QoS and other constraints is adopted, it is crucial to formally and unambiguously define ontologies of metrics, measurement units, currencies, and other terms related to monitoring and management of Web Services. In our opinion, these ontologies should be independent from the works on formal specification of constraints, so that no matter whether constraints are specified in WSOL or DAML-S, they can point to the same ontological definitions. We believe that the Semantic Web community can contribute best to this goal. Next, we discuss our requirements for such ontologies.

4 The Need for Shared Ontologies in Management of Web Services

There is a strong need for several ontologies that would be used in formal representation of QoS and other constraints for Web Services. Some of these ontologies are shown in Table 1.

Table 1. Some ontologies needed for management of Web Services

1.	Ontology of QoS metrics
2.	Ontology of measurement units
3.	Ontology of currency units
4.	Ontology of measured properties
5.	Ontology of measurement methods

Most importantly, the QoS metrics that are used to define QoS constraints have to be ontologically defined because they can be very easily misinterpreted. For example, the term "response time" in performance analysis can be used to denote two different values [10]:

1) "time from the end of request submission till the beginning of response" and
2) "time from the end of request submission till the end of response".

Both are equally good ways to use the term "response time". Some authors use one; some use the other. It is important to formally clarify which definition the given Web Service uses.

Let us now explore what information an ontological definition of a QoS metric should contain. While we try to be comprehensive in this attempt, we welcome further discussion about, and elaboration of, our conclusions. Our conclusions are summarized in Table 2.

Table 2. Requirements for an ontological definition of a QoS metric

1.	Metric name
2.	Short human-readable textual description
3.	Measured property (a link to the ontology of measured properties)
4.	Formulae (from zero to many) how the given QoS metric can be computed from other QoS metrics, each accompanied by a unit conversion rule
5.	Invariant relationships with other QoS metrics

First, an ontological definition of a QoS metric should specify the name of the metric and give a short textual description written for people. Second, the measured property (e.g., time, quantity of information, information transmission rate, …) should be specified. The meaning of these measured properties should also be ontologically defined, in the same or some other ontology.

Specification of dependencies and relationships between QoS metrics is very important. One relationship between QoS metrics is that several QoS metrics can measure the same property. This relationship can be stored implicitly in ontological definitions of QoS metrics. On the contrary, we believe that the formulae for how a particular QoS metric can be computed from other QoS metrics should be specified explicitly in the ontology. In one ontological definition of a QoS metric, several such formulae can be specified. Note that some formulae can be derived from others by substituting one or more unknown QoS metrics with appropriate formulae. Specifying such derived formulae in the ontology would be redundant. However, this often enables easier and faster use of the ontology. Consequently, we suggest specifying the most common formulae explicitly within a definition of a QoS metric, in spite of redundancy. Further, the formulae discussed should be accompanied by appropriate unit conversion rules. An example of such a unit conversion rule is “if the inputs to the formula are in [transaction] and [second], then the result of the computation is in [transactionsPerSecond]”. If the actual input is in milliseconds, the appropriate measurement unit conversion rules from the measurement unit ontology can be performed.

The ontological definition can also specify invariant relationships with other QoS metrics, particularly those that measure the same property. For example, “average response time” should always be less than or equal to “maximum response time”. While such information is probably redundant, it is very important for quick and easy discovery of conflicts. In general, when several QoS metrics are specified in the same service offering (or, for that matter, any contract) checking various dependencies and relationships given in the ontology helps to avoid various conflicts.

The ontology of QoS metrics should be accompanied by an appropriate ontology of measurement units (we will discuss some existing ontologies later). Such an ontology should define three types of units:

1) base units like “second”, “byte”, “bit”;
2) multiples of base units like “millisecond”, “megabyte”, “kilobit”;
3) derived units like “transactionsPerSecond”, “bitsPerSecond”, “bytesPerSecond”.

Synonyms, including abbreviations, should be specified for all three types. For example, “second”, “sec”, and “s” are synonyms. Ontological definitions of all measurement units should contain information about what kind of property is measured, such as time, quantity of information, information transmission rate, etc. When a measurement unit is used for a particular QoS metric, one can check whether the definition of the measurement unit and the definition of the QoS metric refer to the same measured property.

The three types of measurement units differ in what additional information should be specified in their ontological definition. For base units, it is important to specify alternative base units and corresponding conversion rules. For example, one can define “byte” as a base unit and then define it as an alternative to the base unit “bit” with the conversion rule: “byte=bit*8”. Note that, in principle, more that one alternative base unit can be specified. As far as we have explored, conversion rules between base units in QoS metrics for Web Services are relatively simple, as in the previous example. On the other hand, the conversion rules for base units in other areas (used by Web Services, not for QoS management of Web Services) can be more complex (e.g., the conversion between Fahrenheit, Kelvin, and Celsius degrees for measuring temperature). One specific of the area of QoS of Web Services is a relatively large number of additional units that have to be ontologically defined. Some examples are “invocations” (or “numberOfInvocations”), “transactions”, “events”, “failures”, etc. While such units can sometimes be replaced with the special base unit which denotes that the observed value has no named unit attached, this is not always appropriate.

For units which are multiples, it is important to specify the appropriate base unit and the multiplier used. For example, for “millisecond” the base unit is “second” and the multiplier is “milli”. The multipliers and their synonyms should also be defined in the ontology of measurement units. For example, “milli” corresponds to “1E-3” and has “m” as a synonym. Note that, contrary to physics, the range of multipliers used for QoS metrics for Web Services is relatively limited. For example, “byte” and “bit” can be specified with multipliers “Kilo”, “Mega”, “Giga”, and “Tera”, but never with multipliers like “milli”. On the other hand, “second” can be used with multipliers “milli”, “micro”, and “nano”, while we are not aware of its use with multipliers like “centi”, “deci”, “Deka”, “Hecto”, “Kilo”, and larger.

A derived unit for QoS metrics is always a proportion of two sets of units. In other words, a derived unit is proportional to some units and inversely proportional to some other units. Therefore, ontological definitions of derived units should specify a list of one or more units that are proportional and a list of one or more inversely proportional units. For example, “KilobytesPerSecond” is proportional to “Kilobyte” and inversely proportional to “second”.

Special types of measurement units are monetary units representing currencies, and units derived from them. Examples are “CanadianDollars” and “CanadianDollarsPerHour”. Usually countries give a special name to 1/100 (or sometimes 1/1000) part of their currency (e.g., “CanadianCents”) and this information should also be represented in the ontology. Also, some multilingual countries have several synonyms (from different languages) for their currency. A very important special characteristic of monetary units is dynamism of relationships between various currencies. Instead of specifying fixed formula for conversion between different currencies, an ontology could reference one (or maybe more) currency conversion Web Services that can be consulted for up-to-date conversions. Due to this feature, the ontology of monetary units might be separated from the ontology of other measurement units.

An important issue that we want to raise here is development of independent, third-party Web Services for ontological translations between different QoS metrics, measurement units, and currencies. For general usability, these conversion Web Services should be able to communicate using WSDL, not only DAML-S. For example, a Web Service can provide QoS guarantees using “average throughput [invocations/s]” for a particular “number of invocations [invocations]”, while one of its consumers can reason about QoS using “average response time [ms]”. To understand each other, they have to perform an ontological translation. However, in some cases Web Services will not be able perform ontological translations themselves. For example, they might not understand the language (e.g., DAML [8]) used to represent different ontologies. In such cases, the Web Service or its consumer should consult a specialized external ontology translation Web Service for help during the process of service offering negotiation. In general, we believe that an important contribution of the Semantic Web community can be developing semantic infrastructure for Web Services that were not developed specifically for the Semantic Web.

5 A Brief Study of Some Existing Ontologies

The main topics of our research efforts are specification of service offerings for Web Services and exploration of dynamic adaptation capabilities based on the manipulation of service offerings. In our work on WSOL, we would like to reuse ontologies developed by other authors and not to define our own. Unfortunately, our efforts to find an appropriate existing ontology were not successful.

First, we have checked the DAML Ontology Library [11] and especially the DAML representation of the Cyc Upper Ontology [12]. In this ontology, we have found the UnitOfMeasure class, as well as a number of its abstract subclasses. Among the subclasses are UnitOfMoney and UnitOfMonetaryFlowRate, representing currency. We were not able to find descriptions of concrete measurement units and currencies. Consequently, we were not able to find whether this ontology contains clear and easy specification of relationships between measurement units and how it solves dynamism of the relationships between currencies. We would also like to see the Cyc Upper Ontology modularized for easier understanding. For example, the Cyc ontology of measurement units (maybe excluding currencies) should be separated into a separate file. While the Cyc Upper Ontology can be the basis for future work on ontological representation of measurement units and currencies, we do not currently find it “ready to use” for our purpose.

The SHOE (Simple HTML Ontology Extensions) measurement ontology [13] is much simpler than the Cyc ontology. It contains some measurement units (metric system) in four categories: length, time, volume, and weight. No relationships (except belonging to a category) and no derived measurement units are defined. This simple ontology is not appropriate for our purpose.

The work described in [9] enables ontological description of SLA parameters. However, in our opinion, it would be beneficial to separate implementation-independent ontological definition of the metrics used from the descriptions of implementation-dependent elements of an SLA. In addition, their work does not mention ontological representation of measurement units.

The work described in [14] is an ontology for measurement in enterprises, not an ontology of measurement of quality of Web Services. While this ontology can be used by third-party quality measurement Web Services, it does not seem directly applicable for specification of QoS constraints for Web Services.

6 Summary and Challenges for Future Work

In this position paper, we have first briefly presented our work on WSOL and some other works on formal representation of QoS constraints. Then, we have emphasized the need for ontologies in formal representation of QoS and other constraints for Web Services. We have identified the need for ontologies of QoS metrics, measurement units, currency units, and measured properties. We have tried to define our requirements for the ontologies discussed as a useful basis for future work of the ontology community. We would like to see such ontologies developed independently from languages used to specify QoS constraints and contracts between Web Services. Unfortunately, we have not found such information in the existing ontologies that we were aware of. Therefore, an important future work for the Semantic Web community is development of appropriate ontologies that would satisfy these and other requirements. In addition, we would like to see independent, third-party, WSDL-based Web Services performing ontological translations for those Web Services that are not developed for the Semantic Web technology.

Apart from the issues discussed in this paper, there are other possible applications of ontologies in management of Web Services. For example, different measurement methods (e.g., message interception, probing, etc.) can be defined in an ontology. Also, one could think about adding issues related to measurement accuracy, statistics, etc. While there are many possibilities for such work, we believe that addressing the issues discussed in this paper is more urgent.

References

1. Mennie, D., Pagurek, B.: A Runtime Composite Service Creation and Deployment and Its Applications in Internet Security, E-commerce, and Software Provisioning. In Proc. of the 25th Annual International Computer Software and Applications Conference - COMPSAC 2001 (Chicago, USA, Oct. 2001) IEEE Computer Society Press. 371-376
2. Tosic, V., Mennie, D., Pagurek, B.: On Dynamic Service Composition and Its Applicability to E-Business Software Systems. In Proc. of the Workshop on Object-Oriented Business Systems - WOOBS'01 at the 15th European Conference on Object-Oriented Programming - ECOOP 2001 (Budapest, Hungary, June 2001) 95-108. The improved and extended version "On Dynamic Service Composition and Its Applicability to E-Business Software Systems - The ICARIS Experience" is accepted for publication as a book chapter in Corchuelo, R. et al. (eds.) Advances in Business Solutions, Nova Science (2002)
3. Tosic, V., Patel, K., Pagurek, B.: WSOL – Web Service Offerings Language. In Proc. of the Workshop on Web Services, e-Business, and the Semantic Web: Foundations, Models, Architecture, Engineering and Applications (Toronto, Canada, May 2002)
4. Oreizy, P., Medvidovic, N., Taylor, R. N.: Architecture-Based Software Runtime Evolution. In Proc. of the International Conference on Software Engineering 1998 - ICSE'98 (Kyoto, Japan, Apr. 1998) ACM Press. 177-186
5. Tosic, V., Pagurek, B., Esfandiari, B., Patel, K.: On the Management of Compositions of Web Services. In Proc. of the OOWS'01 (Object-Oriented Web Services 2001) workshop at OOPSLA 2001 (Tampa, Florida, USA, Oct. 2001) On-line at: http://www.research.ibm.com/people/b/bth/OOWS2001/tosic.pdf
6. Ferguson, D. F.: Web Services Architecture: Direction and Position Paper. In Proc. of the W3C Workshop on Web Services – WSWS'01 (San Jose, USA, Apr. 2001) W3C. On-line at: http://www.w3c.org/2001/03/WSWS-popa/paper44
7. The DAML Services Coalition: DAML-S: Semantic Markup for Web Services. WWW page. (December 12, 2001) (last accessed: March 14, 2002) On-line at: http://www.daml.org/services/daml-s/2001/10/daml-s.html
8. Defense Advanced Research Projects Agency (DARPA): DAML Language. WWW page. (September 4, 2001) (last accessed: March 14, 2002) On-line at: http://www.daml.org/language/
9. Ludwig, H., Keller, A., Dan, A., King, R. P.: A Service Level Agreement Language for Dynamic Electronic Services. IBM Research Report RC22316 (W0201-112). IBM. (January 24, 2002)
10. 10.Jain, R.: The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling. John Wiley & Sons. (1991)

11. 11.Defense Advanced Research Projects Agency (DARPA): DAML Ontology Library. WWW page. (March 14, 2002) (last accessed: March 14, 2002) On-line at: http://www.daml.org/ontologies/
12. 12.Cycorp. Quantity Vocabulary. WWW page. (Oct. 24, 1997) (last accessed: March 14, 2002) On-line at: http://www.cyc.com/cyc-2-1/vocab/quantity-vocab.html (For DAML representation of the complete Cyc Upper Ontology see also: http://opencyc.sourceforge.net/daml/cyc.daml)
13. 13.Parallel Understanding Systems Group (Department of Computer Science, University of Maryland at College Park): Measurement Ontology 1.0 (draft). WWW page. (April 3, 2000) (last accessed: March 14, 2002) On-line at: http://www.cs.umd.edu/projects/plus/SHOE/onts/measure1.0.html
14. 14.Kim, H. M., Fox, M. S.: Towards Quality Measurement Web Services: An Ontology of Measurement for Enterprise Modeling. In Proc. of CaiSE'02 (Toronto, Canada, May 2002)