Improving Temporal-Awareness of WS-Agreement*

C. Müller, O. Martín-Díaz, A. Ruiz-Cortés, M. Resinas, and P. Fernández

Dpto. Lenguajes y Sistemas Informáticos
ETS. Ingeniería Informática - Universidad de Sevilla (Spain - España)
41012 Sevilla (Spain - España)
{cmuller, resinas, pablofm, aruiz}@us.es, octavio@lsi.us.es

Abstract. WS-Agreement (WS-Ag) is a proposed recommendation of the Open Grid Forum that provides a schema to describe SLAs and a protocol to create them based on a mechanism of templates. However, although it identifies the necessity of specifying temporal-aware agreement terms (e.g. *the response time is 30 ms from 8:00h to 17:00h and 15 ms from 17:00h to 8:00h*), to the best of our knowledge, there are no existing proposals that deal with that necessity. We propose an extension that gives WS-Ag support to temporality. This allows describing expressive validity periods such as those composed by several periodic or non-periodic intervals and it applies not only to the agreement terms themselves but also to other parts of WS-Ag such as creation constraints and preferences about the service properties. In addition, in this paper we propose a *preference XML schema* to describe preferences over any set of service properties using any kind of utility function. In further research we will study a concrete specification for those utility functions.

Keywords: Temporal-Aware, Quality of Service, Service Level Agreement, WS-Agreement, Utility Functions.

1 Introduction

Service oriented architectures are based on the use of loosely coupled services to support the requirements of business processes and users. In this context, service level agreements (SLAs) [12,13,20] can be used to regulate the execution of the services and to provide guarantees related to them.

A SLA usually specifies "*which*" service is offered and "*how*" it is offered. That is to say, it includes requirements and guarantees about functional, and non-functional properties of the services. However, another important question about services is "*when*". Temporality affects orthogonally all aspects of a SLA because it may refer to the entire agreement (e.g. *the agreement expires on 2007/05/31*); to any functional property of the service (e.g. *this operation of the service is available from 8:00h to 18:00h*); or to any non-functional property

* This work has been partially supported by the European Commission (FEDER) and Spanish Government under CICYT project Web-Factories (TIN2006-00472).

B. Krämer, K.-J. Lin, and P. Narasimhan (Eds.): ICSOC 2007, LNCS 4749, pp. 193–206, 2007.

that appears in the SLA (e.g. *the response time is 30 ms from 8:00h to 17:00h and 15 ms from 17:00h to 8:00h*). Therefore, a temporal-aware SLA allows us to express precisely the periods of time in which its terms are valid.

The most significant language to specify SLAs is WS-Agreement (WS-Ag) [12]. WS-Ag is a proposed recommendation of the Open Grid Forum working group (OGF) that provides a schema for defining SLAs and a protocol for creating them based on a mechanism of templates. For compatibility and complexity, WS-Ag only defines the general structure of the agreement. Other aspects such as defining domain-specific extensions or specific languages for expressing conditions are out of the scope of WS-Ag. For this reason the research community has proposed several WS-Ag extensions like [1] and [21]. This is also the case of temporality: WS-Ag recognizes that it is necessary to include temporality in the agreement terms, but for the above mentioned reasons it does not establish how to specify it. However, as far as we know, there is no existing extension to WS-Ag that tackles the problem of temporality.

In this paper, we propose an extension to give WS-Ag support to temporality. To define it, we build on a previous work [18], in which we presented operational semantics on constraint-based temporal-aware matchmaking. We define a *temporal XML schema* and we describe how this temporal schema can be applied to the different elements of WS-Ag.

The advantages of our approach are the following: (i) we apply temporality not only to the entire agreement and the agreement terms but also to other elements of WS-Ag such as the creation constraints, which are used to create agreements based on templates, and business values, which are used to express preferences about the terms of the agreement; (ii) we support expressive specifications of validity periods such as composed intervals like "*From 8:00h to 14:00h and From 16:00h to 18:00h*" and periodical intervals like "*From Mondays to Fridays, from 8:00 to 18:00*", and (iii) as the extension builds on [18], we have a sound foundation on which to develop a constraint-based implementation to give support to the temporal extension.

Moreover, we also propose a *preference XML schema* to describe preferences over any set of service properties using any kind of utility function instead of the constant float utility function which WS-Ag specifies. The specific language for describing those utility functions is currently open and we will study it in further research.

This paper is structured as follows. Section 2 introduces a case study in which temporality is an important feature. Section 3 presents the WS-Ag structure and its temporal-awareness. Section 4 exposes our proposal of WS-Ag extension on temporal-awareness and on preferences descriptions. Section 5 compares the related proposals. Finally Section 6 exposes our conclusions and future work.

2 A Case Study

In general, temporal issues are present in the majority of agreements in real-world scenarios. In this section we explore a particular case where a provider

offers computing services to other organizations; i.e. customers send jobs (data to be processed by a certain algorithm) to be executed in the provider's infrastructure. This specific scenario represents a common situation in research fields with intensive computational requirements [4,15] as it has a wide set of temporal features that can be covered by our model.

In this scenario, a provider is likely to be looking for an optimization in the usage of its resources; that means unused (or underused) resources represent a lack of benefits and, therefore, a low recovery of the initial investment. In doing so, agreement offers should vary in a certain period on the basis of two key elements: (i) The mean time between two consecutive requests (MTBR) in the period and (ii) SLAs already signed with other customers for that period.

Concretely, we focus our case study in the following terms:

- The global validity of the SLA is from october 1/2007 to december 30/2007.
- All Sundays at 23:00h. servers are down for an hour due to maintenance.
- The Provider needs part of its server resources for its own computing necessities from Mondays to Fridays, from 8:00h to 18:00h. Therefore, in such time period, the provider requires that consumers specify in their service requests a MTBR greater or equal to 20 seconds.
- At any other instant, all server resources can be offered to consumers. Thus, at those instants the provider allows more exigent service requests over MTBR from consumers. Concretely the MTBR can be greater or equal to 1 second.
- The service consumer (i.e. the client) must specify in his agreement offer: a request; an algorithm for processing the request; the MTBR; and lastly the temporal execution pattern for the request (that means an estimation about when the service invocations are going to occur).
- The provider prefers receiving demands with a requirement of 20s or more of MTBR from Mondays to Fridays, from 8:00h to 18:00h. Thus, demands which require less MTBR should be satisfied when the provider has all server resources available.
- The provider prefers satisfying only one more exigent demand over MTBR (e.g. only one demand with MTBR=10s) rather than several less exigent demands over MTBR (e.g. 10 demands with 100s of MTBR each one).
- In periods with high MTBR available, the provider prefers customers demanding the *Knapsack algorithm (as first choice), or* Kruskal algorithm (as second choice). In other cases, the provider prefers demands of *Dijkstra algorithm, or* Kruskal algorithm (in this order of preference).

On the consumer side, we consider a case where a certain customer needs to compare two different algorithms with the same requirements on MTBR.

3 WS-Agreement in a Nutshell

3.1 Basic Description of WS-Agreement

In this section we will discuss WS-Ag, a framework for specifying electronic agreements. Concretely, this proposed recommendation specifies an XML-based

language and a protocol for advertising the capabilities of service providers, creating agreements based on agreement offers (with the possibility of further agreement compliance monitoring at runtime).

The interaction protocol comprises of two participants: the agreement initiator (that triggers the beginning of the process) and the agreement responder (that reacts to the initiator's requests). The protocol is divided in three main stages namely: (i) the initiator of the agreement process asks for agreement templates to the agreement responder.(ii) The initiator sends to the responder an agreement offer taking into account the agreement variability contained in the template. (iii) The responder accepts or rejects the agreement offer; additionally, if the responder rejects it, the process may start again.

WS-Ag proposes a structure of the agreement with the following elements:

Name: it identifies the agreement and can be used for reference to it.

Context: it includes information such as the name of the parties and their roles of initiator or responder of the agreement. Additionally, it can refer to an agreement template if needed. In this element, an agreement lifetime can be defined by means of an element called "ExpirationTime".

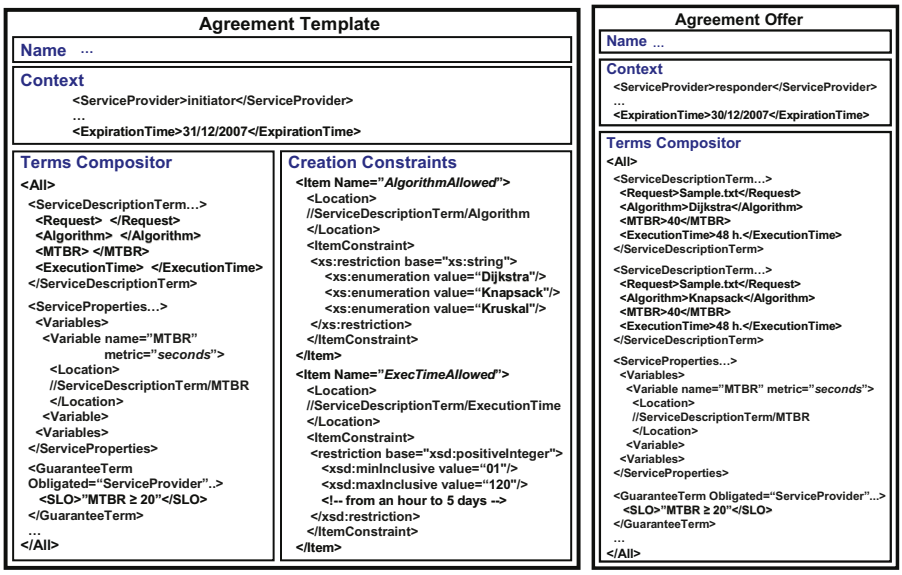
Terms: agreement terms are wrapped by term compositors, which allow simple terms or sets of terms to be denoted by "ExactlyOne", "OneOrMore", or "All". The following are the two main types of terms:

1. Service terms: they provide information to instantiate or identify services and operations involved in the agreement. Additionally, it can comprise information about the measurable service properties.
2. Guarantee terms: they describe the service level objectives (SLO) agreed by the parties. They comprise a SLO specified as a target for a key performance indicator, or as a "CustomServiceLevel" element in a customized way; it also includes the scope of the term (e.g. a certain operation of the service or the whole service itself); a "QualifyingCondition" that specifies the validity conditions under which the term is applied; and information about business properties in the "BusinessValueList" element of the guarantee term such as "Importance", "Penalty" or "Reward" and "Preference" defined as an utility value pointing to a service term.

In order to create agreements, WS-Ag allows to specify templates with the above structure, but including agreement creation constraints that should be taken into account during the agreement creation process. These constraints describe the variability allowed by a party; they can be denoted as general "Constraints", or "Items" pointing to specific locations with their own constraints.

3.2 Temporal-Awareness of WS-Agreement

Concerning temporal issues, WS-Ag identifies two locations to include temporal awareness. On the one hand, lifetime for the entire agreement must be included in



(a) Agreement Template. (b) Agreement Offer.

Fig. 1. An Example of Agreement Template and a possible Agreement Offer

Context into the "ExpirationTime" element (i.e. the last instant where the agreement is valid). On the other hand, WS-Ag recommends the use of "QualifyingCondition" elements for describing validity periods of terms and/or the party preferences. However, the specification document leaves open the specific way these temporal awareness must be exposed for reasons of compatibility and complexity.

The case study presented in the previous section includes several issues with little (or no) support with WS-Ag; in particular, the temporal execution pattern has a high degree of complexity for the WS-Ag recommendation (it includes several temporal expressions) thus we have to reduce it as denoted in the agreement template of Figure 1(a). This implies two main simplifications: (i) specifying the lifetime with an expiration time only (not initial time); (ii) temporal execution pattern for the request (which in the case study is expressed as several validity periods) has to be changed with a simple value of execution time.

Additionally, the need of restarting the server periodically could have been described as creation constraints for the agreement (and thus consumers would have to take these constraints into account in their agreement offers). However, we find that WS-Ag does not allow this temporal description, and therefore we have to reduce the example by restricting only the possible range of execution time in hours (and in addition the algorithms allowed). The possible values for MTBR correspond to the worst case (MTBR$\geq$20s) and the preferences of the provider require the validity period of the case to be included in the example, but for the above reason it is not possible.

Figure 1(b) depicts an offer for such template describing an execution of the same request with two different algorithms with similar MTBR.

4 Our Proposal

We propose a WS-Ag extension for describing temporal properties in SLAs. At first, we specify a generic temporal XML schema which allows to include several forms of validity periods.

4.1 Temporal Schema

We have already studied temporality on web services in previous works. In [18] we presented a constraint-based approach to temporal-aware web services procurement. In [19] we elaborated a study about expressiveness of temporal descriptions for web services. And we have reviewed the kinds of temporal periods defined in the *IETF RFC 3060* [24]. Now we can formulate that validity periods on SLAs can be composed of one or more temporal intervals, periodic or not. There are several types of intervals, namely non-disjoint, disjoint (both mentioned by Allen in [3]), and/or periodical. A non-disjoint interval is composed of a single interval. A disjoint interval is composed of several sub-intervals, so that it does not include all time points between its lower and upper ends. And an interval is periodical if it is repeated regularly.

We have designed an XML schema named "twsag.xsd" for describing these validity periods in practice. An interval is the basic element; different non-disjoint intervals can be grouped together so that more complex intervals can be composed. Several authors [5,17] have proposed a more friendly representation of XML schemas by means of UML class diagrams. Thus, Figure 2 shows an UML class diagram which represents "twsag.xsd"; the three interfaces denote the types of intervals above mentioned: (1) **Interval**: it stands for the basic element; it is comprised of an initial time and a duration (which can be infinite) expressed in seconds, hours, days, or months. (2) **Disjoint**: it stands for disjoint intervals constituted of a set of intervals related by a logic operator (or, and, or xor). (3) **Periodical**: it stands for periodic intervals, be either disjoint or non-disjoint. Its periodicity is comprised of the number of period repetitions (which can be infinite) and a frequency expressed in seconds, hours, days, or months, which denotes the time between two consecutive intervals.

Our proposal allows to include temporality regarding several aspects of agreements. Therefore, we comment them separately: first, temporality on agreement terms and agreement creation constraints in Section 4.2; and later, temporality on preferences in Section 4.3.

4.2 Temporality on Terms and Creation Constraints

Depending on the way validity periods affect the agreement terms, we classify them in two groups: (1) global periods (GP) if validity periods wrap all agreement terms; and (2) local periods (LP) in other cases. We have studied the inclusion of these types of periods in the WS-Ag structure.

WS-Ag specifies the lifetime of agreements by means of an "Expiration Time" in the context. Thus, it only allows a non-disjoint GP, starting from the current

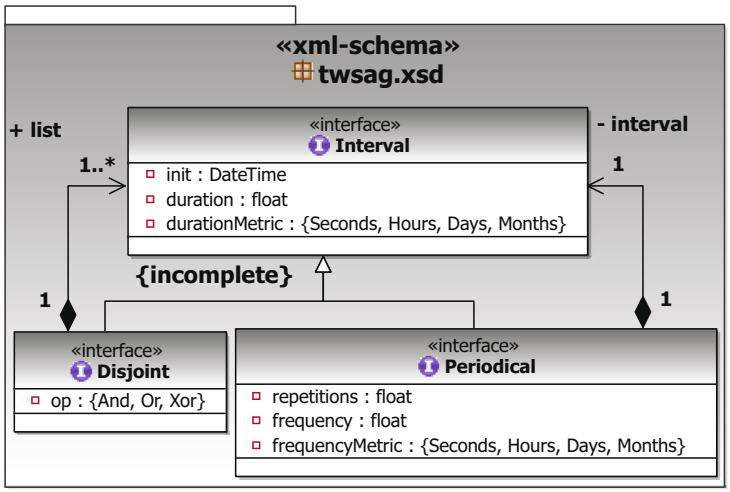


Fig. 2. Schema for Temporal Intervals

date. For a lifetime to be expressed without restrictions, we propose to use the "Any" element, which allows to include any information in the context, for including a new element called "GlobalPeriod" in order to describe it as an "Interval" element of our temporal schema.

WS-Ag recommends to specify temporality regarding agreement terms in the "QualifyingCondition" element. We propose to specify these local periods by means of "Interval" elements of our temporal schema.

Figure 3 shows the global and local periods for the scenario described in Section 2. Figure 4 shows a template and an offer using our WS-Ag extension for describing the validity periods in this case study. In Figure 4(c), note that non-disjoint intervals are put into a single periodical non-disjoint interval which constitutes the agreement offer GP; and periodical disjoint intervals are used to constitute the agreement offer LPs.

It is important to remark that WS-Ag only includes temporal properties in guarantee terms. However, we also need to describe validity periods of service terms. In Figure 4(b), functional properties described in service description terms are active only at specific validity periods (e.g. we must use the service description term with MTBR≥20s, in case of periods with a minimum of 20s of MTBR allowed). Therefore, we make use of term compositors to associate service terms with the guarantee terms which contain the desired validity period.

We also allow to specify temporal properties regarding the agreement creation constraints. There are two ways of describe them: either to allow validity periods on single constraints, e.g. "Provider must allow execution tests with a minimum MTBR of 40s, *48 hours before agreement initiation date*"; or to allow several constraints apart from the validity period definition, e.g. the previous constraints

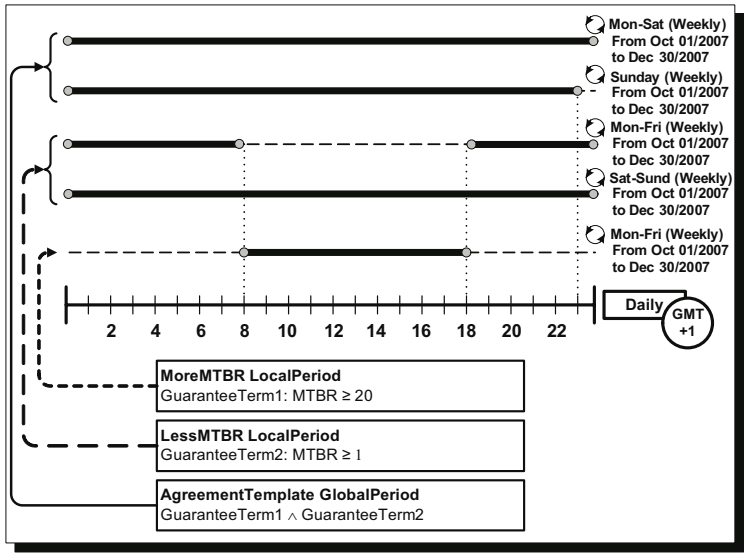


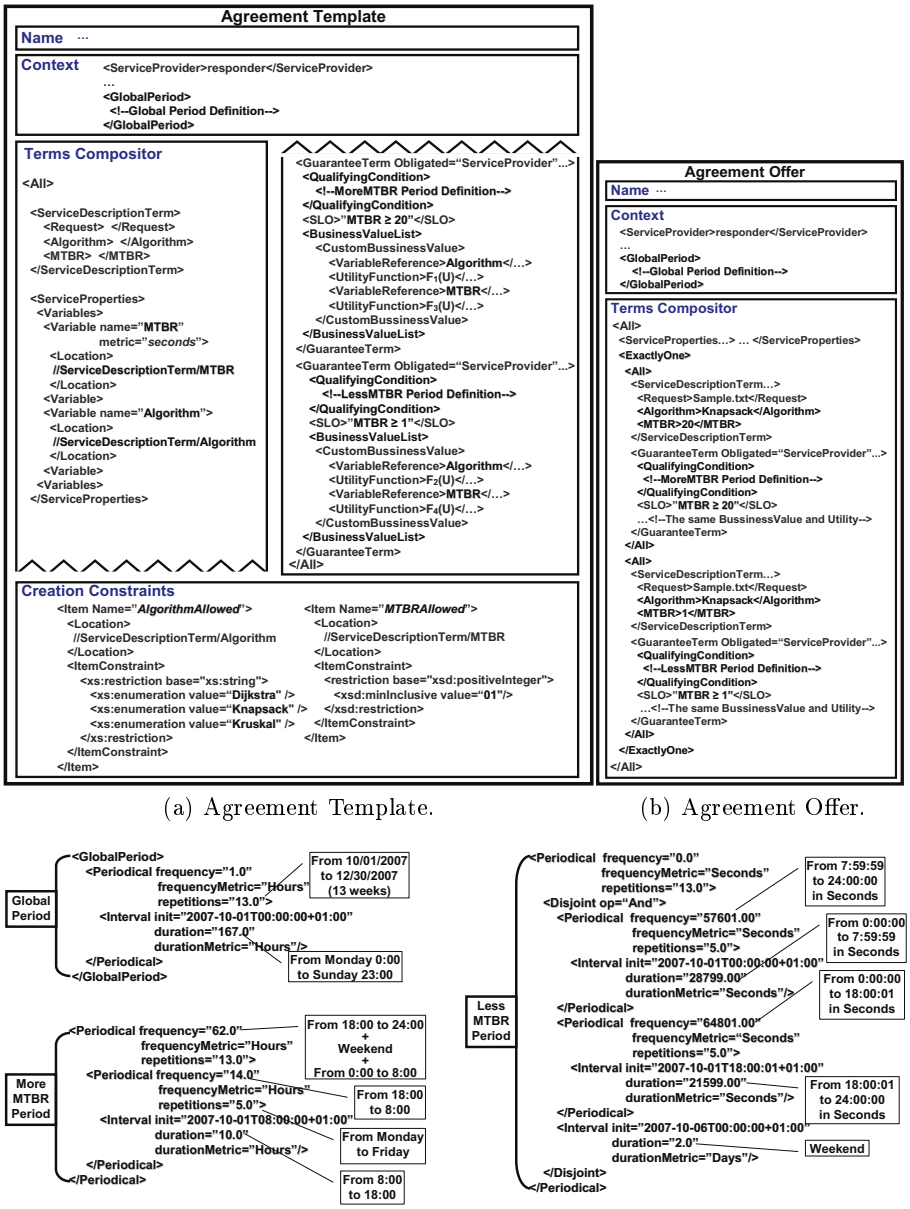
Fig. 3. Global & Local Periods for the Case Study

without validity period: "Provider must allow execution tests with 40s of MTBR", and also "Provider must assure a maximum execution time of 24 hours", both active during the validity period: "*48 hours before agreement initiation date*". For temporality in creation constraints to be allowed, we propose to describe it as an "Interval" element of our temporal schema: (1) a new element under the "Item" of creation constraints, for describing temporal periods on a single constraint (by means of "Any" element of WS-Ag); and (2) the "Constraint" element for temporal periods on several constraints. Figure 5 denotes case (1) with an example of testing requests before an agreement initiation date.

4.3 Temporality on Preferences

In a guarantee term, a validity period described in "QualifyingCondition" involves not only the service level objective, but also the preferences in the "BusinessValueList" element. However, preferences in WS-Ag are described with limitations, because we must specify a float constant value in the "Preference" element in order to describe the utility of a concrete service description term. That forces to define (1) several service description terms with different choices of values in service properties according to preferences, and (2) several guarantee terms, including the constant utility of each service description term on each validity period. Therefore, we obtain constant utility functions anyway.

In order to use any utility function with any number of service properties, we propose to extend the manner of expressing preferences in WS-Ag by using the "CustomBusinessValue" element. Our purpose is to describe the preference



(a) Agreement Template. (b) Agreement Offer.

(c) Period Definitions.

Fig. 4. Agreement Template and Agreement Offer

```
<template...>
...
  <CreationConstraints>
    <Item Name="TestPrevious">
      <Location>
        //ServiceDescriptionTerm/MTBR
      </Location>
      <ItemConstraint>
        <xsd:restriction base="xsd:positiveInteger">
          <xsd:minInclusive value="40"/>
        </xsd:restriction>
      </ItemConstraint>
      <Interval init="2007-09-29T00:00:00+01:00"
                duration="2.0"
                durationMetric="Days"/>
    </Item>
  </CreationConstraints>
  ...
</template>
```

Fig. 5. Example of Creation Constraints with Temporality

of one or more service properties in a concrete validity period with any kind of utility function. Figure 6 shows the structure of our preference XML schema for describing the "CustomBusinessValue" element. It defines utility functions pointing to one or a group of variables, which are described in the corresponding "ServiceProperties" element of WS-Ag.

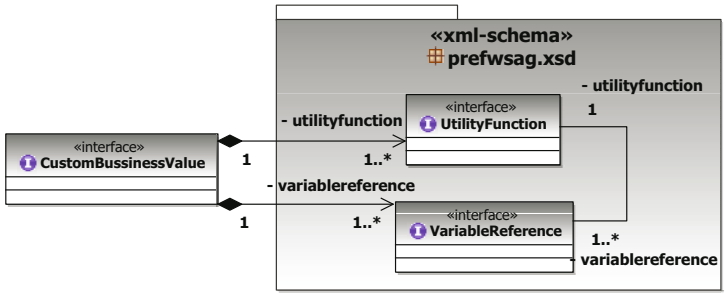


Fig. 6. Schema for Preferences

Figure 4(a) shows the utility function with its name; currently the way for expressing the function is open. For simplicity, in the example we only describe utility functions over one variable. To represent the utility functions, we have to take into account the provider preferences included in the case study. Those preferences are several criteria on the algorithms, on satisfying demands in concrete validity periods, and on satisfying one more exigent demand in MTBR (e.g. a lower value) than several less exigent demands. Figure 7 denotes the utility functions referenced in Figure 4(a).

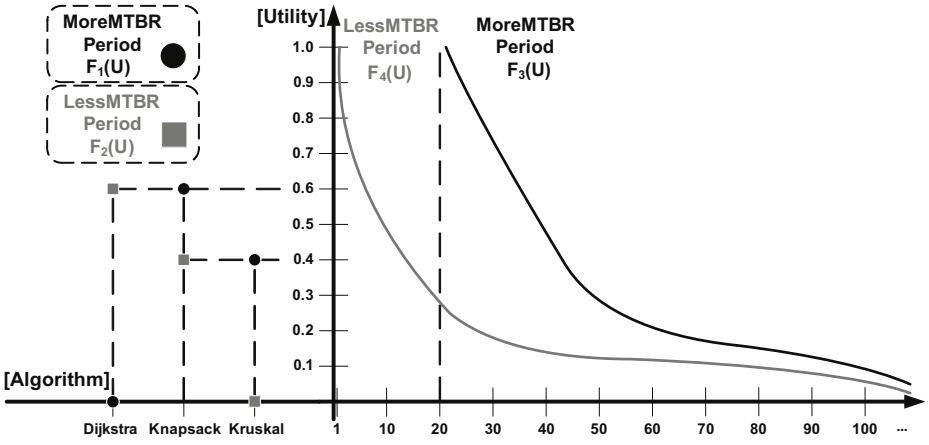


Fig. 7. Utility Functions from the Case Study

5 Related Work

Several authors have studied temporal-awareness on service descriptions. In Table 1 we show a comparative of their proposals, including this paper (those of traditional web at left side, and those of semantic web at right).

Concerning temporal-aware terms, the table denotes that authors who consider GPs, only mention "Non-Periodical" and "Non-Disjoint" intervals, but neither "Periodical" nor "Disjoint" intervals. On the other hand, authors who take into account LPs mention "Periodical" and "Non-Disjoint", but only METEOR-S and WSMO/WSML show interest in "Non-Periodical" intervals. The other authors don't even mention "Non-Periodical" or "Disjoint" intervals in their works. The reason for that lack of "Disjoint" intervals may be due to the fact that they can be expressed by means of several "Non-Disjoint" intervals, though this solution is less expressive. We emphasize WSML(HP) and WSOL because these proposals concern both GPs and LPs in their works. Other proposals like QoSOnt, METEOR-S, and WSMO/WSML have declared that they will study GPs and LPs in their future work.

Only a few of the proposals, among those which are temporal-aware, have taken preferences into account. However, to the best of our knowledge, none of them have studied temporality on preferences and creation constraints. We distinguish two ways to declare the preferences: (1) by comparing the "degree of similarity" between values of service properties from different agreement offers and templates; for example, if a provider specifies in the agreement template that it prefers a value of MTBR of 30s, an agreement offer which requires a MTBR of 32s will be more similar to the template than another offer requiring 20s; and (2) by comparing utility values given by utility functions defined on the service properties, just as we have described above. Both alternatives use weights as a means of incorporating the degree of importance among service properties to the

Table 1. Comparative between Traditional & Semantic Web Proposals

Proposals	Our Proposal.	Lodi et al. [15]	WS-QoS [25]	EWSDL [6]	UDDIe [2]	WSML (HP) [23]	WSOL [26]	WSLA [16]	Gouscos et al. [11]	Trastour et al. [27]	DAML-QoS [7]	Gonzalez. et al. [10]	Li & Horrocks [14]	QoSOnt [9]	WSMO/WSML [8]	METEOR-S [21]
TEMPORAL-AWARENESS ON TERMS																
GP/NP	√	√	√	√	√	√	√			√	√	√	√	∼	∼	∼
GP/P	√															
GP/ND	√	√	√	√	√	√	√			√	√	√	√	∼	∼	∼
GP/D	√															
LP/NP	√													∼	∼	∼
LP/P	√					√	√	√	√							
LP/ND	√					√	√	√	√					∼	∼	∼
LP/D	√															
PREFERENCES																
DoS				√	√							∼		∼	∼	√
UF	√											∼			∼	∼

D=Disjoint, ND=Non-Disjoint, P=Periodical, NP=Non-Periodical
DoS=Degree of Similarity, UF=Utility Functions.
√=feature included, ∼=feature identified as future work.

preferences. EWSDL, UDDIe, and METEOR-S have based their preferences on the degree of similarity, whereas other proposals have only mentioned their interest. Regarding utility functions, Gonzalez et al., WSMO/WSML, and METEOR-S are currently working on incorporating this feature to their proposals.

6 Conclusions and Future Work

In this paper, we have shown how a temporal domain-specific language (DSL) can be used to incorporate validity periods into WS-Ag descriptions, such as qualifying conditions associated to SLOs, template creation constraints during agreement creation process, or preferences over service properties. In order to express these validity periods, we have define a schema which includes several kinds of temporal intervals, from disjoint to periodical. Our temporal DSL would have a very large domain of applications, apart from WS-Ag. In addition we propose another schema which allows the definition of preferences over service properties using any utility function instead of constant float functions as described in WS-Ag. Currently, we abstain from defining a specific language for describing the utility functions.

For future work, we are considering several open issues. First, our temporal DSL should be validated in different scenarios to prove its soundness. In order to do so, it would be needed to develop a proof-of-concept prototype from operational semantics on temporal-aware matchmaking defined in previous works [18,22]. Another future improvement would be defining a concrete DSL to specify advanced utility functions, in order to complete our improvement for the preferences description in WS-Ag.

Acknowledgements

The authors would like to thank the reviewers of the 5^{th} International Conference on Service Oriented Computing, whose comments and suggestions improved the presentation substantially.

References

1. Aiello, M., Frankova, G., Malfatti, D.: What's in an Agreement? An Analysis and an Extension of WS-Agreement. In: Benatallah, B., Casati, F., Traverso, P. (eds.) ICSOC 2005. LNCS, vol. 3826, pp. 424–436. Springer, Heidelberg (2005)
2. Ali, A.S., Al-Ali, R., Rana, O., Walker, D.: UDDIe: An Extended Registry for Web Services. In: Proc. of the IEEE Int'l Workshop on Service Oriented Computing: Models, Architectures and Applications at SAINT Conference, IEEE Press, Los Alamitos (2003)
3. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Communications of the ACM 26(11) (1983)
4. Balaziska, M., Balakrishnan, H., Stonebraker, M.: Contract-Based Load Management in Federated Distributed Systems. In: Proc. of the ACM Symposium on Networked Systems Design and Implementation, San Francisco, California, ACM Press, New York (2004)
5. Bernauer, M., Kappel, G., Kramler, G.: Representing XML Schema in UML - A Comparison of Approaches. In: Koch, N., Fraternali, P., Wirsing, M. (eds.) ICWE 2004. LNCS, vol. 3140, pp. 440–444. Springer, Heidelberg (2004)
6. Chen, Y., Li, Z., Jin, Q., Wang, C.: Study on QoS Driven Web Services Composition. In: Zhou, X., Li, J., Shen, H.T., Kitsuregawa, M., Zhang, Y. (eds.) APWeb 2006. LNCS, vol. 3841, pp. 702–707. Springer, Heidelberg (2006)
7. Chen, Z., Liang-Tien, C., Bu-Sung, L.: Semantics in Service Discovery and QoS Measurement. In: IT Pro - IEEE Computer Society, pp. 29–34 (2005)
8. de Bruijn, J., Feier, C., Keller, U., Lara, R., Polleres, A., Predoiu, L.: WSML Reasoning Survey (November 2005)
9. Dobson, G., Sánchez-Macián, A.: Towards Unified QoS/SLA Ontologies. In: Proc. of the 3^{rd} IEEE International ICWS/SCC Workshop on Semantic and Dynamic Web Processes, Chicago, IL, pp. 169–174. IEEE Press, Los Alamitos (2006)
10. González-Castillo, J., Trastour, D., Bartolini, C.: Description Logics for Matchmaking of Services. Technical Report HPL-2001-265, Hewlett-Packard (2001)
11. Gouscos, D., Kalikakis, M., Georgiadis, P.: An Approach to Modeling Web Service QoS and Provision Price. In: Proc. of the IEEE Int'l Web Services Quality Workshop (at WISE'03), pp. 121–130. IEEE Computer Society Press, Los Alamitos (2003)

12. OGF Grid Resource Allocation Agreement Protocol WG (GRAAP-WG): Web Services Agreement Specification (WS-Agreement) (v. gfd.107) (2007)
13. IBM: Web Service Level Agreement (WSLA) Language Specification (2003)
14. Li, L., Horrocks, I.: A Software Framework for Matchmaking based on Semantic Web Technology. In: Proc. of the 12^{th} ACM Intl. Conf. on WWW, pp. 331–339. ACM Press, New York (2003)
15. Lodi, G., Panzieri, F., Rossi, D., Turrini, E.: SLA-Driven Clustering of QoS-Aware Application Servers. IEEE Transactions on Software Engineering 33(3), 186–196 (2007)
16. Ludwig, H., Keller, A., Dan, A., King, R.P.: A Service Level Agreement Language for Dynamic Electronic Services. Technical Report 22316 W0201-112, IBM (2002)
17. Marcos, E., de Castro, V., Vela, B.: Representing Web Services with UML: A Case Study. In: Orlowska, M.E., Weerawarana, S., Papazoglou, M.M.P., Yang, J. (eds.) ICSOC 2003. LNCS, vol. 2910, pp. 17–27. Springer, Heidelberg (2003)
18. Martín-Díaz, O., Ruiz-Cortés, A., Durán, A., Müller, C.: An approach to temporal-aware procurement of web services. In: Benatallah, B., Casati, F., Traverso, P. (eds.) ICSOC 2005. LNCS, vol. 3826, pp. 170–184. Springer, Heidelberg (2005)
19. Müller, C., Martín-Díaz, O., Resinas, M., Fernández, P., Ruiz-Cortés, A.: A WS-Agreement Extension for Specifying Temporal Properties in SLAs. In: Proc. of the 3^{rd} Jornadas Científico-Técnicas en Servicios Web y SOA (2007)
20. OASIS and UN/CEFAT: Electronic business using XML (ebXML) (2007)
21. Oldham, N., Verma, K., Sheth, A., Hakimpour, F.: Semantic WS-Agreement Partner Selection. In: 15^{th} International WWW Conf., ACM Press, New York (2006)
22. Ruiz-Cortés, A., Martín-Díaz, O., Durán, A., Toro, M.: Improving the Automatic Procurement of Web Services using Constraint Programming. Int. Journal on Cooperative Information Systems 14(4), 439–467 (2005)
23. Sahai, A., Machiraju, V., Sayal, M., Jin, L.J., Casati, F.: Automated SLA Monitoring for Web Services. Research Report HPL-2002-191, HP Laboratories (2002)
24. The Internet Society: Policy Core Information Model - v1 Specification (2001)
25. Tian, M., Gramm, A., Naumowicz, T., Ritter, H., Schiller, J.: A Concept for QoS Integration in Web Services. In: Proc. of the IEEE Int'l Web Services Quality Workshop (at WISE'03), pp. 149–155. IEEE Computer Society Press, Los Alamitos (2003)
26. Tosic, V., Pagurek, B., Patel, K., Esfandiari, B.: Management Applications of the Web Service Offering Language (WSOL). In: I. Systems, pp. 564–586 (2005)
27. Trastour, D., Bartolini, C., González-Castillo, J.: A Semantic Web Approach to Service Description for Matchmaking of Services. Technical Report HPL-2001-183.