Faculty of Informatics - Papers (Archive)

Faculty of Engineering and Information Sciences

1-1-2009

Towards dynamic formation of temporal constraints for the service level agreements negotiation

Azlan Ismail
University of Wollongong, ai423@uow.edu.au

Jun Yan
University of Wollongong, jyan@uow.edu.au

Jun Shen
University of Wollongong, jshen@uow.edu.au

Recommended Citation

Ismail, Azlan; Yan, Jun; and Shen, Jun: Towards dynamic formation of temporal constraints for the service level agreements negotiation 2009, 72-79.
https://ro.uow.edu.au/infopapers/1437

Towards dynamic formation of temporal constraints for the service level agreements negotiation

Abstract

SLAs play an important role for the QoS-driven service composition. Meanwhile, the temporal constraints are one of the main elements in the management of SLA especially in specifying the validity period of the QoS offers. In practice, the temporal constraints should be generated dynamically by taking the resource capability of the provider into account. The generation should consider various parameters that influence the resource capability such as the expected duration of the required Web service, the amount of current utilization, the amount of available resources, the number of required time slots, etc. Therefore, this paper aims to elaborate this issue and present a temporal constraints formation for SLA negotiation framework. This framework is proposed in the context of service selection and SLA negotiation. It provides the foundation towards the dynamic formation. This paper also demonstrates the initial approach of temporal constraints formation.

Keywords

Towards, dynamic, formation, temporal, constraints, for, service, level, agreements, negotiation

Disciplines

Physical Sciences and Mathematics

Publication Details

Ismail, A., Yan, J. & Shen, J. (2009). Towards dynamic formation of temporal constraints for the service level agreements negotiation. IEEE International Conference on Service-Oriented Computing and Applications (SOCA 2009) (pp. 72-79). New York, USA: IEEE.

Towards Dynamic Formation of Temporal Constraints for The Service Level Agreements Negotiation

Azlan Ismail, Jun Yan, Jun Shen
School of Information Systems and Technology,
University of Wollongong,
NSW, 2522, Australia
{*ai423, jyan, jshen*}*@uow.edu.au*

***Abstract*—SLAs play an important role for the QoS-driven service composition. Meanwhile, the temporal constraints are one of the main elements in the management of SLA especially in specifying the validity period of the QoS offers. In practice, the temporal constraints should be generated dynamically by taking the resource capability of the provider into account. The generation should consider various parameters that influence the resource capability such as the expected duration of the required Web service, the amount of current utilization, the amount of available resources, the number of required time slots, etc. Therefore, this paper aims to elaborate this issue and present a temporal constraints formation for SLA negotiation framework. This framework is proposed in the context of service selection and SLA negotiation. It provides the foundation towards the dynamic formation. This paper also demonstrates the initial approach of temporal constraints formation.**

***Keywords*-negotiation; resource constraint; Service Level Agreements; temporal constraint;**

I. Introduction

Web services are self-contained modules - deployed over standard middleware platforms - that can be described, published, located, orchestrated, and programmed using XML based technologies over a network [1]. With the service-oriented paradigm [2], the business processes can be designed and implemented as composite Web services consisting of services from different service providers. Service composition is one of the main concerns in assembling these Web services which can be orchestrated in various compositional structures such as sequential or parallel. In the Web service technology, the composition structures can be specified using composition languages, e.g., WS-BPEL [3].

One of the core challenges towards composing Web services is to ensure the selected Web services collectively satisfy the composition requirements. With the QoS-driven concern [4] [5], the satisfaction is determined based on the aggregation of multiple QoS attributes [6]. Furthermore, the selection decision may also require some objective values to be optimized.

For ensuring the reliability of the composite Web services, the relationship between user and each provider that participates in the composition has to be governed. This is done through a bilateral contract known as Service Level Agreements(SLAs). A SLA document can contain various agreement terms including the duration of the agreement, the QoS values and other relevant policies. In the Web service technology, the SLA document can be specified using the SLA frameworks e.g., WSLA [7] and WS-Agreement [8].

The establishment of SLA is conducted through a process known as negotiation. This process contains two parties where one of them will initiate the negotiation, in this case, the user. As the initiator, the user will send a negotiation request to the provider. Meanwhile the provider will respond with an offer. This negotiation will be implemented in several interactions until both sides achieve the mutually satisfied negotiation which results in the bilateral contract or simply the SLA contract.

In practice, the QoS values that are specified in the offer can be attached with one or more temporal constraints [9]. The temporal constraints may enforce the condition of when the offered QoS value is valid. This is important since the Web service might be dealing with a huge workload at certain period of time. Therefore, the provider cannot guarantee similar quality of service level at all time.

In this work, we perceive the temporal constraints as two constraints [10]; given as start after time constraint, SA and finish before time constraint, FB. SA constraint is used to restrict the validity of QoS value from a time point. Meanwhile, FB constraint is used to restrict the validity of QoS value until a time point. For instance, a provider might give an offer such as a response time less than 1 minute for the invocation time after 9am and the invocation time before 10am. This means, the invocation of the Web service might take longer time before 9am or after 10am.

In this work, we are concerned with the formation of these temporal constraints which should be implemented dynamically. This implies the generation of temporal constraints that is automatically executed and able to consider various information to decide for the temporal constraints value. The generation may take several parameters such as the expected duration and utilization of the required Web service, the amount of current utilization, the amount of available

978-1-4244-5299-6/09/$26.00 (c)2009 IEEE

resource, etc. This dynamic capability is very useful in supporting the automated negotiation process. Furthermore, it can assist the provider in forming the customized offer for the particular user.

To address this issue, this paper begins the discussion with a framework of service selection and SLA negotiation in Section 2. It provides the understanding from the user perspective. In Section 3, we present the framework of SLA formation from a single provider perspective. The discussion set the direction of our work. We then take the first step to model the required information for the temporal constraints formation in Section 4. In Section 5, we present the initial approach to demonstrate how the temporal constraints can be generated. Section 6 remarks major related work, followed by conclusions and future work in section 7.

II. A Framework of Service Selection and SLA Negotiation

In this section, we present the framework of service selection and SLA negotiation to support the service composition. We then elaborate how the framework can assist in dealing with the temporal constraints issue. The explanation of the SLA negotiation provides the understanding on the basic requirements for the temporal constraints formation.

A. The Framework Overview

The framework shown in fig. (1) represents the middle entity that we named as service composition broker or simply broker that assists the user in performing two main processes; service selection and SLA negotiation process.

The goal of the service selection is to select the right Web services [4] [5]. In this work, the right selection is determined based on the information given in the SLA contracts. The main goal of the SLA negotiation is to establish a set of acceptable SLA contracts with the providers. A SLA contract can contain various agreement terms and aspects. In this paper, we focus on the QoS values that are attached with temporal constraints.

The main function of each component in the service selection is identified as follows:

- Selector - The main component that integrates sub processes to enable the service selection.
- Composition model - The component that provides the composition plan which includes the process behavior of the entire composition.
- QoS estimation model - The component that estimates the satisfaction of QoS values against the time-related composition requirements as collectively.
- Consistency model - The component that determines the consistency of the attached temporal constraints between the SLA contracts.
- Impact analysis model - The component that determines which of the SLA contracts needs to be renegotiated along with the suggestion of the renegotiation requirement.

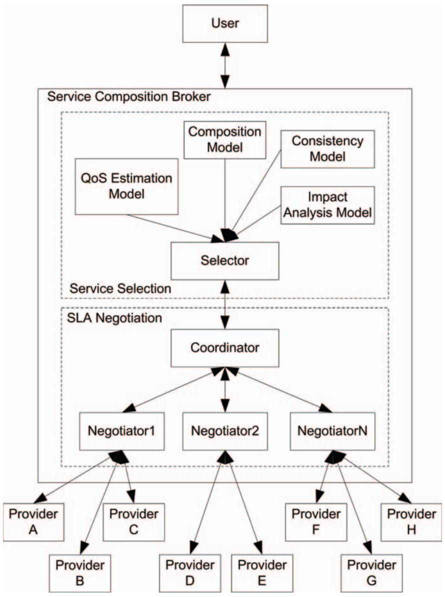

Figure 1. The Main Components of Sevice Composition Broker

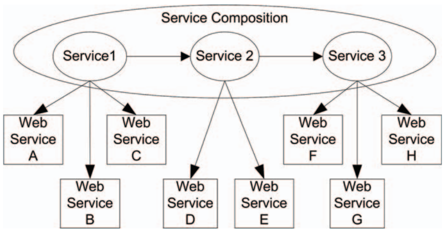

Figure 2. Example of Service Selection for Service Composition

The main function of each component in the SLA negotiation is identified as follows:

- Coordinator - The main component that manages the execution of a set of negotiators.
- Negotiator - The component that establishes the acceptable SLA contracts from a group of providers with similar functionality but possibly with different QoS values.

B. Service Selection with Temporal Constraints Issues

The illustration in fig. (2) shows the role of the service selection in the service composition. Each service in the composition has a set of preferred Web services that have similar functionality but possibly different QoS values with temporal constraints. Thus the service selection will select the right Web services based on the information given in the SLA contracts.

In this paper, the decisions involve in supporting the right selection are identified as follows:

- To determine the initial set of potential Web service from each service group.
- To determine the potential Web services that are consistent between each others based on the SLA contracts.
- To determine the potential Web services that collectively satisfy the time-related composition requirement.
- To determine the potential Web services that need to be renegotiated for their SLA contracts.

For enabling these decisions, the service selection namely the *selector* requires the basic information which is the QoS values and the temporal constraints. This information is retrieved when the negotiation results in a set of acceptable SLA contracts of Web services for each service.

When the acceptable SLA contracts are retrieved, the *selector* requires an initial set of potential Web services for the entire composition. This is determined by evaluating the QoS values and the attached temporal constraints given in the sets of acceptable SLA contracts. The Web service with the highest potential value is chosen as the potential Web service from each service group.

When there are some potential Web services, the *selector* requires the *consistency model* to support the determination of consistent combination [11]. In general, this model will evaluate every combination of potential Web services by determining the consistency between their temporal constraints [12] [13]. This model results in either of these outcomes; consistent combination or inconsistent combination.

When the consistent combination is identified between the potential Web services, the *selector* requires the *QoS estimation model* to support the determination of satisfactory combination. In general, this model will compute the time-related QoS values for the entire composition based on the workflow pattern [14]. Together with the results obtained from the *consistency model*, this model results in either of these outcomes; consistent with satisfactory combination or consistent with unsatisfactory combination.

When the inconsistent combination is identified between the potential Web services, the selector *requires* the *impact analysis model* to support the determination for the renegotiation. In general, this model will compute the negative impact value for every potential Web service in the composition. The potential Web service with the negative impact value beyond a certain limit will be considered for the renegotiation of its SLA contract. Furthermore, the model will also generate an additional requirement for the renegotiation purpose.

C. SLA Negotiation

The role of SLA negotiation for the service selection can be identified as follows:

- to provide the acceptable SLA contracts of each service group.
- to provide the renegotiated SLA contract of the respective Web service.

The negotiation towards providing the acceptable SLA contracts begins with the decomposition of composition requirements. The decomposition task results in individual requirement which may contain information such as the QoS constraints, temporal constraints, etc. Each individual requirement represents each service group.

The *coordinator* will initiate and manage several *negotiators* based on the number of service group. These *negotiators* will be executed concurrently. Each *negotiator* will use this individual requirement to guide the negotiation process with a group of providers.

Herein, the negotiation strategy applies to several interactions. Each interaction will result in an offer from the provider. The decision of accepting the offer is based on the one that satisfy the individual requirement.

Meanwhile, the negotiation towards providing the renegotiated SLA contract is initiated when the renegotiated Web service has been determined. Therefore, the *coordinator* will initiate and manage a set of *negotiators* for the respective providers based on the number of Web service to be renegotiated. Each *negotiator* will use the renegotiated requirement to guide the negotiation process with the respective providers.

III. A Framework of Temporal Constraints Formation for SLA negotiation

In this section, we present the framework of temporal constraints formation from the perspective of a single provider. This framework provides the foundation towards the dynamic formation.

A. The Framework Overview

The framework shown in fig. (3) represents the components that are needed at the provider side which can be divided into two main processes; SLA negotiation process and the temporal constraints formation.

The main goal of the SLA negotiation at the provider side is to support the establishment of SLA contract by providing the required offer. Meanwhile, the main goal of the temporal constraints formation is to generate the temporal constraints value that reflects the provider capability.

The main function of each component in the temporal constraints formation is identified as follows:

- Generator - The main component that integrates sub processes to enable the temporal constraints formation.
- QoS model - The component that provides the expected QoS value for the required Web service.
- Utilization model - The component that provides the expected utilization value for the required Web service.
- Assignment model - The component that decides the right assignment for the job of the required Web service into the job scheduling.

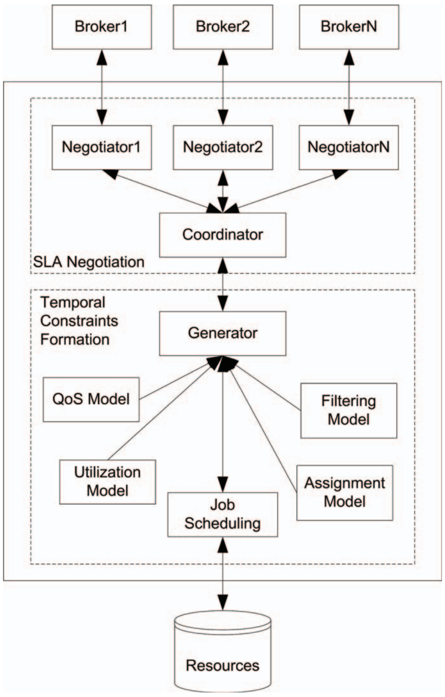

Figure 3. The Main Components of a Single Provider for Enabling SLA Formation

- Filtering model - The component that determines the potential time slots from the job scheduling for the required Web service.
- Job scheduling - The component that provides the resource information specifically the jobs that have been assigned to the schedule.

The main function of each component in the SLA negotiation at the provider side is identified as follows:

- Coordinator - The main component that manages the execution of a set of negotiators.
- Negotiator - The component that establishes the acceptable SLA contracts with the required negotiator from a broker.

B. Temporal Constraints Issues

Temporal constraints are one of the important elements in defining the QoS offer. The provider needs these constraints to express its capability in providing the QoS values within certain period of time. In other words, the temporal constraints are part of the providers requirements towards committing itself to a service composition. This is important since the provider needs to prevent itself from over committing which may end with unguaranteed situation. As a result, the provider has to face with a penalty and its reputation can be affected.

Therefore, this framework provides the foundation towards dynamically generating the temporal constraints value. As there are several types of temporal constraints, we focus on generating two values of the temporal constraints; given as start after time constraint, SA and finish before time constraint, FB.

In a simple model, the temporal constraints can be specified as follows:

$$\forall x, \quad x = y \Rightarrow (T \geq SA) \wedge (T \leq FB)$$

In this rule, the x refers to the QoS parameter, the y refers to the specified value and the T refers to the actual invocation time.

C. The Formation of Temporal Constraints

The formation of temporal constraints in this framework will be initiated upon receiving a set of negotiation requests. In this case, the *generator* determines the ordering of these requests. The simple decision is to take the arrival time value.

In generating the temporal constraints value, the *generator* generally requires two types of information, given as:

- External information - This information is given by the negotiation request and the negotiation component of the provider side.
- Internal information - This information is given by the QoS model, utilization model and the job scheduling.

The details of these information will be discussed in the next section.

As the *generator*, it aims to determine the right value for SA and FB. The right value can be determined when the required job is able to be assigned into the *job scheduling* effectively. The required job refers to the required Web service. Thus, both terms will be used interchangeably. To achieve the assignment of the job, it needs two important decisions:

- determine the potential time slots for the job
- determine the right time slots for the job

The *generator* will require the *filtering model* to determine the potential time slots for the job. The potential time slots can be filtered by certain strategy which can be any or the combination of the following:

- Filtering the time slots based on the QoS characteristic of the job.
- Filtering the time slots based on the utilization characteristic of the job.
- Filtering the time slots based on the individual requirement given by the broker.

Implementing this model can provide the scope for assigning the job. Furthermore, the outcome of this model may be reused for another negotiation request with similar required job. For some situations, the filtering process may results in the right time slots for the job. If this occur, then the time slots are used to determine the temporal constraints value.

When the potential time slots are retrieved, the *generator* will require the *assignment model* to determine the right time slots for the job. The right time slots should be determined by focusing on certain strategy which are as follows:

- Optimizing the provider objectives - The provider may emphasize on the resource environment or some business strategies. The example of the resource concern could be achieving the utilization balance between time slots. The example of the business strategy could be increasing the profit by maximizing the number of jobs to be served.
- Optimizing the user objectives - The user will emphasize on the required Web service. For example, requiring the Web service with earliest finish before time constraint.
- Optimizing both objectives at the same time.

D. SLA Negotiation

The provider also needs the negotiation capability in order to establish the SLA negotiation with the broker. The aim of the negotiation is to respond with an offer for each negotiation request.

The SLA negotiation at the provider side begins when receiving one or more negotiation requests either from similar or different brokers. For this reason, a set of *negotiators* will be initiated to handle the requests. Some *negotiators* will be executed concurrently and some might be executed at different time. Obviously, there is a limit of the number of the *negotiator* that can be executed concurrently. Furthermore, each *negotiator* will record the arrival time of the request. This information can be useful especially in determining the priority of the request.

The execution of these *negotiators* will be controlled by the *coordinator*. The coordinator may pass the requests to the *generator* either one at a time or a group of request. In return, the *coordinator* demands an offer from the *generator* as a response to the broker.

The interaction between the provider and the broker will be executed iteratively. It may terminate when the mutually satisfied negotiation has been reached or vice versa.

IV. Required Information

In this section, we present the information that is required for supporting and enabling the formation of the temporal constraints. The basic information is given by these three main components; the coordinator, the job scheduling, the QoS model and the utilization model.

At certain period of time, there will be a set of negotiation requests received by the provider. The negotiation process namely the coordinator will pass these requests given by G with some additional information to the generator. Therefore, for each element, given as $g_i \in G$, it contains a set of information, given as (S, IR, R, A) which are explained as follows:

- Requested Web service, S - It refers to the Web service that the broker is interested in getting the offer. Furthermore, this information is needed for the generator to determine the jobs related to the Web service.
- Individual requirement, IR - It is a set of QoS requirements that the broker asks the provider to take into consideration in generating the offer. The requirements can be some of the followings:
  - The temporal constraints that restrict the time-related behavior of the requested Web service. Some of the temporal constraints could be a deadline constraint, the duration constraint [12], the bound constraint [15], etc. For instance, assume a deadline constraint $dc \in IR$ is given on the requested Web service, thus, the constraint can be presented as follows:

    $$FB \leq dc$$

    In this example FB refers to the end time to be given in the offer.
  - Other related requirements for the sake of business benefits. For instance, the broker is willing to accept a Web service with longer response time as long as the price is low.
- Negotiation Objective, R - It is used to determine the objective of the negotiation which can be either requiring a new set of offer or adjusting the existing offer.
- Arrival time of the request, A - It refers to the time that the negotiation request arrived at the provider side. This information can be used by the generator for supporting some decision such as to determine the priority of the request, to determine the time slots that should be taken into account, to group certain requests, etc.

The generator also requires the information from the job scheduling J which contains two sets (L, RC) where L is a set of finite time slots and RC is a set of resource constraints. Each $l_i \in L$ holds six elements given as $l_i = (D, B, E, IU, U, O)$. Each of this element is explained as follows:

- Duration, D - It refers to the duration value of a time slot, given as $D(l_i)$.
- Begin time, B - It is time point refers to the begin time of a time slot, given as $B(l_i)$.
- End time, E - It is a time point that refers to the end time of a time slot, given as $E(l_i)$.
- Initial utilization, IU - It refers to the amount of utilization of jobs that have assigned into a time slot, given as $IU(l_i)$.
- Utilization limit, U - It refers to the maximum amount of utilization per time slot, given as $U(l_i)$.

- Job list, O - It refers to the jobs that have been assigned to the slot, given as $O(l_i)$.

Meanwhile, the resource constraints, RC may contain the following constraints:

- Utilization constraint per time slot - Assume there is a new amount of utilization to be assigned on a time slot l_i given as y, thus this constraint can be presented as follows:
$$IU(l_i) + y \leq U(l_i)$$
- Duration constraint per time slot - Assume there is a new amount of duration to be assigned on a time slot l_i given as x, thus this constraint can be presented as follows:
$$x \leq D(l_i)$$

Furthermore, the generator will need another two models namely:

- The QoS model QM to provide the predefined QoS values for the required Web service, given as $QM(S)$.
- The utilization model UM to provide the predefined utilization value for the required Web service, given as $UM(S)$.

V. The Initial Approach

In this section, we focus on the generation of temporal constraints value upon receiving the initial negotiation request from one broker. We make two initial assumptions namely the request is the first negotiation attempt from a broker and there is only one required job to be assigned.

For enabling the generation, a set of information is needed, given as $(G, J, QM(S), UM(S))$.However, this initial approach will not use every sub elements in this set.

With the given information and the assumption, the *generator* can perform the two important decisions namely to find the potential time slots for the job and to find the right time slots for the job.

The *filtering* is required to find the potential time slots. The filtering strategy will be based as follows:

- to find the time slots that are greater than the time defined in the individual requirement.
- to find the time slots that have sufficient amount of utilization to cater the job. The sufficient amount of utilization refers to the total amount of utilization (after considering the utilization value of the required job) that does not violate the utilization limit.

Assuming the expected duration value of the required job may be sufficient for one slot, the decision can be modeled as follows:

$$\forall\ l_i \in L,\ (B(l_i) \geq IR(G)) \wedge (UM(S) + IU(l_i)) < U(l_i) \tag{1}$$

Implementing rule (1) will result in a new set of potential time slots, given as PL.

Definition of Potential Time Slots Set, $\{PL\}$: It contains a set of time slots given as $(l_k, ..., l_m)$ which are filtered from L. There is a possibility where $|PL| = 1$ which means the execution only found 1 time slot. If this occurs, then this slot is decided as the right slot for assigning the new job. If $|PL| = 0$, this means there is no single time slot that has sufficient amount of the utilization.

The *assignment* is needed for finding the right time slots. Assuming $|PL| > 1$, we can further decide which of them is the right time slot for the required job. The decision is based on finding the time slot that has the most available space or the minimum total utilization if the job is to be assigned. The decision can be modeled as follows:

$$\max_{l_i \in PL} (U(l_i) - IU(l_i) - UM(S)) \tag{2}$$

Implementing rule (2) may result in a **solution**. Given $l_x \in PL$ is the right time slot, SA value is determined by $B(l_x)$ and FB value is determined by $E(l_x)$.

Let assume the duration of the required job will require at least k number of time slots, the *filtering* decision can be modeled based on rule (1). However, it results in different set of potential time slots which we named as filtered time slots, given as FL.

Definition of Filtered Time Slots Set, $\{FL\}$: It contains several sets of time slots set, given as $(\{ts_1\}, ..., \{ts_n\})$ where each $|ts_i| = k$. There is a possibility where $|FL| = 1$ which means the execution only found 1 set of time slots. If this occurs, then this set is decided as the right slot for assigning the required job. If $|FL| = 0$, this means there is no available time for the required job.

Assuming $|FL| > 1$, the *assignment* can be implemented to decide which of them is the right slot for the required job. The decision is based on finding the set of time slots that have the most available space or the minimum total utilization if the job is to be assigned. The decision can be modeled as follows:

$$\max_{l_j \in ts_i \in FL} (\sum_{j=m}^{k} (U(l_j) - IU(l_j) - UM(S))) \tag{3}$$

Implementing rule (3) may result in a **solution**. Given $ts_x \in FL$ is the right set of time slots, where $l_{start} \in ts_x$ represents the first slot and $l_{end} \in ts_x$ represents the last slot, SA value is determined by $B(l_{start})$ and FB value is determined by $E(l_{end})$.

Once the right time slots are found, the actual assignment is made by updating the utilization value and the job list of the respective time slots, given as $IU(l_i)$ and $O(l_i)$.

A. Illustration

For the sake of illustration, assuming we have a negotiation request received at 9am. Thus, the information of expected utilization is received from 9am as illustrated in

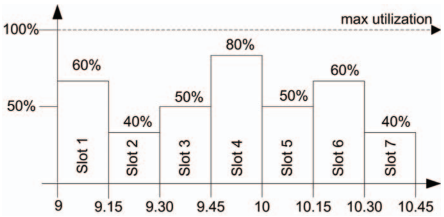

Figure 4. The Utilization Information

fig (4). Assume there is a negotiation requirement that asks for an offer of a Web service that is required to operate at any time starting from 9.40am. Furthermore, the expected utilization for the requested Web service is given as 30% per time slot and the execution time is given as 5 minutes.

Therefore, the provider will first decide all the potential time slots based on rule (1). This results in three potential slots which are *slot 5*, *slot 6* and *slot 7*. Obviously, *slot 1*, *slot 2* and *slot 3* are not considered because their begin time is less than the given time which is 9.40am. Meanwhile, *slot 4* is not considered because its total amount of utilization will violate the maximum utilization if the Web service is to be assigned.

The next step is to decide which time slot is the best slot to assign the job based on rule (2). This results in *slot 7* since it will return the maximum available space if the Web service is to be assigned.

Therefore, the provider will respond to the broker with the execution time of 5 minutes which can be guaranteed with two temporal constraints given as 10.30am for SA and 10.45am for FB.

VI. Related Works

In this section we remark some of the existing works related to the temporal constraints as time-related requirements in the Web service environment.

In the Web service composition, the work by [16] addressed the issue of specifying and analyzing the temporal constraints. The timed automata and duration calculus were used to formally represent the temporal properties. Then several analysis were proposed to check the satisfaction of the composition against the temporal requirements.

In the context of interaction protocol between two Web services, the work by [17] addressed the issue of specifying and verifying temporal constraints specification in OWL-S. The temporal constraints were presented using the time ontology. Then, an extension of Petri Nets was proposed and used to verify the temporal constraints.

The work by [18] addressed the issues of capturing and analyzing the temporal constraints involves in the interaction between two Web services. The timed automata was used to express the temporal constraints. Then, the interactions between Web services were analyzed to determine their compatibility.

In [19], the authors addressed the issue of temporal constraints in Web service procurement which aimed to find the right service offers for the service demands. The temporal constraints were presented using Quality Requirement Language. By utilizing the constraint programming, the temporal constraints were checked for their consistency and conformance.

In the context of SLA, the work by [20] addressed the issue of expressing the temporal constraints either non-periodic or periodic in Service Level Agreements specification, WS-Agreement [8]. They proposed a temporal schema as extension of WS-Agreement.

In conclusion, most of the existing works focus on the issue of specifying and analyzing the temporal constraints. In this work, we are concerned with the formation of temporal constraints by considering the resource capability.

VII. Conclusions and Future Work

In this work, we have discussed the significant of generating the temporal constraints dynamically. Then, a framework for enabling the temporal constraints formation is proposed. Several components are introduced to support the generation of the temporal constraints. The generation will results in two important values namely SA and FB. We also have identified the sets of required information and demonstrate some initial works.

From the broker perspective, it has to deal with temporal constraints if they are attached to the QoS offer. Thus, we have proposed several components, in particular the consistency model, the estimation model and the impact analysis model to support the service selection with temporal constraints concern.

In this work, SLA negotiation plays an important role to govern the relationship between the provider and the user. As the user, the negotiation is needed for obtaining the acceptable offers from several group of providers. Meanwhile for the provider, the negotiation is required for providing the offers to multiple brokers.

Our future work in the context of dynamic temporal constraints formation will extend the initial approach in the following aspects;

- considering different kind of decisions from the provider and the user objectives.
- considering the adjustment of the temporal constraints value.
- developing and evaluating the approach with the SLA negotiation.
- investigating the requirement for dynamically generating QoS values with resource concern.

We will also plan to extend our work on the broker perspective, in particular the impact analysis model since it has a close relation with the adjustment of the temporal constraints.

Acknowledgment

This research is partly supported by the Australian Research Council Discovery Project Scheme under grant number DP0663841 and University of Wollongong Small Research Grant.

References

[1] M. P. Papazoglou, *Web Services: Principles and Technology*. Pearson, Prentice Hall, 2008.

[2] M. P. Papazoglou, P. Traverso, S. Dustdar, and F. Leymann, “Service-oriented computing: State of the art and research challenges,” *Computer*, vol. 40, no. 11, pp. 38–45, 2007.

[3] D. Jordan, J. Evdemon, A. Alves, A. Arkin, et al., “Web services business process execution language version 2.0,” OASIS, Tech. Rep., 2007.

[4] L. Zeng, B. Benatallah, A. H. Ngu, M. Dumas, et al., “Qos-aware middleware for web services composition,” *IEEE Transactions on Software Engineering*, vol. 30, no. 5, pp. 311–327, 2004.

[5] T. Yu and K.-J. Lin, “Service selection algorithms for composing complex services with multiple qos constraints,” in *Service-Oriented Computing - ICSOC 2005*, 2005, pp. 130–143.

[6] M. C. Jaeger, G. Rojec-Goldmann, and G. Muhl, “Qos aggregation for web service composition using workflow patterns,” in *Proceedings of Eighth IEEE International Enterprise Distributed Object Computing Conference - EDOC 2004*, 2004, pp. 149–159.

[7] H. Ludwig, A. Keller, A. Dan, R. P. King, and R. Franck, “Web service level agreement (wsla) language specification,” IBM Corporation, Tech. Rep., 2003.

[8] A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, et al., “Web services agreement specification (ws-agreement),” Open Grid Forum, Tech. Rep., 2007.

[9] A. R.-C. C. Muller and P. Fernandez, “Temporal-awareness in slas: Why should we be concerned?” in *Service-Oriented Computing ICSOC 2007*, 2009, pp. 165–173.

[10] A. Ismail, J. Yan, and J. Shen, “Dynamic service selection for service composition with time constraints,” in *Proceedings of the 20th Australian Software Engineering Conference - ASWEC 2009*, 2009, pp. 183–190.

[11] A. Ismail, J. Yan, and J. Shen, “Verification of composite services with temporal consistency checking and temporal satisfaction estimation,” in *Proceedings of the 10th International Conference on Web Information Systems Engineering - WISE 2009*, 2009.

[12] O. Marjanovic, “Dynamic verification of temporal constraints in production workflows,” in *Proceedings of 11th Australasian Database Conference - ADC 2000*, 2000, pp. 74–81.

[13] J. Chen and Y. Yang, “Temporal dependency for dynamic verification of temporal constraints in workflow systems,” in *Grid and Cooperative Computing GCC 2004*, 2004, pp. 1005–1008.

[14] W. M. P. van der Aalst, A. H. M. T. Hofstede, B. Kiepuszewski, and A. P. Barros, “Workflow patterns,” *Distributed and Parallel Databases*, vol. 14, no. 1, pp. 5–51, 2003.

[15] J. Eder, E. Panagos, and M. Rabinovich, “Time constraints in workflow systems,” in *Advanced Information Systems Engineering*, 1999, pp. 286–300.

[16] R. Kazhamiakin, P. Pandya, and M. Pistore, “Representation, verification, and computation of timed properties in web,” in *Proceedings of the International Conference on Web Services - ICWS ’06*, 2006, pp. 497–504.

[17] R. Liu, G. Dai, C. Hu, and C. Zhao, “A verification method for temporal consistency of service flow,” in *Proceedings of 32nd Annual IEEE International Computer Software and Applications, 2008. COMPSAC ’08*, 2008, pp. 1187–1192.

[18] N. Guermouche, O. Perrin, and C. Ringeissen, “Timed specification for web services compatibility analysis,” *Journal of Electronic Notes in Theoretical Computer Science*, vol. 200, no. 3, pp. 155–170, 2008.

[19] O. Martin-Diaz, A. Ruiz-Cortes, A. Duran, and C. Muller, “An approach to temporal-aware procurement of web services,” in *Service-Oriented Computing - ICSOC 2005*, 2005, pp. 170–184.

[20] C. Muller, O. Martin-Diaz, A. Ruiz-Cortes, M. Resinas, and P. Fernandez, “Improving temporal-awareness of ws-agreement,” in *Service-Oriented Computing ICSOC 2007*, 2008, pp. 193–206.