

# Workflow Composition of Service Level Agreements

M. Brian Blake<sup>1</sup> and David J. Cummings

*Dept. of Computer Science*

*Georgetown University*

*Washington, DC 20057*

*{mb7,djc39}@georgetown.edu*

## Abstract

*Service-oriented architectures enable an environment where businesses can expose services for use by their collaborators and their peer organizations. In such an environment, organizations may have long-standing cooperative agreements representing the existing services that they share. Furthermore, they may have service level agreements (SLAs) that assure the quality of service standards of the existing services. In an ad-hoc workflow scenario, a business may need to perform real-time composition of the existing services in response to consumer requests. In this work, we suggest that, in parallel to traditional web service composition, the business must also compose the corresponding SLAs of the chosen services to generate a guaranteed service level to the consumer. In this paper, we introduce a process for composing SLAs associated with a workflow of web services. This process results in the discovery of the best composite (i.e. workflow) capability while optimizing the underlying service-level attributes.*

## Keywords

Service level agreements, web services, workflow

## 1. Introduction

A service level agreement, SLA, is a technical contract between two businesses, a producer and a consumer. An SLA captures the agreed-upon terms between these two organizations with respect to quality of service (QoS) and other related concerns. Considering a service-oriented computing environment, capabilities are shared via the implementation of web services exposed by a producer organization. The ultimate goal of service-oriented computing is for consumers to access these shared capabilities, on-demand. However, in cases where businesses have longstanding relationships, such as workflow and supply chain environments, peer companies that share services must be able to assure a level of service to their underlying customers.

New standards, such as the Web Service Level Agreement (WSLA) and Web Service Agreement (WS-Agreement) specifications [11][12] enable SLAs to be associated to groups of web services or even specifically

to an individual web service. These specifications define an eXtensible Markup Language (XML) based data model that can be used to extend the Web Service Description Language (WSDL) documents that traditionally describe the web services. These specifications provide a significant opportunity. Organizations can specify QoS-related concerns in concert with the functionality concerns already captured in the WSDL files. As a result, when a new organization searches for a pertinent web service, the SLA-enhanced WSDL file can be used to determine the appropriateness of the service to meet the required business need. Furthermore organizations can use the SLA-enhanced WSDL file to negotiate the QoS terms.

Although these SLA technologies and specifications present new opportunities for service-oriented business processes, there are a number of significant barriers. When a consumer organization must create a new business capability that requires the workflow composition of multiple web services, then that organization will also need to understand the composite impact of the underlying SLAs. Consequently, in addition to composing web services that are functionally compatible, the organization will need to assure the web services are compatible with regards to their service levels. In addition, the product of all the SLAs for a composition of web services must be within the required threshold of *feasibility* as defined by the end users. As the service-oriented computing paradigm increases in popularity, the consumer will have the option of many similar services that may meet a particular requirement. As such, the composition of web services that is most efficient for a particular business purpose will rest on the organization's ability to understand and optimize the corresponding composition of SLAs.

To deal with the aforementioned issues, we introduce the phrase, *workflow composition of SLAs*. Our approach suggests the multi-dimensional optimization of service-oriented QoS attributes to realize an optimized workflow of web services. While the notions of multi-dimensional analysis, optimization, dynamic programming are not new, in this work, we identify the SLA-based attributes that must undergo this analysis. Furthermore we develop a set of principles and the associated process that utilizes the SLA information to enable the optimization process.

<sup>1</sup> For identification purposes, the author also has an affiliation as the Consulting Director of Services Computing and Research for Cleared Solutions Inc, Leesburg, Virginia

In this work, we investigate several research issues relevant to the integration of web services-based workflow:

1. *What SLA measures and principles best support QoS-based web service composition?*
2. *Given multiple web services workflows that represent perhaps duplicate choices of capabilities for a particular requirement, can the SLAs be used as decision support when selecting the choice that is most efficient for a business need?*
3. *Given a group of SLAs and knowledge about current consumer capabilities needs, can an on-demand request be analyzed and guaranteed at a certain service level?*

The paper proceeds in the following section with a discussion of related work. In the subsequent section, the SLA-based QoS assessment values are discussed. In the following section, the technical details of the SLA composition process are introduced, and we show how this approach is integrated with the general web service composition routine. Finally, we evaluate the performance of the integrated process.

## 2. Related Work

There are many related projects that investigate the use of SLAs. Some projects characterize SLA approaches to specific domains, such as military, database management, or information systems [3][6][9]. There is also a large body of work that attempts to automate the management and negotiation of SLAs[4][7].

Of close relationship, Alipio et. al. [1] and Sun et. al. [10] consider an XML-based service level specification approach that is similar to the WS-Agreement approach. Our work also leverages WS-Agreement for providing SLA measures, however the focus here is on composing multiple SLAs while their work focuses on one instance.

Skene et. al. [8] also uses an XML-based approach for specifying SLA measures. Similar to our work, they describe the importance of composing SLAs. Their emphasis is on compatibility between a user requirements and provider constraints. Their approach suggests a promising model-based approach to assuring the compatibility.

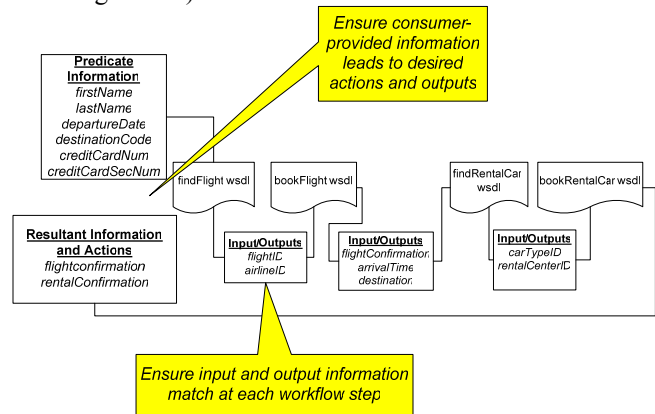
Our work has a close relation to the comprehensive work performed by Zeng [13]. Although both approaches uses QoS measures as an integral part of composition, our approach uses several unique SLA measures that are not QoS-related but rely more on mismatch of services.

In our work, we define specific SLA measures and formally integrate measures across multiple SLAs. We also define a principled process for the composition of SLAs in response to the workflow of web services. The

principles that we identify are suited to integration of web service composition and SLA composition and management. This work expands on related work [2] by concentrating on SLA measures in markup language files as opposed to Unified Modeling Language (UML) models.

## 3. Web Services-based Aspects of SLAs

The typical SLA has a large number of measures and criteria. However, in this work, we attempt to choose the measures that are most closely aligned to web service composition. In fact, many lessons can be learned from the general notion of web service composition. In the general functional notion of web service composition, a basic web services workflow system must ensure that the input information supplied by the consumer ultimately leads to the required actions and outputs required by that consumer. In parallel, the workflow management system must ensure that the predicates and requisites match in each step of the workflow. With respect to the travel reservation scenario as shown in Figure 1, when a user supplies the *predicate information* then the workflow must produce the *resultant information* and actions at its conclusion. In addition, the inputs and outputs must be matched at each step (e.g. between findFlight.wsdl and bookFlight.wsdl).



**Figure 1.** Data-Oriented, Functional Web Service Composition Requirements.

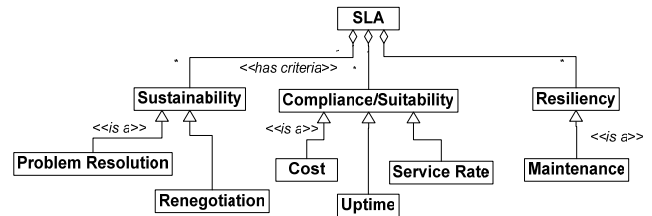
Although the functional web service composition is straight-forward, the assessment criteria associated with SLA terms increases the requirements for composition threefold. We stratify the SLA criteria by introducing three principles associated with composing SLAs, *Compliance (Suitability)*, *Sustainability*, and *Resiliency*. These three principles (illustrated in Figure 2 as a Unified Modeling Language class diagram) are defined below in addition to their underlying SLA measures. Figure 3 puts three principles in perspective of web service composition.

**Compliance (Suitability).** Compliance is the principle that ensures that the consumer receives the requested composite capability at the service level that is required. The functional notion of web service composition (as illustrated in Figure 1) fits within this principle, since the consumer specifies their required outputs of the composition. Considering SLA terms, the composition process must assure that the aggregate *cost*, *uptime*, and *service rate* are compliant with the user requirements. Cost is the sum total price of all services participating in the solution process. Uptime is a guarantee by the service providers that their services will be available a specified percentage of the time per day or month. Finally, service rate is the time it takes to complete the process by adding the response times of each service in the composition.

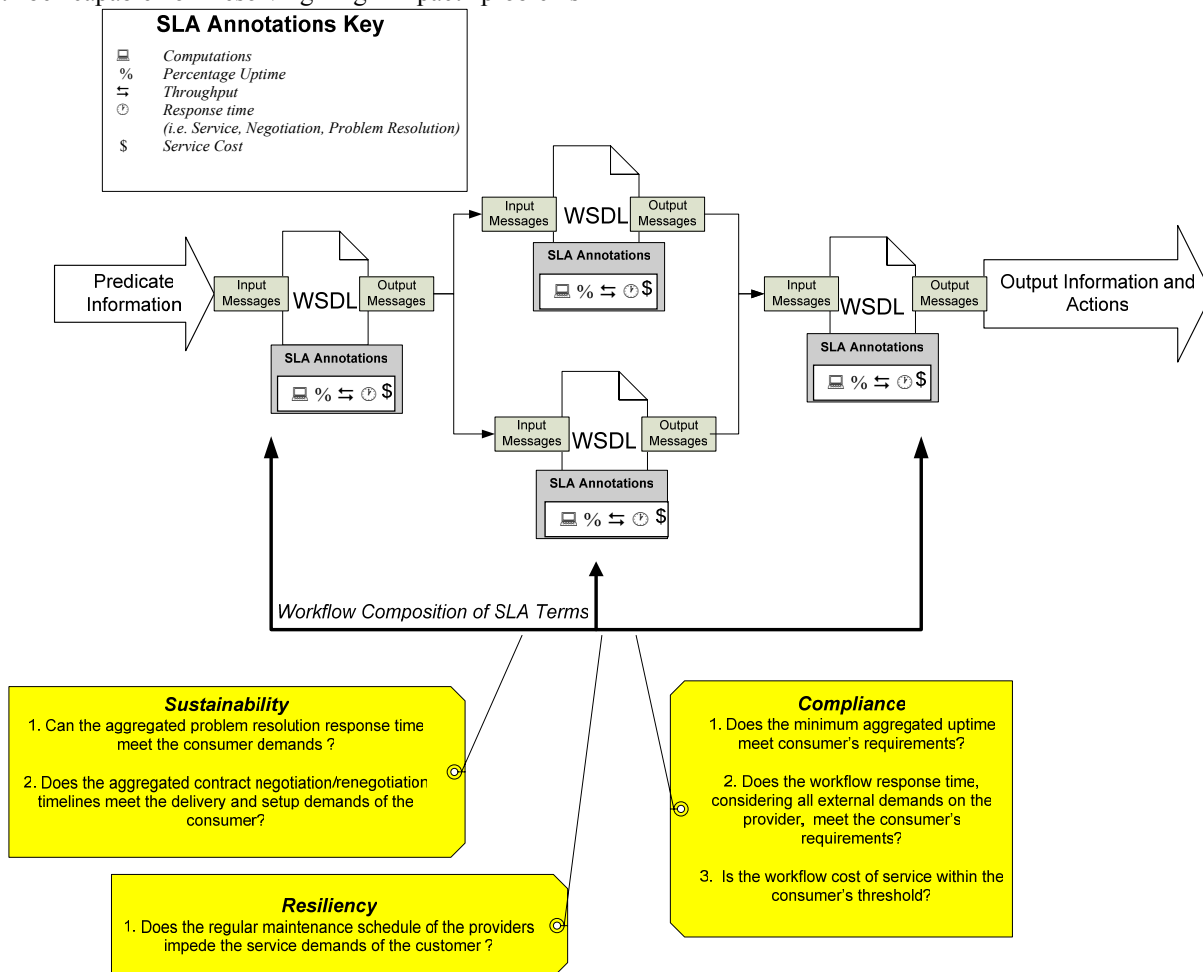
**Sustainability.** Sustainability is the ability to maintain the underlying services in a timely fashion. *Negotiation/re negotiation* and *problem resolution* are strongly correlated to sustainability. A consumer will require assurance that a particular business is capable of agreeing on contract terms (i.e. negotiation/re negotiation) in a timely manner. In addition, the service providers must be capable of resolving high-impact problems

(perhaps identified by the consumer) in a timely manner. Both negotiation and problem resolution times ensure that a consumer can meet the demands of their end users.

**Resiliency.** Resiliency is the principle of a service to perform at high service levels over an extended period of time. If a service is frequently taken off-line for maintenance or if the frequency of updates impedes the predictability of its operation, then that service is not very resilient. Consumers will need adequate notice prior to maintenance downtimes. In addition, resiliency dictates a low frequency of maintenance downtime.



**Figure 2.** A Taxonomy of SLA Measures for Web Services Workflow Composition.



**Figure 3.** Evaluating SLA Terms while Composing Web Services.

#### 4. Generating Composite SLA Measures

When composing web services that are annotated with SLA terms the composite capability must also have aggregate SLA terms. In some cases, aggregating the SLA terms is as straight-forward as adding the measures of each of the dependent services, but in other cases the aggregate measure must be created based on consumer requirements. Figure 3 illustrates the typical questions that arise when attempting to characterize a composite web service based on SLA measures. Since there may be subjectivity with respect to how the aforementioned SLA measures can be aggregated and calculated, we introduce five formal methods for composing the SLA measures with high relevancy to web service composition.

Formally, when a consumer collaborates with a producer within a service-oriented architecture stakeholder capabilities are hosted on a server,  $V$ .  $V$  is the set of servers of all producer and consumers:

$V = \{v_1, v_2, v_3, \dots, v_n\}$ . Each server can be characterized by its performance,  $C_v$ , measured in computations/second and by its uptime percentage  $U_v$ . The throughput,  $T_v$ , of the servers measure in bytes/sec can be further specified for input,  $T_{i_v}$  and output,  $T_{o_v}$ . Finally, a consumer or producer can dictate the pre-notification time,  $P_v$ , that they must inform their constituents prior to downtime associated with server enhancements and repairs. Therefore, a server can be defined as:

$$V = (C_v, U_v, T_{o_v}, T_{i_v}, P_v) \quad (1)$$

A service,  $S$ , can be defined by its required computations,  $C_s$  and by the message length,  $M_s$ , in bytes, that it receives to realize that capability. As such, the set of all services of a SLA,  $A$ , can be defined:  $S_A = \{sa_1, sa_2, \dots, sa_n\}$  and each service can be defined by the tuple:

$$S = (C_s, M_s) \quad (2)$$

An SLA consists of the specific web service-oriented measures, as defined in lower elements in the taxonomy in Figure 2. An SLA for web services workflow composition consists of the uptime,  $U_A$ , specified by the agreement, the allowable service rate,  $\mu_A$ , the specified service response time,  $R_A$ , the subscription cost or price,  $D_A$ , of the service, the maintenance pre-notification time,  $P_A$ , and the expiration/renewal time,  $X_A$ , of the agreement. The SLA can be represented by the tuple:

$$A = (U_A, \mu_A, R_A, D_A, P_A, X_A) \quad (3)$$

When composing a set of SLAs, this approach considers a composite or workflow composition of SLAs defined as  $A_W$ . The composite SLA is calculated by composing the set of all agreements for which the composite SLA is dependent where  $A_D = \{ad_1, ad_2, ad_3, \dots, ad_n\}$ . This calculation must also consider the set of all other SLAs existing on the consumers server that may impact a new composite SLA such that  $A_O = \{ao_1, ao_2, ao_3, \dots, ao_n\}$ . Therefore the composite SLA is a function of the composition of the dependent services while considering other existing services:

$$A_W = COMP(A_D, A_O) \quad (4)$$

##### Uptime

The uptime for the composite SLA is the minimum of:

- The uptime for consumer's server and
- The uptimes agreed in dependent services

The relationship can be shown as:

$$U(A_W) \leq \min(U_V, \min(U(A_D))) \quad (5)$$

##### Service Rate

The time it takes for an individual service to complete its execution is the *task time*. The repeatability of the task time while the service is in commission translates to the *service rate*. For a web service, the service rate is a function of the internet connection, the internal network connection, the hosting hardware, and the software service. The combined service rate is illustrated in Figure 4.

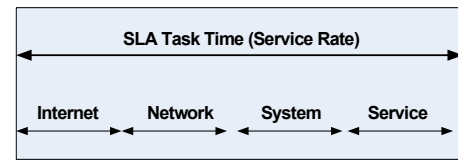


Figure 4. Task Time and Service Rate.

The service rate for this agreement is the minimum of:

- The server throughput divided by the message lengths of service input and output
- The consumer throughput divided by the message lengths of service input and output
- The minimum service rate agreed for the dependent service, minus the sum of the service rates already guaranteed to other consumers. (This is calculated for each dependent service.)

Including the sum of service rates already guaranteed to others is important to avoid the impact of a voluminous load from external operations. Considering the service rate of the consumer,  $J$ , can be calculated as:

$$\mu_J = \frac{T_V}{M_S} \quad (6)$$

and the service rate of all stakeholders,  $I$ , of the consumer is the difference of the service rates of all dependent services and the sum of external services as shown by:

$$\mu_I \leq \mu(A_D) - \text{sum}(\mu(A_o)) \quad (7)$$

then the service rate for the composite SLA is represented as:

$$\mu(A_W) \leq \min(\mu_J, \mu_I) \quad (8)$$

### Maintenance

When a service must be disabled for maintenance, organizations must inform their collaborators. The minimum time of notification before maintenance is calculated as the minimum of:

- The time of notification for this server
- The minimum of all notification times agreed upon for all dependent services

This relationship can be illustrated as:

$$P(A_W) \leq \min(P_V, \min(P(A_D))) \quad (9)$$

### Cost

The cost of using a web service can be aggregated to understand the price of an entire process. The cost of an individual service is the sum of:

- The bandwidth use (frequency times message length) times cost per byte/sec of bandwidth, which is a predefined constant  $BC$ .  $BC$  could be zero if bandwidth costs are negligible to the service provider.
- The computation use (frequency times computations) times cost of computation/sec, which is some predefined constant  $CC$ . As above,  $CC$  could be zero if computation costs are negligible to the provider.
- The sum of the costs of all dependent services.

This results in the following equation:

$$D(A_W) \geq \mu(A_W) * M_S * BC + \mu(A_W) * C_S * CC + \text{sum}(D(A_o)) \quad (10)$$

### Renegotiation

The renegotiation date for this agreement must be after the predefined constant waiting time,  $WT$ , with respect to each other agreement to which this server is a party. If the provider only handles a few services, then they might have a larger waiting time, but one with many services would require less waiting time in order to fit all of its renegotiations on its calendar. This notion of

renegotiation time is illustrated in Figure 5. The equation below must hold for each other agreement  $A_o$  to which the service provider is a party. This test can be shown as:

$$X(A_W) \geq X(A_o) + WT \quad (11)$$

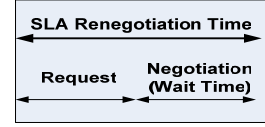


Figure 5. Aspects of Renegotiation Time.

Problem resolution is not formally defined because the calculation is a similar formula as renegotiation. As a definition (shown in Figure 6), problem resolution can be defined as the sum of the time for recognizing the error, the time that it takes for a provider to acknowledge the error, and the actual time for resolving the problem.

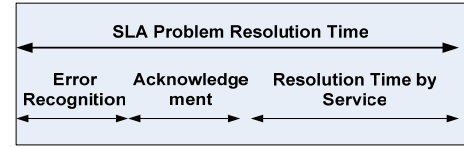
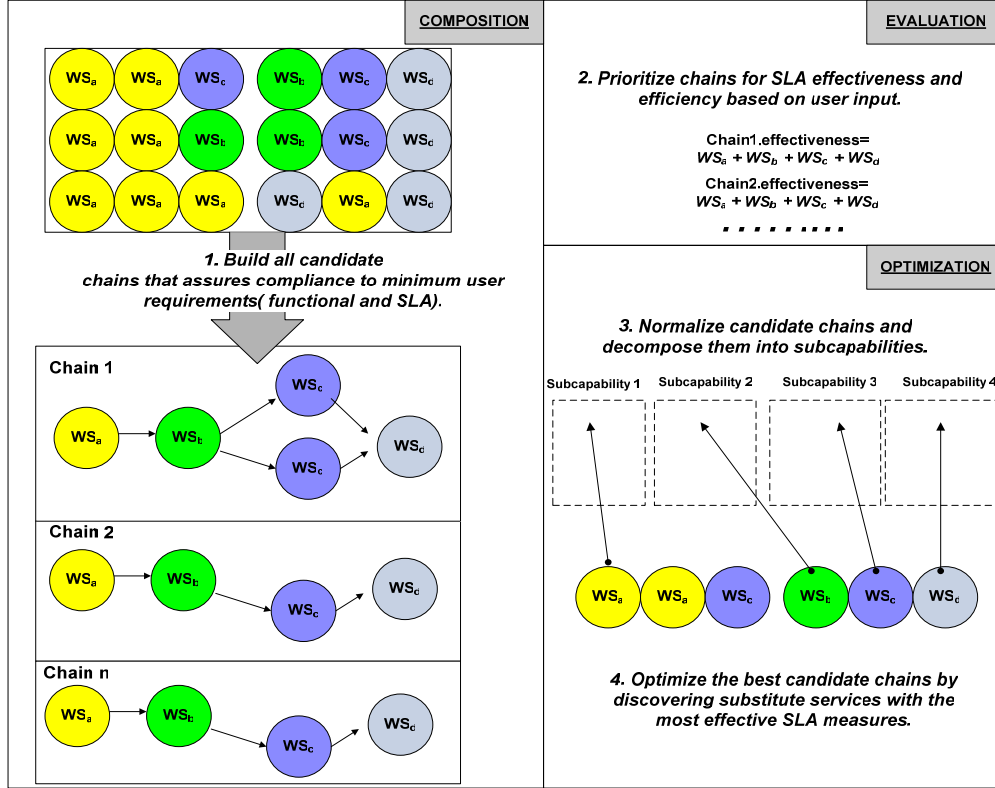


Figure 6. Aspects of Problem Resolution.

## 5. A Process for Integrated Workflow Composition

In this work, we define an integrated process for functional web service composition combined with workflow-based SLA composition. This process can be decomposed into three steps, *composition*, *evaluation*, and *optimization*. These three steps are illustrated in Figure 7. The composition step is similar to the related work in web service composition. Web services are connected via forward-chaining or backward-chaining to match inputs to outputs until the required result is achieved. A variation in this work is that minimum service levels are also validated during this step. For example, if one web service exceeds the minimum requires for an entire chain, then that web service must not be considered in candidate chains. In the evaluation step, user priorities are used to prioritize the list of candidate service chains. Numerous dynamic programming techniques can be used to achieve this step. We present one approach in the following section. Finally, the optimization step attempts to replace services at the sub-process level to generate a “best” service chain. In real-time operations, the composition and evaluation steps are required while the optimization step is best suited for design-time decision support. In this paper, the focus is on these first two steps as we plan to investigate decision support in future investigations.

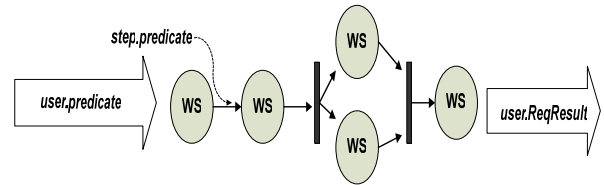


**Figure 7.** Three-Step Process for Integrated SLA Workflow Composition.

### 5.1 Composition: Building Candidate Workflows

In order for SLA measures to be useful for both real-time operations and decision support, the web services workflow generation must integrate traditional web service composition with the SLA composition procedures. The SLA composition procedure must be integrated at each step of the composition and also applied to the workflow as a whole once the full process is generated. We defined the information provided by the user to be the *user.predicate* and the desired outcome to be the *user.results*. The set of information that expands at each step to enable the selection of the service(s) for the next step is defined as the *step.predicate*. At the initiation of a composition routine, the *user.predicate* is equivalent to the *step.predicate*. These relationships are illustrated in Figure 8.

We introduce an integrated procedure that combines standard web service composition with SLA composition. Table 1 shows the pseudo-code of the integrated process. The composition process has a main integrated process, *IntegratedComp()*. The *StepCompose()* process occurs at each step and *WorkflowChk()* process occurs once the required information and actions are realized with the execution of the sequence of web services. The *ComposeSLA()* process is the computation mechanisms that implement the SLA aggregation procedures designed in Section 4. Table 1 shows the pseudo-code of the integrated process.



**Figure 8.** Data Flow in a Composition Routine.

### 5.2 Evaluation: Prioritizing Composite Capabilities using SLAs

In the prior section, candidate service chains are generated that meet the functional and SLA requirements of the user. Nevertheless, at this point, there are still multiple chains that can fulfill a capability. As such, there remains an open requirement to sort the chains based on quality and choose the best chain in the group. Since this approach uses six SLA measures.

In order to prioritize service chains, users are asked to provide a priority,  $Pr$ , for each attribute with respect to their environment, where  $Rr = \{Pr_1, Pr_2, Pr_3, \dots, Pr_n\}$ . The priorities relate to the corresponding set of SLA measures, SLA where  $SLA = \{SLA_1, SLA_2, SLA_3, \dots, SLA_n\}$ . An SLA attribute has the best possible value,  $B$ . Our approach will strongly consider chains that perform favorably with respect to the user's preferences. After evaluation,  $G_S$  and  $G_W$  represent the quality score for the service and workflow, respectively

**Table 1.** Integrated Composition Pseudocode.

<i>IntegratedComp</i> :	Main integrated composition function
<i>StepCompose</i> :	Function that occurs at each step
<i>WorkflowChk</i> :	Process-Level Functional/SLA check
<i>composeSLA</i> :	Function that aggregates SLA measures
<i>PR, RN, CS, :</i>	Problem Resolution, Renegotiation, Cost
<i>SR, UP, MN:</i>	Service Rate, Uptime, Maintenance
<i>Step.Predicate</i> :	Message information per step
<i>ws</i>	WSDL Object with SLA measures
<i>ws.regresults</i> :	Web Service Category (e.g. Business)
<i>service_chain</i> :	Candidate workflow of web services

```

IntegratedCompose
{
  step.predicate = user.predicate
  WHILE (step.predicate != user.regresults)
    { StepCompose() }
  WorkflowChk()
}
StepCompose
{
  FOR ALL candidate ws where
    ws.message ⊆ step.predicate
  THEN Load (ws)
  FOR EACH candidate ws
    IF ((ws.PR ≤ user.PR) || (ws.RN ≤ user.RN) ||
        (ws.CS ≤ user.CS) || (ws.SR ≤ user.SR) ||
        (ws.UP ≤ user.UP) || (ws.MN ≤ user.MN) )
    THEN Drop (ws)
  Update List <service_chain>
  Update List <step.predicate> with ws.outputs
}
WorkflowChk
{
  FOR EACH service_chain
    IF ((chain.outputs ⊆ user.regresults) &&
        (ComposeSLA (ws.PR, ws.RN, ws.CS, ws.UP,
                    ws.MN, ws.SR) < user.SLA))
    THEN ADD (service_chain) to Candidate List
}

```

As such, the quality score for an individual service can be defined by weighing the user's priority with respect to the ratio of the SLA measure to the best possible measure for that attribute. The calculation is defined as:

$$G_s = \sum_{n=0} \left( Pr_n \cdot \frac{SLA_n - B(SLA_n)}{B(SLA_n)} \right) \quad (12)$$

The corresponding quality score for the service chain is:

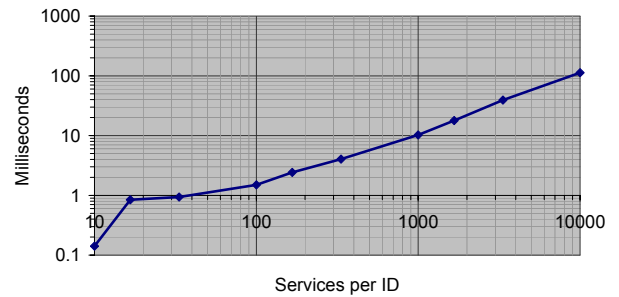
$$G_w = \sum_{m=0} ((G_s)_m) \quad (13)$$

## 6. Performance Evaluation of Integrated Composition Process

In real-time operations, composition will be required to occur within a reasonable response time. Of the three steps in the integrated composition process, the composition and evaluation process will be required to execute in an operational setting. We expect that the optimization sub-process will, at least initially, be limited to design-time decision support settings. As such, it is important to understand the expected response time when manipulating SLA measures for composition and prioritization.

In this work, an initial experiment was performed that evaluates the speed in prioritizing SLA-annotated workflows of web services. We generated random web service objects with unique identification codes. We duplicated identification codes (i.e. ids) such that increasing numbers of services have the same ids. Each web service object was populated with all six SLA measures. Our experimentation searches the repository of web service objects, composes services of the same id, and then prioritizes the resulting chain. This experiment was executed against a repository of sizes from 100 to 100,000 potential services. The workflow size (number of services in a chain) was varied. Figure 9 shows the results of this first experiment. The number of services per chain is varied from 10 to 100,000 services out of a repository of 1,000,000 services. Considering a reasonable workflow of web services should be 100 services or less, a response time of 1.2 milliseconds per service chain is promising.

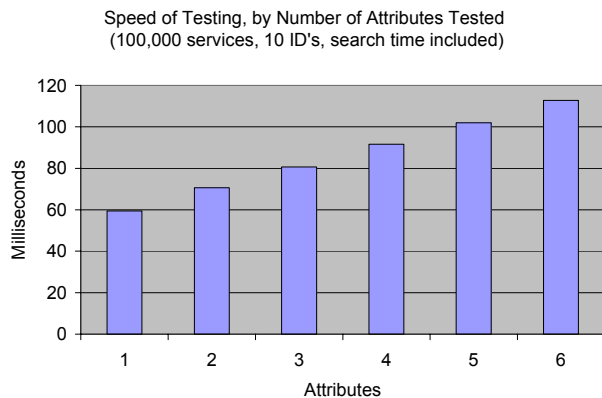
Speed of Testing, by Number of Services per ID  
(constant 100000 services, search time subtracted)



**Figure 9.** The Performance of the SLA Prioritization Function as the Number of Services per Workflow Increase.

In a second experiment, we evaluated the performance by varying the number of SLA attributes that are calculated. In this paper, we suggest six attributes but we expect that other attributes will be required to extend this approach in the future. As such, it is important to

understand the overhead associated with adding a new SLA attribute for real-time operations. Although it is understood that the performance increases as the computing hardware is improved, we are generally interested in how the approach scales as the number of attributes increases in a fixed-size repository. Figure 10 shows the search and calculation time when varying the numbers of attributes calculated. Although this graph has a high concentration of search time (~90%), which is not relevant to the SLA priority calculation overhead, it is clear that the performance degrades favorably (linearly) at about 20 milliseconds per attribute (i.e. the calculation time increases less than 2 milliseconds for the addition of each a new attribute).



**Figure 10.** The Performance Overhead Associated with the Addition of New SLA Attributes.

## 7. Conclusion

In this work, we introduce a method for composing web services that are annotated with SLA measures. The annotated SLA measures are effective in providing decision support when composing the web services. We define a three-step process appropriate for real-time operations and design-time decision support. The innovation of this approach is the identification of SLA measures that are suited for web service composition and management. In addition, we introduce principled methods for calculating those measures and prioritizing them based on user preferences.

In future work, we plan to extend our approach by implement our approach within a real operational setting. We plan to append WS-agreement files to WSDL files and test the approach in a network environment for performance and feasibility. In addition, we plan to extend the description of SLA measures to services that are stored in web service registries. In this way, our approach can be deployed within UDDI registries, and SLA measures can be independently used to annotate businesses and services. As such, UDDI registries can be searched based on user-based SLA criteria.

## 8. Acknowledgements

This work was benefited by the participation of one of primary authors in the Service Level Agreement Technical Exchange Meeting held at The MITRE Corporation on July 2006 in McLean, Virginia. In addition, the service discovery approach/software used in this work was partially funded by the National Science Foundation under award number 0548514.

## 9. References

- [1] Alipio, P., Lima, S., and Carvalho, P., "XML Service Level Specification and Validation", *Proceedings of the 10th IEEE Symposium on Computers and Communications (ISCC 2005)*, pp 975-980, June 2005
- [2] Blake, M.B. "A Lightweight Software Design Process for Web Services Workflows" *4th IEEE International Conference on Web Services (ICWS 2006)*, Sept 2006
- [3] Falkowski, T., Application Service Providing, PhD Dissertation, University of California at Berkeley, USA
- [4] Greenwood, D., Vitaglione, G., Keller, L., Calisti, M., (2006) "Service Level Agreement Management with Adaptive Coordination", In *Proceedings of the International Conference on Networking and Services (ICNS'06)*, July 19-21, 2006, Silicon Valley, USA
- [5] Muller, N.J., "Managing service level agreements" *International Journal of Network Management*, Volume 9 Issue 3 Publisher: John Wiley & Sons, Inc., May 1999
- [6] Reiss, F.R. and Kanungo, T "Satisfying database service level agreements while minimizing cost through storage QoS" In *Proceedings of the IEEE International Conference on Services Computing*, pp13 – 21, July 2005
- [7] Sahai A, Machiraju, V., Sayal, M., A. Moorsel, and Casati, F., Automated SLA Monitoring for Web Services. *Proceedings of the IEEE/IFIP DSOM 2002*. Montreal, Canada Oct. 2002.
- [8] Skene, D. Lamanna and W. Emmerich, Precise Service Level Agreements. In *Proc. of the 26th Int. Conference on Software Engineering*, Edinburgh, UK. pp. 179-188. IEEE Computer Society Press., 2004
- [9] Sorteberg, I. and Kure, O , "The use of service level agreements in tactical military coalition force networks", *IEEE Communications Magazine*, pp 107-114, November 2005
- [10] Sun, W., Xu, Y., and Liu, F., "The role of XML in service level agreements management" In *Proceedings of International Conference on Services Systems and Services Management*, pp 1118 - 1120 June 2005,
- [11] Web Service Agreements (WSAG): [http://www.gridforum.org/Public\\_Comment\\_Docs/Docuents/Oct-2005/WS-AgreementSpecificationDraft050920.pdf](http://www.gridforum.org/Public_Comment_Docs/Docuents/Oct-2005/WS-AgreementSpecificationDraft050920.pdf)
- [12] Web Service Level Agreements (WSLA) (2007): <http://www.research.ibm.com/wsla/>
- [13] Zeng, L., Benatallah, B., Ngu, A. H.H., Dumas, M., Kalagnanam, J., and Chang, H.. QoS-aware Middleware for Web Services Composition. *IEEE Transactions on Software Engineering* 30(5):311-327, May 2004, IEEE Computer Society.