Hierarchical aggregation of Service Level Agreements

Irfan Ul Haq *, Altaf Ahmad Huqqani, Erich Schikuta

Department of Knowledge and Business Engineering, University of Vienna, Rathausstrasse 19/9, 1010 Vienna, Austria

Article info

Article history:
Received 26 April 2010
Accepted 18 January 2011
Available online 21 February 2011

Keywords:
Workflow management
Service Level Agreement
Service composition
Formal model
Grid and cloud Computing

Abstract

IT-based Service Economy requires Service Markets to flourish for the trade of services. A market does not represent a simple buyer-seller relationship, rather it is the culmination point of a complex chain of stake-holders with a hierarchical integration of value along each point in the chain. To enable a Service Economy, Service Markets must be practically realized, which in turn requires an enabling infrastructure to support service value chains and service choreographies resulting from service composition scenarios. In such scenarios, services compose together hierarchically in a producer-consumer manner to form service supply-chains of added value. Service Level Agreements (SLAs) are defined at various levels in this hierarchy to ensure the expected quality of service for different stakeholders. Automation of service composition directly implies the aggregation of their corresponding SLAs. In this paper we elaborate on the requirements of hierarchical aggregation of SLAs corresponding to service choreographies leading to business models such as Business Value Networks. During the hierarchical aggregation of SLAs, certain SLA information pertaining to different stakeholders is meant to be restricted and can be only partially revealed to a subset of their business partners. We introduce the concept of SLA-Views to protect such privacy concerns. We then formalize the notion of SLA Choreography and define an aggregation model based on SLA-Views to enable the automation of hierarchical aggregation of Service Level Agreements. The aggregation model has been designed to comply with the WS-Agreement standard.

1. Introduction

Service Economy to prosper requires IT-based Service Markets for stake holders to do business autonomically and autonomously helping them establish networks of business relationships. Services are traded under formal contracts known as Service Level Agreements (SLA). SLA in Service Oriented Infrastructure (SOI) is an automatically processable contract between a service and its client; the client being a person, organization or yet another service. Services scattered across various Virtual Organizations (VOs) under multiple administration domains can compose together making SLAs between each other to form service choreographies. In novel eBusiness platforms (such as Grids and Clouds) SLA is essentially important for the service consumer as it compensates consumer’s high dependency on the service provider. With the advent of on-demand service infrastructure, there is a high potential for third party solution providers such as Composite Service Providers (CSP), aggregators or resellers [1,2] to tie together services from different external service providers to fulfill the pay-per-use demands of their customers. A cumulative contribution of such Composite Service Providers will emerge as service value chains. In service value chains services corresponding to different partners are aggregated in a producer-consumer manner resulting in hierarchical structures of added value. Service Level Agreements (SLAs) guarantee the expected quality of service (QoS) to different stakeholders at various levels in this hierarchy. This in turn leads to a hierarchical structure of SLAs that may span across several Virtual Organizations (VOs) with no centralized authority. We have termed it as Hierarchical SLA Choreography or simply SLA Choreography, in accordance with the underlying Service Choreography. A major challenge to enable IT-based Service Markets.

* Corresponding author.
E-mail addresses: irfan@email.com (I. Ul Haq), huqqana3@univie.ac.at (A.A. Huqqani), erich.schikuta@univie.ac.at (E. Schikuta).

© 2011 Elsevier B.V. All rights reserved.
thus is to foster these hierarchical service composition scenarios and their underpinning business networks and supply chains. From business’ point of view, the most important asset is the extraction of value from every node of such business networks in a transparent and secure manner.

Service composition directly implies the need of composition of their corresponding SLAs. So far, SLA composition has been considered as a single layer process [3]. This single layer SLA composition model is insufficient to describe supply-chain based business networks. In a supply-chain a service provider may have sub-contractors and some of those sub-contractors may have further sub-contractors making a hierarchical structure. This supply-chain network across various Virtual Organizations may emerge as a Business Value Network. Business Value Networks [4] are ways in which organizations interact with each other forming complex chains including multiple providers/administrative domains in order to drive increased business value. NESSI (Networked European Software and Services Initiative), which is a consortium of over 300 ICT industrial partners, has highlighted the importance of Business Value Networks [4] as a viable business model in the emerging service oriented ICT infrastructures.

In addition to the notion of Business Value Networks, NESSI has pointed out various other possibilities for similar inter-organizational business models; Hierarchical Enterprises, Extended Enterprises, Dynamic Outsourcing, and Mergers to name a few. The process of SLA aggregation in such enterprises is a hierarchical process. Research community has just started taking notice [5] of the importance to describe this hierarchical aggregation. To enable these supply-chain networks as Service Oriented Infrastructures (SOI), the case of the Service Level Agreements needs to be elaborated and its issues resolved. SLA@SOI [5] is a European project that focuses on SLA issues in SOI. On its agenda is the provision of such service aggregators, that offer composed services, manageable according to higher-level customer needs. In SLA@SOI’s vision, service customers are empowered to precisely specify and negotiate the actual service level according to which they buy a certain service. Although SLA@SOI discusses the importance of service chains, but it does not highlight their relevance in terms of value multiplication.

It is not sensible to expose the complete information of SLAs across the whole chain of services to all the stakeholders. Not only because of the privacy concerns of the business partners, but also disclosing it could endanger the business processes creating added value. To achieve this balance between trust and security, we introduce the concept of SLA-Views. The inspiration for this concept comes from the notion of business process views [6,7] and workflow views [8]. We apply the concept of views on SLA-Choreography. Each business partner will have its own view comprising of its local SLA information. The holistic effect of these views will emerge as the overall SLA-Choreography. In this paper we present a formalized approach based on the concept of SLA-Views and adherent to WS-Agreement standard, to automate the aggregation process of hierarchical SLAs in Business Value Networks. The overall contribution of the paper consists of:

- a privacy model based on the concept of SLA-Views,
- a formal description of hierarchical SLA-Choreographies based on SLA-Views in Business Value Networks,
- a formal model for SLA aggregation in hierarchical SLA-Choreographies, and
- the customization of WS-Agreement to support the hierarchical SLA aggregation model.

In Section 2, we give a survey of the related work. Section 3 introduces the hierarchical choreography of SLAs. Section 4 formalizes the concept of SLA View and SLA Choreography. Section 5 describes the formal model of hierarchical aggregation of SLAs and Section 6 discusses the special case of Guarantee Terms. Section 7 highlights some business applications of the aggregation model and section 8 presents a motivational example based on this model. Finally, section 9 concludes the paper with an overview of our achievements and strategy for our future work.

2. Related work

The related work spans across three dimensions: aggregation models of SLAs, formal description of SLAs and the privacy of stake-holders in business cooperations.

2.1. SLA aggregation

A Service Level Agreement is a contract between a service and its client; the client being a person or yet another service. Service composition in workflow also demands SLA composition. Only little research [3,9] has been done towards dynamic SLA aggregation of workflows. Blake and Cummings [3] have defined three aspects of SLAs which are Compliance, Sustainability and Resiliency. Compliance means suitability, i.e. the consumer receives what is expected. Sustainability is the ability to maintain the underlying services in timely fashion. Resiliency directly corresponds to the maintenance of services to ensure their performance over an extended period of time. The authors then subdivide these three categories into six aspects of SLA but this makes their approach rather specific, because it does not cover the whole range of SLA aspects. They put forth a model to compose SLAs of services mapping to a workflow, but they restrict the services to one level only. Frankova [9] has also highlighted the importance of this issue but has just described a vision and not any concrete model. Unger et al.’s work [10] is directly relevant to our focus of research. They focus on aggregation of SLAs in context with Business Process Outsourcing (BPO). They synchronize their work with Business Process Execution Language (BPEL) and WS-Policy. Their model is based on SLO aggregation of SLAs on a single level. One of the limitations of their approach is that they take into account services related to one process in one enterprise because they focus on BPO. Our approach describes cross-VO SLA aggregation and strictly adheres to WS-Agreement. Theilman et al. [11] also take into account the significance of multi-level SLAs and highlight the importance of SLA orchestration. They propose an SLA management framework based on the notions of SLA orchestration, transformation and aggregation. They too take into account
WS-Agreement as their basic SLA standard. In their framework they do not focus on the value driven aggregation of hierarchical SLAs whereas our aggregation model is centered at value generating business networks.

2.2. Formal description of SLA

Aiello et al. [12] present a very sound formal description of SLA. Their approach is based on WS-Agreement. They extend the WS-Agreement standard by introducing a new category of terms called Negotiation Terms. They build automata representation of SLA states to describe the negotiation process. However, their formal model is too vague and they do not explain how this model will describe the sub-entities in WS-Agreement. Unger et al. [10] present a rigorous formal model for SLA aggregation. They follow BPEL and WS-Policy whereas our formal model adheres to WS-Agreement standard.

2.3. Workflow views

For privacy concerns we will define the notion of SLA-Views, which is similar to the concept of workflow views, but is not formally based on it. The concept of workflow views is used to maintain the balance between trust and security among business partners. Schulz et al. [13] have introduced the concept of view based cross-organizational workflows and they call it a coalition workflow. Chebbi et al. [14] provide a comprehensive view based and web services focused approach that is applicable to dynamic inter-organizational workflow cooperation. This means that the cooperation across organizations is described through views without specifying the internal structure of participating workflows. Their concept of contracts is similar to that of SLA, however, SLAs are more dynamic due to negotiation, renegotiation and fault tolerance features. There is some very relevant work done by Chiu et al. [15] in terms of a contract model based on workflow views. They demonstrate how management of contracts can be facilitated. Starting with an example they highlight domains of different participating organizations and then develop a model to identify the corresponding workflow views. They go on further to develop an e-contract model based on plain text format. Service Level Agreements represented in XML format are more structured and flexible than the e-contracts. Furthermore their approach starts with defining views in an inter-organizational workflow and then describing e-contracts to enforce the obligatory communication links in the views. Our model allows SLAs to maintain their individual identity. Therefore, we define views directly on the SLA aggregation structure rather on workflows. Moreover, our approach provides a formal description of hierarchical SLAs and their aggregation model.

3. Hierarchical choreography of SLAs

A Service level agreement is a contract that defines mutual understandings and expectations regarding a service between the service provider and the service consumer. WS-Agreement [16], a standardized SLA language from OGF (Open Grid Forum) [17], defines the structure of agreement as depicted in Fig. 1. The contract should bear an official name. Agreement Context contains information about the initiator, the responder and the provider of the agreement; expiration time of the agreement; and its template Id. Service Terms define the functional attributes of the agreement whereas the Guarantee Terms contain the non functional attributes. Guarantee Terms further describe the conditions, service level objectives and business value list related to the agreement. Business value list may express the importance of meeting an objective as well as information regarding penalty or reward.

Referring to Fig. 1, we can formally define the Service Terms, and Guarantee Terms as part of the encapsulating section Terms.

---

**Fig. 1.** Structure of an agreement in accordance with WS-Agreement specification.
Definition 1 (Service Term). A service term denoted by \( \text{term}_s \) is an element of the set Service Terms denoted by \( \text{STerms} \). A service term is a tuple such that,

\[
\text{term}_s = < \text{name}, \text{value}, \text{type}_a >
\]

where name and value denote the name and value of a service term and \( \text{type}_a \) describes its aggregation type.

We extend the WS-Agreement standard by a new mandatory element, namely \( \text{type}_a \). The \( \text{type}_a \) element corresponds to the aggregation function that helps to automate the aggregation of SLAs. We postpone its definition to the latter part of the paper where we will discuss the aggregation process.

Definition 2 (Guarantee Term). A guarantee term denoted by \( \text{term}_g \) is an element of the set Guarantee Terms i.e., \( \text{GTerms} \). A guarantee term is a tuple such that:

\[
\text{term}_g = < \text{SLO}, \text{condition}, \text{BVL} >
\]

where SLO represents Service Level Objectives, \( \text{condition} \) represents Qualifying Conditions and BVL represents Business Value List. Combining the above two definitions, now we can define the notion Terms in WS-Agreement.

Definition 3 (Term). \( \text{term} \in \text{Terms} \) is a pair such that

\[
\text{term} = (\text{term}_s, \text{term}_g)
\]

where \( \text{term}_s \in \text{STerms} \) and \( \text{term}_g \in \text{GTerms} \).

Following the above definitions, SLA can now be formally defined as:

Definition 4 (SLA). A Service Level Agreement (SLA) denoted by \( \text{sla} \) is a tuple

\[
\text{sla} = < \text{Name}, \text{Context}, \text{Terms} >
\]

where \( \text{Terms} = \bigcup_{i=1}^{n} \text{term}_i \) and Context is a list of strings. Context defines the names of the SLA provider, the consumer and the initiators. It also contains the duration of the SLA. The parameter \( \text{Name} \) denotes the name of the SLA.

A Virtual Organization (VO) in business context is a temporary or permanent, coalition of geographically dispersed organizations expressing high level mutual trust to collaborate and share their resources and competencies in order to fulfill the customers' requests. Web services scattered across various administrative domains, when composed together, are said to form service choreographies. In these service choreographies many service-to-service SLAs are formed. The situation becomes even more complex in Business Value Networks, where services scattered across many such Virtual Organizations (VO) collaborate to enable complex supply chain networks. One way to visualize this hierarchy is by dependency layers. The deeper a service in this chain is, more dependent its ancestors are. A hierarchy of corresponding SLAs pertains to this chain of services. There is no multi-level SLA model that can describe the hierarchical aggregation of SLAs in such Business Value Network. We will call this hierarchical aggregation of SLAs a SLA-Choreography with relevance to the Service Choreography.

In Fig. 2, we present a simplified picture of a cross-VO choreography. The client (that may be a workflow process) is directly connected to some services scattered across three VOs: VO-A, VO-B, VO-C. These services are coordinating with other services to carry out their jobs. This coordination results into service chains, distributed across multiple Virtual Organizations. This scenario can be compared with a simple Business Value Network. The partner services play the producer–consumer roles in this service choreography. All of these services establish Service Level Agreements (SLA), thus giving rise to an SLA-Choreography in connection with the underlying service choreography.

Another way to visualize this SLA Choreography is in terms of hierarchical organization of SLAs. There may be several dependency layers in this SLA-Choreography. The dependency increases along the hierarchy. The aggregated effect of this dependency travels from the very bottom towards the topmost level. This SLA aggregation is depicted in Fig. 2. In this hierarchy the SLAs, which are connected to the client process, are said to exist on level 1. This hierarchy indicates a supply chain type of correspondence among the services. These layers also denote the visibility levels of service providers and the client. The client can see its coordinating services, i.e. its providers and its consumers, with which it is establishing Service Level Agreements. It has no information about the rest of the service choreography. Despite its privacy concerns, a service is dependent on its lower services. The effect of SLAs formed among the services at lower levels is bubbled up through the upper layers.

In this paper we focus on the fundamental problem: To develop a formal model that can describe this SLA Choreography and construct an aggregation model for hierarchical SLAs while protecting the privacy concerns of the stakeholders at the same time. For this purpose we introduce the concept of SLA-Views.
4. SLA views

The concept of Views originates from the field of databases and has been successfully adapted in business workflows [7,14]. In workflows, a view can be a subset of that workflow or can be a representation of that workflow in aggregated or abstracted fashion. We employ the notion of views to represent a subset of SLA-Choreography. As the matter of fact the notion of SLA-Views is related to that of workflow views in a very general sense. In formal sense SLA-Views are different from workflow views. SLA-Choreography is not a workflow; so the rules of workflows are not applicable on it. For instance, in a workflow, rules such as: there should be a single start and single exit or every split should have a join, do not apply on SLA-Choreography.

A view in an SLA-Choreography represents the visibility of a business partner. Every service provider is limited only to its own view. A partner (for example a service) makes two kinds of SLAs: the SLAs for which it acts as a consumer and the SLAs for which it is a provider. For clarity, we name these two types as the consumer-oriented SLAs and the producer-oriented SLAs respectively.

In Fig. 3 SLAs are connected to small circles, which we call aggregation points, by certain edges called dependencies. There are two types of dependencies. Consumer-oriented SLAs are connected to the aggregation points from below by the sink dependencies and the producer-oriented SLAs are connected from above by the source dependencies. To understand the overall picture of the SLA-Choreography, we need to formalize these concepts.

**Definition 5 (Aggregation Point).** An Aggregation Point $ap$ is an object such that

$$ap = \langle \text{aggsla} \rangle$$

where aggsla is the aggregated SLA produced by aggregating the consumer-oriented SLAs connected to it. In Fig. 3 $ap_{-i_2}$ is an aggregation point. An aggregation point is the point where the consumer-oriented SLAs (of the consumer service) are aggregated and on the basis of their aggregated content the service is able to decide what it can offer as a provider. The master–slave relationships in Business Value Networks are directly translated to producer–consumer model with one service provider (enterprise) as a producer and the other as the consumer. So both the producer and the consumer enterprises will have their own aggregation points connected together through their mutual SLA. However, for peer-to-peer relationships, both peers act as producer and consumer of services. This issue can be easily resolved by translating peer-to-peer relationships into producer–consumer model. For this purpose we devise the concept of virtual aggregation point (vap) to automate the aggregation process. Virtual aggregation point is discussed in detail in Section 7.

Now let us define dependencies which have been shown in Fig. 3(a) as edges joining the aggregation point with the producer and consumer oriented SLAs. The Aggregation Point $ap_{-i_2}$ is connected with three consumer-oriented SLAs and one producer-oriented SLA through dependencies.

**Definition 6 (Source Dependency).** A source dependency $dep_{src}$ is a tuple

$$dep_{src} = \langle ap, sla \rangle$$

where $ap$ is the aggregation point and $sla$ is the producer-oriented SLA. In Fig. 3(a) it is represented by the directed edge from the aggregation point $ap_{-i_2}$ to the producer-oriented SLA, $sla_{a_3 - i_2}$.

Each $dep_{src} \in Dep_{src}$, where $Dep_{src}$ is the set of all source dependencies within the SLA-Choreography. Let

$$source : (ap) \rightarrow dep_{src}$$


source($ap$) is the unique $s \in Dep_{src}$ for which a unique producer-oriented SLA exists with $s = (ap_s, sla_i)$. This means that the function source maps each aggregation point $ap$ to a unique SLA through a unique source dependency $s$.

**Definition 7 (Sink Dependency).** A sink dependency $dep_{sink}$ is a tuple

$$dep_{sink} = <sla, ap>$$

where $ap$ is the aggregation point and $sla$ is the consumer-oriented SLA. In Fig. 3, it is represented by the directed edge from the consumer-oriented SLA $i_2 - i_1$ to the aggregation point $ap - i_2$. The aggregation point $ap - i_2$ is connected with three sink dependencies.

Each $dep_{sink} \subseteq Dep_{sink}$, where $Dep_{sink}$ is the set of all sink dependencies within the SLA Choreography. Let

$$sink : (ap) \rightarrow P(Dep_{src})$$

where $P(Dep_{sink})$ is the power set of $Dep_{sink}$.

$sinks(ap)$ is the set $S_{sink} \subseteq P(Dep_{sink})$, i.e. $S_{sink} \subseteq Dep_{sink}$ such that for each $s_i \in S_{sink}$ a unique consumer oriented SLA exists with $s_i = (sla_i, ap_i)$. This means that the function sinks maps a set of consumer-oriented SLAs to a unique aggregation point such that each consumer-oriented SLA $sla_i$ is mapped through a unique sink dependency $s_i$.

**Definition 8 (Dependency).** A dependency $Dep$ is a set that is the union of two sets namely $Dep_{src}$ and $Dep_{sink}$ which are pairwise disjoint, i.e.

$$Dep = Dep_{src} \cup Dep_{sink}$$

$$Dep_{src} \cap Dep_{sink} = \phi$$

Based on these definitions we see in Fig. 3 that the producer-oriented SLA ($a_3 - i_2$) is dependent on the terms of the corresponding consumer-oriented SLAs, aggregated at $ap - i_2$. For example the bandwidth and space aggregated at $ap - i_2$ would be the upper limit of what service $i_2$ can offer to service $a_3$. At the same time service $i_2$ will have to decide about its profit on the basis of the information about total cost in the aggregated SLA. The aggregation point in this sense is also a decision point for a service.

With having all the related concepts formalized, now we are in a position to provide a formal definition of the SLA-View.

**Definition 9 (SLA-View).** An SLA-View denoted by $slaview$ is a tuple such that

$$slaview_i = <sla_i, dep_{sr}, ap_i, SLA_c, Dep_{sm}>$$
where sla, is a producer-oriented SLA, SLAc is a set of consumer-oriented SLAs, dep, is a source dependency, Depsn is a set of sink dependencies, and ap, is an aggregation point. Each aggregation point ap, in the SLA-Choreography corresponds to a unique sla-viewi.

In Fig. 3 the SLA-Views of the client and a service are highlighted.

**Definition 10 (SLA Choreography).** An SLAchor is a tuple such that

\[
SLA_{chor} = <SLA, APoints, Deps>
\]

where SLA is set of all sla within an SLA Choreography, APoints is set of aggregation points ap, and Deps is set of dependencies dep.

Another way to describe the SLA-Choreography terms of SLA-Views is

\[
SLA_{chor} = \bigcup_{i=1}^{n} sla_{view_i}
\]

This means that the whole SLA Choreography may be seen as an integration of several SLA-Views. In terms of Business Value Networks it should be noted that SLA-View defines boundaries of a stakeholder. The aggregation process is performed at every aggregation point. Each aggregation point, which also denotes a dependency level, belongs to one of the service providers. Although each service provider is limited to its own aggregation information, this information is in fact dependent on the aggregation information at lower levels. The sustainability of this business network requires all the stakeholders to trust each other and their ability to maintain their privacy at the same time. SLA-Views maintain a balance between this privacy and trust.

5. Aggregation process

In the aggregation process terms of the consumer-oriented SLAs are aggregated. WS-agreement has no direct support for such an aggregation. So we introduced an attribute for aggregation type, namely “typea” in Definition 1. WS-Agreement gives the liberty to incorporate any external schema. Therefore typea can be made an essential part of the service terms and will describe, how the corresponding service will behave during the aggregation process. We can define typea in a formal way, as follows:

**Definition 11 (typea).** A typea \(\in\) Types is a function that maps a set of tuples to a single tuple, which is the aggregation of that set:

\[
typea : tuples(\text{term}) \rightarrow \text{term}
\]

\[
typea(\text{term}_1, \ldots, \text{term}_n) = \text{term}_{agg}
\]

We define typea as an aggregation function that aggregates n terms into one term. Its result is aggsla in the aggregation point (see Definition 5). Each term in aggsla is computed by applying the type function for that term to the values of the terms for all the dependent (consumer-oriented) SLAs, which define that term. In the present context, we define four types of terms, namely sumtype, maxtype, mintype and neutral, but new types can be added according to the situation, i.e.

\[
\text{Types} = \{\text{sumtype}, \text{maxtype}, \text{mintype}, \text{neutral}\}
\]

These functions are depicted in Fig. 3(b). The function sumtype can be formally defined as follows.

**Definition 11.1 (sumtype).**

\[
\text{sumtype} \in \text{Types}(\iff \text{sumtype} : \text{tuples(\text{term})} \rightarrow \text{term})
\]

\[
\text{sumtype}(\text{term}_1, \ldots, \text{term}_n) = \sum_{i=1}^{n} \text{term}_i
\]

sumtype is an aggregation function that aggregates n number of terms into one term. sumtype is of the type of typea and takes the summation of all terms. Examples include terms for storage space, memory, availability and cost.

**Definition 11.2 (maxtype).**

\[
\text{maxtype} \in \text{Types}(\iff \text{maxtype} : \text{tuples(\text{term})} \rightarrow \text{term})
\]

\[
\text{maxtype}(\text{term}_1, \ldots, \text{term}_n) = \max_{i=1}^{n} \text{term}_i
\]

maxtype is an aggregation function that aggregates n number of terms into one term. It does so by picking up the maximum of these terms, which represents the aggregation of all the input terms. If several terms addressing the same utility are being aggregated and their type has been declared as maxtype, then only the term pertaining to the maximum value will become part of the aggregated
SLA. Examples include latency, which may become a bottle neck for the whole process and an activity with highest latency will directly contribute (though in a negative sense) to the throughput of a workflow sequence.

**Definition 11.3 (mintype).**

\[
\text{mintype} \in \text{Types}(\iff \text{mintype} : \text{tuples}(\text{term}) \rightarrow \text{term})
\]

\[
\text{mintype}(\text{term}_1, \ldots, \text{term}_n) = \min_i^n \text{term}_i
\]

mintype is an aggregation function that aggregates a number of terms into one term. It does so by picking up the minimum of these terms, which represents the aggregation of all the input terms. Similar to \text{maxtype}, when several terms addressing alike utilities are being aggregated and their type has been declared as \text{mintype} then only the term pertaining to the minimum value will contribute to the aggregated SLA. An example can be the bandwidth. In a sequence of activities the activity pertaining to the minimum bandwidth will become the bottleneck for the whole sequence making other activities with higher bandwidth ineffective.

**Definition 11.4 (neutral).**

\[
\text{neutral} \in \text{Types}(\iff \text{neutral} : (\text{term}) \rightarrow \text{term})
\]

\[
\text{neutral}(\text{term}_i) = \text{term}_i
\]

neutral is an aggregation function that includes all the input terms separately without any processing. This function is applied on those terms which cannot be mixed with other terms and need to be preserved in the aggregation process as separate terms. The terms declared as neutral are unaffected through the aggregation process and are just copied in the aggregated SLA. They represent services which are independent from similar services, for example identity of some valuable data in a certain organization or discount in a specific service, etc.

So far we have defined only four types of terms, but it is important to realize that this enumeration can be extended without affecting the generic definition of the \text{type} function. In certain cases, especially for calculating the reward and penalty expressions, logical operations will also be required. Similarly we can define logical functions such as AND, OR, and XOR to integrate the service level objectives or other constituents of Guarantee Terms to form rule-based aggregation expressions. These logical operators are equally useful for the Service Description Terms as well. For instance, a service provider who wants to aggregate resources of varying qualities but would also like to segregate them under different levels of SLAs, may use \text{ORType} aggregation function for this purpose. An example could be a reseller who buys computational resources of different speeds and qualities from different vendors and aggregates them using \text{ORType} function so that later, he can offer SLAs of different levels such as gold silver or bronze, etc. to its consumers.

6. Aggregation of guarantee terms

Guarantee Terms (GTs) can also be aggregated together similar to the Service Description Terms (SDTs) as described in the previous section. However there are certain peculiarities to be considered when it comes to the Guarantee Terms. First of all, Guarantee Terms are optional terms in context with the WS-Agreement standard. Secondly, even if two aggregating SDTs have GTs associated with themselves, the GTs may refer to different service level parameters, i.e. they provide guarantees in form of Service Level Objectives (SLOs) about different properties of the service.

An example is a service consumer who wants to aggregate two similar storage services. The aggregation of SDTs will give him the total sum of available disk space. But what if two vendors are providing GTs describing entirely different aspects, e.g. access time and availability of service? These two aspects are not related, hence cannot be aggregated. To solve this problem there are many feasible approaches:

- A solution to this problem can be found, if the SLA negotiation process somehow facilitates the consumer to ask for guarantees upon the desired properties of services thus helping him setting up the identical guarantees with its different service providers. Not only this type of mechanism is very difficult to achieve, but by restricting the variability of SLA contents, it also turns out to be contrary to the automation requirements of the process for which it was originally designed for.
- A similar approach can be based on the renegotiation for a revision of SLA with new guarantees. This approach may not be successful every time because the service provider may not be in a position to offer the required type of guarantee.
- Some popular (or straightforward) guarantees may be standardized to be always offered for the relevant services by all the service providers. This approach will definitely improve the situation but there will be always new services with innovative properties expressed by guarantees unseen in the past.
- If the guarantees translate to the quality of service, then in some situations it may be desirable to use \text{ORType} aggregation in order to segregate the services on the basis of their guarantees. For this purpose the service terms should be declared as \text{ORType}.
- The most straightforward and safe method is to leave the guarantees disaggregated and the situation should be reported to the service provider to take some decision. In this way, we may allow each service provider in the supply chain to figure out and set up its own Guarantee Terms during the aggregation process based on its personal business rules.
The last approach also conforms to our formal model. We assume the aggregation point to be the decision center of the service provider as well. Within this decision center, the aggregation of SLAs is performed to facilitate the formation of business objectives of the service provider. Therefore, when a stake-holder in the supply chain acts as a service provider, it needs to layout its business strategy at least once before starting the provision of services. Our aggregation model thus only promises a semi-automatic aggregation of Guarantee Terms. In that context, the aggregation of Guarantee Terms becomes a business issue and is interlinked with the business goals of the service provider. This approach also resolves another very crucial issue of aggregating reward and penalty expressions. Within the aggregation point, the reward and penalty expressions must be expressed in accordance with the business rules of the service provider. A subset of those business rules may be dedicated especially to facilitate the aggregation process.

7. A case for hierarchical aggregation of SLAs in business applications

NESSI, in its Grand Vision and Strategic Research Agenda (SRA) [4], defines Value Networks as the ways in which organizations interact with each other to drive increased business value. Fig. 4 shows their example Business Value Network (BVN) where the Enterprises A and D have been shown to collaborate on the development of a new product. Enterprise A has subcontractors B and C whereas the enterprise has E and F as subcontractors. The Enterprises A and D form a peer-to-peer relationship between themselves.

So far, we have discussed the aggregation of SLAs in context with the composition of services in a producer–consumer manner along service value chains. This service level SLA aggregation model can be scaled up to enterprise level. It can conveniently describe both master–slave and peer-to-peer relationships in Business Value Networks. Master–slave relationship can be simply mapped on the producer–consumer model where an SLA is formed between the service provider and the client. However, in peer-to-peer relationships, the participating enterprises are acting as the service provider and the client at the same time. To form a WS-Agreement compliant SLA between them, one party can either be treated as a service provider or a service consumer in context with some service. Therefore a peer-to-peer relationship needs to be dissolved into two producer–consumer relationships with a separate SLA associated with each of them. Here we would like to define a Virtual Enterprise Organization (VEO). According to NESSI’s definition [4] VEOs are formed when two or more administrative domains overlap and share resources. Relationships among different enterprises within a VEO can be master–slave or peer-to-peer or a combination of both. We apply the concept of VEO to peer-to-peer relationships in Fig. 4. If we consider the enterprises A and D to form a Virtual Enterprise Organizations (VEO), their SLAs are aggregated at a virtual aggregation point (vap) that represents this VEO. The virtual aggregation point is important to be represented, because it describes the SLA view of the resulting VEO, which is different from the SLA views of A and D. The shared functionality of the VEO is described in the aggregated SLA computed within the vap-[AD]. Note that the big brackets have been adopted to highlight the jointly contained capabilities of enterprises A and D. The terms of services are aggregated through aggregation functions described in Section 5. The terms marked as neutral are not merged and kept separate in the aggregated SLA. The virtual aggregation point also denotes the decision point of the resulting VO and policies such as distribution of revenue and

![Fig. 4. A business value network and its corresponding SLA choreography with different Enterprises’ Views.](image-url)
cost of offered services will also be decided inside it. From a practical perspective there are numerous issues, such as trust, security and heterogeneity related to SLA aggregation among peer-to-peer enterprises. We provided a conceptual framework including a third party trust manager to address these issues [18].

Other NESSI models such as Hierarchical Enterprises and Extended Enterprises [4] can be easily described through our model too. The concept of inter-Cloud or Cloud of Clouds [19] is becoming very popular these days, which realizes the virtual collaboration of Clouds. Such a virtual collaboration among the Cloud maps straightforwardly on our SLA aggregation model.

8. Motivational scenario

In the following section we will present a motivational scenario of an ad-hoc business value network, which is enabled by the aggregation mechanism presented above. Arfa is a graphics designer and she has just finished designing an animation involving thousands of high resolution images. Now she needs to carry out hi-tech multi-media operations such as rendering and editing. She plans to utilize online services to accomplish these tasks. A media workflow service allows Arfa to define a series of activities involving video rendering, compression etc. Afterwards she would like to host the final compressed video on a dedicated server to visualize the output of her work. From Arfa’s view-point, she is only aggregating two services, i.e. “rendering workflow” service and the “hosting service” identified as the “Platform as a Service” (PaaS) and the “Infrastructure as a Service” (IaaS) respectively by her Cloud based service providers. What Arfa’s View does not cover is the fact that both of these services are themselves result of an aggregation of even more basic services thus extending a supply chain type of structure beneath them. The rendering workflow subdivides into services such as the “media engine” and the “computing infrastructure” provided by different service providers in a public Cloud. On further investigation it may be revealed that the “media engine” is composed of even more basic services such as the “graphics engine” the “sound engine”, and the computing Infrastructure too resells different qualities of computing services with varying response times and calculation speeds. The list goes on. The SLA-Choreography resulting from this simple scenario is shown in Fig. 5. There are two services, namely the rendering workflow service and the hosting service. The host video service downloads the video from a specified location, archives it and makes it available online. An authenticated user can play the video in a YouTube like style.

The SLA-Choreography resulting from this scenario is depicted in Fig. 5. The aggregation functions described in Fig. 3(b) are being applied in the scenario shown in Fig. 5. It is evident that the resolution provided to the end-client is the minimum of the hosting service and rendering workflow service. So at the aggregation point ap-client, the aggregation function Min will choose only minimum of
the two resolutions. On the same grounds the total cost that the client has to pay is the sum of the cost incurred on hosting and the cost spent on rendering workflow, because cost has been declared as "sumtype". We take the liberty of importing an external schema into WS-Agreement's Service Description Terms' section. The following chunk of Schema allows this.

```
<xs:complexType name="ServiceDescriptionTermType">
  <xs:complexContent>
    <xs:extension base="wsag:ServiceTermType">
      <xs:sequence>
        <xs:any namespace="#other" processContents="strict"/>
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

The above schema enables us to include an XML structure of elements adhering to any external Schema. This makes it possible to incorporate the aggregation type (typea) element inside a Service Description Term. A simple schema to accomplish this can be written as follows.

```
<?xml version="1.0" encoding="utf-16"?>
<xs:schema
  xmlns:myns="http://schemas.xyz.com"
  xmlns="http://www.mynamespace.com"
  targetNamespace="http://www.mynamespace.com"
  xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:simpleType name="aggregationType">
    <restriction base="xs:string">
      <enumeration value="MinType"/>
      <enumeration value="MaxType"/>
      <enumeration value="SumType"/>
      <enumeration value="Neutral"/>
    </restriction>
  </xs:simpleType>
  <xs:element name="Resolution">
    <xs:complexType>
      <xs:sequence>
        <xs:complexType name="ResolutionXY">
          <xs:sequence>
            <xs:element name="ResolutionX" type="xs:integer"/>
            <xs:element name="ResolutionY" type="xs:integer"/>
          </xs:sequence>
        </xs:complexType>
      </xs:sequence>
      <xs:attribute name="aggregationType" type="xs:aggregationType"/>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

Then the service Description Term namely "resolution" for the Enhance-Video service may be expressed as follows.

```
<wsag:ServiceDescriptionTerm wsag:Name="Resolution" wsag:ServiceName="Enhance-Video">
  <myns:ResolutionXY>
    <myns:ResolutionX>1920</myns:ResolutionX>
    <myns:ResolutionY>1080</myns:ResolutionY>
  </myns:ResolutionXY>
  <myns:aggregationType>mintype</myns:aggregationType>
</wsag:ServiceDescriptionTerm>
```

The aggregationType (i.e. typea) declares Resolution as a minType term. When it will be aggregated with other minType terms, only the minimum of these terms will become part of the aggregated SLA. Other aggregation types listed in the schema can be expressed and aggregated in a similar fashion.
9. Conclusion

We presented a view based formal model to describe hierarchical Service Level Agreements in supply chain scenarios such as Business Value Networks. SLA-Views help to maintain balance between trust and privacy. Our model identifies basic aggregation constructs that are used in the aggregation of SLAs. The whole aggregation process stays in compliance with the WS-Agreement standard. We also highlighted the significance of this work in context with the IT-based service economy and on-demand service provision especially with reference to Cloud Computing. In our other work we also discussed a validation framework based on the aggregation model explained in this paper.

There are many interesting questions that need answers: What trust model will bind together the Business Value Networks? Who will manage this SLA-Choreography? How to monitor and validate this SLA-Choreography? Although these questions are related to our overall research agenda but are beyond the scope of this paper.

In future, we will continue our work on implementing a secure aggregation and validation framework for SLAs in heterogeneous Virtual Organizations.

Appendix A. List of Improvements in the Journal Version

There have been following modifications in the original BPM2009 version.

1. The abstract has been modified and motivated with the notion of IT based Service Economy.
2. Section 1 has been improved with further elaborations about IT based Service Economy, On demand service infrastructure and Cloud based business processes.
3. Section 2 has been slightly improved with some relevant discussion regarding the SLA@SOI project.
4. Section 5 has been improved with some additional notes as part of the last paragraph.
5. Section 6 is an entirely new section which was not part of the BPM2009 version.
6. Section 8 discusses an entirely new motivational scenario which is different as compared to the one in the previous version.
7. Section 9 also discusses the relevance of this research with our recently published work.

References


Dr. Irfan Ul Haq has recently completed his PhD studies at the Department of Knowledge and Business Engineering, University of Vienna. He received his Masters degree in Physics with specialization in Electronics from University of the Punjab, Pakistan in 1997. He has also completed coursework of MS in Software Engineering from National University of Science and Technology Islamabad, Pakistan. He has also completed two Post Graduate Diplomas in Computer Science and Computer Systems of accumulative duration of 2 years. He has published over 20 papers in international conferences, edited books and journals. His research interests include Grid and Cloud Computing, Workflows, Artificial Intelligence and Semantic Web.
Altaf Ahmed Huqqani is a PhD candidate at the Department of Knowledge and Business Engineering, University of Vienna. He received Masters degrees in Physics as well as in Systems Engineering from the Quaid-i-Azam University Islamabad. He has contributed in providing computing facilities to several Virtual Organizations like Atlas, Biomed, LHCB, Alice relating to different experiments under the umbrella of LCG (LHC Computing Grid). His research interests include Grid and Cloud Computing.

Prof. Dr. Erich Schikuta is Deputy Head of the Dept. of Knowledge and Business Engineering. His research interests are now in the area of parallel and distributed computing (with a specific focus on Grid Computing and Service-Oriented Architectures), neural network simulation and database systems, and he is author of more than 100 peer reviewed papers. The main impact of his work in the area of CS was, beside his work on MPI, the establishment and labeling of the research area of Grid Clustering methods, which set ground for a number of highly renowned clustering and data mining algorithms. His expertise is reflected by his appointment as expert evaluator and reviewer of the European Commission (FP5, FP6, and FP7) and several other European research and funding organizations (e.g. INRIA). Further he has been a member and chair of many program committees of conferences.