Rule-Based Workflow Validation of hierarchical Service Level Agreements

Ul Haq, Irfan; Paschke, Adrian; Schikuta, Erich; Boley, Harold

Publisher's version / Version de l'éditeur:

Proceedings of the 4th International Conference on Grid and Pervasive Computing (GPC 2009), 2009-05-08

Rule-Based Workflow Validation of Hierarchical Service Level Agreements

Irfan Ul Haq[1], Adrian Paschke[2], Erich Schikuta[1], and Harold Boley[3]
[1]Department of Knowledge and Business Engineering, University of Vienna, Austria
{irfan.ul.haq, erich.schikuta} AT univie.ac.at
[2]Institute of Computer Science, Freie University Berlin, Germany
paschke AT inf.fu-berlin.de
[3]Institute of Information Technology, National Research Council, Canada
harold.boley AT nrc.gc.ca

Abstract

Business-to-business workflow interoperation across Virtual Organisations (VOs) brings about possibilities for novel business scenarios. In such business scenarios, parts of workflows corresponding to different partners can be aggregated in a producer-consumer manner, making hierarchical structures of added value. Service Level Agreements (SLAs), which are contracts between service providers and service consumers, guarantee the expected quality of service (QoS) to different stake holders at various levels in this hierarchy. This hierarchical SLA choreography and aggregation poses new challenges regarding its description, management, maintenance, validation, trust and security. In this paper we focus on the design and assessment of an agent-enabled, rule-based validation framework for the hierarchical SLA aggregation, corresponding to cross-VO workflow cooperation.

1 Introduction

The work presented in this paper aims at dynamic and automated cooperation of business workflows in a service-enriched environment such as the Grid. During a business-to-business workflow composition across Virtual Organisations (VOs), Service Level Agreements are made among different partners at various points of the choreography. A Service Level Agreement (SLA) is a formally negotiated contract between a service provider and a service consumer to ensure the expected level of a service. The service consumer can be a client or another service. The partners in an SLA include the client, the Virtual Organizations (VO) and the services. Workflow composition also implies the composition of their corresponding SLAs. So far, SLA composition in workflows has been considered [5] as a single-layer process. This single-layer SLA composition model is insufficient to describe coalition workflows [23] where a multilayered aggregation of services is required that results in supply-chain type of business networks. This supply-chain business network, spun across various VOs, may result in a so-called Business Value Network.

Business Value Networks [2] are ways in which organizations interact with each other forming complex chains, including multiple providers/administrative domains, in order to drive increased business value. In a supply chain, a service provider may have sub-contractors and some of those sub-contractors may have further sub-contractors, resulting in a hierarchical structure. This leads to a hierarchical structure of SLA contracts between the different supply chain partners. Since this SLA hierarchy may span across several VOs with no centralized authority, in the rest of the paper we will call it *Hierarchical SLA Choreography* or simply SLA Choreography, in accordance with the underlying Service Choreography.

A service provider may not want to disclose certain information about their personal SLAs. Not only is it usually unnecessary to reveal the information about a business partner's sub-contractors, but this could also endanger business processes creating added value. These issues have been addressed in terms of *workflow views* [23, 7, 15], where every service provider is limited to only their own view. Extending this approach, we introduced the concept of *SLA Views* [11]. SLA Views also complement the notion of distributed trust among the various partners in a coalition workflow [12].

We have shown in our earlier work [11] how SLA Views contribute to the process of hierarchical SLA aggregation across SLA choreographies. This aggregation of SLA choreographies frequently requires validation for maintenance and fault tolerance purposes. As the aggregation details are obscured at different levels of hierarchy, a distributed top-down validation mechanism is a good strategy

for the complete validation of a hierarchical SLA aggregation.

In this paper we present an agent-enabled rule-based runtime validation framework for hierarchical SLAs, which allows the provisioning, delivery, and monitoring of services in coalition workflows as well as their highly dynamic and scalable consumption.

The framework is based upon:

- the Rule Responder Architecture [21],
- the findings of the *RBSLA* project [20],
- the formal model of SLA Views [11], and
- the distributed trust model [12].

Section 2 introduces the relevant models contributing to our validation framework by introducing the Rule Responder architecture, RBSLA, SLA Views, and Distributed Trust. In section 3 we describe the runtime validation framework for Hierarchical SLA Aggregation, and in section 4, our Delegation-of-Validation approach. Section 5 gives a survey of related research and finally, Section 6, the conclusion and future work.

2 Validation of SLA Aggregation in the Cross-section of Models

Validation of hierarchical SLA aggregation corresponding to cooperative workflows is a distributed problem. The service choreography may be distributed across several Virtual Organizations and under various administrative domains. This hierarchical choreography of heterogeneous services is only possible through a well defined distributed trust schema. There must be a privacy model too, to restrict the information of a service provider that it does not want to reveal to the consumer. One of the very relevant challenges in this regard, discussed in detail in [11], is the step-wise aggregation of SLAs for the series of service providers at different levels in the service chain. The complete information of aggregated SLA at a certain level in the service chain is known by the corresponding service provider and only a filtered part is exposed to the immediate consumer. This is the reason why during the validation process, the composed SLAs are required to be decomposed in an incremental manner down towards the supply chain of services and get validated in their corresponding service providers's domain. A validation framework for the composed SLAs, therefore, faces many design constraints and challenges: a trade-off between privacy and trust, distributed query processing, and automation to name the most essential ones. The aforementioned challenges bring in a cross-section of models depicted in figure 1. The privacy concerns of the partners are ensured by the SLA-View model [11], whereas

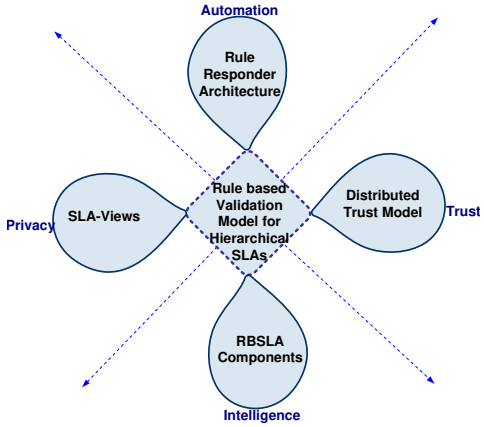

Figure 1. Validation as a Cross-section of Models

the requirement of trust can be addressed through a distributed PKI (Public Key Infrastructure) based trust model. There are two rule based systems contributing in terms of automation and intelligence. Rule Responder [22] weaves the outer shell of the validation system by providing the required infrastructure for the automation of role description of partners as well as steering and redirection of the distributed validation queries. The knowledge representation techniques from the RBSLA (Rule based Service Level Agreements) project [20] contribute at the core of validation system. Different parts of the WS-Agreement compliant SLAs can be transformed into corresponding sets of logical rules, which can compose together during the process of SLA composition and can be decomposed into separate queries during the process of validation. We will discuss these models one by one to find out how they contribute to our proposed validation approach.

2.1 Rule Responder Architecture

Rule Responder (http://responder.ruleml.org) is a rule-based enterprise service middleware for distributed rule inference services and intelligent rule-based Complex Event Processing on the Web. It utilizes modern enterprise service technologies and Semantic Web technologies with intelligent agent services that access external data sources and business vocabularies (ontologies), receive and detect events (complex event processing), and make rule-based inferences and autonomous pro-active decisions and reactions based on these representations (enterprise decision management). For a description of the syntax, semantics and implementation of the underlying logical formalisms and its usage in IT Service Management (ITMS) see [18]. Rule Responder adopts the approach of multi agent systems. There are three kinds of agents:

- Organisational Agents
- Personal Agents
- External Agents

A virtual organization is typically represented by an organizational agent and a set of associated individual or more specific organizational member agents. The organizational agent might act as a single agent towards other internal and external individual or organizational agents. In other words, a virtual organization's agent can be the single (or main) point of entry for communication with the "outer" world (external agents). Similar to an organizational agent, each individual agent (personal and external) is described by its syntactic resources of personal information about the agent, the semantic descriptions that annotate the information resources with metadata and describe the meaning with precise business vocabularies (ontologies) and a pragmatic behavioural decision layer which defines the rules for using the information resources and vocabularies/ontologies to support human agents in their decisions or react autonomously as automated agents/services. The flow of information is from External to Organisational to Personal Agent. Figure 2 shows the Rule Responder agents contributing to SLA validation. Two external agents outside of VO invoke the organizational agent by sending HTML and SOAP messages. Typical examples of external agents are web browser, client service or a workflow tool. It must be highlighted that the overall collaboration between VOs is based on choreography, while the internal collaboration model within a VO (one closed enterprise service network) can be either choreography with no central authority or an orchestration with orchestration workflows defined in the organizational agent as under control of a central authority within this particular VO. Rule Responder can span across several VOs and can support both of the collaboration models. In our scenario Rule Responder provides the rule-based enterprise service middleware for highly flexible and adaptive Web-based service supply chains.

Rule Responder utilizes RuleML [6] as Platform-Independent Rule Interchange Format. The Rule Markup Language (RuleML) is a modular, interchangeable rule specification standard to express both forward (bottom-up) and backward (top-down) rules for deduction, reaction, rewriting, and further inferential-transformational tasks. It is defined by the Rule Markup Initiative, an open network of individuals and groups from both industry and academia. Figure 2 shows Enterprise Service Bus (ESB), the Mule open-source ESB [16], as Communication Middleware and Agent/Service Broker to seamlessly handle message-based interactions between the responder agents/services and with other applications and services using disparate complex event processing (CEP) technologies, transports and protocols. ESB provides a highly scalable and flexible applica-

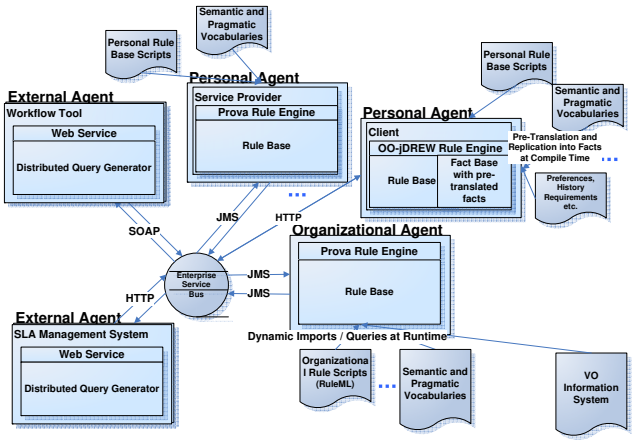

Figure 2. Rule Responder Services for SLA Validation

tion messaging framework to communicate synchronously but also asynchronously with external services and internal agents which are deployed on the bus. A large variety of more than 30 transport protocols provided by Mule can be used to transport the messages. Rule Responder supports Platform-dependent Rule Engines as Execution Environments. Each agent service might run one or more arbitrary rule engines to execute the interchanged queries, rules and events and derive answers on requests and reactions on detected events. Currently the Prova [18] and OO jDREW [4] rule engines are implemented as two rule execution environments.

2.2 RBSLA

The Rule Based Service Level Agreements (RBSLA) [20, 19, 18] project focuses on sophisticated knowledge representation concepts for service level management (SLM) of IT services. At the core of its contract and service level management tool are rule-based languages to describe contracts such as service level agreements or policies in a generic way. The research draws on basic knowledge representation concepts from the area of artificial intelligence (AI) and knowledge representation (KR) and as well as on new standards in the area of web services computing and the semantic web. A particular interest is the investigation of expressive logic programming techniques and logical formalisms such as defeasible logic, deontic logic, temporal event/action logics, transaction and update logics, description logics as a means of deriving formal declarative contract specifications with which to reason about ideal and actual behaviours relating to agreed contract norms (permissions, obligations and prohibitions and their viola-

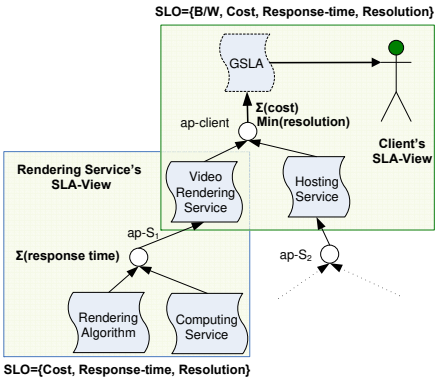

Figure 3. Example Scenario for SLA Views

tions (contrary-to-duty obligations) or exceptions (defeasible prima facie obligations). The important advantages of our approach are the automated verification, validation and consistency checks of large possibly distributed and interchanged rule sets, the automated chaining, (scoped) reasoning and execution of rules and distributed contract modules as well as the flexibility in the dynamic extension with new contract rules (dynamic transactional updates). This facilitates contracts which are flexible and thus able to adapt in order to meet changes to service requirements dynamically with the indispensable minimum of service execution disruption at runtime, possibly permitting coexistence of differentiated contract variants and simplifying contract management and contract execution.

2.3 SLA Views

The concept of Views comes from the databases and has been very successfully adapted in business workflows. Workflow views are employed to separate different administrative domains in workflow coalitions [23].

An 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 structure. Therefore the views of SLA Choreography are quite different from the workflow views. A view in an SLA Choreography represents the visibility of a business partner. Every service provider is limited only to its own view. In figure 3, two different views have been highlighted in an example scenario where a client requires to render and host his videos by using online web services. The Rendering and Computing Service S_1 is restricted to its view and the client is also shown here to have its own view. This scheme can be generalized for all the other partners of this SLA Choreography. A partner (for example a service) makes two kinds of SLAs: the consumer-oriented SLAs and the producer-oriented SLAs. In figure 3, SLAs are shown to be connected to small circles, representing the *Aggregation Points* via certain edges called *Dependencies*. Consumer-oriented SLAs are connected to the aggregation points through the *sink dependencies* and the producer-oriented SLAs are connected through *the source dependencies*. This means that the whole SLA Choreography may be seen as an integration of several SLA-Views. For a rigorous formal model elaborating SLA Views, please see [11].

During the aggregation process, terms of the consumer-oriented SLAs are aggregated. WS-agreement has no direct support for such an aggregation but it gives the liberty to incorporate *any* external schema. We introduce an attribute for aggregation type namely, "$type_a$". The attribute $type_a$ 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 $type_a$ in a formal way, as follows:

Definition (Aggregation function type$_a$) A $type_a \in Types$ is a function that maps a set of tuples to a single tuple which is the aggregation of that set.

$$type_a : tuples(term) \rightarrow term$$

$$type_a(term_1, ...term_n) = term_{agg}$$

We define $type_a$ as an aggregation function that aggregates n terms into one term. Each aggregated term 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. We can define different types of terms namely sumtype, maxtype, mintype, andtype, ortype, and neutral but new types can be added according to the situation. The aggregation process is an incremental process, with aggregation functions applied at each step i.e. every SLA view in the chain [11].

2.4 Distributed Trust Model

We need to choose a suitable trust model that integrates seamlessly with our aggregation and validation model. During service choreography, services may form temporary composition with other services, scattered across different VOs. Whose parent VO will act as the root CA in this case? Public Key Infrastructure (PKI) is a popular distributed trust model that offers certificate containing the name of the certificate holder and the holder's public key, as well as the digital signature of a Certification Authority (CA) for authentication. The public keys are distributed among all the trusted parties, packaged in digital certificates, building trust chains. A solution for dynamic ad hoc networks is the inclusion of a *Third Party Trust Manager* acting as a root CA. We propose a PKI based trust model with a third

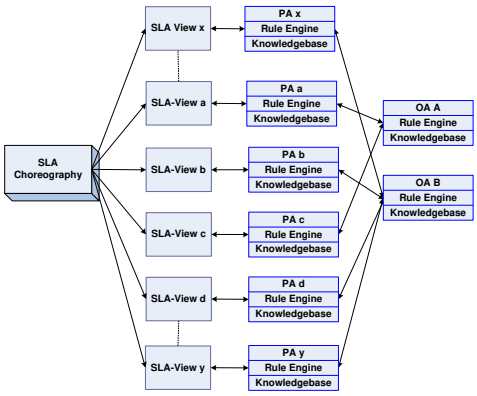

Figure 4. Every SLA-View corresponds to a Personal Agent

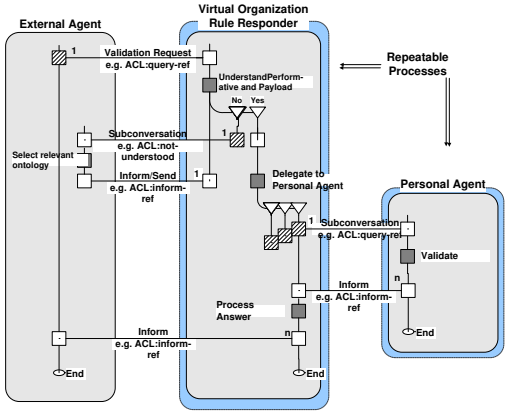

Figure 5. Role Activity Diagram for a simple Query-Answer Conversation

party trust manager that will act as a root CA and authenticate member VOs. Some of those authenticated members may further authenticate other members and services and so on. The authentication layer in each VO middle-ware may be based on Grid Security Infrastructure (GSI) where all resources need to install the trusted certificates of their CAs. GSI uses X.509 [14] proxy certificates to enable Single sign-on and Delegation. With Single Sign-On, the user does not have to bother to sign in again and again in order to traverse along the chain of trusted partners (VOs and services). This can be achieved by the Cross-CA Hierarchical [14] [26]Trust Model where the top most CA, called the root CA provides certificates to its subordinate CAs and these subordinates can further issue certificates to other CAs (subordinates), services or users. SLA views integrate very closely with the trust model to maintain a balance between trust and security. While the trust model promises trust and security, the SLA views protect privacy.

3 Rule based Validation Framework for Hierarchical Aggregation of SLAs

Service Level agreements are frequently validated throughout their life cycle. Runtime Validation ensures that the service guarantees are in complete conformance with the expected levels. WS-Agreement [3] defines a detailed structure of Guarantee Terms with the most important constituents being: Service Level Objectives that define the desired quality of service, Qualifying Conditions that express assertions over service attributes, and Penalty and Reward expressions. These terms are represented as logical rules following the RBSLA specifications. These rules are composed together during the process of SLA aggregation [11], introduced in the section 2.3. The process of validation is performed by using these rules as distributed queries. During the validation process, queries are decomposed making their premises as subgoals. This backward chaining propagates throughout the SLA Choreography. If all the subgoals are satisfied then the validation is successful.

Due to the consumer-oriented aggregation structure of SLA choreography, we propose a top-down validation framework. A top-down validation approach has several advantages in connection with its implementation:

- interfaces can be validated before going into details of modules,
- in case of a problem on higher levels, one does not need to go into lower levels,
- since in the view based SLA aggregation, the top level represents the client's perspective therefore this approach can better translate the on-demand validation queries initiated from the client.

Figure 4 depicts how the Rule Responder and SLA-Views work together to enable this scheme.

Each SLA-View that in fact represents a service provider in the SLA Choreography, is connected to a Personal Agent (PA). SLA choreography is composed of various SLA views. A PA receives the queries from the Organizational Agent (OA) and having the complete information of its consumer oriented SLAs in its knowledge-base, performs the local validation and delivers back the responses on behalf of the service providers.

The complete request pattern starting from the External Agent has been depicted in figure 5. OA intercepts the query at the boundary of a VO and redirects it towards the corresponding PA. Rule Responder architecture supports various multi-agent communication protocols including Agent

Communication Language (ACL) [1]. The trust model facilitates the distributed query to travel across various domains through a single sign-on and delegation mechanism. Referring to this multi-agent architecture coupled with the notion of SLA Views and the distributed trust, the validation process is termed as the *Delegation of Validation*.

4 Delegation of Validation

The aggregation of SLAs is a distributed mechanism and the aggregation information is scattered throughout the SLA choreography across various SLA views. To be able to validate the complete SLA aggregation, the validation query is required to traverse through all the SLA views lying across heterogeneous administrative domains and get validated locally at each SLA view. The multi-agent architecture of Rule Responder provides communication middle-ware to the distributed stake-holders namely the client, the VOs and various service providers. The *Delegation of Validation* process empowered by the *single sign-on and delegation* properties of the distributed trust model, helps the distribute query mechanism to operate seamlessly across different administrative domains.

Now we explain how the Guarantee Terms from a WS-Agreement, expressed as rules, are transformed into distributed queries. We discussed in the section 2.3 how the aggregation functions are applied on the basis of aggregation type of a service term, identified by $type_a$ attribute. SLOs can also be aggregated as conjunctive premises of derivation rules. It is also important to realize that the SLOs refer to an established SLA and their ranges are meant to be guarded in order to maintain desired levels of service. Lets revisit the scenario depicted in Figure 3.

In the scenario, the user is interested to render her videos and then host them on the web. Her requirements include a maximum cost of 45 Euros, maximum response time of 5 seconds, minimum resolution of 640X480 pixels and the minimum bandwidth (from hosting service) of 50 Mbps.

In figure 6, we have depicted this scenario from validation point of view. The user-requirements are shown on the top of the figure, expressed as a derivation rule composed of SLOs of the final aggregated SLA. The agents OA and PA representing the Rule Responder architecture, are shown to automate the distributed query processing. For the sake of simplicity, we have outlined the Rule Responder architecture just from agent-oriented perspective, and have abstracted various essential details such as the Rule-bases, the knowledge resources and the role of Enterprize Service Bus (ESB). The predicates lt and gt denote lesser-than and greater-than respectively. The user requirements are expressed as a set of premises in the following derivation rule:

```
SLO():- ~gt(Cost,45,euro),
```

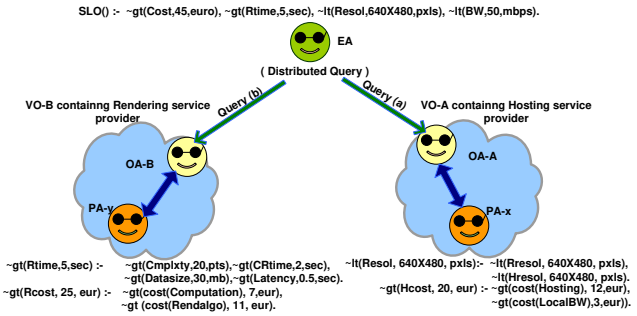

Figure 6. Validation through distributed query decomposition

```
~gt(Rtime,5,sec),
~lt(Resol,640X480,pxls),
~lt(BW,50,mbps).
```

During the validation process, this rule will be decomposed such that each premise will become a subgoal. This subgoal will be sent as a message to the PA corresponding to the next SLA view in the hierarchy where it will emerge as a conclusion of one of the rules in the local rule set, thus forming a distributed rule chain. The initial steps of decomposition procedure are depicted at the bottom of the figure. In the figure, Organizational Agents (OA) have been shown to receive and track the distributed query whenever it enters a new VO. For each service provider, there is a Personal Agent (PA). A PA, after finishing its job, should report to the corresponding OA that will redirect the distributed query to the service provider's PA that comes next in the hierarchical chain. The process continues until the query has found all the goals expressed in terms of logical rules. Active rules tracking these goals or SLOs, are then invoked locally within the administrative domains of the corresponding SLA views. The true or false results are conveyed back following the same routes.

To validate all the guarantee terms of the final (client's) aggregated SLA, the aggregation chunks within all the SLA Views, scattered through the whole SLA Choreography, are required to be validated. In our scenario, OA-B receives a subgoal $\sim gt(Rtime, 5, sec)$ representing the requirement that the total response time of the system should not be more than 5 seconds. This SLO depends on several factors such as the complexity of the rendering algorithm, size of the data, latency and response time of the computational hardware which is expressed as the new subgoal:

```
~gt(Rtime,5,sec):-~gt(Cmplxty,20,pts),
~gt(CRtime, 2,sec),~gt(Datasize,30,mb),
~gt(Latency,0.5,sec).
```

The SLO expressing the cost will be divided between the two service providers as shown in the Figure 6. The

service cost at the level of OA-A should be less than 20 and is dependent on the sum of the cost for hosting and the cost for local bandwidth. The varying upper limit of cost at different levels reflect the profit margins of different providers e.g. the provider in OA-A has a profit margin of 5 Euros.

It should be noted that in accordance with the WS-Agreement standard, there are three arguments in each SLO, denoting: the SLO name, its value and its unit respectively. The delegation of validation, continuing across various levels, reaches the SLA views originating the corresponding SLOs, and the SLOs get validated there. At each level, the corresponding reward and penalty conditions are also checked and if required, appropriate action is taken. The distributed Rule Responder agent architecture acts as an enabling technology for the SLA Views concept. One of its important features is that we can implement principles of autonomy, information hiding and privacy with the agent approach. For instance, the details how a particular service level objective is measured and computed in a personal agent might be hidden (e.g. a third-party monitoring service) and only the result if the service level is met or not might be revealed to the public. Another important aspect is that the monitoring/validation might run in parallel, i.e. several service provider (PAs) might be queried by an OA in parallel via messaging. For instance, a complex SLOs might be decomposed by the OA into several subgoals which are then sent in parallel to the different services (PAs) which validate them.

Qualifying Conditions and penalty and reward expressions can be expressed through Event Condition Action (ECA) rules. For example, if we want to express the statement "If the response time of the service is larger than 60 seconds then there is a penalty of 5 Euros", we can write its equivalent in WS-Agreement as follows:

```
<wsag:Penalty>
  <wsag:AssesmentInterval>
    <wsag:TimeInterval> 60
    </wsag:TimeInteval>
    <wsag:Count> 1 </wsag:Count>
  </wsag:AssesmentInterval>
  <wsag:ValueUnit> Eur </wsag:ValueUnit>
  <wsag:ValueExpr> 5 </wsag:ValueExpr>
</wsag:Penalty>
```

This can also be represented by ECA rules:

```
timer(sec,T):- Timer(T),interval(1,min).
event(Violate):- ping(servic1,RT),RT>60.
action(Penalty):- penalty(Obligation,5).
```

Now combining together:

```
ECA(monitor):- timer(sec,T),
event(violate), action(penalty).
```

The above rule is activated according to the timer(sec, T) which is defined by the following rule, invoked after every minute:

```
timer(sec,T):-Timer(T),interval(1,min).
```

Similar approach can be used for the renegotiation, fault tolerance and breach management processes. During renegotiation, the distributed query traverses in the same way towards the service providers, offering those terms which are desired to be renegotiated. During fault tolerance and breach management, violations are localized through a similar invocation of the distributed query. The combination of ECA rules and using derivation rules to implement the different parts of an ECA rule provides high expressiveness and can be very easily transformed in a rule based markup language such as RuleML [6]. RuleML allows to declaratively implement the functionality of each part of a Reaction Rule (event, condition, action etc.) in terms of derivation rule sets (with rule chaining), thus making them processable in autonomic and autonomous way.

5 Related Work

The concept of Workflow Views is utilized to maintain the balance between trust and security among business partners [24]. Schulz et al [23] have introduced the concept of view based coalition workflows. Chiu et al [8] present a meta model of workflow views and their semantics based on supply chain e-service but their model lacks an integrated cross-organizational perspective. Other authors [24, 13], however, do propose a global view or a decomposition process based on the views. But none of them have focussed on the dynamic workflows in their approach. Chiu et al [9] describe a contract model based on workflow views. They construct an e-contract model that defines e-contracts in plain text format.

A little research has been carried out towards dynamic SLA composition of workflows [5] [10, 25]. The research area corresponding to the management of such aggregated SLAs is still wide open. Ganna Frankova [10] has highlighted the importance of this issue but she has just described her vision instead of any concrete model.

RBSLA [20] transforms SLAs into logical rules to automate their management and monitoring. The authors discuss knowledge representation of SLAs with complex business rules and policies. RBSLA [19, 20] uses a combination of Horn Logic, Deontic Logic and ECA (Event-Condition-Action) rules. RBSLA also covers many related areas such as the breach management, authorization control, conflict detection and resolution, service billing, reporting, and other contract enforcements. RBSLA employs query driven, backward reasoning for SLA management.

Oldham et al [17] have extended WS-Agreement by building a rule based ontology on the WS-Agreement. Their SWAPS schema [17] transforms constructs from the Guarantee terms into predicate based markup language. They admit that their schema is limited to a specific domain.

The Grid Security Infrastructure (GSI) and the security modules of middle-ware, provide a set of security protocols for achieving mutual entity authentication between a user (actually a user's proxy) and resource providers [26]. GSI uses X.509 proxy certificates (PCs) to enable Single sign-on and Delegation [14].

6 Conclusion and Future Work

In this paper, we presented the design of a validation framework for hierarchical SLA aggregations corresponding to cross-VO workflow compositions. This rule based validation framework employs a top-down validation mechanism based on distributed query processing. The validation framework assumes the hierarchical aggregation of SLAs [11]. The validation framework also assumes unique consumers for the providers of a value chain in the hierarchy. In the future, we plan to implement the distributed rule based validation system based on RuleML, through iterative development phases, and adhering to the WS-Agreements standard.

References

[1] Agent Communication Language Specifications. http://www.fipa.org/repository/aclspecs.html, last access: Feb 25, 2009.

[2] NESSI-Grid, last access: July 12, 2008.

[3] Web Service Agreement (WS-Agreement). GFD.107 proposed recommendation, last access: July 12, 2008.

[4] M. Ball, H. Boley, D. Hirtle, J. Mei, and B. Spencer. The OO jDrew Reference Implementation of RuleML. In *RuleML 2005*, Galway, 2005.

[5] M. B. Blake and D. J. Cunnings. Workflow composition of service level agreements. International Conference on Services Computing (SCC2007), 2007.

[6] H. Boley. The Rule-ML Family of Web Rule Languages. In *4th Int. Workshop on Principles and Practice of Semantic Web Reasoning*, Budva, Montenegro, 2006.

[7] I. Chebbi, S. Dustdar, and S. Tata. The view based approach to dynamic inter-organizational workflow cooperation. *Data and Knowledge Engineering*, 56:139–173, 2006.

[8] D. Chiu, S. Cheung, S. Till, K. Karalapalem, Q. Li, and E. Kafeza. Workflow view driven cross-organisational interoperability in a web service environment. *Information Technology and Management*, 5:221–250, 2004.

[9] D. Chiu, K. K. Q. Li, and E. Kafeza. Workflow view based e-contracts in a cross-organisational e-services environment. *Distributed and Parallel Databases*, 12:193–216, 2002.

[10] G. Frankova. Service Level Agreements: Web services and security. *Springer Verlag, Berlin Heidelberg*, pages 556–562, 2007.

[11] I. U. Haq, A. Huqqani, and E. Schikuta. Aggregating hierarchical service level agreements in business value networks, submitted in a conference and available at http://www.pri.univie.ac.at/irfanulhaq/.

[12] I. U. Haq, A. A. Huqqani, and E. Schikuta. A conceptual model for aggregation and validation of SLAs in Business Value Networks. accepted in the 3rd International Conference on Adaptive Business Information Systems , Leipzig, Germany, 2009.

[13] Q. Li, D. Chiu, Z. Shan, P. Hung, and S. Cheung. Flows and views for scalable scientific process integration. First International Conference on Scalable Information Systems, Hong Kong, 2006.

[14] A. Lioy, M.Marian, N.Moltchanova, and M. Pala. PKI past, present and future. *International Journal of Information Security, Springer Berlin*, page 2006, pages 1829.

[15] D. R. Liu and M. Shen. Workflow modeling for virtual processes: an order-preserving process-view approach. *Information Systems*, 28:505–532, 2002.

[16] Mule. Mule Enterprise Service Bus, http://mule.codehaus.org/display/MULE/Home, 2006.

[17] N. Oldham, K. Verma, A. Sheth, and F. Hakimpour. Semantic WS-Agreement partner selection. Proceedings of the 15th international conference on World Wide Web, Edinburgh, Scotland, 2006.

[18] A. Paschke. *Rule-Based Service Level Agreements - Knowledge Representation for Automated e-Contract, SLA and Policy Management*. Idea Verlag GmbH, Munich, 2007.

[19] A. Paschke and M. Bichler. SLA representation management and enforcement. The 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service, 2005.

[20] A. Paschke and M. Bichler. Knowledge representation concepts for automated SLA management. *Int. Journal of Decision Support Systems (DSS)*, March 2006.

[21] A. Paschke, H. Boley, A. Kozlenkov, and B. Craig. Rule responder: RuleML-based agents for distributed collaboration on the pragmatic web. Proceedings of the 2nd international conference on Pragmatic web Tilburg, The Netherlands, 2007.

[22] A. Paschke, B. Harold, A. Kozlenkov, and B. Craig. Rule Responder: A RuleML-Based Pragmatic Agent Web for Collaborative Teams and Virtual Organizations, http://ibis.in.tum.de/projects/paw/, 2007.

[23] K. A. Schulz and M. E. Orlowska. Facilitating cross-organisational workflows with a workflow view approache. *Data and Knowledge Engineering*, 51:109–147, 2004.

[24] M. Shen and D. R. Liu. Discovering role-relevant process-views for disseminating process knowledge. *Expert Systems with Applications*, 26:301–310, 2004.

[25] T. Unger, F. Leyman, S. Mauchart, and T. Scheibler. Aggregation of Service Level Agreement in the context of business processes. Enterprise Distributed Object Computing Conference (EDOC '08) Munich, Germany, 2008.

[26] S. Zhao, A. Aggarwal, and R. D. Kent. PKI-based authentication mechanisms in grid systems. International Conference on Networking, Architecture, and Storage, 2007.