Evaluation framework for service level management in federated service management context

Tuomas Nurmela[1]

[1] TietoEnator Processing & Network Oy, Espoo, Finland
Tuomas.Nurmela@tietoenator.com

Abstract. Web Services are being adopted for integration, yet to enable a full-blown service-oriented architecture approach for federated service management, a comprehensive support infrastructure is required. Focusing on service level management in the service management arena, this paper describes a frame of reference and uses it to evaluate service level management –related support infrastructures in research projects from a service federations perspective. While all the projects surveyed provide useful insights, areas of research still remain.

Keywords: service federations, web services, service level agreements

1 Introduction

Web Services are fast becoming the approach for software interoperability and service oriented architecture implementation. If utilized in a manner following service-oriented architecture (SOA), the approach is said to provide benefits such as adaptation to local and environmental changes, management of environment heterogeneity and autonomous decisions over service implementation [5]. While the basic SOA model requires just a service directory, in order to actually realize these benefits, a more comprehensive support infrastructure is required. The support needs to include development and design time support as well as runtime support, covering also runtime collaboration and interoperability at technical, semantic and pragmatic levels in federated services approach [1,2]. Service level management (SLM) is part of the web service management arena in this B2B middleware area, enabling moving from price-based competition to service differentation through measurable differences in service capabilities. The focus on measurable metrics is one of the things that separates it from the wider context of QoS parameters in general. To establish itself, SLM needs both SLA languages [17] and design and runtime support infrastructures.

From a business IT service management perspective SLM is the process that contains all the activities relating to service level agreements (SLAs, negotiated or selected service agreements) and their management [6]. In business environments, SLM as a process loosely speaking contains the activities of defining SLAs, negotiating SLAs (or buyer selection based on classes of service), monitoring and evaluation of SLAs and managing breaches of SLAs. SLM also contains the notion of pricing the different levels of service and reporting the service trends and breaches to the customer. This business-centric approach can be seen as the central difference between thinking about technical QoS and management of SLAs.

As can be seen, the SLM process activities are nearly the same as for eContracting [2,8]. However, the difference lies in the scope: in open, dynamic environments, eContracting is required to negotiate and agree the common process between collaborators (e.g. when forming a virtual breeding environment) and between a virtual organization instance and customers when forming an external contract and ensuring that what is agreed will be honored by all parties. Likewise, issues such as the capability to utilize the support infrastructure in a federation is required, although in SLM, the focus is only on managing the SLA commitments. SLM also complements the present work on extended service-oriented architectures (SOA) [3][4], while taking into account autonomy of members and adaptation to local and environmental changes [5].

This paper introduces a frame of reference for service level management in web services context in Section 2 to enable comparison of approaches. Section 3 provides a short review of a number of support infrastructures that are partial solutions for SLM. The paper concludes with a summary of the reviews, noting the especially usable contributions for SLM in federated service management while suggesting additional issues that need to be developed further.

2 Service level management

The discussion of service level management is dependent on the type and scope of agreements as well as the agreement management lifecycle. In terms of *types of SLAs* [6], service providers typically create both internal SLAs and external SLAs. *Internal SLAs* define the requirements between service producers. *Operational Level Agreements (OLAs)* codify what is expected of different units within the service provider company that offers the service to customers. If the service provider utilizes a third party as sub-contractor to provide the service, an *underpinning contract (UC)* is created between the third party and the provider. *External SLAs* codify what is being offered to the external customer. A central tenant is that internal SLAs relating to the service (whether OLAs and UCs) are more stringent than external SLAs. SLAs contain among other things *SLA parameters* (e.g. availability), with each having a service level objective (SLO), i.e. target value for the given SLA parameter.

The different types of SLAs relate especially to organizational form, i.e. whether the virtual organization is a temporary organizational structure like a consortium or a more permanent structure such as a partnership [7]. The virtual organization in practice requires means of either aggregating the SLAs to determine the composite SLA for the whole service (offers-based approach) or using the external SLA in the contractual agreement with the customer to make negotiation demands on the potential members in the virtual organization (reverse-auctioning approach). The latter assumes the service provider either takes the risk that fulfillment of service is not really possible or uses an already existing virtual breeding environment as the basis for negotiation, without having negotiated the details with participating members.

Alternatively service providers could approach the issue as a risk management scenario and include SLA breach-related monetary compensation to service pricing without regard to actual requirements. However, intuitively this does not lead to long customer relationships given that customer probably cannot negotiate the actual financial loss as part of the breach management payoff.

SLA contract scope needs to be considered in addition to considering the different roles that may be related to producing the service. The SLAs can either deal with technical metrics or it can deal with business metrics as part of the eContract. Ideally the technical metrics can be aggregated to business metrics. Yet the business metrics are domain dependent. Therefore, the mapping is problematic.

Figure 1 describes a suggestion for minimal content with in regard to different types of SLA and eContracting. Possibility for separation of SLA management from the eContracts provides benefits in terms of reuse and breadth of situations to which the language can be applied. The separation of technical metrics from business metrics supports system modularity and specification of third party roles in order to manage a specific area of responsibility (e.g. monitoring and evaluation of purely technical SLA parameters). This approach would benefit from indicating dependencies between different metric types.

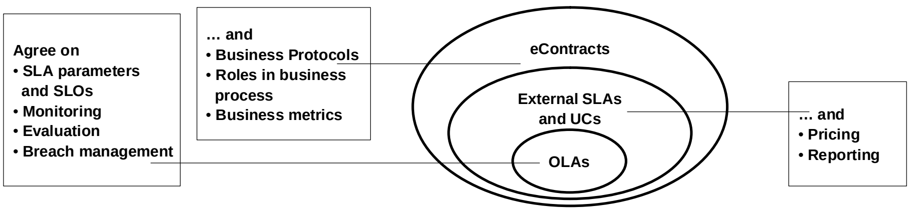

Figure 1: Minimal scope of contract content from SLM perspective.

In addition to the contents of the agreements and the scope of content amongst the involved parties, the service level management lifecycle has to be determined. In the following, the steps of template design, SLA-enhanced process design, negotiation and selection, monitoring, evaluation, breach and bonus management and reporting are identified. The lifecycle is captured in Figure 2. This is loosely based on the ITIL SLM process description [6] and the eContracting process [8].

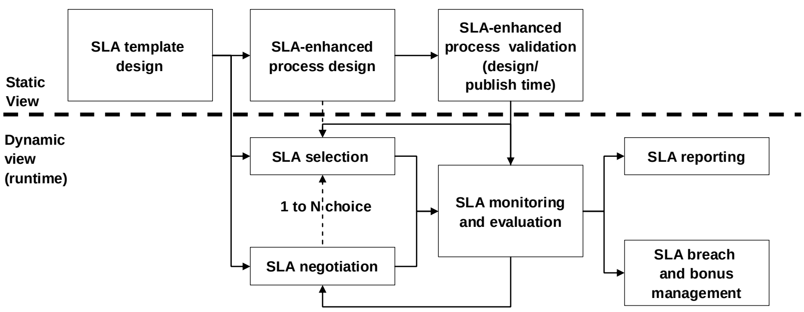

Figure 2: Frame of reference for SLM.

The SLA template design consists of defining the XML elements of the SLA. If the SLA is to be negotiated, SLOs are dynamically established. Only SLA parameters and parameter boundaries need to be defined. Alternatively, if a class of service–approach is used, classes need to be defined. This means defining the SLA parameters and the SLOs prior to offer of the service. The class of service approach is beneficial in the sense that possible conflicting technical demands (e.g. minimal latency but assured delivery) can be screened and will not need runtime resolution. However, because customer specific requirements cannot be matched, it fits better to environments focusing only on technical metrics. The template design is particularly impacted by the SLA language design choices.

The SLA-enhanced process design relates to utilization of composite services: SLAs may be involved at design time of the process (composite service), especially if the process is private and therefore only internal SLAs are involved. SLA-enhanced process design requires that process design tool supports SLAs.

After creation of the process, the SLA-enhanced process design may be *validated* at design or web service publishing time. This requires extending the type repository to include SLA validation support.

At runtime, after deployment of service, the consumer either *negotiates* the required SLOs or *selects* an appropriate class of service. In the case where services are provided in an open market, it is possible that the web service consumer participates in an auction for the best possible web service. This would require a negotiation mechanism with support for multiparty negotiation. Alternative approaches include the capability to select an identical service from each service provider and only provide payment for the fastest [9]. In addition, the offered services can be provider resource-constrained. In this case the negotiation may be may revolve around multiple consumers competing in an auction for single provider resources.

As can be seen, the SLA determination can be modeled as a full-blown auction or bargaining scenario. However, this is typically not required in practice, because of SLA having limited scope. Likewise, the negotiation can be separated under a separate negotiation protocol.

The monitoring of SLA parameters contains at least two issues: 1) the monitoring can be done either in-band or out-of-band and 2) link between monitoring and evaluation can be passive, reactive or proactive [8]. *Out-of-band monitoring*, following a typical probe-approach, is suitable for performance metrics. *In-band monitoring* on the other hand can be located on the service host providing host or on a separate tier consisting of e.g. access control, message routing and XML firewall protecting the service. Especially non-performance based metrics utilize in-band monitoring. *Passive monitoring link* merely refers to logging monitoring data at run time. Evaluation is done later as a separate action. *Reactive monitoring link* provides the means for evaluation of SLO breaches for corrective actions. *Proactive monitoring link* would support the use of internal thresholds prior to SLO breach and actions that would try to ensure breach of SLO would not happen. Evaluation therefore includes threshold evaluation in addition to SLO breach evaluation.

The evaluation of SLOs can be based on different *modes*, being *event-based* (with e.g. schedules) or *request-based*. Likewise it can support *complete evaluation* (i.e. utilize all available monitoring data) or *statistical evaluation* (i.e. evaluate only a sample of monitoring data). *Evaluation accuracy* is dependent on the monitoring data sources: for an example, if availability data source consists of trouble tickets, a human element is involved. On the other hand, in case of an end-to-end poller, frequency of polling establishes accuracy.

SLA breach management governs breaches for both proactive thresholds and SLOs. As such it is closely tied to the monitoring link. For an example with passive monitoring link, breach management is typically done a posteriori by people. While little research on automated breach management is available, intuitively (i) this is done by consumer and/or provider and (ii) not all possible mechanisms fit the different monitoring link (i.e. reactive or proactive) types. Intuitively a number of mechanisms are possible, including compensations, renegotiation, redesigning the process, forcing a virtual organization evolution, making monetary compensations and making reputation evaluations.

SLA bonus management could provide additional monetary or reputation bonuses based on over-performance of a member. If no bonus management is utilized, degradation of service is a provider option, though this is suitable only in completely automated services.

SLA reporting in all likelihood needs to provide both operational reporting and management reporting. This is especially important for the next evolutions of workflow systems, which suffered in comparison to ERPs due to lack of reporting facilities [10].

3 SLM support infrastructures

In the following, examples of different types of support infrastructures for SLM are discussed. The following attributes can be used to review these: 1) context of the solution based on background and approach, 2) solution scope to determine completeness of support in regards to the SLM frame of reference, 3) relationship to particular SLA language, 4) use of selection of class of service or negotiation of SLA and whether this mechanism is integrated to the support infrastructure, 5) main concepts used, 6) monitoring mechanism used, 7) evaluation mechanism used, 8) reporting capabilities, 9) breach and bonus management capabilities, 10) pricing support, 11) SLA support for web services composition and 12) extensibility and interoperability of the approach with other solutions.

3.1 Web Services Offering Infrastructure (WSOI)

Web Services Offering Infrastructure (WSOI) [11] [12] has been developed to implement the Web Services Offering Language. The implementation has not been deployed.

WSOI is basically an XML parser and a SOAP engine extension, which provides the in-band processing of WSOI artifacts. The *XML parser* provides means to validate WSOL files as well as transform the WSOL service offerings and related files (ontologies, WSDL files) to the XML document object model (DOM) trees and separate symbol tables. These contain the data required for semantic validation of the WSOL extracted from the ontologies.

While currently not available, the research group is looking to extend this functionality with a *code generator* that provides interceptors (WSOI specific handlers) for QoS measurement, evaluation and accounting. Separate Web Service Offering Descriptors (WSOD) are used to define the chaining order of the interceptors. The parser and code generator are depicted in Figure 3a.

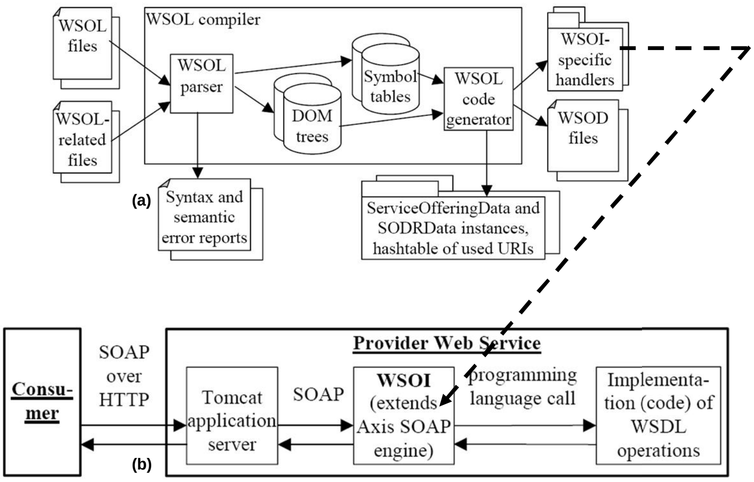

Figure 3: WSOL parser and code generator (a) and WSOL architecture (b) [11].

Regardless of whether the WSOI handlers are manually or automatically created, they are currently in practice Apache Axis handlers. These provide the means for SOAP request, response and error message interception and manipulation in Axis. Through this WSOI is able to both provide the WSOL service offers (predefined classes of service) as well as monitor the service requests and replies in-band. The monitoring information can be used to evaluate fulfillment of service offerings. Figure 3b contains the architecture overview.

From deployment perspective no discussion is provided about whether all provider web services should really incorporate the extended SOAP engine, or if this should be placed e.g. on a corporate SOAP intermediary (“SOAP router”). This could process all WSOI extensions and hide the additional functionality from the actual application servers providing the web services.

While WSOL supports pricing, there seems to be no discussion on integrating payment compensations into the solution. On the other hand reporting and web services composition support seem to be outside the scope of WSOI. From the architectural perspective, there seems to be no reason why payment support through billing integration and reporting could not be extended to support the implementation. It is possible that the lack of real world deployments is the reason why these areas of implementation are not discussed. Composition support is a separate issue altogether as WSOL makes implementation of this challenging [11, pp. 63].

3.2 TrustCoM SLA management

TrustCoM is a research project (www.eu-trustcom.com) carried out in the EU 6th Framework program (Networked Business and Governent). The project started in 2004 and is to conclude in the first half of 2007. The project focuses on a holistic evaluation of virtual organization (VO) concept. TrustCoM uses WSLA [13] to describe SLAs. As WSLA only focuses on describing the SLA content, TrustCoM SLA negotiation is done by a separate negotiation protocol.

The TrustCoM middleware [14] SLA management sub-system can be partitioned among participants. These include *local SLA management services*, which contain SLA monitoring and management. A separate trusted third party (TTP, i.e. WSLA supporting participant) provides an *SLA evaluator service*. These are depicted in Figure 4a.

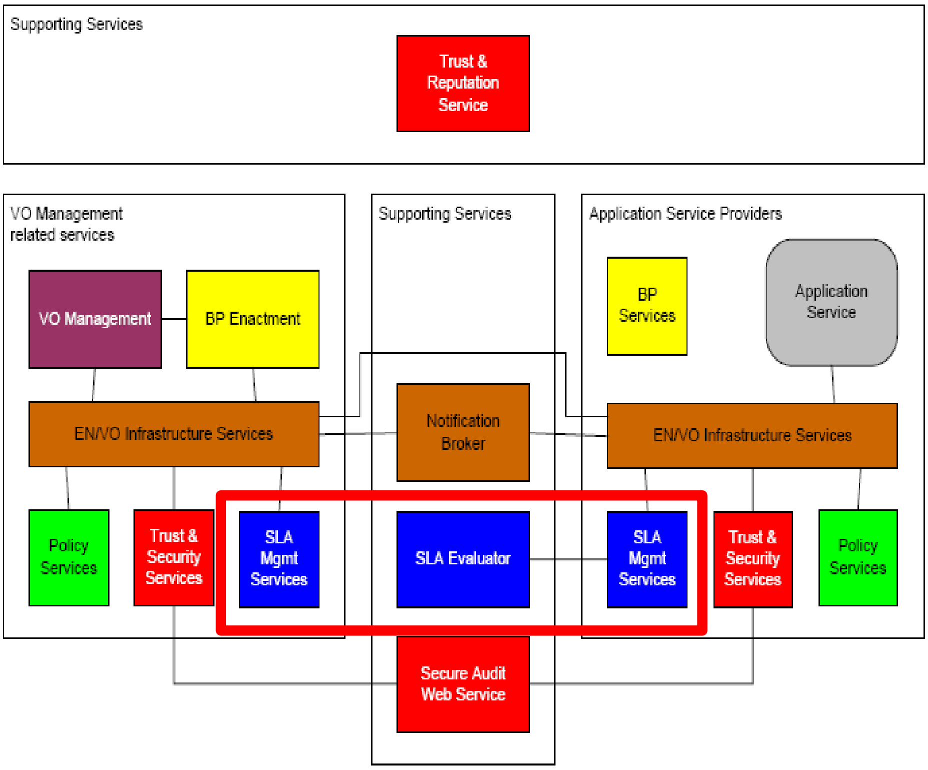

Figure 4a: TrustCoM architecture with SLA management subsystem highlighted [14].

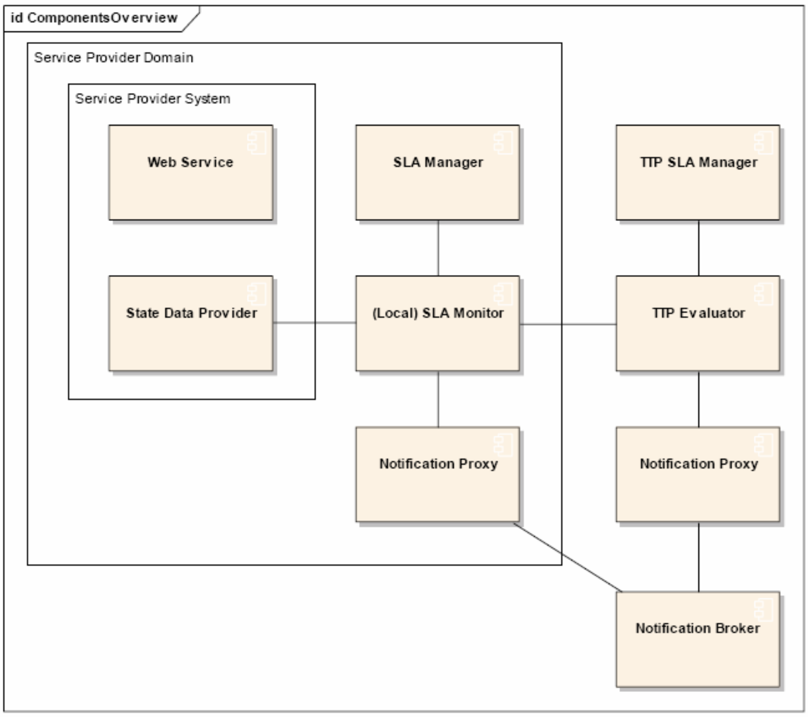

Figure 4b: SLA management subsystem [14].

TrustCoM SLA management sub-system (depicted in more detail in Figure 4b) focuses on monitoring and evaluation, utilizing the notification service within the TTP notification broker and VO infrastructure services subsystem. Breach management through notification event handling can be provided by separate subsystems. One possibility would be the reputation services under Trust and Security Services subsystem. Another possibility would be VO reconfiguration under the Business Process Enactment and Orchestration subsystem.

An SLA is instantiated through a negotiation process initiated by the TTP SLA management service. Currently TrustCoM does not seem to contain a comprehensive negotiation processing logic, but rather just exchanges predefined WSLA descriptions.

Local SLA managers utilize the SLA provided to configure local monitoring. Currently monitoring of resources is based on (i) using Windows Management Instrumentation for reading Windows performance counters, registry information etc (ii) Java Management Extensions used in the application services themselves and (iii) the GANGLIA monitoring tool available for both Linux and Windows for unified systems monitoring. These are collective called the State Data Provider.

The monitoring data from the State Data Provider is handled by the local SLA monitoring service, which calculates the composite metrics as per WSLA design. These are provided to the evaluator based on a schedule (push) or as reply to request (pull). The TTP SLA evaluator compares the measurement results to the WSLA service level objectives. The SLA Evaluator utilizes the TrustCoM Notification service to publish notifications for both breaches and fulfillments of service, without regard to what is being done with these. The assumption is that the related parties (e.g. reputation service, VO manager) subscribe to SLA Evaluator notifications and handle the events.

TrustCoM provides a very extensible approach, by using the notification subsystem to integrate various parts of the system. This suits the current project organization well, as design and implementation of sub-systems is divided among a number of parties. If the notifications are to be used more thoroughly, a more comprehensive pricing and reporting mechanism would be included and some approach to support SLAs in composition of services, TrustCoM would be as close to a complete solution for federated services as one would intuitively imagine. As it stands, the first version of the support infrastructure still provides a foundation for practical applied research.

3.3 Web Services Management Network

The Web Services Management Network (WSMN) [15] [16] was produced as a pilot project by HP. This uses an SLA language supporting SLA template design. Thiis is used with WSFL to describe the business processes to which WSMN SLAs relate.

WSMN members each have the same capabilities which are used to create an overlay network that communicates with out-of-band messages based on a number of WSMN specific protocols. Each WSMN member represents a different organization, WSMN only focuses on the public processes (i.e. process organizational interfaces). In order to identify particular business protocol process flows, a (separate or business protocol provided) global identifier (GUID) is used by the members.

The WSMN architecture contains an SLA engine, measurement engine, business correlation engine and a message handler for WSMN protocols. These are depicted in Figure 5a along with related applications.

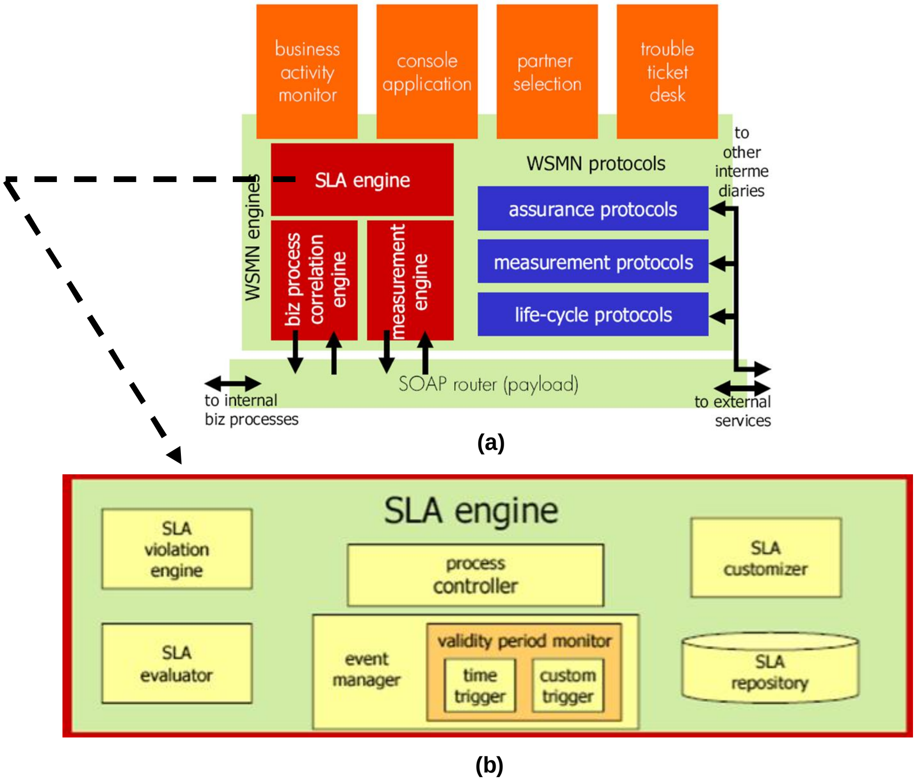

Figure 5: WSMN intermediary (a) and SLA engine (b) [16].

The *SLA engine* comprises multiple components. A *process controller* manages the initiation and flow control. During initiation it provides the *SLA customizer* to configure an event manager and input the SLOs to an *SLA repository*. The SLA repository also contains SLO validity period information. The validity period of the SLOs defines their evaluation period, which is not necessarily right after the initiation. *The Event manager* is responsible for both managing timed measurements and event-based measurements during the SLO validity period. Measurements are pushed to the *SLO evaluator*, which compares the measurements to the SLA repository SLO information. SLO breaches are pushed to a separate *SLA violation engine*, which has the required logic for breach management. SLA engine is depicted in Figure 5b. Note that, compared to WSLA notions of signing and supporting party in TrustCoM, all functionality of these parties is provided by a single WSMN member functionality.

The *Business process correlation engine* (BPC engine) communicates with a workflow engine API or logs to establish runtime measurements of the public process. The BPC engine can also be used to manage events to modify to the process execution if an API is available. The HP prototype utilizes HP Process Manager as the workflow engine and its Java API to manipulate execution (possibly to e.g.

replace misbehaving members). The *measurement engine* on the other hand provides environment measurements by abstracting a unified management interface for different types of external probes (e.g. SNMP agents, WMI data providers etc).

Beyond BPC and the measurement engine, WSMN external information regarding the private process can be gathered from a separate *business activity monitor (BAM)*. All WSMN members have a model of the business process in a separate model repository which defines the measurement targets. These are utilized to store the measurement data in an operational database, which is utilized by the SLA engine.

WSMN message handler is responsible for three classes of out-of-band protocols to manage the federation (i.e. partners) and SLAs. *Lifecycle protocols* are a set of four basic partner connectivity and cooperation protocols. Initiation protocols provide a means to initialize the management network. The initiation protocol has a hardcoded bootstrap mechanism; difficulties with secure negotiation protocol are not discussed. Keepalive protocols provide means to monitor WSMM members. Clock synchronization protocol provides time synchronization among WSMN members. A teardown protocol provides means for explicit signoff. Possible use of existing protocols or the details of the protocols are not described in the articles.

Obviously lifecycle management is not an integrated part of the SLA management. E.g. in TrustCoM it is handled by the Business Process and Enactment and Orchestration subsystem in conjunction with other subsystems.

Measurement (exchange) protocols (MEPs) provide means of exchanging measurement information amongst WSMN members. The authors note that although WSLA supports bilateral agreements (i.e. two signing parties), it is possible that the web service is related to a third party that is more than a supporting party (e.g. in the case of service provider providing composed services without using choreography to generate a shared view of the public web services interfaces). Their view is that in this case, instead of making internal SLAs and a separate external SLAs or decomposing the SLA to subsets with indicated responsible parties, the SLA measurements should be exchanged with a separate MEP in order to evaluate the SLA.

The MEPs approach seems to be contain unaddressed problems, yet these are not discussed by the authors. These seem to stem from the decoupling of SLA signing parties from the members participating in the execution. This has legal and trust management implications. From a legal perspective, if legal binding of contract is established between SLA signing members what is the contractual obligation of a WSMN members participating but outside the SLA? From a trust perspective, how would consumers establish trust to these external members outside the SLA?

Assurance protocols provide means for management network optimization and integration to interface systems related to management. Negotiation protocol provides SLA negotiation. Trouble ticket exchange protocol provides means to integrate the WSMN node to trouble ticket systems. The authors note the assurance protocols are outside the focus of their work. Details of the protocols are not provided. Utilization of service directories, reputation services and TTP negotiation services to support functionality of this layer is merely mentioned.

The WSMN implementation also contains a visualization tool for reporting. This can utilize the WSMN intermediary databases (both in memory and RDBMS) to provide a view to all metrics gathered as part of SLA monitoring. Overall, whereas TrustCoM focuses to a great extent on enabling virtual organization related specific topics (dynamic reconfiguration, negotiation of partners and impact to SLAs), WSMN seems to be more focused on integrating workflow systems and providing an exchange point between service management and workflow engine performance.

3.4 Summary

A comparison of the support infrastructures is provided in Table 1 on the basis of review in previous subsections. The focus here throughout has been on runtime functionality of the support infrastructures. As noted in the SLM frame of reference, issues such as process design and SLA-aware matchmaking functions need support in federated services context. These were not comprehensively discussed in any of the support infrastructures reviewed and therefore have not been addressed here.

Attribute	WSOI	TrustCoM	WSMN
Background and approach	Network QoS, Runtime support infrastructure	Service management, Runtime support infrastructure	Service management, Runtime support infrastructure
SLA language	WSOL	WSLA	"WSMN SLA"
Solution completeness	Partial	Partial	Complete
Scope of agreements	External SLAs	External SLAs	eContracts
Negotiation vs. selection	Selection integrated	Negotiation; by separate mechanism	Negotiation; by separate mechanism
Main elements	Apache Axis handlers, (XML parser and code generator)	SLA manager, SLA monitor, TTP SLA evaluator	SLA engine, measurement engine, business correlation engine, WSMN message handler
Monitoring capabilities	In-band monitoring	In-band and out-of-band local monitoring, external out-of-band monitoring	In-band monitoring, possible to integrate to BAM
Evaluation capabilities	Integrated to monitoring	External SLA evaluator TTP service	Integrated SLO evaluator service
Reporting capabilities	Unknown	Unknown	Separate console application
Pricing support	Yes	Yes	Unknown
Breach management capabilities	Monetary compensation, Management operations	Management operations; separated	Through workflow engine control
SLA support for web services composition	No	No	No
Extensibility and interoperability	Not defined	Separate subsystems for monitoring and evaluation	Integrated system with interfaces to BAM and trouble ticket system

Table 1: Comparison of web-services SLM-related support infrastructures.

4 Conclusions

The presented support infrastructures all provide some aspects, which can be used in the design of SLM subsystem for federated service management. In particular, WSOI provides support for class of service approach, which might also be suitable for VO internal SLA relationships, to establish the composition matchmaking prior to actual agreement of external SLAs (i.e. SLAs between the business customer web service and the composition web service provider).

TrustCoM on the other hand has for a modular approach, which supports division of work, including separation of negotiation, local and external monitoring and evaluation. Likewise, with its event-based integration of subsystems, breach management can later complement these extensively.

WSMN on the other hand considers direct interaction of SLA engine and external applications such as the business process enactment engine. The possibility of controlling an instrumented workflow based on SLA evaluation results is evident.

In general, the work reviewed is best suited to support the design of the SLM support infrastructure for federated service management. None of the solutions are a perfect fit. They lack of explicitly defined and implemented composition support. Also, they could benefit from separation of different levels of SLA negotiation and separation of external SLA (i.e. customer-VO) negotiation from internal SLA negotiation. Finally, support for service pricing and billing systems integration has been left outside the discussion.

Acknowledgments. This paper presents a part of a survey performed for the CINCO group as part of the author's thesis work.

References

1. Lea Kutvonen, Toni Ruokolainen, and Janne Metso, "Interoperability middleware for federated business services in web-Pilarcos", International Journal of Enterprise Information Systems, 3(1):1-21, January 2007
2. Lea Kutvonen, Janne Metso, and Sini Ruohomaa. From trading to eCommunity population: Responding to social and contractual challenges. In *Proceedings of the 10th IEEE International EDOC Conference (EDOC 2006)*, Hong Kong, October 2006.
3. Mike P. Papazoglou, "Service oriented computing: concepts, characteristics and directions", In *4th International Conference on Web Information Systems Engineering (WISE'03)*, 2003.
4. M. P. Papazoglou, D. Georgakopoulos, "Service oriented computing", Communications of the ACM, Vol 46, Issue 10, 2003, pp. 25-27.
5. Mundimar P. Singh, Michael N. Huhns, *Service-Oriented Computing: Sematincs, Processes, Agents*, John Wiley & Sons, 2005.
6. OCG, *ITIL Service Delivery*, The Stationary Office, 2001.
7. Luis M. Camarinha-Matos 1 and Hamideh Afsarmanesh 2," Virtual Enterprise Modeling and Support Infrastructures:Applying Multi-agent System Approaches" in M. Luck et al (eds), *Multi-Agent Systems and Applications*, ACAI 2001, LNAI 2086, 2001, pp. 335-364.
8. Z. Milosevic, A. Berry, A. Bond, K. Raymond, "Supporting business contracts in open distributed systems," In *2nd International Workshop on Services in Distributed and Networked Environments*, 1995.

9. Heiko Ludwig, "Web Services QoS: External SLAs and Internal Policies, Or: How do we deliver what we promise?" IBM research center report, 2003.
10. Jorge Cardosa, Robert M. Bostrom, Amith Sheth, "Workflow Management Systems and ERP Systems: Differences, Commonalities, and Applications", Kluwer, Information Technology and Management 5, 2004, pp.319-338.
11. Vladimir Tosic, Service Offerings for XML Web Services and Their Management Applications, PhD Thesis, Carleton University, Department of Systems and Computer Engineering, August 2004
12. V. Tosic, W. Ma, B. Pagurek, B. Esfandiari, "Web Service Offering Infrastructure – A management infrastructure for XML Web Services ", 2004, pp. 817-830
13. Heiko Ludwig et al., "Web Service Level Agreement (WSLA) Language Specification", Version 1.0, revision wsla-2003/01/28. available from: www.research.ibm.com/wsla/WSLASpecV1-20030128.pdf
14. TrustCoM, "TrustCom Basic Reference Implementation", Deliverable D19, 2004
15. Akhil Sahai, Vijay Machiraju, Mehmet Sayal, Li Jie Jin, Fabio Casati, "Automated SLA monitoring for web services", HP Research report, HPL-2002-191, 2002
16. Vijay Machiraju, Akhil Sahai, Aad van Moorel, "Web Services Management Network – an overlay network for federated service management", HP Research report, HPL-2002-234, 2002
17. Tuomas Nurmela, Lea Kutvonen, "Service level agreement management in federated virtual organizations", accepted for publication in 7th IFIP International Conference on Distributed Applications and Interoperable Systems (DAIS 07), 2007