
بسم الله الرحمن الرحيم

SUDAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

End-to-End Service Level Agreement Monitoring Framework

A thesis submitted in fulfillment of the
Requirements for the award of the degree of
Doctor of Philosophy (Computer Sciences)

ALMAHDI IBRAHIM KHOJALI MOHI ELDEEN

Supervisor: Prof. Dr. Robert Michael Colomb

Co-Supervisor Dr. Abdelgaffar Hamid Ahmed

February 2018

بسم الله الرحمن الرحيم

يَا أَيُّهَا الَّذِينَ آمَنُوا أَوْفُوا بِالْعُقُودِ ۚ أُحِلَّتْ لَكُم بَهِيمَةُ الْأَنْعَامِ إِلَّا مَا يُتْلَىٰ عَلَيْكُمْ غَيْرَ مُحِلِّي الصَّيْدِ وَأَنتُمْ حُرُمٌ ۗ إِنَّ اللَّهَ يَحْكُمُ مَا يُرِيدُ (1)

صدق الله العظيم

[المائدة :١]

O you who have believed, fulfill [all] contracts. Lawful for you are the animals of grazing livestock except for that which is recited to you [in this Qur'an] - hunting not being permitted while you are in the state of ihram. Indeed, Allah ordains what He intends.

DEDICATION

I dedicate this thesis to

My mother Batool,

My father Ibrahim,

My sons Ibrahim and Mohammed,

My daughters Rzan and Aisha

For their devotion and endless support.

ACKNOWLEDGEMENT

I cannot express enough thanks to our leaders Shams Aldeen, Mohammed, Shama, Aisha and Kamal Almahdi for their continued advice and encouragement.

Also, I cannot express enough thanks to all SUST staff and my colleagues for their help to reach this an academic degree.

I want to deeply thank my brother Nasir for financial support and my wife, Kwakib for doing the role of father beside all house activities.

A lot of thanks to my supervisor, Prof. Robert M. Colomb and co-supervisor Dr. Abdelgaffar Hamid Ahmed for Providing success factors during research and writing.

ABSTRACT

The current trend in modeling and designing systems follows a new paradigm called Service-Oriented Architecture (SOA) In this approach the functionality of the system is assigned to loosely coupled services where integration between heterogonous systems is possible. This situation highlights the necessity of conducting a contract which called Service Level Agreement (SLA). SLAs in SOA framework is still new, but recently it became imperative due to the high demand for services in SOA systems to be provided cross over organizations. SLAs are meaningless without monitoring the quality of the service which two parties or more agreed upon. Sometimes service provider, in turn, gets the service from other providers (supply chain) to deal with this situation, four End-to-End performance guarantee strategies have been proposed, Audit trail at each stage, the third-party do audit trail, End-to-End performance predicate, and cost versus performance trade-off. Also, the End-to-End monitor has been designed using the third party do audit trail strategy to monitor the performance and to perform audit trail in case of SLAs violation detected. Furthermore, Web Service Agreement Language (WSLA) is extended using UML profile to be more informative. Also, metric ontology has been developed to figure out semantic heterogeneity problem. End-to-End monitoring has a number of aspects such as SLAs in a supply chain are not public, Information exchange problem and End-to-End performance measuring problem. Proposed End-to-End SLAs monitoring framework resolves all these issues. Thus our proposed solution is distinct.

المستخلص

الاتجاه الحالي في نمذجة وتصميم الانظمة يتبع طريقة جديدة تسمى المعمارية خدمية التوجه . في هذه الطريقة تحدد وظائف النظام بتكامل مجموعة من الخدمات التي يمكن استخدامها من أنظمة مختلفة حيث ان الارتباط بين هذه الخدمات هو ارتباط حربمعنى يمكن ان ترتبط مع بعضها دون الحوجة لمعرفة نظام التشغيل الذي تعمل تحته أو أي تقنية برمجية تستخدم. هذه الظروف أبرزت ضرورة إبرام عقد. وهذا العقد يسمى الاتفاق على مستوى الخدمة. الاتفاق على مستوى الخدمة في إطار المعمارية خدمية التوجه أصبح مهما جدا وذلك للحوجة الماسة لإبرام هذا النوع من العقود بين المنظمات التي تتبادل الخدمات المختلفة في أنظمة المعمارية خدمية التوجه. الإتفاق على مستوى الخدمة ليس له أي معنى بدون مراقبة جودة الخدمة التي أتفق عليها طرفين أو أكثر. كما نعلم الخدمات تقدم من مقدم الخدمة الى المستهلك , ولكن أحيانا الخدمة تقدم عبر سلسلة, من مقدم الخدمة الى موردين الى المستهلك. للتعامل مع مثل هذه الظروف أقترحت أربعة إستراتيجيات لضمان الأداء من مقدم الخدمة للمستهلك وهي مراقبة الاداء في كل نقطة, طرف ثالث يقوم بالمراقبة , إستراتيجية الاسناد وإستراتيجية الموازنة بين التكلفة والأداء.أيضا صمم مراقب لمراقبة السلسلة من المقدم الى المستهلك ليراقب ويجمع المعلومات المطلوبة ويتعقب العمليات في حالة إكتشاف أي خروقات لبنود الاتفاق , أستخدمت استراتيجية الطرف الثالث في تطوير هذا المراقب. كذلك إستخدمنا UML profile لجعل لغة الاتفاق على خدمات الصفحات تعبر عن خصائص لم تكن تستطيع أن تعبر عنها. أيضا صمم Metric Ontology لحل مشكلة عدم التجانس. مراقبة الاداء من المقدم للمستهلك لها جوانب عديدة مثل عقود الاتفاق ليست متاحة للجميع , مشكلة تبادل المعلومات , قياس الاداء. ان إطار العمل المقترح حل كل هذه المشاكل مما جعله تصميما متميزا.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS /ABBREVIATIONS

SOA	Service-Oriented Architecture
ICT	Information and Communication Technology
SLA	Service Level Agreement
QoS	Quality of Service
WSLA	Web Service Level Agreement
WS	Web Service
ISP	Internet Service Provider
KPI	Key performance indicators
WSDL	Web Service Description Language
SOAP	Simple Object Access Protocol
UDDI	Universal Description, Discovery, and Integration
OGF	Open Grid Forum
XML	Extensible Markup Language
SCM	Supply Chain Management
BLAs	Business Level Agreements
BLOs	Business Level Objectives
MBE	Model Base Engineering
UML	Unified Modeling Language
OMG	Object Management Group
W3C	World Wide Web Consortium
OWL	Web Ontology Language
RDF	Resource Description Framework
MOF	Meta-Object Facility
OTP	One-Time Password

SMS	Short Message Service
CRM	Customer Relationship Management
SFA	Sales Force Automation
B-to-B	Business to Business
B-to-C	Business to Customer

CHAPTER 1

INTRODUCTION

1.1 Background

Service-Oriented Architecture (SOA) is a widely used approach in modeling and designing service-based systems. In SOA, the functionality of the system is assigned to loosely coupled services where integration between heterogonous systems is possible. To provide quality of service in such environment, one needs to conduct a contract between service provider and service consumer. In ICT industries, this contract is often called a Service-Level Agreement (SLA). SLA is a contract or an agreement between two parties, a service consumer and a service provider, to define the obligations between them. SLA typically specifies the definition of the service, problem management, and performance measurements. SLA is meaningless without monitoring because the Quality of the Service (QoS) could not be evaluated. Monitoring SLA is needed to avoid failure and to ensure that a service satisfies the pre and post conditions (Winkler et al. 2008). Moreover, monitoring SLA help in service recovery if required. Finally monitoring help organizations to manage its resources. Many studies had been published in monitoring SLA, but few of them deal with a supply chain SLAs and even these few studies did not explain how to monitor End-to-End SLAs in details.

1.2 Problem Statement

Monitoring SLAs requires collecting statistical information to measure the QoS. It further complicated by the fact that a service may be provided through a supply chain. The consumer does not receive a service from a provider directly instead he receives the service from the supplier who in turn receives the service from the provider. There are three parties—consumer, supplier, and provider, the research questions are:

- How do End-to-End SLAs be monitored?

- How to do audit trail in case of SLAs violation?
- What is the appropriate strategy to provide End-to-End performance guarantee?

1.3 Research Objectives

- To investigate in End-to-End performance guarantee strategies
- To create metrics ontology to resolve metrics conflict problem.
- To establish WSLA profile to represent derived, predicted and predicated association which links between Business to Business metrics and Business to Customer metrics (multi-values association).
- To build End-to-End SLAs monitoring framework to help SLAs monitor designers to develop End-to-End SLAs monitor.
- To evaluate the proposed approach.

1.4 Research Scope

There are many Quality of Service (QoS) such as performance, security, availability, usability, etc. This thesis concerns only with two QoS availability and performance.

1.5 Contributions

End-to-End monitoring has a number of aspects such that SLAs in a supply chain are not public, information exchange problem, End-to-End performance measuring. Proposed End-to-End monitor design resolves these issues. To figure out interoperability among autonomous systems namely semantic heterogeneity problem, a metric ontology is developed. Also, a UML profile is presented for modeling multi-valued association.

1.6 Thesis Outlines

The thesis structured as follows. Chapter 1 introduces the research motivations, problem statements, research objectives, scope in addition to contribution. Chapter 2 presents SLA aspects, SLA monitoring, and supply chain management. SLA parameter, SLA metrics, metrics collection approaches are provided in chapter 3. In chapter 4 ontology concepts are explained. Chapter 5 detailed audit trail and

investigated End-to-End performance guarantee strategy. Case studies are described in chapter 6. Proposed solution is provided in chapter 7. Results are discussed in chapter 8.

CHAPTER 2

SERVICE LEVEL AGREEMENT

2.1 Introduction

When you purchase a service, how do you know you are getting the QoS as agreed upon? If you buy a camera, you can know pretty quickly whether it works because a camera has specifications and warranty. If it doesn't, then you can convince the store easily that it needs repair under warranty. When the camera is repaired, you can verify that the camera now works. The same is true in a Business to Business (B to B) environment. If an organization buys materials, the materials are inspected on delivery and only accepted if they meet their specifications. The work is done throughout the supply chain. The camera store buys the cameras from a manufacturer, who buys parts from suppliers, who in turn purchase materials from other suppliers. At each stage, the products are inspected for conformance to specifications. How about services, suppose you are an organization and what you are buying is internet service from an Internet Service Provider (ISP). The service is needed because organization's employees need to access websites from many places in the world, including let us say Amazon.com for your ISP to enable a query on Amazon.com, and the ISP needs the cooperation of:

- Telecommunication Company.
- National or regional internet backbone (Domain Name Server, etc).
- The telecommunication company linking the ISP to the backbone.

This entire supply chain cooperates in producing the service, which lasts only a second or two. There may be thousands of instances of the service per day. How to specify the service provided by your ISP? How to apportion blame for a service failure? Complex services are determined by a contract between the provider and customer. In ICT industries, this contract is often called a Service-Level Agreement (SLA). SLAs are fundamental to success and a basic of a good client relationship. An SLA enables a service providers and consumers to evaluate the QoS.

2.2 Quality of Service

Quality of service (QoS) plays a vital role in service selection, but it is complicated to define the QoS term because it includes a countless of features and a fine details

or distinction that depend on the system been described. QoS is fundamental for the discovery, selection, and composition of the services. Therefore it requires an in-depth investigation to provide a complete service model. There are two categories of qualities that can be specified in SLAs: measurable and immeasurable. Measurable qualities can be measured automatically using metrics such as security, flexibility, usability and the most critical quality of service required in web services are availability besides performance. Immeasurable attributes are those that cannot be measured automatically from a given viewpoint for example, determining the cost of changing a service (modifiability) is difficult to automate (Dobson 2005).

2.3 SLA Definition

SLA is an essential artifact defines the obligations between the service provider and service consumer in which services and the level of quality are specified. SLA is a prediction agreement between two parts (Al-sagaf 2012). SLA typically defines the definition of the service what you are buying, nonfunctional requirement measurement for example availability and performance, problem management what to do if something goes wrong, service consumer and provider duties, warranties which specified regarding the performance measurements, disaster recovery. It could be mentioned a service level agreement (SLA) is a contract or agreement that formalizes a business relationship, or part of the relationship, between two parties. Most often it takes the form of a negotiated contract made between a service provider and a consumer and defines a price paid in exchange for an entitlement to a product or service to be delivered under specific terms, conditions, and with particular financial guarantees.

2.4 Importance of Service Level Agreement

Service Level Agreement is Important because it sets boundaries and expectations by constituting a single document that contains the terms of the agreement as understood by both parties. With the SLA in place, it is much more difficult for either side to claim ignorance if the agreement breaks down. SLA addresses five essential aspects. Firstly what the provider is promising. Secondly how the provider will deliver those promises. Thirdly who will measure delivery, and how? Fourthly what happens if the provider fails to provide the QoS. Fifthly how the SLA will change over time. SLA contain key performance indicator, by having these

indicator established, it is accessible to achieving customer satisfaction. SLA drives internal processes by setting a clear, measurable standard indicators objectives become more transparent and more comfortable to measure. Finally, SLA makes the relationship between customer and provider bright and positive for SLA includes defined penalties which make the customer understands that the provider truly believes in its ability to provide a level of QoS as described in an SLA.

Organizations turn to third-party outsourcing for many reasons such as to obtain expertise or to reduce costs, application maintenance and help desk operations (Nadeem 2005). Companies delegate their IT-intensive business processes to an external provider who, in turn, owns, administrates the selected process (es), based upon defined and measurable performance metrics. In this situation, it is essential to implement a contract that allows managing the service efficiently (Schmidt 2000). SLA is critical to ensure sufficient outsourcing engagements. So SLA is a crucial factor in long-term success.

2.5 SLA Example

Suppose a large organization X wants to outsource its employee transportation in a given city to a taxi company. Any travel within the city on official business not using the employee's transport will be done by the selected taxi company.

2.5.1 Service Definition

Carrying employees of company X with baggage up to 100 kg from any destination to any other destination within the cities of Khartoum and Omdurman. At least 1000 and not more than 5000 movements per calendar month. The contract says what the service is. It carries people and their baggage, but not, for example, unaccompanied parcels or more massive shipments of goods. It also doesn't include carriage to Port Sudan. The number of trips allows the service provider to organize taxis and drivers.

2.5.2 Performance Measurement and Warranties

There is need to define some indicators of customer satisfaction. Service consumer hope the service to be on time, so a good performance indicator would be the time elapsed between making an ad hoc call and the arrival of a taxi.

2.5.3 Warranties

Warranties associated with these indicators would be a maximum waiting time. Say 15 minutes for an ad hoc call and 5 minutes after a pre-booked time. Since there

will be many service instances during the life of the agreement, the warranty would likely be expressed in statistical terms, say an average wait time of 10 minutes for an ad hoc call, with 95% of service instances less than 20 minutes. Pre-booked times average 3 minutes late with 95% less than 10 minutes. Averages will be over a longish period, say a month. For the warranties to have any force, they must be accompanied by financial penalties and perhaps rewards. This brings into play two additional factors: the price the contractor will charge, and factors outside the control of the contractor. Because service requests are to some extent random, the shorter waiting time in the warranty the greater excess capacity required, resulting in increased costs for the contractor as idle time will increase. The waiting time warranty will be negotiated with the customer along with the service price and the penalties for failure to meet the guarantees. The contractor will not want to be responsible for waiting time outside their control. For example severe traffic congestion. The warranty will need to be worded to exclude these situations, and some method will have to be agreed to measure traffic congestion. The drivers should know their way around the area, or at least have reliable access to directions. Also drivers should be clean and presentably dressed, and the taxis must be honest and in good repair. Both a provider and a consumer have to think how to measure these indicators. One way is by complaints. If penalties apply, then there would need to be some form of dispute resolution procedure to prevent frivolous claims.

2.5.4 Customer Duties

The consumer is co-produces in the service. At least the customer places the booking and gives an address for the destination. The agreement might specify that the customer must make a pre-booked request at least two hours before the booked time. Otherwise, the booking must be ad hoc. This gives the contractor time to outsource a pre-booking if none of their taxis are available. It might also specify that the customer not brings any food or drink into the taxi, and not smoke. The upper limit of 5000 calls per months would go here. The agreement might allow the parties to negotiate for more calls in a given month.

2.5.5 Disaster Recovery

Disaster recovery (DR) and business continuity refer to an organization’s ability to recover from a disaster and unexpected event and resume operations. Organizations must have a set of policies, tools, and procedures to enable the repair or

continuation of service. This kind of clause is standard in IT-based services because of the possibility of failure of the computing or communication infrastructure. But there is no scope for disaster in the taxi service case.

2.5.6 Termination of Agreement

There will be a section in the SLA that allows one party or the other to terminate the agreement before its normal expiry if warranties are consistently not met or problems cannot be resolved. In the taxi example, the customer might be able to terminate the agreement if taxis are late more than a certain number of times in a month, or if taxis do not arrive at all after a valid booking has been made. The contractor might be able to terminate the agreement if the customer consistently fails to make payment. After defining a contract an SLA representation language is required.

2.6 SLA Languages

A service-level agreement is a part of a standardized service contract where a service is formally defined using any of service levels agreement languages such as web service level agreement (WSLA) and WS-Agreement.

2.6.1 WSLA Language

WSLA framework was proposed by IBM in 2001 and it is based on XML and defined as an XML schema. Primarily, the WSLA allows the creation of machine-readable SLAs (Bianco et al., 2008). WSLA allows authors to specify the performance metrics which associated with a web service application, desired performance targets, and actions that should be performed when performance is not met (Wikipedia 2018). However, the WSLA language is extensible to deal with other service-based technologies and other technical fields such as network storage. WSLA language encompasses a set of standard extensions that allow WSLA authors to define complete agreements that relate to Web services and include guarantees for response time, throughput and other common metrics (Ludwig et al. 2003) see figure below.

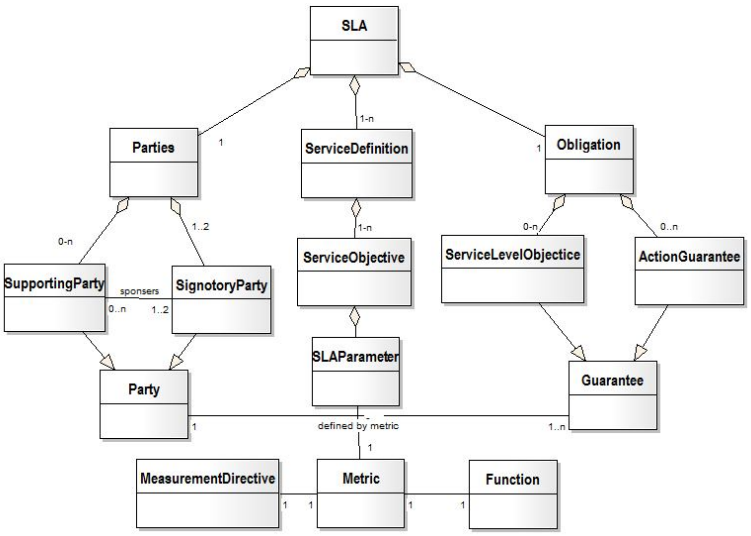


Figure 2-1 WSLA Metamodel

Figure 2-1 shows that WSLA document contains three main parts SLA parties, service definition, and obligation. Parties describe the parties involved in the management of the Web Service. Service definitions describe the services the WSLA is applied to. Obligations define the service level that is guaranteed concerning the SLA Parameters specified in the service definition section. Service level objectives contain a formal expression of the guaranteed condition of a service in a given period. Action guarantees represent promises of parties to do something, for example, to send a notification in case the guarantees are not met.

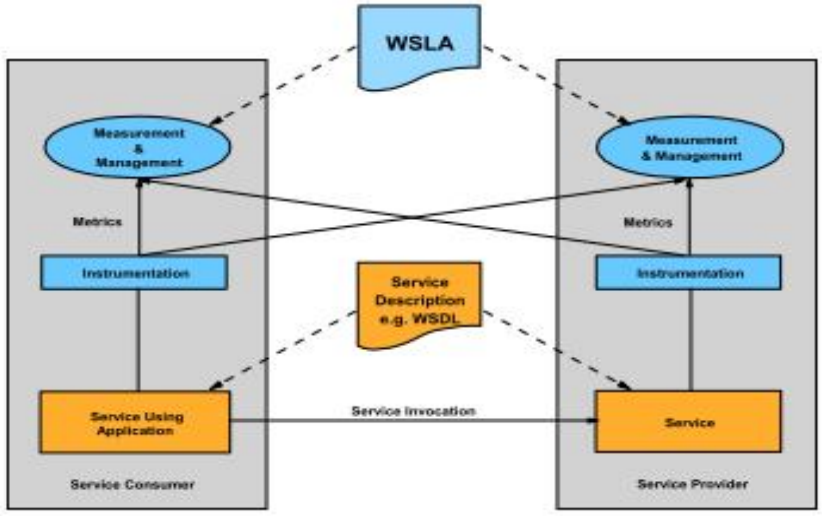


Figure 2-2 Role of a Web Service Level Agreement

A complete WSLA document is composed of all the information negotiated and agreed upon by the two parties. In many scenarios, one of the parties (e.g., the service provider) will define most of the content of a WSLA, and a service customer may merely agree to such information, and provide additional specifications. The authoring process can be off-line. Alternatively, the WSLA creation can be negotiated in an online process. A template can be published in a registry such as Universal Description, Discovery and Integration (Ludwig et al. 2003).

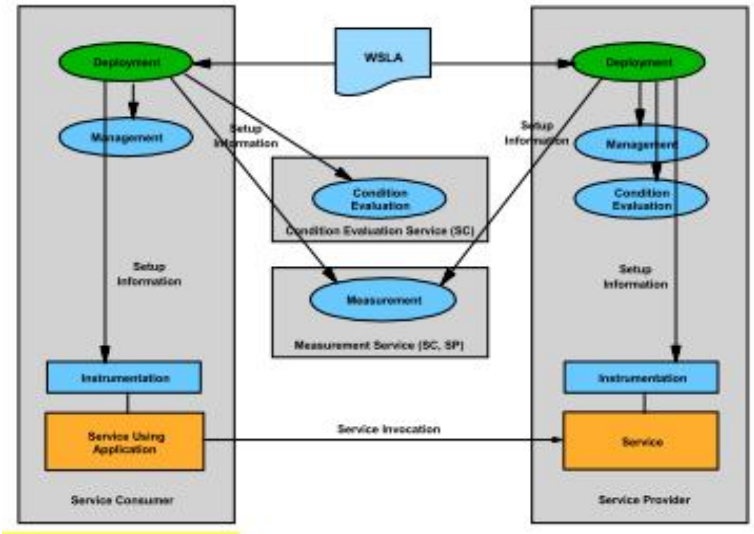


Figure 2-3 Deployment Process

2.6.2 WS-Agreement Language

WS-Agreement is web Services protocol for establishing an agreement between two parties, such as between a service provider and consumer, using an extensible XML language for specifying the nature of the contract, and agreement templates to facilitate discovery of compatible agreement parties. The specification consists of three parts a schema for defining a contract, a schema for specifying an agreement template, and a set of port types and operations for managing agreement life-cycle, including creation, expiration, and monitoring of agreement states. WS-Agreement refers to all aspects of agreement content as terms, and all agreement terms are

listed as WS-Policy assertions. It is left to the parties to make sure that the essential elements of an agreement are captured as agreement terms (Andrieux et al. 2007).

Andrieux and others (2007) argue that the goal of WS-Agreement is to provide the mechanisms needed to enable Web Services applications to specify agreement terms for the usage of their service. Explicitly, this specification defines both the context and terms under which agreement applies. Agreement Context contains metadata about the involved parties and services. Agreement Term contains domain-specific Web Service agreement information. Also, WS-Agreement includes a core set of grammar elements to indicate how the contained agreement terms apply.

2.6.3 WSLA vs. WS-Agreement

Like the WSLA, the WS-Agreement provides an XML schema that defines the overall structure of an agreement document. Also, the WS-Agreement specification defines a protocol for negotiating and establishing agreements dynamically based on web services (a set of WSDL Definitions).A critical difference between the WS-Agreement and WSLA is that the structure of a WS-Agreement is highly extensible. It contains several sections where intended users are expected to define domain-specific elements and properties. Another difference between the WS-Agreement and WSLA is that the former does not provide a means to specify the metrics associated with parameters used in the agreement. Instead, metrics are defined in any structure required by a domain-specific extension. WSLA and WS-Agreement used to represent an SLA. This representation is required to manage and control services which provided through a supply chain.

2.7 Supply Chain Management

It is clear that SLA is a contract between service provider and service consumer, but the consumer in many cases gets the service from the supplier who in turn receives the service from the provider, so the service provided through a supply chain. A supply chain is a system of organizations, people, activities, information, and resources involved in moving a product or service from supplier to ultimate customer (Wikipedia 2015).

The Supply chain management (SCM) is concept implemented in many areas such as business and supported by an information system. Supply chain management has

been defined as the "design, planning, execution, controlling, and monitoring the supply chain activities with the objective of creating net value, building a competitive infrastructure, leveraging worldwide logistics, synchronizing supply with demand and measuring performance globally (Wikipedia 2015). SCM draws heavily from the areas of operations management, logistics, procurement, and information technology, and strives for an integrated approach (Bartsch and Frank 2013).

Supply chain management is essential to company success and customer satisfaction. SCM boost customer service. Also, SCM reduces operating costs. Moreover, SCM improves financial position by increases profit leverage, decreases fixed assets and increases cash flow.

Supply chain management requires monitoring the supply chain activities and measuring performance. Supply Chains Management facing many challenges such as interoperability, semantic heterogeneity, customer preferences. To enhance QoS, providers have to redesign their supply network and meet their customers QoS in a way that's transparent for customers (David 2014). Interoperability and semantic heterogeneity will be discussed in more details in chapter 4.

2.8 Summary

SLA is a contract which defines the QoS that the provider must provide to the consumers. Sometimes service provided through a supply chain. To manage the supply chain, monitoring SLAs is required. Next chapter will explain these sufficient details.

CHAPTER 3

SLA MONITORING

3.1 Introduction

The goal of monitoring contractual Service Level Agreements (SLAs) is to measure the quality of service, to evaluate whether the provider provides the QoS to the consumer as agreed upon or not. Monitoring contractual SLAs involves collecting statistical information to assess whether the provider is delivering the level of QoS stipulated in a contract signed between the provider and the consumer (Molina-Jimenez et al. 2004).

3.2 SLA Monitoring Motivation

Service Level Agreements need to be monitored at runtime to ensure that the Business Level Agreements (BLAs) and Business Level Objectives (BLOs) are indeed satisfied in the realized business workflow and allow the organization to adjust its business processes best to the environment. Service monitoring can play an important role in cutting testing costs. Ameller and others (2008) show that monitoring SLAs is useful for such tasks. Monitoring SLA help provider and consumer to avoid failures by replacing an unavailable service with another service of the same functionality. Also, monitoring enforces the SLA compliance between the provider and the consumer. In addition to ensuring that a service satisfies the pre- and post conditions. Finally monitoring SLA enabling recovery activities if required.

3.3 SLA Monitoring Strategies

Generally monitoring SLAs strategies can be classified into two types passive and active as shown in figure 3-1. Active or online done at runtime while passive or offline using simulation mechanisms. The active monitoring approach generates test traffic periodically or on-demand, and then measures the performance of test packet or response. Passive monitoring approach captures the traffic by mirroring or splitting and analyzes the captured packets. Table 3-1 explains advantages and limitation of active and passive monitoring.

Monitoring strategy	Advantages	Limitations
Active monitoring	- Detect violation before it happens - Accurate results	- Cause overload
Passive monitoring	- No overload	- Less accurate

Table 3-1 Monitoring Strategies Advantages and Limitations

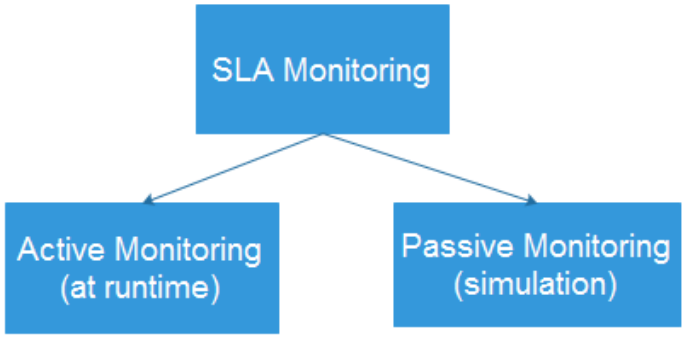


Figure 3-1 SLA monitoring Strategies

3.4 QoS Required in Web Service

A lot of attributes can be measured such as maintainability, portability, usability, and reliability. But all these attributes are software design characteristics; they are not supposed to change during execution time. Ameller and others (2008) explained that the primary requirements for supporting QoS in Web services are availability and performance. Availability is the quality aspect of whether the Web service is present or ready for immediate use. Availability represents the probability that a service is available. While performance is the quality aspect of Web service, which is measured in terms of throughput and latency. Higher throughput and lower latency values represent the excellent performance of a Web service. To describe the QoS, parameters are needed.

3.5 SLA Parameter and Metrics

Parameters are the main elements of the description of a service. A parameter describes an observable property of a service whose value can be obtained from a source of measurement. SLA Parameter can be used as a guarantee of an SLA. SLA parameters are specified by a set of metrics. These metrics determine the measures that need to be gathered in order to verify whether the SLA parameters are being met. To facilitate SLA management, SLA parameters must be reasonable. They should motivate involved parties to act in a manner that is mutually beneficial. Also, SLA parameters should be attainable. Parameters that are beyond the control of either party should not be included. Furthermore, they must be measurable (Bianco et al. 2008).

3.5.1 SLA Metrics

Bianco and others (2008) explain that metrics are used in process control, software process improvement, business strategy implementation, and basically any field where data has to be collected in order to verify whether goals are being met. Metrics defines service properties. Properties extracted from a service providing system or computed from other metrics and constants. Metrics are the principal instrument to describe precisely what SLA Parameters mean by specifying how to measure or calculate the parameter values. They reflect the commitments made in the contract and the SLAs, and they allow continuous tracking of the service being delivered and determine whether service delivery conforms to the agreed-upon SLA. Determining metrics requires a long thought process because if done wrong it may actually do more harm than good. Literature in these fields indicates that creating metrics is a difficult task, thus there are many criteria for establishing right metrics.

3.5.2 Criteria to Select Metric and Metrics Usage

There are many criteria to select metrics such as it must be easy to be understood by users. And it must be bias from different technologies used. Also, it must be agreed upon at least by the provider and its customers, and preferably by a third party as well. In addition to being derived from a formal specification that is the basis of the contractual commitment. Be part of a community process and not

owned or biased by the provider. Be useful for diagnosis, forecasting, and what-if scenarios. Equally important to being associated with a service level objective.

Metrics are a central component in the monitoring of SLAs. Metrics allow continuous tracking of the service being delivered and determine whether service delivery conforms to the agreed-upon SLA. The consumer and provider can use their metrics to monitor the performance and react or make decisions on the service based on the monitoring results. Furthermore, the consumer and provider can use their metrics to monitor the performance to improve the QoS. So metrics are useful for both provider and consumer.

3.5.3 SLA Parameter and Associated Metric Example

```
<SLAParameter name="TransactionRate"type="float"unit="transactions / hour">
    <Metric>Transactions</Metric>
    <Communication>
        <Source>ACMEProvider</Source>
        <Pull>ZAuditing</Pull>
        <Push>ZAuditing</Push>
    </Communication>
</SLAParameter>
```

The example shows an SLA parameter named "TransactionRate" that is based on the metric "Transactions". ACMEProvider is in charge of providing this value. The SLAParameter's Communication Type represents information on how the current values of an SLA parameter can be interchanged between parties. This interaction can be either proactive, that is, an update is sent to defined parties each time a new value is computed (push), or other parties can retrieve the current value of an SLA parameter whenever they want (pull). For the interaction of SLA parameter values, it is assumed that standard operations are used to facilitate this interaction. The "ParameterUpdate" operation is the standard operation for "pushing" values to other parties; the GetSLAParameterValue operation facilitates the "pull". The interfaces of both operations are defined as Web service in Web Service Description Language (WSDL).

3.5.4 Metrics Collecting Approaches

There are several issues involve in a metric collection for example what is suitable metrics collection approache passive (packet sniffing) or active (packet interception, probe with synthetic operations). From what point or points of view (provider, service consumer or network in between) are the metrics to be collected? Also, who is in charge of collecting the metrics? Besides what information can be deducted from the collected metrics? With these questions in mind and without paying attention to implementation details, metric collection techniques can be divided into four general categories (Molina-Jimenez et al., 2004). The first approach is service consumer instrumentation. In this scheme; the metrics are collected by the service consumer. The second one is a provider instrumentation approach. In this scheme, the provider is in charge of collecting the metrics about the performance of its own resources. Trusted third party instrumentation is the third approach. In this approach trusted third-party periodically probes the provider to measure its response. The fourth approach is a network packet collection with request-response reconstruction approach, in this schema, a metrics collector is installed somewhere in the path between the provider and the service consumers.

3.5.5 Web Service Performance Metrics

In this age where consumer and provider are won and lost in a second, continual evaluation and optimization of web properties is essential. There are two categories of web performance metrics, server Side metrics (Business to Business B to B) such as Server throughput and Latency. And client Side metrics (Business to Customer B to C). Computing web service metrics affected by web service composition.

3.6 Web Service Composition Types

There are two types of Web service composition Static and Dynamic Compositions. Static service composition can be divided into two approaches. The first approach, referred to as Web services orchestration, in this approach there is a central coordinator (the orchestrator) which is responsible for invoking and combining the single sub-activities. The second approach referred to as Web services choreography. In this approach, there is no a central coordinator, but it defines

complex tasks via the definition of the conversation that should be undertaken by each participant.

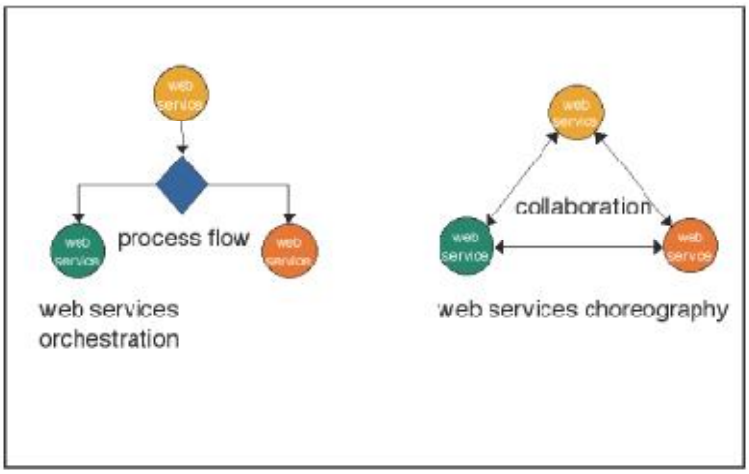


Figure 3.2 Orchestration and Choreography

Web Services are designed to provide interoperability between different applications. The platform and language independent interfaces of the web services allow the smooth integration of heterogeneous systems (Bianculli and Ghezzi 2007). Web languages such as UDDI, WSDL and SOAP define standards for service discovery, description, and messaging protocols. The dynamic composition of services requires the location of services based on their capabilities and the recognition of those services that can be matched together to create a composition. (Sirin et al., 2003).

3.7 Web Service Composition Patterns

Zeginis in (2009) explains that web services can be composed using different patterns.

3.7.1 Sequence Pattern

In the the sequence pattern web services are executed in sequence.

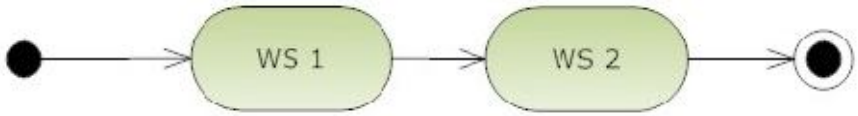


Figure 3.3 Sequence Pattern

3.7.2 Parallel pattern

The parallel pattern indicates that the web services can be executed in parallel.

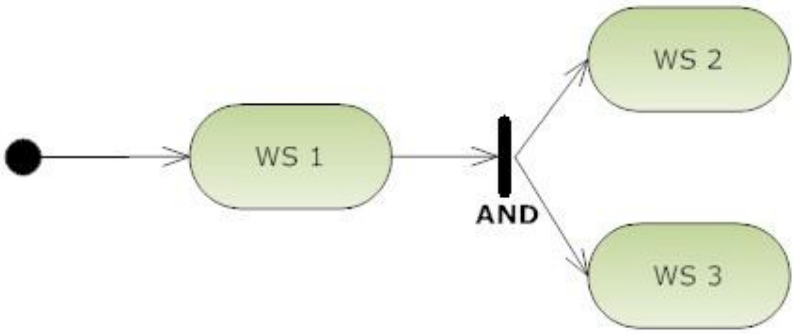


Figure 3.4 Parallel pattern

3.7.3 Loop pattern

In loop pattern there is a certain point in the composition block is executed repeatedly.

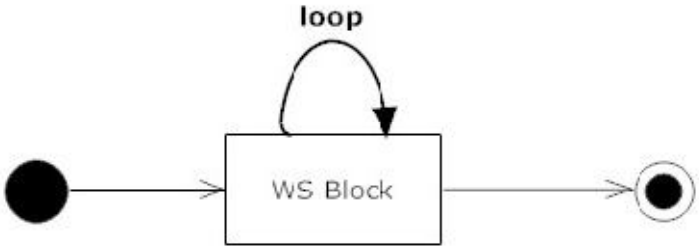


Figure 3.5 Loop Pattern

3.8 Performance Measuring

To measure the quality of service indicators for customer satisfaction are required. For example, if the consumer concern about the performance, so there is need to performance indicator. To get good indicators defining these qualities become essential. Frank Schulz in (2010) defines availability, response time and throughput. Daniela (2007) also identifies availability, response time and throughput in addition to other QoS. Their definitions are closely related, and it is clear that Daniela defines QoS in the context of service-oriented architecture.

1- Availability: The availability of a service within a given time interval can be defined as the ratio between the sum of durations during which the service could be invoked divided by the total duration of the time interval.

2- Response Time: The response time of a single service request can be defined as the time between service request and service response. Depending on whether network latency is taken into account, the time on consumer side between sending the service request and receiving the response, for provider side response time is the time between receiving the

service request and sending the response. Service level objective for performance are defined for individual service invocations (e.g., the completion time of each service call less than 2 seconds).

3- Throughput: Throughput or capacity of a service is the number of a service requests that can be processed by the system within a given interval. For example, a service may respond up to 100 service requests per one hour.

Warranties associated with these indicators would be a response time for example 15 seconds since there will be many service instances during the life of the agreement, the warranty would likely be expressed in statistical terms, say an average response time. Averages will be over a longish period, like a day or month. In order for the warranties to have any force, they must be accompanied by financial penalties and perhaps rewards. A provider and a consumer have to think of how to measure these indicators. If penalties apply, then there would need to be some form of dispute resolution procedure.

Performance can be measured with time-related attributes, with throughput, and with scalability. Given a service s, an operation o belonging to service s and a request r for the operation o, Daniela (2007) defines processing time, wrapping time, execution time, latency, response time and throughput as follow:

3.8.1 Processing Time

Processing time tp(s, o) is the time required for the execution of the operation. It does not include the network communication time, but simply the time required for the operation to be processed.

3.8.2 Wrapping Time

Wrapping time tw(s, o) is called XML processing time. In fact, it includes the time used for unwrapping the XML structure of the request r, wrapping the request r and sending it to the destination.

3.8.3 Execution Time

Execution time te(s, o) is the time required for execution operation o, i.e. the time needed for unwrapping the XML document (tw), plus the time needed

for processing the output (tp), plus the time for wrapping the result in like SOAP envelope (tw). In a formal definition the execution time is:

$$t_e(s,o) = t_p(s,o) + 2 \times t_w$$

3.8.4 Latency

Latency tl(s, o) is the time required by the SOAP message to reach the destination. This value depends on the network capacity and load during the conveyance of the message.

3.8.5 Response Time

Response time tr(s, o) is the time required for sending a message from a client to the service s provider until the response for the message arrives back to the client. This value is provider-specific since the provider gives a different priority to the clients of his services. The formula for the response time is:

$$t_r(s,o) = t_e(s,o) + 2 \times t_l(s,os)$$

3.8.6 Throughput

Throughput tp(s, o) measures the number of requests r for an operation o that can be processed by the service s in a given interval of time. Its value depends on provider capability. Its formula is:

$$t_p(s,o) = \frac{\#OfRequests}{timeInterval}$$

3.9 End-to-End Performance Measuring

Computing all metrics for composite service is complicated, as it is confusing and inaccurate to define formulas for all the metrics. Response time and throughput for composite service are computed (Zeginis 2009) because they are most commonly

used metrics in addition to performance and availability are user-oriented metrics the end user can easily understand them.

3.9.1 Computing Response Time

The response time is B to C metric. Thus it is the time a service provider needs to serve a consumer request. It is usually defined in milliseconds or seconds and is the most commonly used metric for web services.

For the sequential pattern, the response time is defined as the sum of the response times of the constituent web services:

$$\text{RT}(\textit{Sequential Supply chain}) = \sum_{i=1}^{n} \text{RT}(S_i)$$

For the parallel, the response time of the compositions is defined as the maximum response time of the constituent providers.

$$\text{RT}(\textit{Parallel Supply chain}) = \max\{\text{RT}(S_i)\}$$

3.9.1.1 Computing Response Time in Case Of Loop

Consider a composite service C consisting of execution of service A followed by service B. If execution of C involves several (K) executions of B for each execution of A.

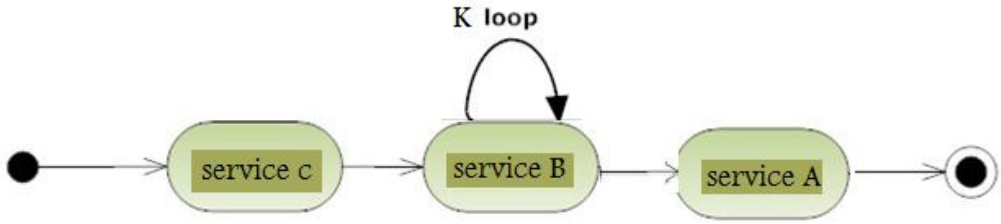


Figure 3-6 Loop in Composite Service

Then if the executions of B are in parallel, the average response time of C is still average response time of A + average response time of B. There are just more executions of B than of A. If the executions of B are in series, then:

> Average response time of C = average response time of A + K * average response time of B.

3.9.2 Computing Throughput

Throughput is business to a business metric, so throughput measured in transactions per some time period. Consider a composite service C consisting of execution of service A followed by service B. If service A has a throughput TA, and service B has throughput TB, then the composite service C cannot have a throughput higher than the smaller of TA and TB. If TA is lower, in that case, service B only gets TA transactions per second, no matter how many transactions B is capable of, it can only process TA. This is true of any linear chain of services. The End-to-End throughput can't be any more than the smallest of the link throughputs.

If service C consists of a single execution of service A, which involves N executions of service B (TB1,…, TBn), then the composite throughput depends on whether the executions of B occur in parallel (like a checking the weather at several locations), or in series (like building a multi-city flight plan). If in parallel the composite is the same as for the single execution of B case, since TB1, TB2, … TBn is all the same and occurs all at once. In series, the effective throughput of B is the TB1/N, since N executions of B are needed. TC is, therefore, the smaller of TA and TB1/N.

More complex QOS measurements will involve combinations of probabilities and will depend on whether the QoS of the various service components is statistically independent of each other.

3.9.3 Computing the Probability of Composite Service

In case studies which had been described in chapter 6, controlling the value of certain variable such as response time was difficult, thus the results were vary from one performance of the experiment to the next, even though most of the conditions were the same. This situation highlights the necessity to use probability theory to solve uncertainty problem. To compute probability there is need to determine sample space S which consists of all possible outcomes of a random experiment. For each web service, different values of response time (i.e. can be accessed through log file) represent sample space. Also, events need to be specified. An event is a subset A of the sample space S, it is a set of possible outcomes, and for example, different response time values are events.

A probability can be used to determine the event occurrence. It is convenient to assign a number between 0 and 1. If an event will occur surely, then the probability is 100% or 1. If the event will not occur, the probability is zero otherwise the probability is value between 0 and 1 (Spiegel et al. 2009).

3.9.3.1 Approaches to Estimate the Probability of Events

Spiegel and others summarize that there are two approaches to estimate the probability of an event: classical approach and frequency approach. In Classical approach, if an event can occur in h different ways out of a total of n possible ways, all of which are equally likely, then the probability of the event is h/n. But if after n repetitions of an experiment, where n is huge, an event is observed to occur in h of these, then the probability of the event is h/n. This is called frequency approach. Frequency approach is suitable for our case study because of the nature of randomized data which located over a long period (month or year) in the audit record. Audit record exists in provider side, and it represents information history see chapter 5 for more information about audit record (Spiegel et al. 2009).

3.9.3.2 Set of Events Probability

For any web service, there are multiple activities (a1, a2, …, an) among which only one activity can be executed. Each of these activities ai have a pi probability to be executed. From audit record, response time can be computed for each request as well as frequency for each event h among the total requests n in a certain period. From this information, the probability for each event can be calculated using the frequency approach. The probability of the event is h/n. Then the event with maximum probability is selected. And this web service can guarantee response time not greater than a selected event.

3.9.3.3 Independent Events

If the probability of B occurring is not affected by the occurrence or nonoccurrence of A, then A and B are independent events. Two events A and B are said to be statistically independent.

$$P(A \cap B) = P(A)P(B)$$

Also three events A1, A2, A3 are independent if they are pairwise independent.

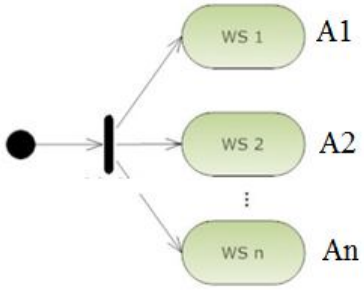


Figure 3-7 Independent Events

$$P(A_j \cap A_k) = P(A_j)P(A_k)\ j \neq k \qquad \text{where} \qquad j,k = 1,2,3$$

And

$$P(A_1 \cap A_2 \cap A_3) = P(A_1)P(A_2)P(A_3)$$

Both of these properties must hold in order for the events to be independent. Independence of more than three events is easily defined.

$$P(A_j \cap A_k) = P(A_j)P(A_k)\ j \neq k \qquad \text{where} \qquad j,k = 1,2,3 \ldots n$$

And

$$P(A1 \cap A2 \cap \ldots An) = P(A1)P(A2) \ldots P(An)$$

The QoS for various service components is statistically independent. If the response time of any web service is not affected by other web services above formula can be used to compute the probability of these independent events.

3.10 End-to-End Quality of Service Monitoring

Providing End-to-End quality of service is a big challenge because it requires a method of coordinating. In literature this direction is in mature with a few efforts. Ta and others (2006) propose algorithm to monitor End-to-End Quality of Service using a novel adaptive stratified sampling with optimum allocation to make the QoS monitoring less intrusive and more efficient but they did not pay any attention to other End-to-End QoS monitoring problems such as End-to-End performance guarantee, and metric .conflict,.

Another related work with a different goal is introduced by Bertolino and others in (2008). They concentrated on the problem of how one can invoke the real warehouse services during development time for testing purposes for example really buying goods. And they ignore the supply chain of SLAs issues.

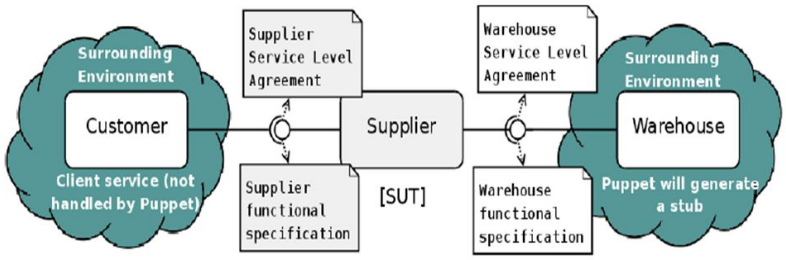


Figure3-8 Customer-Supplier-Warehouse

Al-sagaf and Dayang (2012) developed standard design and standard instrumentation process for monitoring SLA. Their monitor can be used in monitoring a supply chain of SLAs because a supply chain monitoring requires standard instrumentation. But also they did not pay attention to supply chain management problems.

Al Falasi et al. (2013) argue that cloud computing as technology has evolved from a convergence of Grid computing, Service Oriented Architecture (SOA), and Web services paradigms. It has adopted the SOA concept with clearly defined web services interfaces. They promoted the idea of infinite resource provisioning; grounded by remote resource provisioning introduced by Grid computing. Challenges in Cloud SLA management come from the need to provide different SLAs, for different consumers to integrate with their own business processes. SLAs in cloud computing requires a precise and specific definition of SLA parameters and metrics, dynamic SLA negotiation, on-demand service monitoring, in addition to precise enforcement measures. Al Falasi et al. proposed an SLA management model for federated cloud environments. They introduced the Sky Framework with two modules. The framework aims to facilitate the provisioning of composite Cloud services by promoting collaboration among Cloud vendors, and through the adoption of the social networking infrastructure. Within each of the modules, there is a specialized focus, socialization, and federation, that is administered through a Sky Broker that overlooks both modules. For every established SLA, an instance of the Monitoring Agent is created, and two monitoring services are initiated detection service and evaluation service. With all of the complexities that are involved with multiple SLAs, monitoring agents for each established SLA and a monitoring coordinator that oversees all monitoring of the combined SLAs. These monitoring agents will be responsible for two different services: detection service and evaluation service. The combinations of all different monitoring roles help trigger

the proper measures to correct situations or to enforce the SLAs. The overall model proposes an SLA specification and monitoring schemes that administrate the social relationships between Cloud services within a federation framework of Cloud services through the Sky model.

Monitoring SLAs in a federated Cloud depends highly on the collaboration among Cloud providers and Cloud consumer. Monitoring linked SLAs requires a consistent specification of SLA parameters, dynamic SLA negotiation, and multi-level SLAs monitoring, in addition to a reliable enforcement measure.

Cloud providers and services are often selected more dynamically than in traditional IT services, and as a result, SLAs need to be set up, and their monitoring implemented to match the same speed. Monitoring SLAs in this context is complicated because different Cloud providers expose different management interfaces and SLA metrics differ from one provider to another. Mohamed et al. (2017) develop rSLA framework that enables fast setup of SLA monitoring in dynamic and heterogeneous Cloud environments. The rSLA framework is made up of three main components: the rSLA language to formally represent SLAs, the rSLA Service, which interprets the SLAs and implements the behavior specified in them, and a set of Xlets-lightweight, dynamically bound adapters to monitoring and controlling interfaces. The rSLA framework can monitor data, evaluating SLOs, and execute enforcement and reporting actions. Beside SLA frameworks, an essential part of a solution to automate SLA management is a formal, machine-interpretable representation of the SLA. Several specifications have been proposed in the Web service and Grid context such as the Web Service Level Agreement language (WSLA),1 the Web Services Offer Language (WSOL) and Web Service Agreement (WS-Agreement) standard of the Open Grid Forum.

They build a framework which enables the dynamic setup of service quality management but to use their framework you must use rSLA language for formally specifying SLAs. To resolve metric conflict problem they build adapter which developed in implementation phase.

Rizvi et al. (2017) propose a model in which a third-party can assist consumers to ensure that they are receiving the promised services from their chosen providers. They call it a three-step approach because customers need three steps to evaluate the service-level agreement (SLA). The three primary steps of their model are an initial review of any valuable information, an assessment of specific cloud metrics.

They use both cloud auditors and cloud brokers to evaluate the services provided which help consumers in making the best decisions. Providers can benefit from these kinds of trust models. Also, they proposed the third party to monitor SLAs, but they did not declare how this third party will solve a supply chain issues such as metric conflict.

3.11 Summary

To the best of our knowledge and based on the summarized review of existing work described in section 3.10, there is no holistic End-to-End SLAs monitoring. Due to the complexity inherited with such environments, End-to-End SLAs monitoring should manage a complete SLA life cycle, be able to hide the complexity of SLA management from both consumers, and providers, reflect the composite nature of End-to-End environment and be able to identify source of service violation in a chain of SLAs. Our proposed SLA management model aims to address these issues. Next chapter presents model-based engineering and Ontology.

CHAPTER 4

MODEL-BASED ENGINEERING& ONTOLOGY

4.1 Introduction

Recently software developers use more efficient development processes that are requirements-driven and architecture-centric instead of traditional development processes that are document-based and code-centric (Perisic 2014b).

Model-based Engineering (MBE) is an engineering approach that uses models as an integral part of the technical baseline that includes the requirements, analysis, design, implementation, and verification of system and product throughout the acquisition life cycle. This chapter includes two crucial topics model-based engineering and Ontology.

4.2 Model-Based Engineering

Using model has a far greater impact in developing software than code-based techniques. Model-Based engineering approach enables the development of software and systems in all stages. MBE reduces the time for the acquisition. Also, MBE minimizes the time to implement planned and foreseen changes in systems. In addition to MBE enhance reliability and interoperability by building platform-independent high-level models specified by UML and interfaces throughout the life cycle. Not all but also MBE reduce development time and cost to design, develop, deliver, support capabilities, and enhance reliability. MBE uses model to describe the functionality of the systems and a metamodel to conceptulize the model.

4.3 Model-Based Engineering Model and Metamodel

There are many terms used in model-based engineering such as model and metamodel so what is the relationship between them. A model is a formal specification of the function, structure and behavior of a system within a given context, and from a specific point of view (or reference point). The primary factor

that determines the effectiveness of modeling is model clearness for the user. The models are developed through extensive communication among product managers, designers, developers and users of the application domain. As the models approach completion, they enable the development of software and systems. While metamodel is a model of a model, and metamodeling is the process of generating such metamodels. Metamodeling or meta-modeling is the analysis, construction, and development of the frames, rules, constraints, models, and theories applicable and useful for modeling a predefined class of problems. As its name implies, this concept applies the notions of metamodeling and modeling in software engineering and systems engineering (Wikipedia 2015).

The metamodel is a conceptual model for the syntax of a modeling system. Metamodel specifies the schema for the repository. Repository stores model instances. Constraints expressed as queries on the repository. Repository supports the model creation, editing, rendering, browsing, etc. Metamodel usually has classes model and instances model.

4.3.1 Unified Modeling Language and UML Profiles

Unified Modeling Language (UML) is designed to provide a standard way to visualize the design of a system. It was adopted as a standard by the Object Management Group (OMG) and has been managed by this organization. In 2005 the Unified Modeling Language was also published by the International Organization for Standardization (ISO) as an approved ISO standard. UML is general-purpose modeling language, for customizing UML to particular domain and platform UML profile is used.

UML profile is an extension mechanism to the UML standard. A profile in the Unified Modeling Language (UML) provides a generic extension mechanism for customizing UML models for particular domains and platforms. The profile is a profile package that extends a reference metamodel (such as UML) by allowing to adapt or customize the metamodel with constructs that are specific to a particular domain, platform, or a software development method. Profile uses the same notation as a package, with the addition that the keyword «profile» is shown before or above the name of the package.

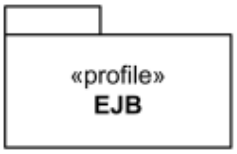


Figure 4-1 Profile in UML

A profile can define classes, stereotypes, data types, primitive types, enumerations. The profiling mechanism is used to establish a system of sub metaclasses of the metaclass Class, so different kinds of classes can be represented. It enables to represent the classes diagram as the more informative model see stereotype server using the profile as shown in figure 4-2 UML class becomes more informative, now it can represents the role.

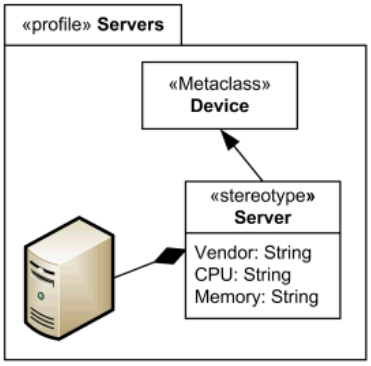


Figure 4-2 Profile Servers

One profile might reuse some or all parts of another profile, to extend already existing profiles. Multiple profiles could be applied to the same model.

The constraints that are part of the profile are evaluated when the profile has been applied to a package. These restrictions need to be satisfied in order for the model to be well-formed.

4.3.2 A stereotype

The stereotype is the primary extension construct, which is defined as part of the profile and extends some Metaclass. A stereotype is extensibility mechanism in UML. They allow designers to extend the vocabulary of UML in order to create new model elements, derived from existing ones, but that have specific properties that are suitable for a particular problem domain or otherwise specialized usage.a

stereotype is rendered as a name enclosed by guillemets (« ») and placed above the name of another element. For instance, in a class diagram stereotypes can be used to classify method behavior such as «constructor» and «getter». Despite its appearance, «interface» is not a stereotype but a classifier. All objects that can have instances are classifiers.The stereotype is a profile class which defines how an existing metaclass may be extended as part of a profile. It enables the use of a platform or domain-specific terminology or notation in place of, or in addition to, the ones used for the extended metaclass.

A stereotype cannot be used by itself, but must always be used with one of the metaclasses it extends. A stereotype cannot be extended to another stereotype. A stereotype uses the same notation as a class, with the keyword «stereotype» shown before or above the name of the stereotype. Stereotype names should not clash with keyword names for the extended model element (OMG 2007).

4.3.3 Using UML Profiles to Extends Metaclass

A metaclass is a profile class and a packageable element which may be extended through one or more stereotypes. A metaclass may be shown with the optional stereotype «Metaclass» shown above or before its name (all lower-case «metaclass» was used in UML versions prior to 2.4).

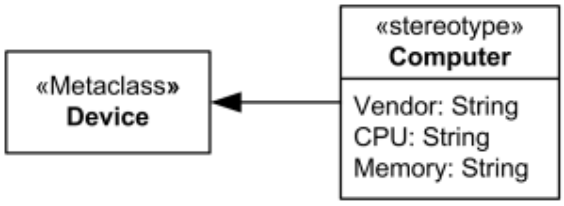


Figure 4-3 Stereotype Computer Extends Metaclass Device

There are several reasons to customize a metamodel such as to give a terminology that is adapted to a particular platform or domain, Give a syntax for constructs that do not have a notation, Give a different notation for already existing symbols, Add semantics that is left unspecified in the metamodel Add semantics that does not exist in the metamodel, and Add constraints that restrict the way you may use the metamodel (OMG 2011).

4.4 End to End Performance Guarantee Profile

To provide End-to-End performance guarantee through the supply chain, some computations are needed. The supplier should only use metrics that can be derived or predicted or computed from metrics that used between the supplier and the provider. To explain this step end to end performance guarantee profile is developed.

End-to-End performance guarantee profile defines five stereotypes, Kind, Suskind, Supply Chain, End-to-End, and Predicated. These stereotypes extend UML class and association to represents different metrics in a supply chain of SLAs to provide end to end performance guarantee. The metamodel elements are indicated by classes stereotype <<metaclass>>. The notation of extension is an arrow pointing from stereotype to the extended class, where the arrowhead is shown as a solid triangle see figure 4-4.

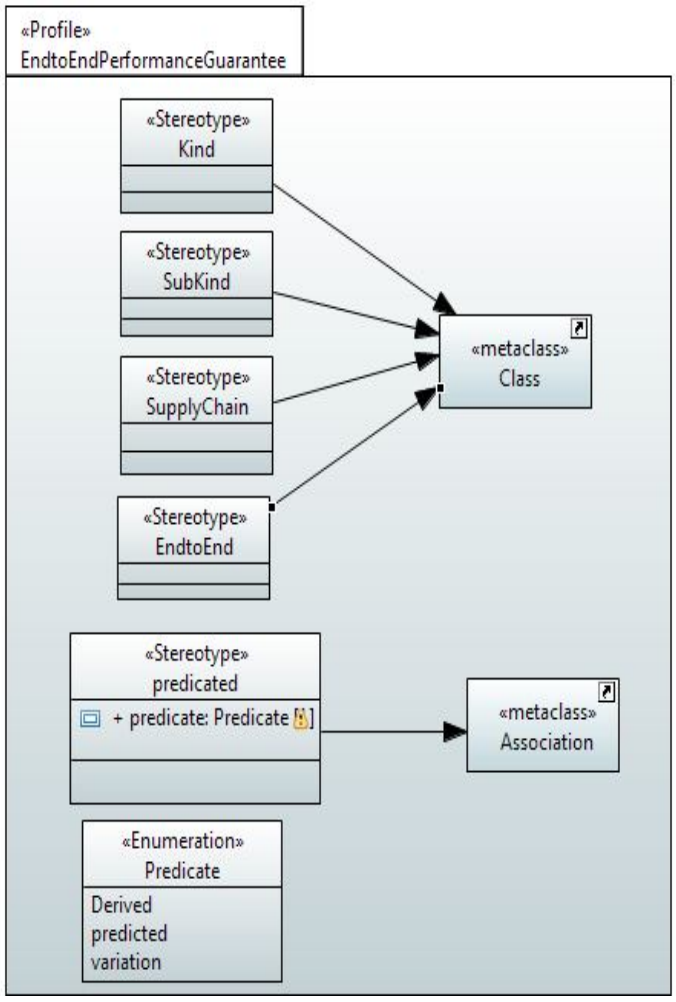


Figure 4-4 End-to-End Performance Guarantee profile

4.4.1 Using End-to-End Performance Guarantee profile

Dependency relationship stereotyped is used as <<apply>> to show the using of profile in specific domain or application, for example, figure 4-5 shows WSLA uses EndtoEndPerformanceGuarantee profile. WSLA can, therefore, describe diagram which illustrates two classes linked by association <<stereotype>>. Notice the value of tagged value and the value assigned to it see figure 4.6 and figure 4-7.

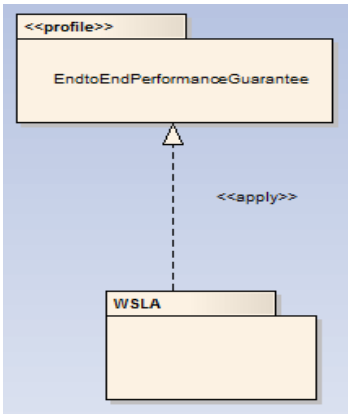


Figure 4-5 WSLA Uses End-to-End Performance Profile

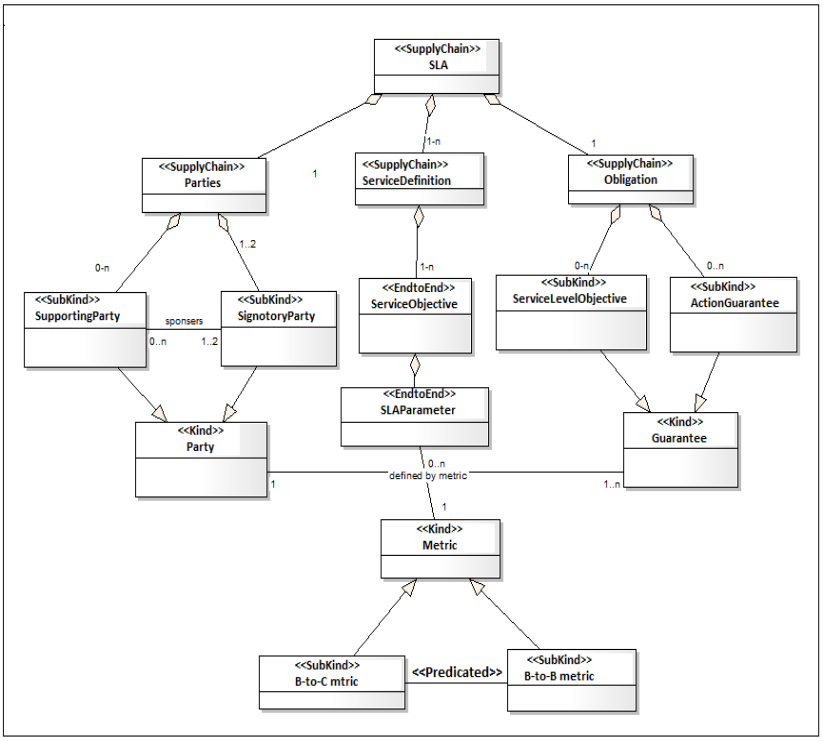


Figure 4-6 WSLA Metamodel Using Proposed Profile

4.4.2 Tagged Values and Stereotypes

Metrics in the supply chain can be categorized into two type B-to-B metric and B-to-C metrics and to provide End-to-End performance guarantee B-to-C metrics should be derived, predicted or computed from B-to-B metrics. Figure 4.7 below shows that B-to-C metric response time can be obtained from B-to-C metric transaction rate.

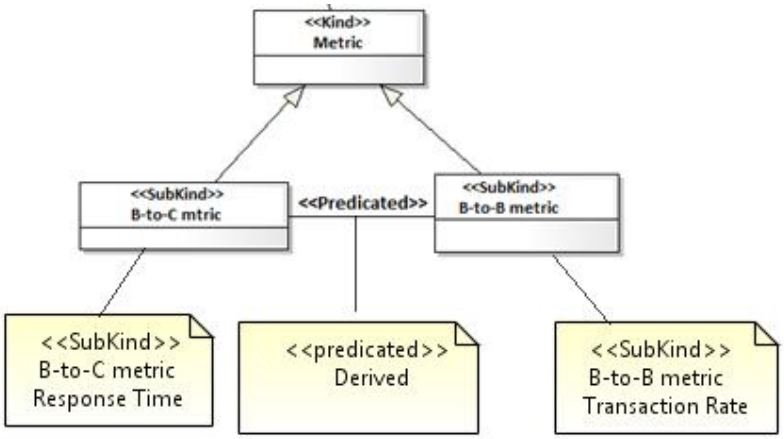


Figure 4-7 Tagged Values Related to Metrics and its Association

4.4.3 WSLAUP Features

WSLA metamodel has been extended using End-to-End performance guarantee profile. This extension makes WSLA model more informative and expressive in representing SLAs metrics for WSLAUP provides semantic representation for SLA metrics. Furthermore using WSLAUP, you can distinguish between different types of metrics involved in SLAs.

4.5 Summary of UML Profile

A profile in the Unified Modeling Language (UML) provides a generic extension mechanism for customizing UML models for particular domains and platforms. Make the model more informative. A stereotype is a specialization of a UML metaclass. A stereotype says I can take a basic modeling element and give it more meaning. Stereotypes may be used to classify and extend associations, inheritance relationships, classes, and components. The profiling mechanism leads to different kinds of classes to define a system of sub metaclasses of the metaclass Class.

4.6 Introduction to Ontology

Providing and interoperation among internet services require complex communication. The information systems often create aspects of the reality they share. Anyone who does anything in this world will typically interact with several businesses or organizations supported by information systems, this situation highlights the necessity to resolve interoperability. Recently services provided by

many systems these systems should interact together. Having all these information systems publically and accessible on the web open many additional possibilities. If doing something involves interaction with many businesses and organizations, each supported by information systems, it is reasonable to expect many issues such as interoperability among autonomous systems, namely semantic heterogeneity.

4.6.1 Semantic Heterogeneity

Structural semantic heterogeneity can be resolved using views, so preserves the autonomy of the local systems. Fundamental semantic heterogeneity requires that at least one of two incompatible systems must be changed, thus violates autonomy.
This Chapter purposes to describe a collection of things which exist in the environment of the interoperating systems. Such a collection is called Ontology. Associated with the description of each thing is an agreement about the semantics of that thing, how it is to be interpreted when it is used in a message (Colomb, 2007).

4.6.2 Benefits of Using Ontology

"Whatever problem you are trying to solve, you need an ontology"

(Colomb, 2007)

Ontology has many benefits. Firstly, it provides a rich vocabulary for describing the systems , so they can assist us in designing and understanding systems. Secondly, it facilitates interoperability. Thirdly Ontology eases reasoning. Also using ontology would add, for example, the ability to constrain possible combinations of properties in order to create a more accurate model of the world (Glen Dobson et al., 2005).

4.6.3 Ontology Representation Language

To represent Ontology a vocabulary is needed. Such a vocabulary is called an ontology representation language. Ontology is a sort of data model, so there is a number of competing representation languages derived from different modeling traditions. The underlying semantics of all these languages is derived from set theory and the predicate calculus. The central concepts used are:

- Class: A set of individuals. UML Class.

- Individual: A single, but possibly complex thing. An individual belonging to a class is called an instance of that class.
- Property: A binary relation among classes. UML association or attribute. The two ends of the property are called domain and range, with the property being read from the domain to the range.
- Subclass: A class whose individual members are also members of another class (the superclass).
- Participation: A class participates in a property if the class is an end of the property.
- A derived property can be defined by composing two properties. The range of the first is the domain of the second. See figure below.

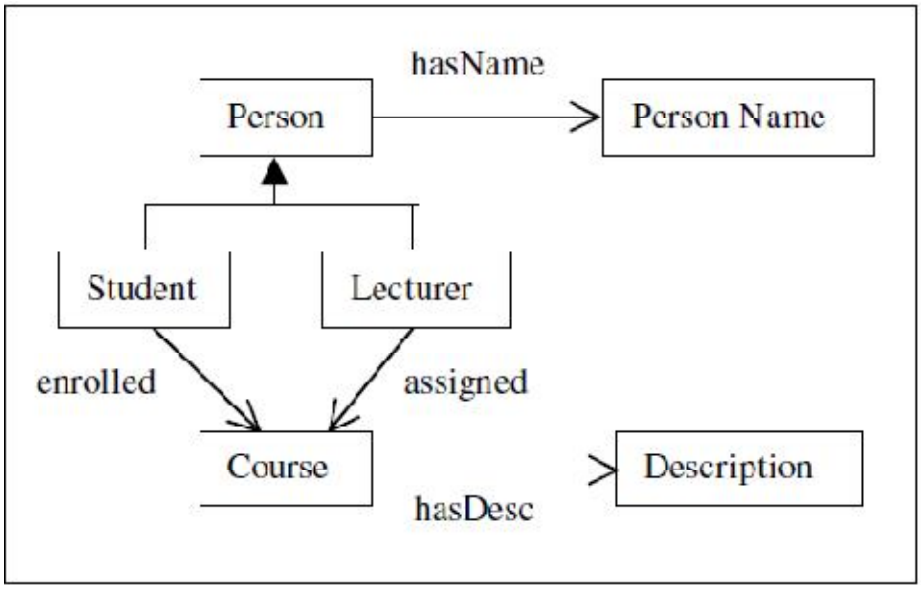


Figure 4-8 Ontology Example (Robert 2007)

The classes are Person, Student, Lecturer, Course, Person Name, and Description. The properties are enrolled, assigned, hasName, hasDesc. The classes Student and lecturer are subclasses of Person. Individuals are instances of any of the classes. The properties are shown as having direction. The class away from the arrow is called the domain; the class at the arrow is called the range. UML has directional or navigable binary associations where the corresponding ends are called source and target respectively.

4.6.3.1 Web Ontology Language

"an extension of the current web in which information is given well-defined meaning, better-enabling computers and people to work in cooperation" (Lee et al., 2001)

World-Wide Web Consortium's Resource Description Framework (RDF) language can be used as ontology representation language, but it has some deficiencies. The W3C has addressed some of these deficiencies with the Web Ontology Language OWL which designed as a specialization of RDFS.

OWL is the Web Ontology Language, designed for publishing and sharing ontologies via the web. OWL is based on the Resource Description Framework (RDF), which can also be regarded as a simple ontology language (Glen Dobson et al, 2005).

Although RDFS is appropriate representation for an ontology than UML because it clearly identified individuals and has a suitable small granularity. But, RDFS has no convenient mechanism to represent an ontology as an engineered object, with boundaries, versions and the like. Additional structure, including engineered objects, can be expressed in an extension of RDFS called Web Ontology Language, or OWL

4.6.3.2 OWL Metamodel

One way to understand the structure of OWL and its relationship to RDFS is by a metamodel as in Figure 4-9. This metamodel is a sort of ontology represented in a subset of UML called the Meta-Object Facility, or MOF. The MOF is used to define the structure of modeling systems, in particular, UML.

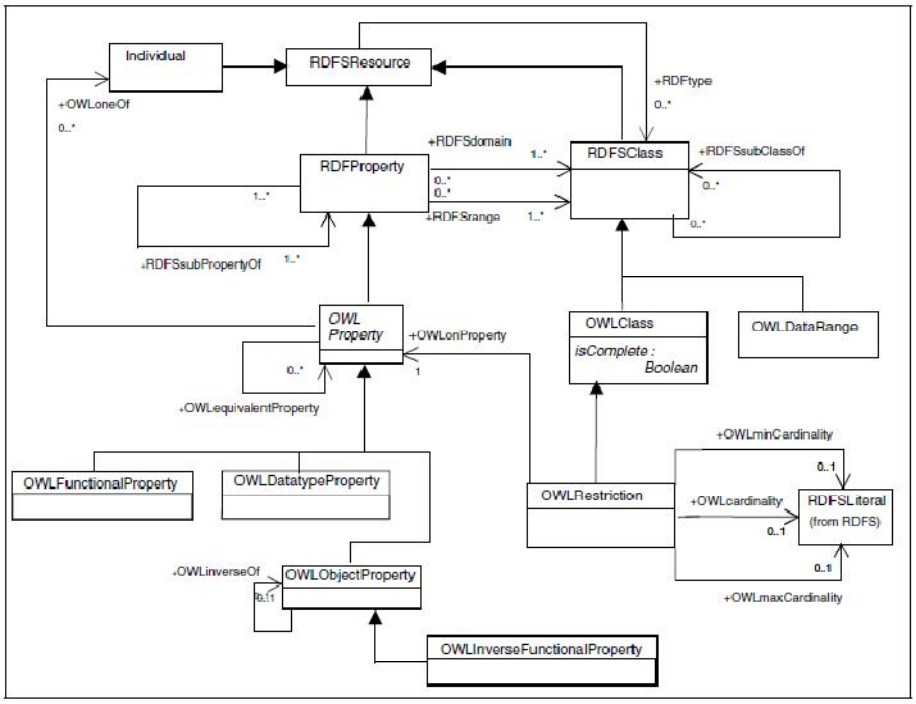


Figure 4-9 MOF Metamodel for RDFS and OWL

The key constructs of OWL are Individual, OWLClass, and OWLProperty, respectively specializations of RDFSResource, RDFSClass, and RDFProperty. So an OWL Individual is an RDFSResource, anOWLClass is an RDF class, and an OWLProperty is an RDFProperty.

4.7 Summary

Chapter 4 explains two types of semantic heterogeneity, structural and fundamental semantic heterogeneity. Ontology is strongly required to resolve a fundamental semantic heterogeneity chapter 7 provides more details about metric ontology which resolves a fundamental semantic heterogeneity problem which may occur in monitoring a supply chain of SLAs. The question is how to do audit trail in this supply chain? And what is a suitable strategy to monitor a supply chain of SLAs? Next chapter tries to answer these two questions.

CHAPTER 5

AUDIT TRAIL AND END TO END MONITORING STRATEGIES

5.1 Introduction

This chapter includes two parts. Part one gives a brief introduction to the concept of the audit trail, audit trail components and objectives of audit trail and types of auditing. The second part presents End-to-End performance guarantee strategies which include audit trail at each stage, third-party do audit trail, End-to-End performance predicate and cost versus performance trade-off strategy.

5.2 Audit Trail

An audit trail is the most part of the monitor because it determines the level to which QoS is fulfilled. Auditing is the systematic, independent and documented process for obtaining audit evidence to evaluate the QoS. The audit criteria are usually determined by the audit scheme or certification scheme which is used to perform the audit. Certification is one of many ways to address audits. Logging is the recording of data related to the operation and use of a service. Log file entries are mean by which audit trail can be realized, significant to service customers when analyzing event such as performance violation and service failures as well as in monitoring the customer's day-to-day use of the service. It is necessary for there to be service level objective relating to logging and monitoring in order to adequately describe the service and its related capabilities.

Audit trail main components are information collector which gathers require information and store it in audit record. The second component is auditor or violation detector that responsible to provide audit evidence to support audit report. The purpose of an audit is to enhance the confidence between parties. Audit trail help service provider and consumer to detect SLAs violation before it takes place so

they can avoid failure. Thus the trust between SLA provider and consumer will increase.

5.3 Types of Auditing

There are two types of auditing. Internal auditing which done by the organization itself, while the external auditing is done by another organization or done by an expert in auditing. Internal auditing can be done by consumer or provider, while external auditing must be done by the third party. In the proposed solution a consumer does internal auditing and the results recorded in audit records, if the violation is going to occur report must be sent to the third party to perform external auditing.

5.4 End-to-End Performance Guarantee Strategies

To increase the QoS provided through a supply chain one of End-to-End performance guarantee strategy must be chosen, but measuring performance in this situation facing many challenges. It can be easy the ultimate consumer can measure the time between the initiation of a request and the receipt of the response, at least, in this case, it is possible to determine whether there is a problem, namely that the response time is more than is expected. But If the SLAs are between pairs of partners in the supply chain SLA1 between consumer and provider and SLA2 between provider and supplier, there is no agency responsible for End-to-End performance this lead to dispute because the results of audit trail which done by consumer may not be accepted by provider, also the results of audit trail which done by provider may not be trusted by supplier. In addition to, there is no any type of End-to-End performance guarantee. To provide End-to-End performance guarantee, four primary strategies can be used to monitor supply chains of SLAs, these strategies are audit trail at each stage; the third party do audit trail, End-to-End performance predicate, and cost versus performance trade-off.

5.4.1 Audit Trail at Each Stage (Principle A)

To monitor a supply chain of SLAs, audit trail at each stage can be placed. In this scenario all consumers have to do internal audit trail to ensure that they get the service as agreed upon see figure 5-1. Consumer periodically records the time of

requesting and receiving the service and the audit results stored in audit records. Then consumer compares audit results with SLAs criteria. If a violation occurs, the consumer will send a notification to the service provider. But audit trail at each stage strategy has weaknesses:

- Audit trails results may not be available for all partners, and this situation leads to disputes.
- Audit trail at each stage strategy does not guarantee End-to-End performance for there is no trusted party responsible to manages and controls these SLAs.

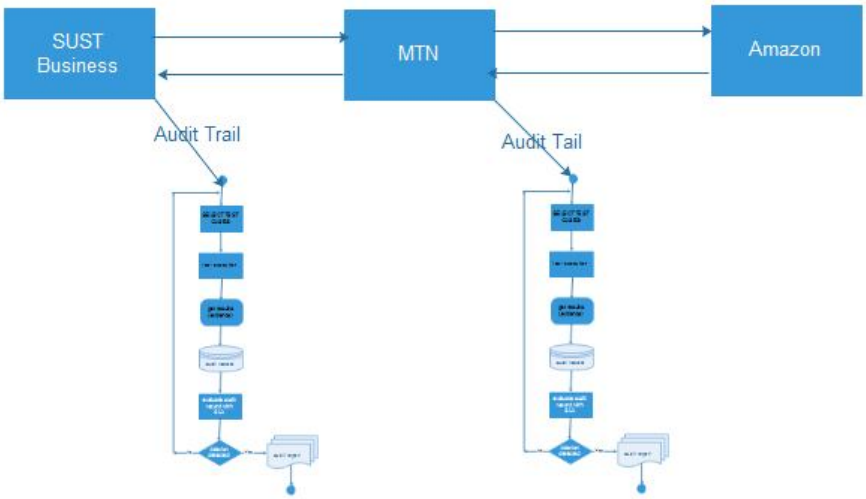


Figure 5-1 Audit Trail at Each Stage Strategy

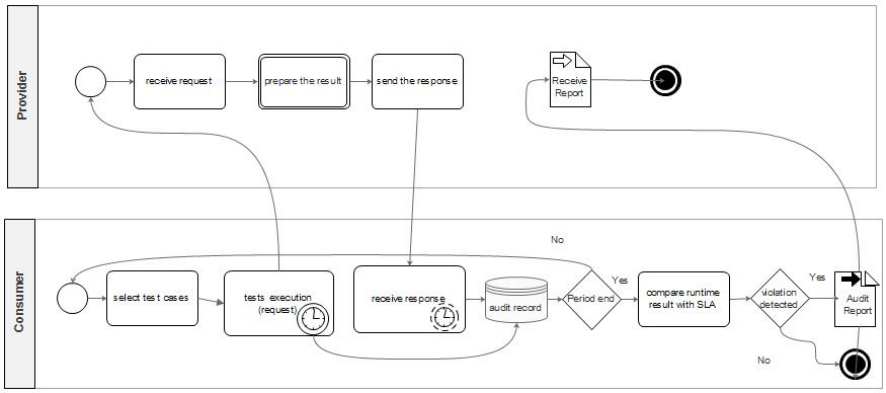


Figure 5-2 Consumer Do Audit Trail

5.4.2 Third Party Do Audit Trail Strategy (Principle B)

One possibility for there to be some sort of exchange which guarantees End-to-End performance, each link in the supply network would have an SLA with the trusted third party assigned by all partners, and the trusted third party would monitor performance at each step. The service step would not necessarily be performed through the trusted third party, but each service step would report to the trusted third party with sufficient information to measure performance. So this strategy can guarantee End-to-End performance because the trusted party is responsible for managing and controlling supply chain of SLAs. Also, the trusted third party will exhaust disputes.

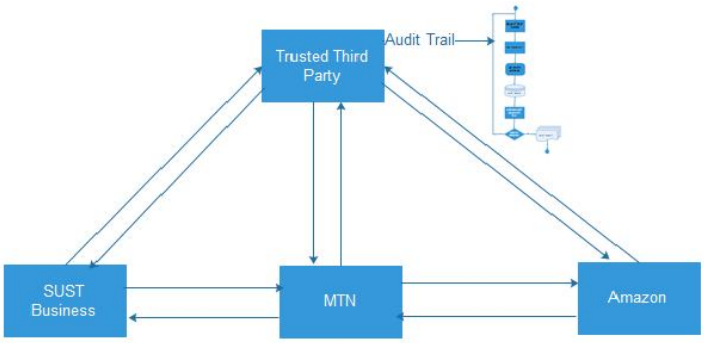


Figure 5-3 Third Party Do Audit Trail

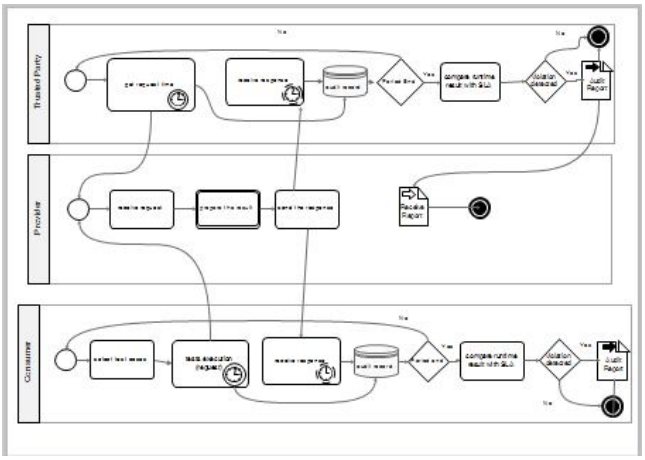


Figure 5-4 Third Party and Consumer Do Audit Trail

But there would need to mechanisms in place to prevent or detect deliberate or accidental misreporting and to resolve disputes. In this case, the trusted third party could monitor End-to-End performance.

A customer of the exchange could perform its own monitoring, so there would need to be a mechanism for reconciling the customer's measurements with the trusted third party measurement. The first solution is when the consumer feels that the performance is going to violated send notification to the trusted third party triggering him to do audit trail during this period. The second solution is trusted third party, and the consumer do audit trail at the same period, and this solution may increase overload, so the first solution is better in this regard, but it is less accurate.

5.4.3 End-to-End Performance Predicate Strategy

(Principle C)

A related possibility is for there to be an End-to-End performance predicate. Each step in the supply chain would know how much of the for example the delay they contribute, subtract that from the End-to-End predicate, and impose the resulting more restrictive predicate on its own suppliers. This process could be repeated. There is no trusted party monitor the performance as a result disputes will occur in addition to this strategy does not guarantee the quality of End-to-End performance.

5.4.4 Cost versus Performance Trade-Off Strategy

(Principle D)

Another strategy is for each stage to have a cost versus performance tradeoff, and to either negotiate or bid for a segment of the End-to-End performance. Thus the ultimate customer would then have a range of options at different prices. But no End-to-End performance guarantee and disputes may occur. Table 5-1 explains advantages and limitation of each strategy.

Strategy	Advantages	Limitations
(Principle A)	- Straightforward	- Disputes problem
(Principle B)	- Guarantee End-to-End performance. - Exhaust disputes.	- Accidental misreporting. - Information exchange. - Metric conflict - End-to-End performance measuring.
(Principle C)	- Very simple - Provide End-to-End predicate	- Disputes problem
(Principle D)	- Provides a range of options at different prices for consumer.	- Disputes problem.

Table 5-1 End-to-End Monitoring Strategies Comparison

5.5 Summary

There are four End-to-End performance guarantee strategies; audit trail at each stage, third party do audit trail strategy, End-to-End performance predicate strategy, and cost versus performance trade-off strategy.

To get the benefits of End-to-End performance guarantee the appropriate strategy should be selected in addition to suitable metrics. What happens if different metrics are used in this supply chain of SLAs? Next chapter explains this problem in details.

CHAPTER 6

THE FIRST CASE STUDY

MTN CRM CASE STUDY

6.1 Introduction

Descriptive research studies especially case study research deal with collecting data and testing hypotheses or answering questions concerning the current status of the subject of investigation. It deals with the question “WHAT IS” of a situation. It concerns with determining the current practices, status or features of situations. This chapter presents two case studies MTN CRM and A one-time password (OTP) authentication via SMS.

6.2 Motivations

MTN CRM and A one-time password authentication via SMS have been selected for many motivations. MTN CRM and A one-time password are very simple that the reader can easily understand them and at the same time they reflect the thesis problems.

6.3 Case Study Strategy

Case study research strategies include four essential stages start with selecting and defining the appropriate case study, and then run the case using the proposed solution, after that collecting data. Finally extracting results and discussing it under the light of research problem. Each one of these four stages includes many activities see figure below.

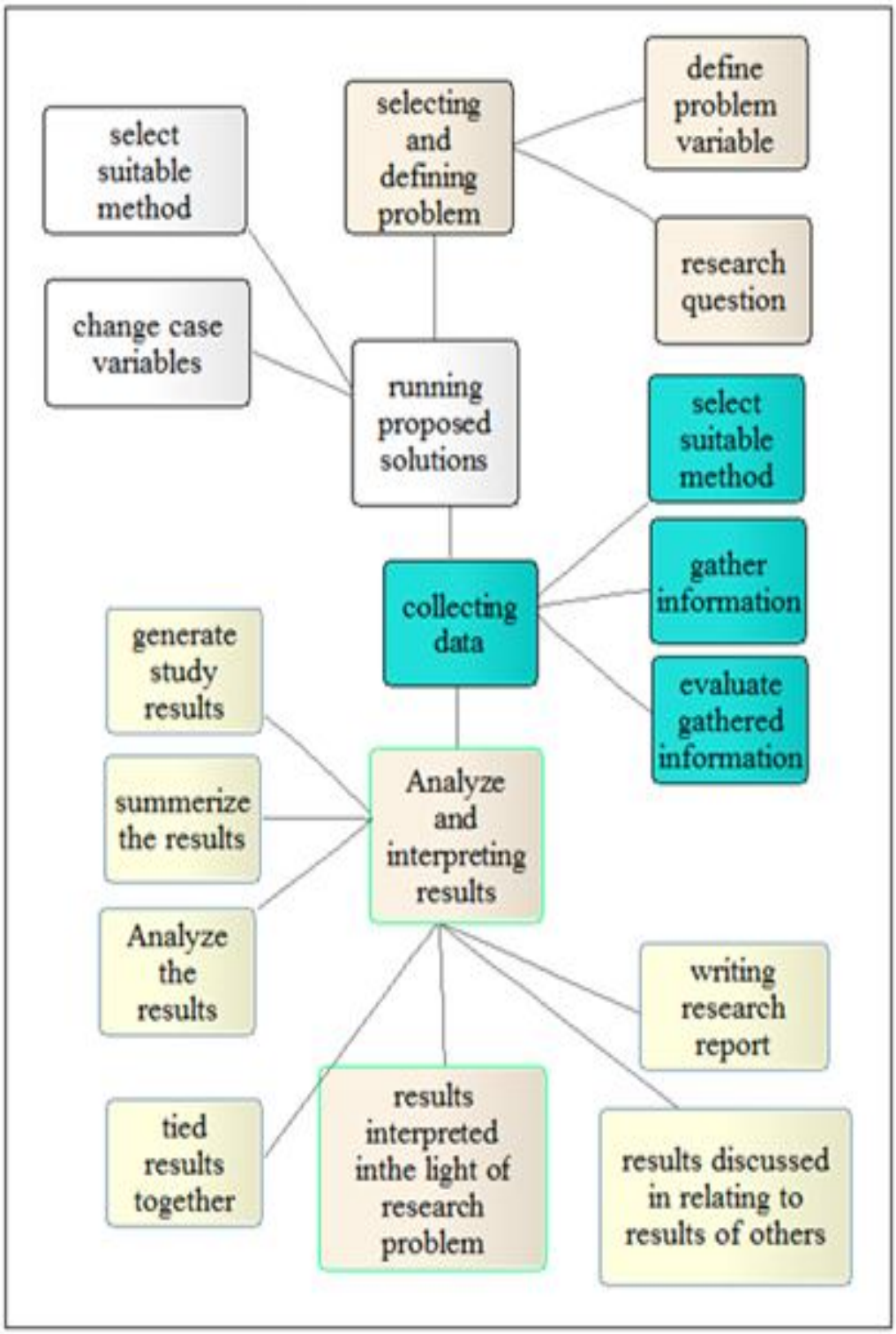


Figure 6-1 Case Study Strategy

6.4 Customer Relationship Management

Customer relationship management (CRM) is a model for managing a company's interactions with current and future customers. It involves using technology to organize, automate, and synchronize sales, marketing, customer service, and technical support. Customer Relationship Management is specific software that allows a company to measure and control contacts with customers. CRM can be used for controlling contacts with a customer either by phone, fax, mail, and e-mail. The data collected can be used for research and analysis of the customer relationship. The Customer Relationship Management is the procedure that is crucial for every business. As the customer is the most important part of the business, the CRM is the procedure that analyzes the contact with the customers in a call center for example.

6.5 CRM Types

There are many CRM types such as sales force automation, marketing and Appointments. Salesforce automation (SFA) uses software to control and manage sales process by recording every stage in the sales process. CRM systems for marketing track and measure campaigns over multiple channels, such as email, search, social media, telephone, and direct mail. CRM systems for marketing track clicks, responses. Appointment CRMs automatically provide suitable appointment times to customers via e-mail or the web. Sales analytics monitor client actions and preferences and augment sales forecasts and help measure the effectiveness of marketing campaigns. For small businesses, a CRM may merely consist of a contact management system which integrates emails, documents, jobs, faxes, and scheduling for individual accounts.

6.6 CRM Important Characteristics

CRM track and report every customer interaction, describing the customer's purchase, interest or demand. It also report the changing of customer needs and the way the business reacts efficiently to them. In addition to CRM can work as a universal instrument for collecting data about the service requests, order entry, satisfaction, and billing. Also, CRM measure the performance of the business on the

basis of internal benchmarks. Finally, CRM facilitate the working processes by emphasizing on the positive and exclude the harmful practices in customer's relations center.

6.7 Products and Services MTN Offers

For every owner of a growing enterprise, MTN business provides many services such as mobile fleet management to keep track of everything. Also, MTN provides SMS communications solution which can be tailored to specific business needs. In addition to MTN provides Sales Force management which can be used to automate sales processes, cuts costs, increases revenue, boosts performance and can dramatically improve customer relationships. MTN CRM allows you to do many tasks. Firstly using CRM, help organization to communicate with customers via bulk or personalized text messages. Secondly find customer records quickly and track customer preferences accurately. Thirdly by using MTN CRM, you can answer questions and resolve issues instantly, from anywhere. Fourthly MTN CRM allows you to manage your workforce from end to end effectively. Fifthly CRM useful to connect teams to data that will help them gain insights and add value to every relationship. Sixthly using CRM make it easy to access business applications from anywhere. Thus MTN CRM, in general, Optimizes business operations. MTN CRM Services provided in this case study are capabilities such as load, update, or view all information relating to the customer, products, and sales representatives, including:

- Customer information
- Customer previous order history
- Customer latest orders and quotes
- Product details
- Warehouse on-hand product lists
- Latest product pricing details
- Competitor information
- In addition to reports relating to issues like product sales, product orders, customer order lists and more can be generated.

The application on the mobile device allows for the sales personnel on the road to have online access to back-end systems and update information regarding any sales

call on their mobile phones. Further to the above, the mobile application also allows mobile sales personnel to:

- Place orders
- Record/Log red flag issues per customer or per product
- Record competitor information
- View latest available products
- View customer order history
- MTN CRM Supply Chain

MTN gets the service (customers and products information) from Amazon and provides the service to the customer (SUST business). This situation highlights the necessity to create two SLSs.SLA between MTN and Amazon and another SLA between MTN and the final consumer.

MTN as a service supplier get the service from Amazon and provide it to the consumer SUST business, so MTN requirements are:

1. Measuring the performance of service to evaluate whether it complies with the quality of service (QoS) that the customer expects.
2. To avoid SLA failure MTN wants to receive notification in case of SLA violation is going to take place.
3. If the violation occurred, MTN needs to know why the failure happened and who is responsible.
4. MTN hopes to avoid the overload which can be made by statistical information collection.
5. All these operations have to be done automatically this the vital point that MTN hopes to implement.

Amazon, in turn, provides the service to MTN, so an Amazon wants all MTN requirements.

SUST as a service consumer will measure the performance of service to evaluate whether it complies with the quality of service (QoS) as agreed upon. And it hopes that the process of measuring the performance to be done automatically.

Trusted third party with excellent experience in monitoring SLAs is assigned by three parties MTN and Amazon and SUST business to monitor these End-to-End SLAs. The third party is responsible for monitoring two SLAs to ensure that the results are trusted both by the provider and consumer.

6.8 SLA between Consumer and Supplier

As mentioned in service definition SUST business as a consumer can load, update, or view all information relating to the customer, products, and sales representatives. AS a service description SLA parameter used in this SLA is response time. Metric to measure SLA parameter is Average response time. SLA1 between SUST and MTN says that MTN responsible to provides average response time less than or equal 5 seconds during the contract period which starts on 30/4/2015 and ends on 30/4/2016.

```
<service name="getProductInfo" xsi:type="WSDLSOAPserviceDescriptionType">
    <SLAParameter name="ResponseTime" type="int" unit="seconds">
        <metric>averageresponsetime</metric>
        <source>MTN</source>
        </SLAParameter>

    <obligation>
        <serviceLevelObjective name="conditionalSLOForResponseTime">
        <obliged>MTN</obliged>
        <validity>
            <start>30-4-2015</start>
            <end>30-4-2016</end></validity>
        <expression>
            <implies>
                <expression>
                    <predicate xsi:type="less than or equal">
                    <SLAParameter>responsetime</SLAParameter>
                    <value>5</value>
                    </predicate>
                </expression>
            </implies>
        </expression>
        </serviceLevelObjective>
    </obligation>
</service>
```

SLA1 between SUST business and MTN

6.9 SLA between Supplier and Provider

To provide the service to the consumer SUST business as agreed upon, MTN should initiate a contract with Amazon. As shown in SLA2 bellow Amazon is responsible for providing products information to MTN with performance measured by the response time metric, more specific average response time metric is used and it must be less than 15 milliseconds.

```
<service name="getProductInfo" xsi:type="WSDLSOAPserviceDescriptionType">
    <SLAParameter name="ResponseTime" type="int" unit="millisecond">
        <metric>ResponseTimemetric</metric>
        <source>Amazon</source>
        <function xsi:type="average" resultType="double">
    </SLAParameter>

    <obligation>
        <serviceLevelObjective name="conditionalSLOForTransactionRate">
        <obliged>Amazon</obliged>
        <validity>
            <start>30-4-2015</start>
            <end>30-4-3016</end></validity>
        <expression>
            <implies>
                <expression>
                    <predicate xsi:type="less">
                    <SLAParameter>responseTime</SLAParameter>
                    <value>15</value>
                    </predicate>
                </expression>
            </implies>
        </expression>
        </serviceLevelObjective>
    </obligation>
</service>
```

SLA2 between MTN and Amazon

6.10 Issues Arise from the Case Study

In MTN CRM there are many problems, but the most important and clear of them is the management of the supply chain which includes two significant problems the first one is End-to-End performance guarantee and the second issue is SLAs metrics conflict. Now let us go through the case study to explain these issues in more details.

- SLA1 and SLA2 use the same metric response time but in SLA1 the metric measure by seconds while it measures by milliseconds in SLA2 the question now is how to resolve the conflict (units conflict).

THE SECOND CASE STUDY

A ONE-TIME PASSWORD (OTP) AUTHENTICATION VIA SMS

6.11 Introduction

A one-time password (OTP) is an automatically generated numeric or alphanumeric string of characters that authenticates the user for a single transaction or session. One Time Password technology increases security when logging in to a secure online environment or performing financial transactions. OTP messages are sent through SMS text messages or voice messages (IVR) and will be delivered on the handset of the user within 10 seconds or less. OTP text messages used in general to secure employee login sessions to a digital company portal. In addition to a variety of authentication benefits, OTP is accessible to everyone who owns a mobile or fixed phone. No doubt OTP is more cost-effective than the use of hardware tokens and very easy to use (UTHealth 2015).

6.12 Mobile Banking

Mobile banking is a service provided by a bank or other financial institution that allows its customers to conduct a range of financial transactions remotely using a mobile device such as a mobile phone or tablet, and using software, usually called an app, provided by the financial institution for the purpose. Mobile banking is typically available on a 24-hour basis. Some financial institutions have restrictions on which accounts may be accessed through mobile banking, as well as a limit on the amount that can be transacted (MariSol 2015). The types of financial transactions which a customer may transact through mobile banking include obtaining account balances and list of latest deals, electronic bill payments, and funds transfers between customers.

Figure 6-2 One-Time Password Message Sending

6.13 SLA between Consumer and Supplier

Real state commercial bank outsources its one time password transferring to a Smart Solution company. Smart Solution Company must transfer one time password to the bank's clients in Sudan. Some indicators of bank clients satisfaction should be identified for example the service must be on time, so a good performance indicator would be time elapsed between making a request and arrival of onetime password. Warranties associated with these indicators would be a maximum waiting time. Say 10seconds for a one-time password request. Since there will be many service instances during the life of the agreement, the warranty would likely be expressed in statistical terms, say response time 10 seconds for 95% of service instances.

```
<service name="sendOneTimePassWord" xsi:type="WSDLSOAPserviceDescriptionType">
    <SLAParameter name="ResponseTime" type="int" unit="seconds">
        <metric>averageresponsetime</metric>
        <provider>Smart Solution Company</provider>
        <consumer>Bank</consumer>
        <recipient>Bank Client</recipient>
    </SLAParameter>

    <obligation>
        <serviceLevelObjective name="conditionalSLOForResponseTime">
        <obliged>Smart Solution Company</obliged>
        <validity>
            <start>15-4-2016</start>
            <end>15-4-2016</end></validity>
        <expression>
            <implies>
                <expression>
                    <predicate xsi:type="less than or equal">
                    <SLAParameter>responsetime</SLAParameter>
                    <value>10</value>
                    </predicate>
                </expression>
            </implies>
        </expression>
        </serviceLevelObjective>
    </obligation>
</service>
```

SLA1 between Bank and Smart Solution Company

As shown in SLA1 Smart Solution Company has obligation to transfer the Password to the Bank client in average response time less than or equal 10 seconds.

Having warranties on performance measures is one thing, but in any complex interaction, problems can arise that may not have been planned for. It is generally better to have some procedure to manage issues rather than rely solely on warranties and penalties.

In the one-time password case, the bank might be able to terminate the agreement if Smart Solution Company is late more than 15 seconds in a month. Smart Solution Company might be able to end the deal if the real state commercial bank fails to make payment.

6.14 SLA between Supplier and Provider

To provide the service to real state bank Smart Solution Company has a contract with telecommunication company, this SLA says telecommunication Company must transfer the one-time password message to the bank clients.

A good performance indicator would be transaction rate as Smart Solution Company hires leased line from Telecommunication Company and leased line capacity known by a number of messages per time of unit, in this contract Telecommunication Company provides leased line Smart Solution Company can use to transfer 1000 messages per day. Also both of telecommunication company and Smart Solution Company concern with the number of messages they deliver to the bank clients because both of them want to know their profits. Warranties associated with this indicator would be average transaction rate 1000 messages per day for 80% of the time.

```
<service name="sendOneTimePassWord" xsi:type="WSDLSOAPserviceDescriptionType">
    <SLAParameter name="TransactionRate" type="int" unit="message/day">
        <metric>AverageTransactionRate</metric>
        <provider>MTN</provider>
        <consumer>Smart Solution Company</consumer>
    </SLAParameter>

    <obligation>
        <serviceLevelObjective name="conditionalSLOForResponseTime">
        <obliged>Smart Solution Company</obliged>
        <validity>
            <start>15-4-2016</start>
            <end>15-4-2016</end></validity>
        <expression>
            <implies>
                <expression>
                    <predicate xsi:type="greater than or equal">
                    <SLAParameter>transaction rate</SLAParameter>
                    <value>1000</value>
                    </predicate>
                </expression>
            </implies>
        </expression>
        </serviceLevelObjective>
    </obligation>
</service>
```

SLA2 between Smart Solution Company and Telecommunication Company

According to the contract between MTN and Smart Solution, MTN must provide 1000 transaction per day to Smart Solution.

6.15 One-time Password Message Scenario

Bank client send request to get a one-time password to complete the process of money delivery. The client sends request to the bank and waits for the one-time password he/she could use the password if he/she get it in less than or equal 20 seconds for the delivery session ends in 30 seconds as security policy commitment. If the client receives the password in more than 20 seconds, he/she will fail to complete the delivery process, and he/she has to send another request. The client wants response time less than or equal 20 seconds. After the bank receives the OTP request, directly the bank sends it to Smart Solution Company in less than or equal 5 seconds, so the request will be processed in the bank in less than or equal 5 seconds as a latency. Smart Solution Company receives the OTP message. Smart Solution company must send the message to appropriate Telecommunication company SMS server(MTN, ZAIN or Sudani) in less than or equal 5 seconds. Telecommunication Company receives the message from Smart Solution Company and transfers it to the client in less than or equal 5 seconds. The bank client concern with response time to be less than or equal 20 seconds.

If the service provided as described, the bank would get many benefits such as permutation in addition to delivery process fees. But if the average response time exceeds 20 seconds for more than 5% of total passwords request this is mean the bank fails to provide the delivery service. The bank could advise or force the clients to chose one of the telecommunication companies MTN, ZAIN or Sudani according to performance monitoring reports. In order for the warranties to have any force, they must be accompanied by financial penalties and perhaps rewards.

6.16 Problems Arise from OTP Case Study

In addition to all the problems mentioned in the first case study, there are many problems come to mind from this case study. First Smart Solution Company is responsible for transferring one-time password to the bank's clients in Sudan so it must monitor the supply chain from the end to end and this situation highlights the necessity of End-to-End performance guarantee. Second SLA1 and SLA2 use a different metric to measure the performance. Response time is used in SLA1, while transaction rate used in SLA2, so the question is how to resolve this conflict (metrics conflict).

6.17 Summary

In this chapter, two case studies have been provided. MTN CRM and A one-time password. These case studies aim to reflect the problems and issues that arise when designing End-to-End performance monitor such as End-to-End performance guarantee and metrics conflict. Chapter 7 shows how to solve these problems by running these case studies using proposed solution.

CHAPTER 7

PROPOSED SOLUTION

7.1 Introduction

Chapter 6 presents two case studies MTN CRM and a one-time password authentication via SMS. These two case studies raise problems that reflect thesis problem. Supporting-third party do audit trail is the appropriate strategy to provide End-to-End performance guarantee. But supporting third-party do audit trail strategy has a number of aspects. This chapter shows how the proposed solution solves these issues.

7.2 Case Studies Participants and Roles

There are shared concepts and points in two case studies. Participants and their roles are the same in two cases. The consumer who receives the service, the supplier who gets the service from the provider and provides it to the consumer, the provider who offers the service to the supplier, the supporting third party who receives the measured metrics from other participants to monitor the supply chain and these are all the participants and their roles in the supply chain. How does supporting third-party monitor this supply chain of SLAs?

Figure 7-1 Case Studies Participants and Roles

7.3 End-to-End SLAs Management

To monitor End-to-End SLAs, measuring the functionality is needed. Measurement service which receives metrics from the system's instrumentation and sends it to the third party. Instructions on how to measure a particular system parameter are defined in the measurement directives see figure 7-2 below. The role of the measurement function is to compute high-level metrics, e.g., the average response time of a complete cluster of servers in a particular period.

The set of metrics that are used as guarantees to the SLA are made available by the measurement function as SLA Parameters.

The condition evaluation function evaluates the guarantees of an SLA. Guarantees are defined as predicates over SLA Parameters.

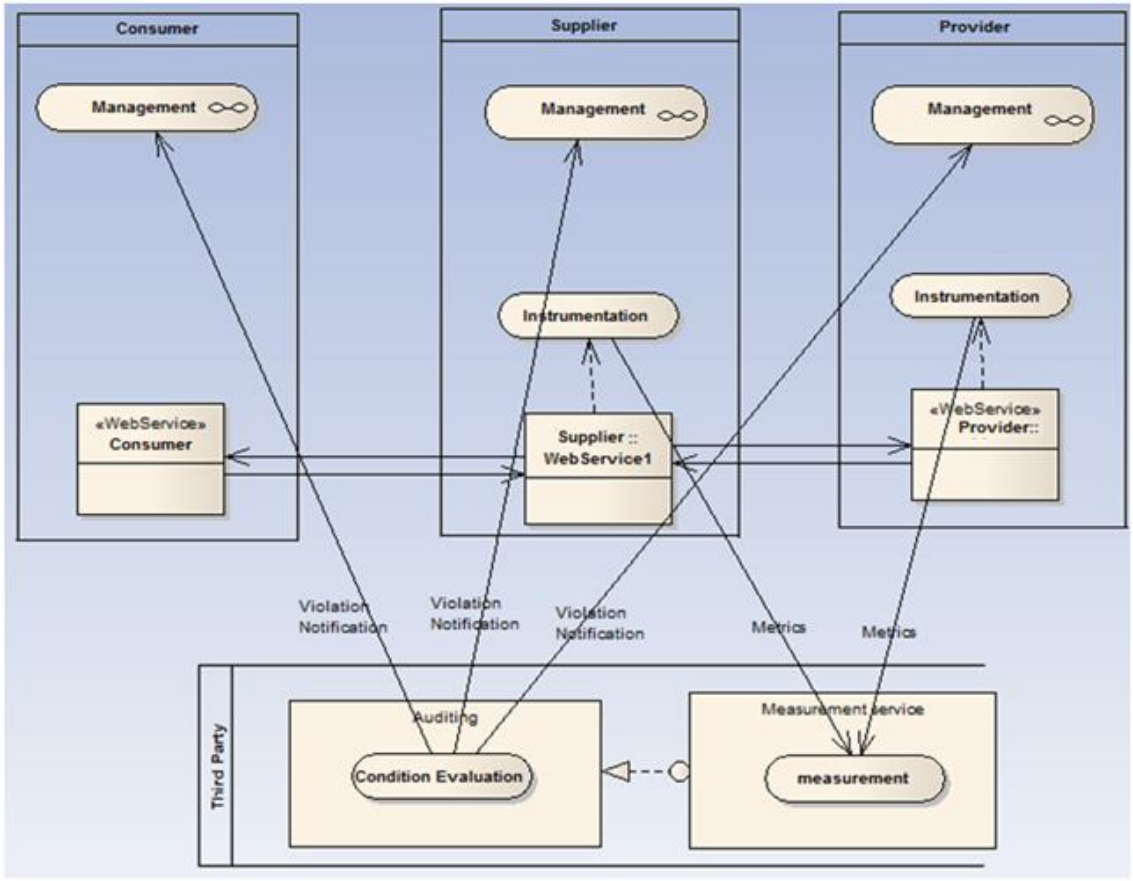


Figure 7-2 Supporting Third Party Do Audit Trail

7.4 Managing SLAs Using Third Party Strategy

Supporting third-party strategy provides End-to-End performance guarantee. Also, this strategy improves the QoS delivered to the ultimate consumer because the supporting third party can help consumers in the supply chain to find alternative solutions in case of violation. Furthermore, the supporting third party can help consumers and providers to discover and publish their services (Directory). Finally, the supporting third party can help to achieve the quality assurance, and it can give a quality certificate for every party who provide the service as agreed upon or as described. For all these advantages our proposed framework uses third-party strategy.

7.5 Overview of End-to-End Performance Monitoring Framework

End-to-End performance monitoring framework composes two stages: preparation stage and run-time stage. Figure 7-3 shows the architecture of the framework. Preparation stage which indeed required in End-to-End performance monitoring because in this situation many problems arise, so tasks at this stage solve them. Preparation stage comprises three tasks: the first task is receiving machine-readable SLAs from all parties. The second task is building metric ontology to avoid metric conflict problem. The last function in preparation stage is computing End-to-End performance to ensure that the supply chain can provide expected performance. Preparation stage can be passed as input to the run-time phase. Runtime stage includes four processes: gathering metrics, metrics aggregation, performance analysis which compares collected metrics with agreed upon to check if SLAs had been violated or not. According to the results of performance analyzer decision maker will choose a suitable decision. To use End-to-End performance monitoring framework these conditions must be met.

- All parties in the supply chain have to send SLAs to trusted third party.
- SLAs must be created using machine-readable languages such as WSLA and WS-Agreement this condition is needed for it is step to full automation.

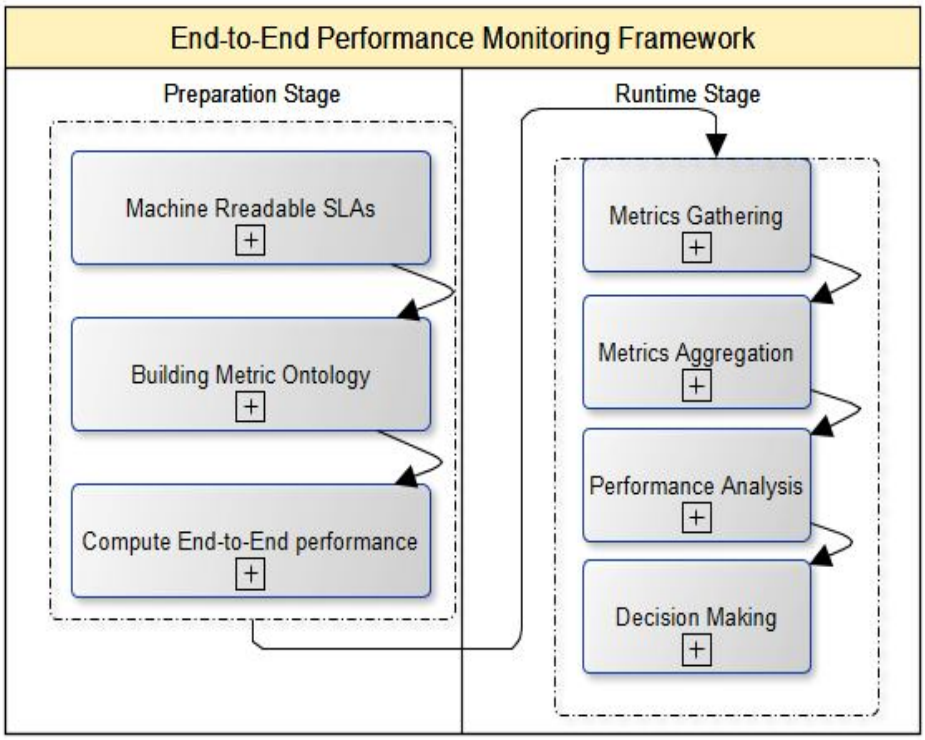


Figure 7-3 End-to-End Performance Monitoring Framework

The most advantage of the proposed framework it is a step to full automation. All tasks suggested by End-to-End performance monitoring framework can be done automatically. End-to-End performance monitoring framework had been developed to collects needed data related to the performance metrics in addition to the description of how to gather and process these data are of great help supporting analysts and engineers in making their decisions.

End-to-End performance monitoring framework figures out interoperability among autonomous systems. In the supply chain of SLAs different terms may be used for example different units of measuring for the same metric, different words used for the same terms(latency=delay) also the same word may have a different meaning. The proposed framework solves these problems by constructing metric ontology. Using the ontology server make it easy to query the ontology as shown in the appendix.

7.6 End-to-End Monitoring Aspects

End-to-End monitoring of service quality is interesting and relevant, but the problem of End-to-End monitoring has a number of aspects:

7.6.1 SLAs in A supply Chain are not Public

The SLAs in a supply chain are not necessarily public, so even though there may be audit trails at each stage, they may not be available. In a proposed solution, every party must commit to sending all information to the third party to monitor the supply chain.

7.6.2 Information Exchange Problem

The question now how can these parties exchange information? They can communicate using Simple Object Access Protocol (SOAP) message. SOAP invents no new technology; it builds on key Internet standards HTTP plus XML. SOAP is a simple messaging framework for transferring information between peers over Web in a distributed environment using XML. It is essential for application development to allow Internet communication between programs. Today's applications communicate using Remote Procedure Calls (RPC) between objects like DCOM and CORBA, but HTTP was not designed for this. A better way to communicate between applications is over HTTP because HTTP is supported by all Internet browsers and servers. SOAP was created to accomplish this. Also, SOAP message structure supports information exchange that needed to monitor the supply chain of SLAS.

7.6.3 Disputes Problem

The last section explains that there is a need for there to be some sort of exchange which does guarantee End-to-End performance, Each link in the supply network would have an SLA with the supporting third party, and the third party would monitor performance at each step. The service step would not necessarily be performed through the third party, but each service step would report to the third party with sufficient information to measure performance. All parties could perform its own monitoring, so there would need to be a mechanism for reconciling the customer’s measurements with the third party, and resolving disputes.

7.6.4 Resolving Dispute Problem Using Logs

To solve dispute problem logs are used. Logs are stored on providers which can then be analyzed. These logs contain useful information like:

- Date/time of access to your content
- The protocol used etc.
- HTTP Status
- Turn around time

These logs can be analyzed and managed by using third-party tools to measure the QoS. Supporting third party can do an audit trail (evidence) to test the correctness of log file information. Event logs record events which took place in the execution of a system in order to provide an audit trail that can be used to understand the activity of the system and to diagnose problems. An audit trail (also called audit log) is a security-relevant chronological record, set of records, and/or destination and source of records that provide documentary evidence of the sequence of activities that have affected at any time a specific operation, procedure, or event. The process that creates an audit trail is typically required to always run in a privileged mode, so it can access and supervise all actions from all users; a typical user should not be allowed to stop/change it (Wikipedia 2015).

7.7 End-to-End Performance Measuring

In some cases, the consumer sends a request to a supplier who in turn sends a request to provider ultimately back to the consumer. As shown in figure 7-1 services are composed in a sequential pattern. To compute response time RT and throughput T the following formulas can be used:

$$\mathrm{RT}(\textit{Sequential Supply chain}) = \sum_{i=1}^{n} \mathrm{RT}(S_i)$$

$$\mathrm{T}(\textit{sequential supply chain}) = \frac{1}{\sum_{i=1}^{n} \frac{1}{\mathrm{T}(S_i)}}$$

7.8 Metric Ontology

In the supply chain of SLAs, different terms may be used (different metrics, various functions) to measure SLAs parameter, this situation may solidify End-to-End performance monitoring.

- Different **units** of measuring for the same metric.
- Different words used for the same terms(latency=delay)
- The same word may have a different meaning.

So how does third-party deals with this situation? To solve these types of conflict metric ontology is built see figure 7-4.

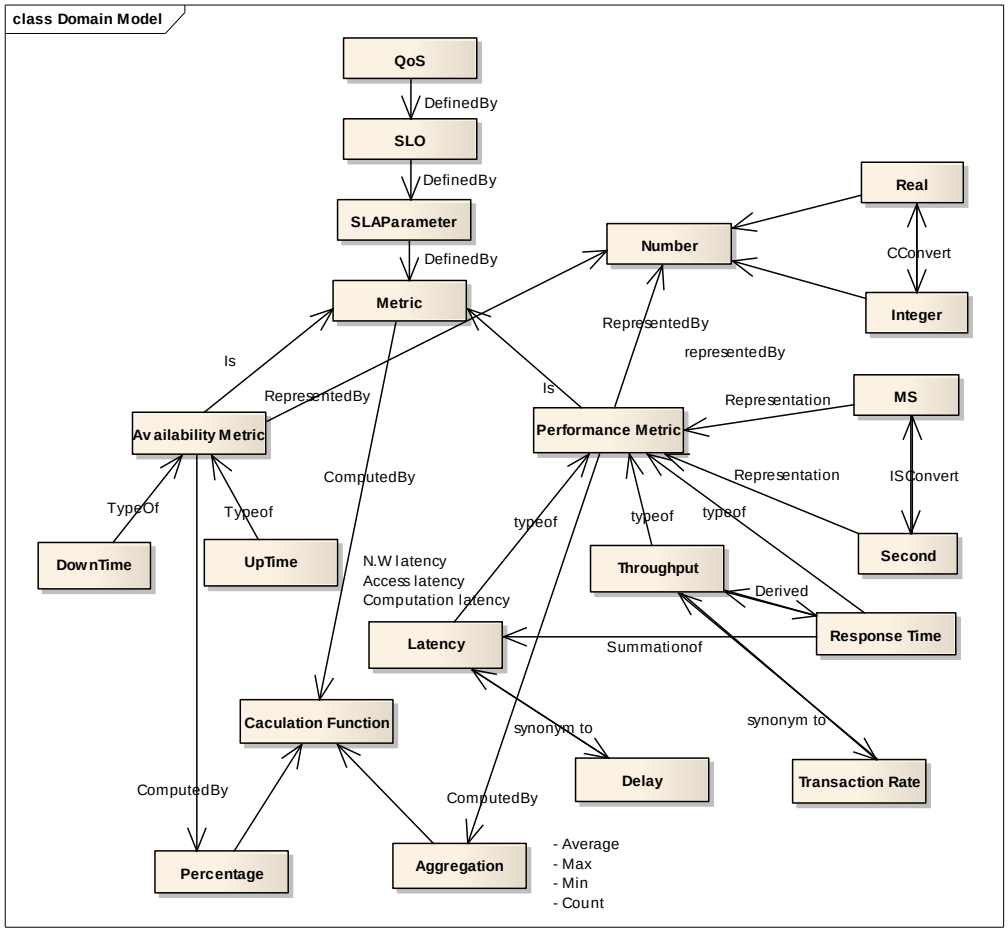


Figure 7-4 Metric Ontology Built from A collection of Imported Packages

This ontology shows that QoS is defined by service level objective which in turn determined by SLA parameter and SLA parameter defined by the metric. The most used metrics for performance and availability are response time, throughput, and latency. Response time is shown as a concept class with representations Millisecond and Seconds. The two representations are shown interconnected by a bidirectional property Convert. Convert designates a collection of methods which given an instance represented in some unit and convert it to another representation.

The dimension issue is made more complicated by the fact that a specification that a response time measured in time unit is not sufficient for interoperation. A program needs to know not only that the class is in seconds or milliseconds, but how the response time is represented as numbers (integers, real,…). Response time is a summation of different types of latency. Network latency, Access latency. And

computation latency is instance of latency. Delay is a synonym for latency. Also, throughput is a synonym for transaction rate. Uptime and downtime are types of availability metrics. Availability metrics computed by percentage.

7.8.1 Querying Metric Ontology

Metric Ontology can be used to solve many problems such as metric conflict. For example, if some party use term delay while other party use latency instead of delay, metric Ontology solves this type of conflict because delay and latency are declared as a synonym classes thus one can makes a query using delay or latency.

7.9 Summary

This chapter presented proposed framework which suggested to monitor the supply chain of SLAs. Ontology metrics solves metrics conflict problem. This ontology solved two type of metric conflict a synonym problem and unit conflict problem which may occur in case of using different units for the same metric. Also selected End-to-End SLAs monitoring strategy solves most of the supply chain management problem in addition to the extra advantages which can be provided by using this strategy. Next chapter will explain these results in more details.

CHAPTER 8

SUMMARY OF RESULTS AND FUTURE WORK

8.1 Introduction

This chapter discuss the summary of the results and findings obtained by the researcher. An overview of the most crucial research and contributions of the thesis are presented. In addition to, a future work that can be done in the area of End-to-End SLAs monitoring.

8.2 Summary of Results

This study aims to provide an End-to-End performance guarantee which can be achieved by End-to-End monitoring strategies. And researcher found that the supporting third party do audit trail (Principle B) is the appropriate strategy for many reasons. Firstly, third party's audit trail strategy provides End-to-End performance guarantee. Secondly, this strategy improves the QoS delivered to the ultimate consumer. Thirdly, the supporting third party can help providers and consumers to publish and discover services. Finally, the supporting third party can help all parties in the supply chain to achieve the quality assurance. But the problem of End-to-End monitoring has a number of aspects. Although the proposed framework is developed to work in SOA environment, it can work in cloud computing platform because the situation is the same, in addition to cloud computing is SOA based.

8.3 Research Contributions

The first contribution is proposing an End-to-End monitoring strategies to provide End-to-End performance guarantee. these strategies include Audit Trail at each stage (Principle A), the third party do audit trail strategy (Principle B), End To End Performance Predicate Strategy (Principle C), and Cost Versus Performance Trade-Off Strategy (Principle D).

The second contribution is extending the WSLA metamodel using End-to-End performance guarantee profile. This extension makes WSLA model more informative

and expressive in representing SLAs metrics for WSLAUP provides semantic representation for SLA metrics.

Also, this thesis proposed formulas to measure End-to-End performance through a supply chain.

Finally, thesis proposed End-to-End SLAs monitoring framework to help SLAs monitor designers to develop End-to-End SLAs monitor.

8.4 Future Work

Supporting third-party can help all parties in the supply chain to achieve the quality assurance more work is to be carried out in this area. Also, to merge End To End Performance Predicate strategy with trusted support party strategy. Trusted third party can perform the subtract operation in addition to managing and controlling all the supply chain.

REFERENCES

Ahmed Al-sagaf, A.A. and Jawawi, D.N.A., Standard Monitor Design for SLA Parameters in SOA. 2012.

Al Falasi, A., Serhani, M.A. and Dssouli, R., 2013, December. A model for multi-levels SLA monitoring in federated cloud environment. In Ubiquitous Intelligence and Computing, 2013 IEEE 10th International Conference on and 10th International Conference on Autonomic and Trusted Computing (UIC/ATC) (pp. 363-370). IEEE.

Ameller, D. and Franch, X., 2008, February. Service level agreement monitor (SALMon). In Composition-Based Software Systems, 2008. ICCBSS 2008. Seventh International Conference on (pp. 224-227). IEEE.

Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Nakata, T., Pruyne, J., Rofrano, J., Tuecke, S. and Xu, M., 2007, March. Web services agreement specification (WS-Agreement). In Open grid forum (Vol. 128, No. 1, p. 216).

Bartsch, Frank. "Supply Chain Management (SCM)". BB Handel. Retrieved 19 June 2013

Bertolino, A., De Angelis, G., Frantzen, L. and Polini, A., 2008. Model-based generation of testbeds for web services. In Testing of Software and Communicating Systems (pp. 266-282). Springer, Berlin, Heidelberg.

Bianco, P., Lewis, G.A. and Merson, P., 2008. Service level agreements in service-oriented architecture environments (No. CMU/SEI-2008-TN-021). Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst.

Colomb, R. 2007a. Ontology and the Semantic Web. Frontiers in Artificial Intelligence and Applications, vol. 156. IOS press.

Colomb, R., Service Science, Management and Engineering, Module 7 Service-Level Agreements 2010.

D. Bianculli and C. Ghezzi. "Monitoring Conversational Web Services". InProcs of IW-SOSWE, pages 15-21, Dubrovnik, Croatia, September 2007.

David Berrios, 2014, challenges-in-supply-chain-management[online], available: <https:// bus.wisc.edu/Mba/supply-chain-management/blog/2014/03/06/ challenges-in-supply-chain-management>/[accessed 13-8-2015].

H. Schmidt, “Service Level Agreements based on Business Process Modeling”, HP OpenView University Association Workshop, 2000.

HugeDomains, SLA why it is important[online], available:<http://www.Acquisitioninstitute/uploads/ sla_why_it_is_important.pdf>/ [accessed 13-8-2015].

Lee, T.B., Hendler, J. and Lassila, O., 2001. The semantic web. Scientific american, 284(5), pp.34-43.

Ludwig, H., Keller, A., Dan, A., King, R.P. and Franck, R., 2003. Web service level agreement (WSLA) language specification. Ibm corporation, pp.815-824.

MariSol Federal Credit Union, Mobile Banking 101, available:< https://www.marisolcu.org/pdf/Mobile%20Banking%20101%20-%20User%20Guide%202016.pdf, - >/ [accessed 13-8-2015].

Mohamed, M., Anya, O., Tata, S., Mandagere, N., Baracaldo, N. and Ludwig, H., 2017. rSLA: An Approach for Managing Service Level Agreements in Cloud Environments. International Journal of Cooperative Information Systems, 26(02), p.1742003.

Molina-Jimenez, C., Shrivastava, S., Crowcroft, J. and Gevros, P., 2004, July. On the monitoring of contractual service level agreements. In Electronic Contracting, 2004. Proceedings. First IEEE International Workshop on (pp. 1-8). IEEE.

Nadeem Ahmed, 2005, The Importance of Service Level Agreements in Outsourcing Contracts[online],available:<http://libguides.scu.edu.au/c.php?g=356711&p=2406583>/[accessed 13-8-2015].

OMG, 2011b. OMG Unified Modeling Language TM (OMG UML), Superstructure. , (January). Available at: http://www.omg.org/spec/UML/2.4/Superstructure.

OMG, 2007, OMG unified modeling language (OMG UML), Superstructure, V2. 1.2.

Perisic, B., 2014. Model driven software development-state of the art and perspectives. Invited Paper, INFOTEH, pp.1237-1248.

Rizvi, S., Roddy, H., Gualdoni, J. and Myzyri, I., 2017. Three-Step Approach to QoS Maintenance in Cloud Computing Using a Third-Party Auditor. Procedia Computer Science, 114, pp.83-92.

Schulz, F., 2010, June. Towards measuring the degree of fulfillment of service level agreements. In Information and Computing (ICIC), 2010 Third International Conference on (Vol. 3, pp. 273-276). IEEE.

Simon, E. and Graham, M., 2011. Toward reusable SLA monitoring capabilities. School of Computing Science, Newcastle University, UK.

Sirin, E., Hendler, J. and Parsia, B., 2003, April. Semi-automatic composition of web services using semantic descriptions. In 1st Workshop on Web Services: Modeling, Architecture and Infrastructure (pp. 17-24).

Specification, O.A., 2007. OMG unified modeling language (OMG UML), Superstructure, V2. 1.2. Object Management Group.

Spiegel, M.R., Schiller, J.J., Srinivasan, R.A. and LeVan, M., 2009. Probability and statistics (Vol. 2). New York: Mcgraw-hill.

Ta, X. and Mao, G., 2006, September. Online end-to-end quality of service monitoring for service level agreement verification. In Networks, 2006. ICON'06. 14th IEEE International Conference on (Vol. 2, pp. 1-6). IEEE.

UTHealth, Self Service Password Reset (SSPR), available:< https://www.uth.edu/it/sspr/>/[accessed 13-8-2015].

Wikipedia, Audit trail, available: < https://en.wikipedia.org/wiki/Audit_trail, Audit trail >/[accessed 13-8-2015].

Wikipedia, log file, available: < https://en.wikipedia.org/wiki/Computer_data_logging, Log file >/[accessed 13-8-2015].

Wikipedia, Service-level agreement, available https://en.wikipedia.org/wiki/Service-level_agreement, [accessed 14-8-2018] .

Wikipedia, Supply chain, available:< https://en.wikipedia.org/wiki/Supply_chain, Supply chain > /[accessed 13-8-2015].

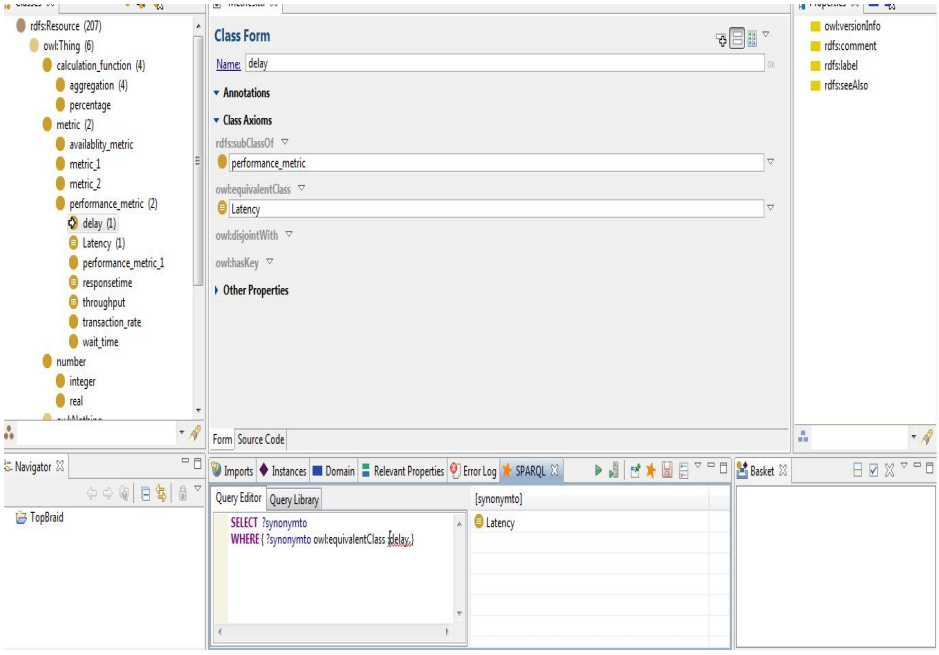
Wikipedia , Supply chain management, available:<https://en.wikipedia.org/wiki/Supply_chain_management>/[accessed 13-8-2015].

Winkler, M . , J. Cardoso, and G. Scheithauer, "Chall enges of Business Service Monitoring in the Internet of Services". iiWA S2008, ACM, A ustria. 2008. P P : 6 1 3-6 1 6.

Zeginis, C., 2009. Monitoring the QoS of Web services using SLAs–Computing metrics for composed services. Master's Thesis, University of Crete Computer Science Department, Heraklion.

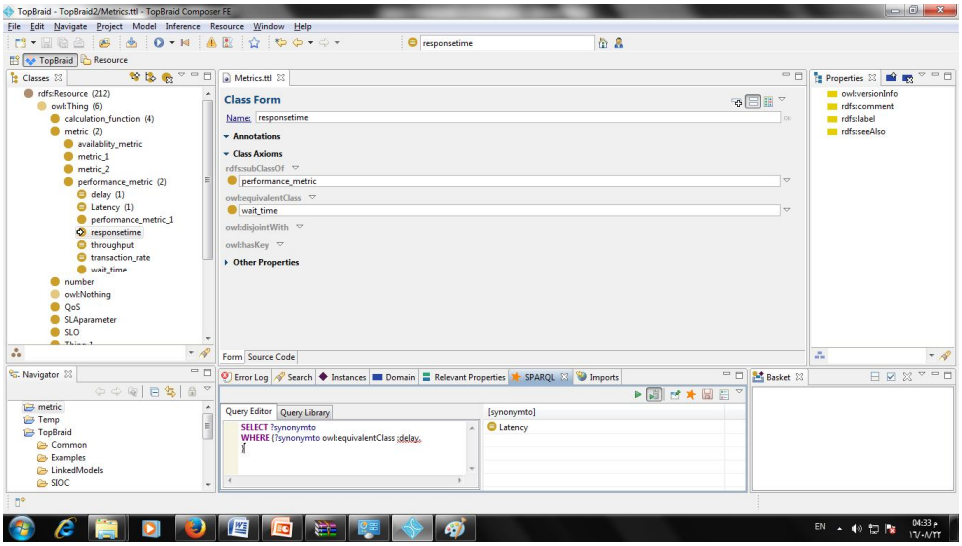
APPENDIX A

Querying Metric Ontology

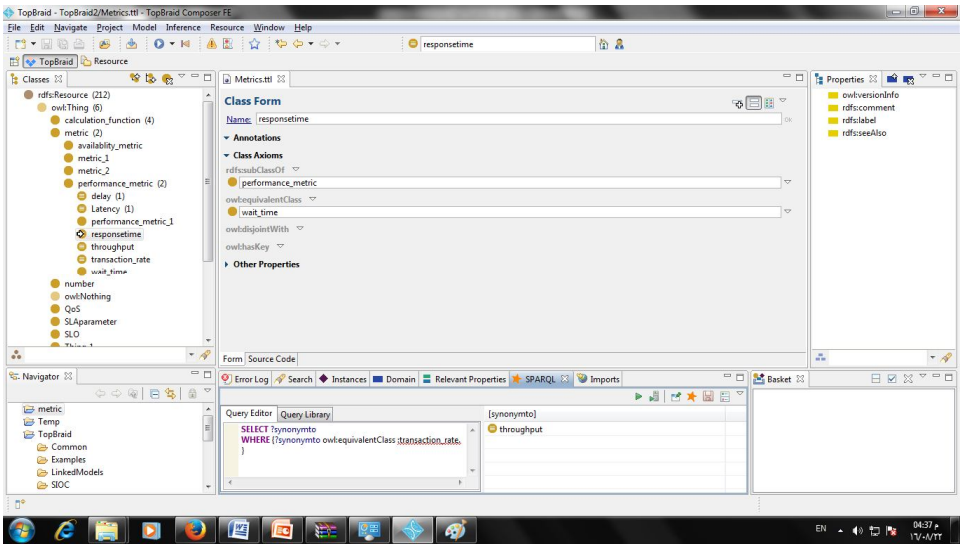


Snapshot of metric Ontology query using TopBraid tool

Latency and delay are defined as equivalent classes so now the party can use any one of these terms delay or latency without any conflict because delay and latency are declared as a synonym to each other.



Snapshot of metric Ontology query synonym problem



Metric Ontology query synonym problem

APPENDIX B

Metric Ontology Exported to Java RDFS/OWL Schema Class

```
package ;

import com.hp.hpl.jena.rdf.model.Property;

import com.hp.hpl.jena.rdf.model.Resource;

import com.hp.hpl.jena.rdf.model.ResourceFactory;

/**

 * Vocabulary for http://example.org/unnamed

 *

 * Automatically generated with TopBraid Composer.

 */

public class metricontologyinjava {

 public final static String BASE_URI = "http://example.org/unnamed";

 public final static String NS = BASE_URI + "#";

 public final static Resource _seconds =
ResourceFactory.createResource(NS + "_seconds");

 public final static Resource seconds =
ResourceFactory.createResource(NS + "seconds");

 public final static Resource Latency =
ResourceFactory.createResource(NS + "Latency");

 public final static Resource QoS = ResourceFactory.createResource(NS
+ "QoS");

 public final static Resource SLAparameter =
ResourceFactory.createResource(NS + "SLAparameter");
```

```
 public final static Resource SLO = ResourceFactory.createResource(NS
+ "SLO");

 public final static Resource Thing_1 =
ResourceFactory.createResource(NS + "Thing_1");

 public final static Resource Thing_2 =
ResourceFactory.createResource(NS + "Thing_2");

 public final static Resource Thing_3 =
ResourceFactory.createResource(NS + "Thing_3");

 public final static Resource Thing_4 =
ResourceFactory.createResource(NS + "Thing_4");

 public final static Resource Thing_5 =
ResourceFactory.createResource(NS + "Thing_5");

 public final static Resource aggregation =
ResourceFactory.createResource(NS + "aggregation");

 public final static Resource availablity_metric =
ResourceFactory.createResource(NS + "availablity_metric");

 public final static Resource average =
ResourceFactory.createResource(NS + "average");

 public final static Resource calculation_function =
ResourceFactory.createResource(NS + "calculation_function");

 public final static Resource count = ResourceFactory.createResource(NS
+ "count");

 public final static Resource delay = ResourceFactory.createResource(NS
+ "delay");

 public final static Resource downtime =
ResourceFactory.createResource(NS + "downtime");
```

```
 public final static Resource integer =
ResourceFactory.createResource(NS + "integer");

 public final static Resource max = ResourceFactory.createResource(NS
+ "max");

 public final static Resource metric =
ResourceFactory.createResource(NS + "metric");

 public final static Resource metric_1 =
ResourceFactory.createResource(NS + "metric_1");

 public final static Resource metric_2 =
ResourceFactory.createResource(NS + "metric_2");

 public final static Resource millisecond =
ResourceFactory.createResource(NS + "millisecond");

 public final static Resource min = ResourceFactory.createResource(NS +
"min");

 public final static Resource number =
ResourceFactory.createResource(NS + "number");

 public final static Resource percentage =
ResourceFactory.createResource(NS + percentage");

 public final static Resource performance_metric =
ResourceFactory.createResource(NS + "performance_metric");

 public final static Resource performance_metric_1 =
ResourceFactory.createResource(NS + "performance_metric_1");

 public final static Resource real = ResourceFactory.createResource(NS +
"real");

 public final static Resource responsetime =
ResourceFactory.createResource(NS + "responsetime");
```

```
 public final static Resource second =
ResourceFactory.createResource(NS + "second");
 public final static Resource throughput =
ResourceFactory.createResource(NS + "throughput");
 public final static Resource time_unit =
ResourceFactory.createResource(NS + "time_unit");
 public final static Resource transaction_rate =
ResourceFactory.createResource(NS + "transaction_rate");
 public final static Resource uptime =
ResourceFactory.createResource(NS + "uptime");
 public final static Resource wait_time =
ResourceFactory.createResource(NS + "wait_time");
 public static String getURI() {
 return NS;
 }
}
```