An analysis of web service SLA management infrastructures based on the C-MAPE model

Y. Tang*
Department of Computer Science and Technology,
Xi’an Jiaotong University,
28 Xianning Road West, Xi’an, Shaanxi, 710049, China
E-mail: yztang@mail.xjtu.edu.cn
*Corresponding author

H. Lutfiyya
Department of Computer Science,
Middlesex College, University of Western Ontario,
London, Ontario, N6A 5B7, Canada
E-mail: hanan@csd.uwo.ca

V. Tosic
NICTA and The University of New South Wales,
NICTA, Locked Bag 9013, Alexandria NSW 1435, Sydney, Australia
and
Middlesex College, The University of Western Ontario,
London, Ontario, N6A 5B7, Canada
E-mail: vladat@computer.org

Abstract: Quality of service (QoS) management is becoming an integral part of current web services. Contracts are playing a very important role in MAPE-based (monitor, analyse, plan and execute) web service QoS management. Therefore, we propose a general model, contract-MAPE (C-MAPE) model, and based on this model, review five major projects in this area. The analysis results show that these major research projects are quite much following C-MAPE model, whereas they give different support on different aspects. We also analyse possible ways to achieve integrated QoS management, for web services and their compositions. This paper contributes to the understanding of automatic processing of formal contracts, as well as of the relationship between contracts, QoS management systems, and web service provisioning systems.

Keywords: contracts; service level agreements; SLA; web services; web service management; quality of service; QoS; QoS management; management infrastructures; MAPE loop; business process integration; business process management.


Biographical notes: Yazhe Tang is an Associate Professor at Computer Science and Technology Department of Xi’an Jiaotong University. He received his PhD from Xi’an Jiaotong University, China. His research focuses on the areas of web service management, network management, network security and grid management.

Hanan Lutfiyya is a Full Professor at University of Western Ontario, Canada. Her research interests are autonomic computing, distributed applications and system management, policy-based management, QoS management and pervasive computing.

Vladimir Tosic is a Researcher at NICTA, an Adjunct Research Professor at the University of Western Ontario, Canada, and a Visiting Fellow at the University of New South Wales, Australia. He received his PhD from Carleton University, Canada. He won the 2001 Upsilon Pi Epsilon/IEEE Computer Society Award for Academic Excellence. He previously worked as an Assistant Professor at Lakehead University, Canada and a Software Engineer in the OpenView Software Division of Hewlett-Packard Company in Germany. His research interests are business-driven management of information technology systems and comprehensive description and middleware for management of web services.
1 Introduction

Web services are an increasingly important technology for application development in e-commerce, peer-to-peer (P2P) and business-to-business (B2B) areas. These applications often have quality of service (QoS) requirements (performance or availability requirements). Thus, management of web services (MOWS) is essential for successful deployment of applications that are developed based on this technology.

There has been considerable research related to the QoS MOWS. Different projects emphasised different aspects of web service management. The most important aspects are:

1 specification of QoS requirements and guarantees, which are usually represented in a contract
2 a management infrastructure that performs monitoring and (hopefully) control activities.

Although different architectures for web service management adopt various choices and designs, we believe that they have the same underlying model. In particular, we find that they are based on various contracts (formal agreements) for specification of QoS requirements guarantees and that they implement some or all of the monitor, analyse, plan and execute (MAPE) management activities identified by Kephart and Chess (2003). Therefore, we propose the contract-based MAPE (C-MAPE) model for study, analysis and comparison of web service QoS management architectures. Note that while Tian et al. (2004b) and Tosic (2004) also contain overviews of this area, this paper presents a general management model and analysis missing in past publications. Since there are already several surveys and critical analyses of contract-based web service QoS specification languages, most notably Tosic and Pagurek (2005) and Patel (2003), this paper will focus on critical analysis of web service QoS management infrastructures.

The rest of the paper is organised as follows. In Section 2, the general C-MAPE model is presented and the importance of contracts in this model is discussed. In Section 3, five most important web service QoS management projects are analysed in detail, based on the C-MAPE model. In Section 4, we discuss the analysis results from Section 3 and also other important topics, e.g., integrated management. Section 5 provides some general comments and concluding remarks.

2 C-MAPE model

MAPE is introduced in the autonomic computing project of IBM, which models the management architecture as a control loop (MAPE-k loop) with four common functions:

1 monitoring refers to the retrieval of the information about the run-time behaviour of the systems and applications executing on the systems
2 analysing refers to examination of collected information, e.g., comparison of collected information with pre-defined thresholds
3 planning refers to decision-making about which actions should be taken if QoS requirements are not satisfied
4 executing refers to the carrying out of the actions, e.g., taking actions to bring the system back to its normal state if not all QoS requirements are being met.

Note that here the ‘k’ refers to a common knowledge base that represents the knowledge about a problem domain. Examples of knowledge include topology, system logs, performance metrics, etc.

3 Analysis based on the C-MAPE model

Specific challenges in web service management include the following:

1 Applications using web service technologies are internet based distributed applications. The performance of these applications partially relies on the underlying internet network and applications or services provided by other businesses. Unfortunately, internet infrastructure and services provided by other businesses (in different administrative domains and opposing different management goals) can not be managed completely.
2 Applications based on web services technologies are often consumer oriented. Different consumers may have different QoS requirements. These requirements will often be encapsulated in a contract. Thus the ability to distinguish between different consumers and the ability to specify contracts are needed.

Therefore, for web service QoS management, we propose the C-MAPE model, which stands for contract-based MAPE model. Service level agreement (SLA), which is most used in web services area is a special type of QoS contract. The SLA specifies QoS requirements and guarantees between providers and consumers. SLA usually includes:

1 The involved parties (Keller and Ludwig, 2003). This should distinguish between providers, consumers and third parties, which are usually responsible for service measurement or service evaluation.
2 The time period that the SLA applies to. This usually includes the begin and end times of the SLA.
3 QoS service level objectives (SLO), which are used to describe the QoS requirements and guarantees, e.g., response time of the service that provider ‘A’ serves for consumer ‘B’ should be less than two seconds. The contents of the SLO consist of the metrics that need to be measured or calculated and the constraints that need to be evaluated.
Measurable metrics and how they can be used to calculate a parameter that is specified in an SLO.

Other information that includes penalties, prices, scope, etc.

Note that in this paper, a contract usually means an electronic contract. So we will use SLA, contract and electronic contract interchangeably in this paper.

Contracts are part of knowledge ('k'), but they are very important and emphasised in this paper for the following reasons:

1. A contract is the starting point of web service QoS management. The content of a contract is used for management configuration, deployment and implementation, and it will largely influence MAPE elements: an SLO in the contract defines a goal of QoS management. Parameters and metrics in contracts specify data to be collected, and where and how to collect them.

2. Contracts have a life cycle (Keller and Ludwig, 2003) that includes contract negotiation, contract deployment, contract measurement (Monitor in MAPE), contract evaluation (Analysis in MAPE), contract enforcement (Plan and Execute in MAPE) and contract termination. This life cycle plays a significant role in QoS management. Support for the life cycle is a major factor in our discussion and analysis of web service management architectures.

3. As we will discuss later, formal contracts and the corresponding compilers or other tools could automate configuration and deployment of management systems. Most of the projects on web service management are based on formal or semi-formal contracts. This is important since web services and the corresponding management systems are both dynamic. When a new contract is signed and the related service is deployed, the management system must be configured quickly. This is impossible if there are no formal contracts and corresponding tools or facilities to automate the configuration and deployment process.

Figure 1 depicts the C-MAPE model. It shows the importance of contracts in web service QoS management. For instance, if there is an SLO specifying 'If service response time is more than 20 seconds, then e-mail the administrator Bob', then the management system will know:

1. the QoS parameter that should be monitored is 'response time'
2. the management system will do the analysis work to see if 'response time is more than 20 seconds'
3. in the case of a delayed response, the management system will plan and execute certain actions – in this case, 'e-mail the administrator Bob'.

This section critically analyses the representative research in web services QoS management: WS-agreement/creation and monitoring of agreements (Cremona) (Ludwig et al., 2004; Andrieux et al., 2004), web service level agreement (WSLA) (Keller and Ludwig, 2003; Dan et al., 2004), web service offering language/web service offering infrastructure (WSOL/WSOI) (Tosic, 2004; Patel, 2003), WS-QoS (Tian et al., 2004a; Gramm et al., 2003) and web service management network/web service management language (WSMN/WSML) (Sahai et al., 2002; Machiraju et al., 2002). They were chosen for the following reasons:

1. they are regarded as the most typical and influential architectures
2. relatively speaking, they are considered the most complete architectures in that they have contract specification languages and corresponding management infrastructures.

The purpose of this analysis is to find common requirements of such systems; evaluate their advantages/disadvantages; and determine possible future directions. The analysis will be mainly based on the C-MAPE model. We will study questions such as whether the architecture contains all the functional building blocks of MAPE and to what extent it supports MAPE elements. Also, we will consider contract-related issues such as contract negotiation, contract verification, contract deployment, contract adaptation and contract termination. Table 1 shows these analysis criteria and their meaning. It is the legend for Tables 2 and 3.
<table>
<thead>
<tr>
<th>Analysis criteria name</th>
<th>Meaning</th>
<th>Legend</th>
</tr>
</thead>
<tbody>
<tr>
<td>M</td>
<td>Collecting management data</td>
<td>Yes or no</td>
</tr>
<tr>
<td>A</td>
<td>Analysing the management data</td>
<td>Yes or no</td>
</tr>
<tr>
<td>P</td>
<td>Planning to take actions</td>
<td>Yes or no</td>
</tr>
<tr>
<td>E</td>
<td>Executing the actual actions</td>
<td>0 No support</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 Notification</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 Price or penalty</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 Having interfaces for complex actions</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 Actual execution of control operations</td>
</tr>
<tr>
<td>Substitutability of contract languages</td>
<td>Can the architecture support more than one contract language?</td>
<td>Yes or no</td>
</tr>
<tr>
<td>Negotiation of contracts</td>
<td>To what level does the architecture support contract negotiation?</td>
<td>0 Fixed contracts</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 Choice between options</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 Limited creation of new options;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 Negotiability of content (not format)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 Fully negotiable about format and content</td>
</tr>
<tr>
<td>Automatic negotiation of contracts</td>
<td>To what level does the architecture support automatic negotiation of contracts?</td>
<td>0 Manual</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 Semi-automated</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 Automated</td>
</tr>
<tr>
<td>Contract verification</td>
<td>Support for contract syntax, semantic consistency checks and guarantees that the infrastructure has the modules and resources to support the contracts</td>
<td>1 Language syntax checks</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 Conflicts checks of QoS metrics and SLOs</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 Existing checks of supporting modules in the infrastructure for QoS metrics monitoring</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 Mapping QoS guarantees to underlying resources</td>
</tr>
<tr>
<td>Automatic deployment according to the contract</td>
<td>Are there compilers or tools to generate the configuration file or even code from formal contracts?</td>
<td>Yes or no</td>
</tr>
<tr>
<td>Contract adaptation</td>
<td>The ability to change contracts dynamically</td>
<td>0 Renegotiate from scratch;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 Automatic switching to another contract</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 Arbitrary modification of current contract</td>
</tr>
<tr>
<td>Contract Termination</td>
<td>Flexibility in terminating a contract</td>
<td>1 Shut down</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 End of session</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 Time-based</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 Event-based</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 Switch to another contract</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 Explicit request but non negotiable</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 Fully negotiable</td>
</tr>
<tr>
<td>Contract deactivation/ reactivation</td>
<td>The ability to temporarily disable/ enable the use of the contract for consumers and to handle affected consumers</td>
<td>Yes or no</td>
</tr>
<tr>
<td>Mapping between QoS guarantees in contracts and resources</td>
<td>Certain SLOs specified in contracts will need a certain amount of resources to guarantee it. Do the contract and the corresponding infrastructure support this kind of mapping?</td>
<td>Yes or no</td>
</tr>
</tbody>
</table>
3.1 Analysis based on MAPE related criteria

Table 2 shows the analysis results based on the MAPE related criteria. These results imply that all of the examined projects support monitoring and analysing but give various supports to planning and executing corrective actions. In WSLA (Keller and Ludwig, 2003), two services are responsible for monitoring and analysing the collected data. The service for monitoring is measurement service, and the service for analysing data is condition evaluation service. The measurement service is configured by retrieving the service deployment information (SDI) from a contract (SLA). It measures SLA parameters such as response time or availability. The condition evaluation service accomplishes the analysis by comparing measured values of the parameters with the thresholds, both defined in the SLO. WSLAs management service fulfills the tasks of planning and executing actions. It makes simple planning decisions (based on policy-like ActionGuarantee clauses in SLA) and takes actions to correct the problem when the condition evaluation service sends a notification about the violation of an SLO. The corrective actions are now limited to opening a trouble ticket or sending an event to the management system. WS-agreement (Andrieux et al., 2004) defines structure of agreements templates and agreements, basic protocols for establishing agreements and a simple interface to monitor agreements. This simple interface for monitoring agreements is the way to perform monitor and analysing tasks in WS-agreement. Through this interface, the runtime status of every term (analogous to SLO) of an agreement can be introspected and analysed. Cremona (Ludwig et al., 2004) is an architecture for the WS-agreement implementing infrastructure. Among all the examined architectures in this paper, Cremona is the only one that relates the agreement (contract) to the underlying resources of the service providers. The compliance monitor checks whether guarantees are met then decides what actions will be taken. Then it invokes the compliance manager interface to take corrective actions and change the system configuration.

<table>
<thead>
<tr>
<th>M</th>
<th>A</th>
<th>P</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSLA</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>WS-agreement/Cremona</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>WSOL/WSOI</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>WSMN/WSML</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>WS-QoS</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

WSOI (Tosic, 2004) is based on extensions of Apache Axis, an open source SOAP engine. From WSOI (Patel, 2003) files, the WSOL compiler under development will automatically generate WSOL-specific handlers. Among them, QoS-measurement WSOI handlers and constraint-evaluation WSOI handlers are responsible for monitoring and analysing, respectively. The WSOL-specific handlers are inserted into the configurable chains to intercept SOAP message, measure and monitor the performance of service offerings (SOs). WSOL/WSOI supports planning and executing actions by specifying prices and penalties in management statements and calculating these prices or penalties at runtime.

WSMN (Machiraju et al., 2002) is an intermediary-based management architecture. Machiraju et al. (2002) argues that the SOAP router is an ideal place to embed the intermediaries. WSMN engines for measurement and SLA management are designed for the task of monitoring and analysing. WSML specifies in evalAction clauses what actions should be taken. These actions are usually notifications.

In WS-QoS (Tian et al., 2004a; Gramm et al., 2003), there are two kinds of scenarios related to monitoring and analysing. One happens before a consumer invokes the web service operations. The service broker will test the service whose provider claims to support certain QoS guarantees. This is a comparison and matching process, but can be viewed as a special kind of monitoring and analysing. On the other hand, QoS channels will do the tasks of monitoring and analysing at runtime. In this scenario, the consumer puts the requirements for the performance data into a SOAP header. Accordingly, the server will ‘understand’ the requirements and deliver the required information to the consumer. Therefore, the consumer can get the state of the server, such as the processing time of current request and worse case and average values for processing requests of different classes. The actual monitoring and analysing activities are done at consumer-side WS-QoS monitor. WS-QoS does not support automatic planning and executing corrective actions. These tasks are left to people (e.g., administrators).

3.2 Analysis based on contract related issues

Table 3 shows the analysis results based on the contract related issues defined in Table 1.

All architectures (except Cremona, which is a common architecture and does not define the actual contract language) adopt XML for contract representation and XML schema for grammar definition. All of these XML-based languages contain basic contract elements such as parties, SLOs and related parameters or metrics. There are some differences between them, including the following:

1. The contents of contracts of the different architectures differ. For example, WSOL (Patel, 2003) contracts contain not only QoS requirements and guarantees, but also functional constraints and access control constraints, while other contracts are only concerned with QoS issues. Further, WSOL does not contain a definition of QoS metrics, but outsources it to external ontologies. For example, a simple reference to a metric in WSOL may look like QoSMetricOntology: ResponseTime. In WSLA (Keller and Ludwig, 2003) and WSML (Sahai et al., 2002), QoS metrics are fully defined in the contracts, while in WS-QoS (Gramm et al., 2003), the metrics are defined in the XML schema.
of the contracts. Thus in WS-QoS, if a new metric needs to be added, the schema must be modified first.

2 To ease the effort of composing contracts, various reuse mechanisms are introduced in different architectures. WSOL uses constraint groups, inheritance of SOs, templates and inclusion mechanisms to perform specification reuse. For example, every SO definition can contain an attribute named extends, which specifies the name of a previously defined SO that the new SO extends through single inheritance. WSLA can support reuse by grouping specification and templates, while WSML can only support reuse by templates. Note that in WS-QoS, there is no explicit contract signed between consumers and providers. Instead, both consumers and providers specify their own QoS definitions. A management component called service broker makes comparisons to determine which service offer is suitable for the consumer. Once a web service is chosen, its QoS guarantees will be regarded as the contract’s QoS guarantees.

WS-agreement (Andrieux et al., 2004) specification only defines the overall structure of agreements and agreement templates. Thus, it is the only architecture that supports more than one contract language.

Contract negotiation can be very complex due to the large variety of QoS parameters and metrics. WSLA (Keller and Ludwig, 2003), to some extent, realises negotiation of a contract’s content but not format. WSLA does this by putting pre-specified, fixed information and negotiable elements as well as their choices into a WSLA template, which can act as a starting document for two parties to conduct partial negotiation. WS-agreement (Andrieux et al., 2004) supports templates based negotiation. WSOL/WSOI (Tosic, 2004; Patel, 2003) proposes the concept of ‘class of service’, modelled as a SO in WSOL. An SO is used to describe a discrete variation of the complete service and its QoS supported by one web service. All of SOs of one web service share the same WSDL description while differ in constraints and management statements. Thus service consumers can choose one SO from several options. WSOL/WSOI also supports creation of new SOs as simple variations of existing service offerings, based on WSOL reusability elements and attributes, and on leveraging the modular architecture of WSOI. In WS-QoS, the comparison and selection of one web service are special kinds of choices between contract options.

As seen in Table 3, none of the discussed architectures supports automatic negotiation of contracts. While WSLA realises a template-based contract negotiation with some degree of automation, fully automatic negotiation of contracts requires much more. We will discuss this topic in Section 4.2. As for contract verification, all of the discussed research support syntax is check by means of parsers or other tools.

Only Cremona (Ludwig et al., 2004) supports mapping service guarantees to the underlying resources. The service provider side agreement service role management (ASRM) will translate the guarantees of the agreement requested into specific resource requirements. Automatic deployment (configuration) of management systems is of great importance since dynamic web services deployment requires dynamic configuration of the web service management system. In WSLA (Keller and Ludwig, 2003), the deployment service is responsible for retrieving the configuration information from the contract (SLA) and distributing it either in full or in appropriate parts to the involved components, such as the measurement service or the condition evaluation service. For example, the configuration information sent from the deployment service to a measurement service contains the following

1 the information about the operation ParameterUpdate of the condition evaluation service that this measurement service will be corresponding with

2 the SLA parameters that this measurement service should send out to the condition evaluation service

3 the metrics that it needs to calculate for the parameters or it can get from other services specified by measurement directives.

Table 3

<table>
<thead>
<tr>
<th></th>
<th>Contract negotiation</th>
<th>Automatic negotiation of contracts</th>
<th>Automatic deployment according to contract</th>
<th>Contract adaptation</th>
<th>Contract termination</th>
<th>Contract deactivation/reactivation</th>
<th>AA</th>
<th>BB</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSLA</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>Yes</td>
<td>0</td>
<td>1,3,4</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>WS-agreement/Cremona</td>
<td>3</td>
<td>0</td>
<td>4</td>
<td>Yes</td>
<td>0</td>
<td>1</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>WSOL/WSOI</td>
<td>2,1</td>
<td>0</td>
<td>1</td>
<td>Yes</td>
<td>1</td>
<td>1,2,3,5</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>WSMN/WSML</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Yes</td>
<td>0</td>
<td>1,3</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>WS-QoS</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>Yes</td>
<td>0</td>
<td>1,3</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Note: AA means substitutability of contract languages, BB means mapping between QoS guarantees in contracts and resources.
In WSOL/WSOI (Tosic, 2004), from WSOL files, the WSOL compiler under development will automatically generate WSOL-specific handlers and configuration files in a format called web service offering descriptor (WSOD). Note in this case, the configuration is realised by creating the code and the configuration files and distributing them to appropriate locations in Axis process chain. In WSMN/WSML (Machiraju et al., 2002), a certain degree of automatic deployment and monitoring is realised by components in the SLA engine, e.g., SLA customiser, SLA evaluator and event manager. While there is no explicit contract and no automatic configuration of monitoring systems in WS-QoS (Gramm et al., 2003), it does make an automatic configuration of the underlying networks following the specification in consumers’ network QoS requirements. In Cremona (Ludwig et al., 2004), the management system can derive a specification called agreement implementation plan (AIP) from the agreement. These AIPs can be interpreted by the provisioning system to automatically configure the web service management system and the service itself.

Contract adaptation is the ability to change the contract on the fly. WSOL/WSOI (Tosic, 2004; Patel, 2003) is the only architecture that explicitly supports automatic switching to another contract (SO) at runtime if there are multiple SOs for the web service. Two kinds of switching are introduced in Tosic (2004): consumer-initiated and provider-initiated. Consumers, providers and third parties exchange messages and cooperate to perform the switching.

Contract deactivation/reactivation is another dynamic adaptation mechanism that only WSOL/WSOI (Tosic, 2004) suggests and supports. First, an attribute named ‘autoManipulation’ (Patel, 2003) is introduced in a WSOL SO specification. It indicates whether the provider needs a confirmation from the consumer before it deactivates the SO the consumer is using; Second, all affected consumers are switched to an appropriate replacement SO using provider-initiated switching. The old SO can be stored as ‘preferred’ for automated switching after its reactivation.

Contract termination is not as simple as it is often perceived to be. A common termination method is to specify an expiration time for the contract. WSLA (Keller and Ludwig, 2003), WSOL/WSOI (Tosic, 2004), WSMN/WSML (Sahai et al., 2002; Machiraju et al., 2002) and WS-QoS (Gramm et al., 2003) support this method. WSLA (Keller and Ludwig, 2003) and WSOL/WSOI (Tosic, 2004) also support contract termination when a specific event occurs. In addition, WSOL/WSOL (Tosic, 2004) supports the termination of a contract by the end of a session or by switching to another SO. Contract termination method in Cremona is not concrete since WS-agreement does not specify the contract language.

4 Discussion

In Section 3, we compared and analysed, based on the C-MAPE model, the different aspects of the representative projects. We came to a conclusion that current research on web service QoS management is strong in automatic contract deployment, contract measurement (monitoring) and contract termination. There is rather weak research and progress on contract negotiation, contract adaptation, contract verification, contract deactivation/reactivation and planning of corrective actions, while few contributions can be seen on executing corrective management actions (notification can be regarded as one solution but it is not enough). In addition, support for the integrated MOWS, applications servers and underlying networks, composite web service QoS management as well as web service compositions are still not fully addressed. In this section, we will discuss these topics and indicate the possible directions for future work.

4.1 How to realise plan and execute

What makes planning and especially executing difficult is that there are no management interfaces between web service management systems and the service providers’ provisioning systems. So when a SLO is found to be violated, the management system (e.g., condition evaluation service in WSLA) cannot interact with the service provisioning system through management interfaces to adjust the resources allocation at runtime. Recently, there has been some progress in this area. In Cremona, a service provider can offer services by agreements, based on current state of resource usage, and manage resources at runtime according to the agreements. In other words, Cremona associates agreements (an abstract model of contracts) with providers’ underlying resources.

Another possible solution for this problem could be web services distributed management (WSDM) (Bullard et al., 2005) and the corresponding management using web services (MUWS) (Vambenepe, 2005) and MOWS (Sedukhin, 2005). Basically, MUWS uses web services techniques to construct a management framework for management of IT resources. MUWS can provide a vendor-neutral and platform-independent management infrastructure. MOWS is regarded as an application and extension of MUWS. From the perspective of MOWS, web service management systems can be seen as ‘manageability consumer’ while the managed web services can be seen as ‘manageable resources’. Thus, the web service management systems could invoke operations to retrieve information or even change certain management information and affect the state of the managed web services, as long as the corresponding managed web services can support these operations through web service interfaces.
4.2 Future work on contract related issues

Contract negotiation has been the subject of research in e-commerce and B2B for several years. However, automatic contract negotiation remains a challenge. Template-based, decision support system (DSS) based, group support systems (GSS) based and autonomous agent system based contract negotiations are the primary research topics on this area. Contract negotiation can be conducted by combining approaches that integrate multiple technologies mentioned above. Automatic contract negotiation needs further support from learning and adaptation technologies such as artificial intelligence (AI). Recently, a number of research projects related to multi-attribute (multi-dimension) negotiation have been conducted. Basically these research projects are focused on constructing a utility (evaluation) function, which measures the goodness of a service by calculating an expression that comprises all the attributes for consideration. For example, Faratin (2000) propose a multi-attribute representation and evaluation model based on an evaluation function, which is defined as weighted sums of score function values. de Paula et al. (2001), Kowalczyk and Bui (2001) and Mikhailov and Tsvetinov (2004) also made contributions to this area by either extending Faratin’s model, or introducing constraint satisfaction problem into negotiation, or using a fuzzy logic modification of the analytic hierarchy process (AHP) as an evaluation tool for tackling the uncertainty and imprecision of service evaluations. SLA negotiation, which is a kind of multi-attribute negotiation, has the following specific factors that require further study:

1. There are many different attributes, for example, response time, throughput, availability, etc.
2. Attributes are mutually dependent. Sometimes there is conflict between two attributes.
3. For a service provider, there are limited resources to be allocated for enforcing the SLAs. That is, any new SLA negotiation has to take into account other SLAs that have been signed by the same provider.

Contract verification, as shown in Table 3, is also in its infancy. All research can only support syntax checks of the contracts. Semantic checks like conflict checks have not yet been addressed. We note that verification of contract guarantees and underlying resources capacity is a similar problem (discussed in Subsection 4.1), and Cremona has made some progress in this area.

Dynamic contract adaptation is also an important subject. Dynamic and arbitrary modification of current contract on the fly looks very attractive and will bring great flexibility in contract management. However, there are issues related to the used business models and technical capabilities for adapting provisioning infrastructures.

4.3 Integrated management

Consider the example of a web service whose requests are supposed to have a response time of less than one second has to wait for more than 60 seconds. There can be multiple causes for such behaviour, such as:

a. the web service is not running very well
b. the container application server has some problems
c. the response message had a prolonged stay in the transportation network because of congestion.

Consequently, the QoS management system has to be an integrated one to pinpoint the exact problem. That is, everything that is on the service path must be managed (at least monitored if not controlled). This is not a trivial task since underlying networks, operating systems and application servers are increasingly heterogeneous and are not always fully prepared to support QoS (Jin and Nahrstedt, 2004).

Among the research work we examined, only WS-QoS (Tian et al., 2004a; Gramm et al., 2003) addressed this topic. WS-QoS supports integrated web service QoS management by defining network layer QoS parameters in a QoS schema and runtime mapping of those parameters onto the QoS-aware network configuration. A component called QoS proxy is responsible for this kind of mapping. Note there are one assumption and one simplification:

1. the assumption is that the underlying transport network must support network QoS, such as DifferServ, ATM, etc.
2. the simplification is that, to deal with various underlying QoS mechanisms, QoS parameters in schema are defined as priorities, rather than absolute values.

A client side QoS proxy is placed between the web service client and network interfaces, while a server side QoS proxy is put between the network interface and the web service. The requirements for network level QoS are put into the SOAP header, which is parsed and mapped onto the current QoS-aware network by the QoS proxy on both the server and client sides. The limitation of this approach is that internet protocols currently used for web services do not support QoS.

Another interesting research is presented by Wang et al. (2004). Based on concepts from QoS management frameworks defined by ISO, W3C and OMG, an integrated QoS management architecture is presented for the publish/subscribe pattern of enterprise service oriented architecture. This work essentially follows the C-MAPE model, with contracts negotiated by clients and a set of management components, with resources reserved and initialised by resource management components and monitors set up to get QoS parameters. The main point of this integrated QoS management architecture is that there are central QoS management and resource management components that have full control of all the available resources within an enterprise, e.g., memory, CPU, network. Note although that this work is not exactly under web service environment but rather under service oriented architecture, it illustrates an outline for intranet-based
integrated service QoS management. The limitation of this work is that it is not clear how it could scale to internet-wide B2B compositions of web services where the participating businesses are not willing to allow full quality of class control (e.g., resource allocation) by third parties.

4.4 Composite web service management support

Management of composite web services is challenging. For the management of QoS properties, e.g., response time, an immediate question would be how the individual response time of one element service contributes to the response time of the composite service as a whole.

When business considerations are brought into the QoS management, the situation will become more complicated. Fox example, web service provider ‘S’ would prefer not to let consumer ‘C’ know the details of the service composition (business consideration), i.e., making the composite service behave as one single web service from the consumer’s point of view. Now, ‘S’ is responsible for the QoS guarantee violations caused by all the element services it uses.

The C-MAPE model still works in this scenario, but with new challenges. We will discuss contract-related issues first. For example, during contract negotiation phase, how can web service ‘S’ find appropriate element services to take part in the composite service and satisfy QoS requirements of consumer ‘C’? An interesting idea was introduced in Ludwig (2003): The consumer can invoke synchronously many web services with the same function and then only pay the best one (here ‘best’ could mean fastest, cheapest, etc.). Another important consideration in contract negotiation phase is whether there should be one global contract for a web service composition or a group of contracts each one only between two communicating element web services. Obviously, signing a contract between every two communicating web services will be a simpler way, while an all-in-one contract could probably be optimised to get better efficiency or benefit. The disadvantage of the latter one is there is no counterpart for BPFLAWS (Andrews et al., 2003) in web service management area. This implies that the management of the composite web service is difficult to optimise.

As for contract verification, the semantic check has to guarantee not only the consistency of statements in one contract, but the consistency between different contracts, e.g., is the sum of the processing time equal to or less than the consumer’s response time requirement? Contract adaptation is very important for composite web services. Research topics include, for example, how to automatically change QoS contracts or participating web services in order to maximise the QoS of the whole composition web service. Even contract termination is much more complicated. For example, if every contract is terminated based on the time period specified in contracts, the synchronisation of the clock of all involved parties will be a key task.

As for the MAPE model, the new challenges come from the plan and execute phases. In brief, if the element web services of a composite web service can share the measurement data with each other, there may exist optimised corrective actions in plan and execute phases for the whole composite web service. The management system could find these optimised solutions by correlating and analysing all the measurement data based on the topology of the composite web service.

5 Conclusions

QoS MOWS is becoming increasingly important as various web services are developed and deployed within an organisation or across organisational boundaries. There have been many research projects on this topic. Among them, WSLA (Keller and Ludwig, 2003), WSOL/WSOI (Tosic, 2004; Patel, 2003), WSMN/WSML (Sahai et al., 2002; Machiraju et al., 2002), WS-QoS (Tian et al., 2004a; Gramm et al., 2003) and the newly emerging WS-agreement/Cremona (Ludwig et al., 2004; Andrieux et al., 2004) are of great importance and large influence.

We proposed a general model, C-MAPE, for reviewing and analysing representative research work. The purpose is to examine what they have done and what they have not done on contract related and MAPE issues. The results show clearly that current representative projects strongly support formal contract definition, automatic contract deployment, contract measurement (monitoring), contract evaluation (analysing) and contract termination. Further, these projects provide limited support for contract negotiation, contract adaptation, contract verification, contract deactivation/reactivation and planning of corrective actions. However, they make few contributions to executing corrective actions. In addition, we discussed and emphasised that the association between contracts and service provisioning systems is the key point for realising planning and executing management actions. We also analysed possible ways to achieve integrated QoS management, for web services and their compositions. Finally, we outlined the need and possible directions for future research on contract related issues, such as automatic contract negotiation, semantic contract verification, and dynamic contract adaptation.

With the definition of the C-MAPE model and the corresponding analysis of current research, we contributed to the understanding of automatic processing of formal contracts, as well as of the relationship between contracts, QoS management systems, and web service provisioning systems. Since QoS management is often business-critical for web services, the importance of this area will grow in the future.

Acknowledgements

This work is supported by National Natural Science Foundation of China (NNSFC) under Grant No. 60403029.
References


