ON-LINE CONTROL OF SERVICE LEVEL AGREEMENTS

Manoel Camillo Penna and Rafael Romualdo Wandresen
Pontifical Catholic University of Parana, PPGIA, Rua Imaculada Conceicao, 1155 80215-901 Curitiba PR Brazil {penna, wandresen}@ppgia.pucpr.br

Abstract: Service Level Agreement (SLA) is used as the corner stone for building service quality management (SQM) systems. SLA and the processes associated with them, establish a two-way accountability for service, which is negotiated and mutually agreed upon, by customer and service provider. It defines a set of service level indicators and their corresponding Service Level Objectives (SLO), which defines a threshold for the indicator value. Service quality assessment can be accomplished in two modes, on-line and offline. Off-line service level evaluation is performed only at the end of the period of service delivery, whereas on-line service evaluation supports continuous supervision of service quality. This paper presents a method for on-line control of SLA that evaluates indicator value each time an event that changes its value occurs. The method also computes the deadline to reach the corresponding SLO, what is important for pro-active control.

Key words: Service Quality Management, Service Level Agreement

1. INTRODUCTION

In recent years there were many research efforts on service quality management (SQM). Many authors explored service management under infrastructure viewpoint, for example, quality of service in active networks[1,2] and IP networks[3,4,5,6]. Many experiments were also target to the construction of generic SQM platforms[7,8,9]. From those, we borrow some of the architectural concepts presented in section 2 where we introduce a three layer functional architecture. This architecture is presented in order to provide a framework for conceptual understanding of involved concepts. Particularly we simplify the management layer and just consider inference and control functions.

Inference and control functions can be implemented in two modes, on-line and off-line (see Figure 1). In on-line mode, the indicators are evaluated in the very moment an event that changes an indicator value arrives to the system. In off-line mode, service level indicators are evaluated at pre-defined periods, for example, daily, weekly or monthly. Both modes are necessary: on-line mode takes only into account data available on arriving of incoming events, in contrast with off-line mode, which takes into account all information collect during the assessment period. The last can consider data that is not available when events arrive to SQM system, for example events related to faults that can not be imputed to service provider, allowing more precision on indicator calculus. See Lewis[10] and Wandresen[11] for a more detailed discussion on two modes.

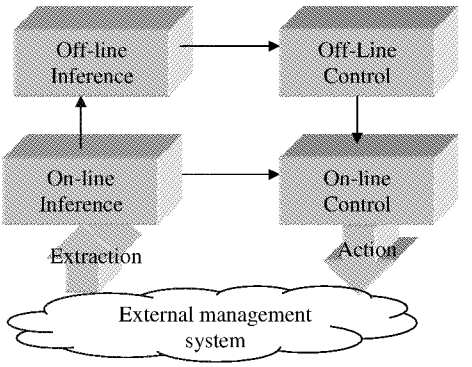

Figure 1. Inference and control functions for service quality management

The main concern of this work is with the questions related to on-line inference and control functions in SQM systems. In section 2 we present a functional architecture for SQM systems. The goal is to provide a conceptual basis for sections 3, where we discuss a method and depicts an algorithm used for on-line inference and control. Finally in section 4 we present some conclusions and future work.

2. SQM FUNCTIONAL ARCHITECTURE

In this section we present a functional architecture for SQM systems, which is organized on three logical layers: data collection layer, inference and control layer and presentation layer. At the first layer, extraction and mediation functions interact with external management systems to obtain data for SQM. Typically, extraction function gets data to calculate indicators and mediation function gets information to populate SQM database.

Extraction functions get external data and interact with SQM database to obtain the necessary information to construct service fault events (SF events) as described in section 3. Inference algorithms collect service fault events, calculate new indicators' values and deadlines, and deliver them by means of service quality events (SQ event), which are used at presentation layer to construct service level reports and supervision panels. The whole architecture is illustrated on Figure 2.

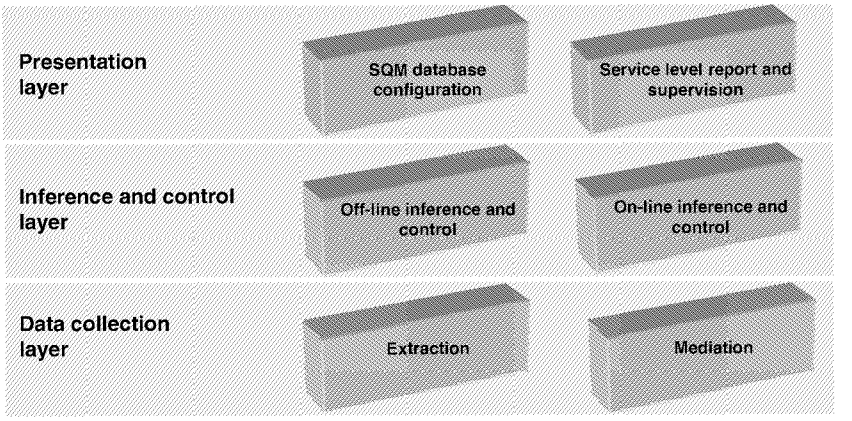

Figure 2. SQM functional architecture

2.1 Presentation layer

Presentation layer includes graphical user interface for SQM database configuration and result presentation functions. SQM database configuration includes service, customer and SLA registration, whereas service level reports and supervision panels present results.

2.1.1 SQM database configuration

SQM system collects and handles a large set of information to achieve its goal, and organize them in five repositories: service inventory, contract repository, policy repository, supervision repository and indicator repository. Service inventory contains the service list and respective attributes. It must be flexible allowing modeling of service information according to business needs. Service inventory also stores references to components of the infrastructure that supports the service (service elements). This is necessary when algorithms that compute indicator values rely on information originated at service elements, as their availability or other technical parameters.

Contract repository relates service instances to customers. In many cases this information already exists in other corporate information systems and should be obtained from them by integration. Mediation function is the architectural component for integration. Contract repository also stores SLA information, which relates service instances to a set of indicators and thresholds. SLA provides the basis for SQM by establishing a two-way accountability for service, which is negotiated and mutually agreed upon by customer and service provider. An SLA is determined by a set of service level indicators and their corresponding thresholds. A threshold defines an edge value for the indicator that is meaningful for SQM purposes. Several thresholds can be associated with one indicator, but for simplicity, we will consider only two in this paper: (i) service level objective (SLO), which is the value against with the indicator, will be matched at the SLA assessment time; (ii) alert threshold, which points, when reached, a risk of offending the agreement.

Policy repository stores the rules responsible for automatic control of service level objectives and supervision repository stores information necessary for monitoring purpose. Inference algorithms compute indicators' values and store them at indicator repository.

2.1.2 Service level report and supervision

Effective service level reporting is the medium of communication that demonstrates the value of service and can serve as an excellent management tool. Reporting can be broadly divided into tow categories: service level reporting and service supervision.

A service level report presents the performance obtained during service deployment within a pre-defined period. It presents, in a structured way, the measured indicators' values and compares them with the quality thresholds established in the agreements. They show the values stored in the indicator repository by off-line inference functions. When the service quality goals are not attained, the service level report includes the failure causes and shows the corrective actions that had been taken. However, quality goals may not be attained due to circumstances out of service provider control. In this case they should not be considered by service level evaluation algorithms, that is, service fault events which can not be imputed to the service provider must be excluded from calculus.

Service supervision is accomplished by presentation of two panels: indicator and alarm panel. Indicator panel presents all managed indicators in an organized way, by clients and service. It allows a managerial valuation of the service offer through the identification of those who have offended SLA or with offense risk. Alarms panel presents in a framed way the services

alarms. They present events that cause modifications on service state and show deadlines for service degradation. As discussed before, an SLA defines several indicators for one service. The service state is defined as being the worst state among them. Service state evaluation depends on SLM events produced by on-line inference algorithm, and keeps the worst indicator state as the current service state.

In this paper service supervision uses two thresholds for each indicator, alert threshold and service level objective (SLO). They determine three states for each indicator: normal state means indicator current value is better than alert threshold; warning state means its value is between alert threshold and SLO; and violated state means that the value is worst than SLO.

2.2 Inference and control layer

Indicator evaluation and automatic triggering of management actions happen at the inference and control layer. Inference functions compute indicators' values; compare them with service level thresholds and fires control actions in order to pursue management objectives. Typical control actions are updating supervision panels, sending alert messages and starting management interactions with external systems. Data for indicator evaluation is obtained from external management systems by extraction function at the data collect layer.

2.2.1 Off-line inference and control

Off-line inference performs indicator evaluation by means of a scheduling mechanism that drives periodically corresponding computing algorithms. It reads service fault events prepared by extraction function and stores the calculated values in indicator repository. Computed values must remain stored, at least, up to the end of the assessment cycle. For example, all service fault events occurred last month must remain stored until this monthly assessment is performed, that is, service level reports are delivery and accept by customers. This is necessary because some events can be considered non-pertinent at assessment time even if they were considered at collection time.

Off-line control actions are fired through rules evaluation stored in policy repository. A rule contains a condition and an action. Conditions are a logical expressions made by variables (indicators) and by logical and arithmetical operators. When the condition is evaluated to TRUE, the corresponding action is fired. Off-line control actions unlink computing procedures, for example, reconfiguration of the network that supports service delivery or a routine for penalties and bonus account if the agreement is violated. Off-line

control actions to update the indicator panel are also fired by changes in service state.

2.2.2 On-line inference and control

On-line inference performs indicator evaluation for alarm and alert generation. It is based on an event's handling mechanism: for each incoming event, indicator's value is updated, considering the information existing in service fault event. Also, a new deadline for indicator state change is computed. An SQ event is raised for each state change.

2.3 Data collection layer

Data collection layer gets the external systems data and can be divided into tow categories: extraction and mediation. Extraction functions get data for indicator account whereas mediation functions get information to fill service and SLA repositories.

2.3.1 Extraction function

Extraction functions interact with outside management systems to collect information used for indicator account. Extraction algorithms depend on data to be collected, including its origin, shape, access mode and previous treatment that it should receive. Data can be stored in log files, databases, spreadsheets or general files. It is also possible that data is not available at collection time, and a graphical interface should be provided.

As data origin can be multiple and heterogeneous, extraction function must perform the following tasks: convert multiple unities from origin data to a unified unity; synchronize data production periodicity making them available at appropriate frequency to the off-line account algorithms; summarize collected data and exclude the ones that are not necessary, ensuring that those data will get to the inference and control layer, according to their needs; and build, send and store SF events to account algorithms.

2.3.2 Mediation function

Mediation function interacts with outside systems to collect and store information at the SQM system repositories. They import customer, service, and SLA data to SQM databases when necessary.

3. ON-LINE CONTROL OF SLA

3.1 Method for indicator and deadline evaluation

This section presents a method and for on-line control of service level agreements. They can be applied to SLA established on indicators whose values depend on service fault events (SF events), which indicate the beginning and the end of a service unavailable interval. It also outlines the algorithm to implement the method.

Extraction functions build SF events by handling data reported by external management system. When building SF events, extraction functions relate infrastructure alarms with a customer-service pair. Moreover they assure events are delivered according to the following rules: don't send duplicated events; don't send an event with UP notification type before the corresponding event with DOWN notification type; provide a growing identification for events related to the same customer-service pair.

The method is based in mathematical functions that describe indicator's behavior based on occurrence of events throughout time. Figure 3 illustrates the idea: the horizontal axis represents SF events occurrences (e1 and e2) on time (t0 and t3), and the vertical axis represents the indicator value. It has a starting value (Q0) and two known thresholds: alert threshold (Q1) and service level objective (Q2). The occurrence of e1 event (at t0) characterizes the beginning of an unavailability interval for the service, and affects the indicator value according to its formula. The occurrence of e2 event characterizes the end of the unavailability interval, from when the indicator will have its value unchanged until the beginning of the next unavailability interval. With this information it is possible to compute t1 and t2, which are the moments when indicator value will reach thresholds Q1 and Q2, respectively.

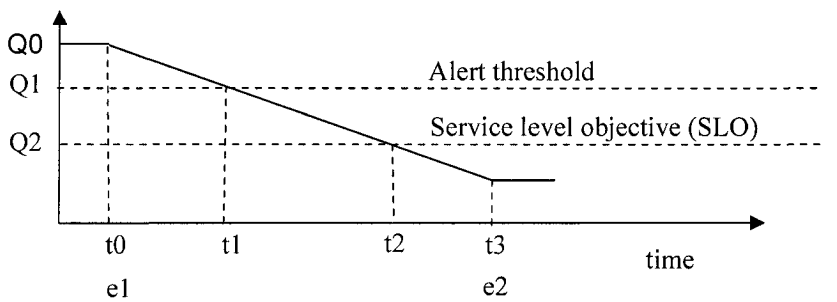

Figure 3. Indicator value and deadline to reach thresholds

3.2 Information flow

The structure of SF event is shown in figure 4. Event identifier is a grown number that uniquely identifies the event for a customer-service pair. Notification type says if the event corresponds to the beginning or the end of an unavailability interval, carrying respectively DOWN or UP value. The same event identifier is used in two related DOWN and UP notifications. The event also carries service and customer identifiers and a timestamp informs the time and date of event occurrence.

event identifier	value
notification type	value
service	value
customer	value
timestamp	value

Figure 4. Service fault event

SQ events are sent when service quality condition is changed (see figure 5). Event identifier is a grown number that uniquely identifies the event for a customer-service pair. The same identifier is used for all SQ events generated when handling SF events related to a DOWN-UP pair, that is, having the same identifier. The SQ event also carries service and customer identifiers, informs what indicator is being reported, the indicator value and state, the next threshold to be reached (Q1 or Q2), and the deadline to reach it.

event identifier	value
service	value
customer	value
indicator	value
value	value
state	value
next threshold	value
deadline	value

Figure 5. Service quality event

Some intermediary data is stored in a control register, which structure is presented in figure 6. It contains the event identifier, service and customer identifiers, a timestamp for last DOWN event occurred for this service-customer pair (DT), the number of SF events for this service-customer pair (NSF), and the previous value calculated for each indicator.

event identifier	value
service	value
customer	value
DT	value
NSF	value
previous value [i]	value

Figure 6. Control register

3.3 Indicators and formulas

To demonstrate the method we will consider mean time to restore service (MTRS) indicator, which is one of the most important and used service level indicators. Initially we present its definition (equations 1) and graphic (figures 7); then we deduce the formula that calculate its current value (equations 2) and the formula that calculate the deadline to reach the corresponding threshold (equations 3). Mean time to restore service (MTRS) is the mean of all unavailability intervals (TRS) observed during the evaluation period.

$$MTRS = \frac{\sum_{n} TRS_n}{n} \tag{1}$$

The value of MTRS indicator depends on the occurrence of SF events. A SF event with DOWN notification type starts an unavailability interval when MTRS value starts increasing as shown in figure 10. SF events *e1* and e3 have notification type DOWN, and start two unavailability periods, whereas *e2* and *e4*, with UP notification type, end the corresponding periods. *PMTRS* is the indicator value at the end of the previous unavailability period and *dl_1* is the deadline when indicator value will reach the alert threshold (*Q1*).

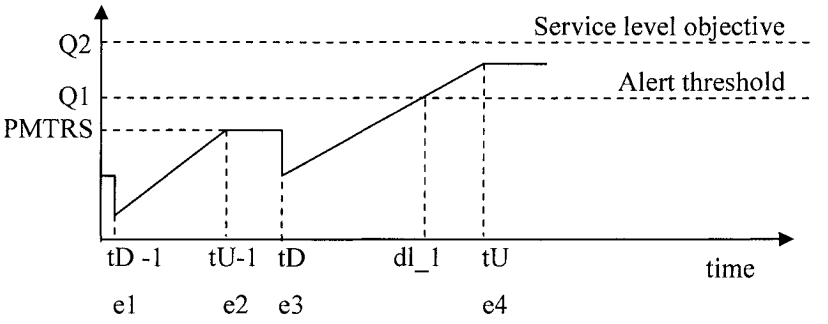

Figure 7. MTRS indicator value and deadline to reach thresholds

Equation 2 calculates MTRS value at the end of an unavailability interval (*tU*), started at *tD*. The value is calculated from MTRS previous value (PMTRS) at the end of the last unavailability interval.

$$MTRS_{(n)} = \frac{\sum_{n} TRS_{(n)}}{n} \quad \text{or} \quad \sum_{n} TRS_{(n)} = MTRS_{(n)} \times n$$

Considering event (n+1):

$$MTRS_{(n+1)} = \frac{\sum_{n} TRS_{(n)} + TRS_{(n+1)}}{n+1}$$

Substituting $\sum_{n} TRS_{(n)}$ by $MTRS_{(n)} \times n$

$$MTRS_{(n+1)} = \frac{MTRS_{(n)} \times n + TRS_{(n+1)}}{n+1} \quad \text{or}$$

$$MTRS = \frac{PMTRS \times n + (tU - tD)}{n+1} \qquad (2)$$

Equation 3 computes the deadline to reach threshold *Qi* at the beginning of an unavailability interval, that is, at the reception of a SF event with DOWN notification type and timestamp *tD*. The deadline to reach *Qi* is deduced from (2) by substituting SA by *Qi* and *tU* by *dl_i*.

$$dl\_i = tD + (n+1) \times Qi - n \times PMTRS \tag{3}$$

3.4 Algorithm

The algorithm to implement the method is quite simple, and is outlined bellow. It is, however, necessary to implement a very sophisticated mechanism to handle incoming and outgoing events. A mechanism, named event dispatcher, receives and processes incoming SF events, then builds and sends corresponding SQ events. It calls an evaluation component to calculate the values according to equations 2 and 3.

Event dispatcher reads incoming SF events and verifies its notification type. For SF events with DOWN notification type it calls the evaluation component to compute the deadline to reach next threshold by applying equation 3. Then it prepares an SQ event and sets a timer that expires at computed deadline. When deadline expires, the associated SQ event is sent to be presented at service alarm panel. When the notification type is UP, the corresponding timer is unset. A new SQ event is built to clear previous alarm condition, and also to inform new indicator's value, computed according to equation 2.

4. CONCLUSION AND FUTURE WORK

This work presented a method for on-line service level management. It is the base for an algorithm that computes indicator's value for each incoming event and also, that computes the deadline for reaching the next service level threshold. The information rendered on-line by the algorithm in SQ events is important because it allows pro-active management. Operational management actions, as reconfiguring the service, can be started based on it. An SQM system with the outlined architecture was fully implemented and is in use in two telecommunication operators in Brazil. The algorithm has been implemented and a benchmark against off-line methods is available in Wandresen[11].

The service level indicators considered in this paper are all based on SF events, which are related intervals where service is unavailable. Such intervals are characterized by two timestamps, one at the beginning and the other at the end of the unavailability period. The extension of the algorithm for other kind of service level indicators can be considered. The only output of the algorithm is SQ events. Another important extension is to consider control actions based on policies.

REFERENCES

1. R. Boutaba; A. Polyrakis. Projecting Advanced Enterprise and Service Management to Active Networks. IEEE Network – February 2002.
2. M. Brunner; B. Platner; R. Stadler. Service Creation and Management in Active Networks. Communication of the ACM – April 2001.
3. G. Cortese; P Cremonese; A. Diaconescu; S. D'Antonio; M. Espósito; R. Fiutem; S. Romano. Cadenus: Creation and Deployment of End-User Services in Premium IP Networks. IEEE Communications Magazine, January 2003 – Vol. 41 n.1.
4. R. Maresca; M. D'Arienzo; M. Espósito; S. Romano; G.Ventre. An Active Network Approach to Virtual Private Networks. In Proceedings of ISC2002, July 2002.
5. A. Kittel; R. Bhatnagar; K. Hum; M. Weintraub. Service Creation and Service Management for Advanced IP Network and Services – An Experience Paper. In Proceedings of IM2001
6. Dinesh Verma. Supporting Service Level Agreements on IP Networks. New Ridres, Indianapolis, 1999.
7. A. Keller; G. Kar; H. Ludwig; A. Dan; J. Hellerstein. Managing Dynamic Services: A Contract Based Approach to a Conceptual Architecture. Journal of Network Operations and Management, 2002
8. A. Keller; H. Ludwig. The WSLA Framework: Specifying and Monitoring Service Level Agreement for Webservices. Journal of Network and Systems Management, 11(1), 2003.
9. P. Trimintzios; I. Andrikopoulos; D. Goderis; Y. T'Joens. An Architectural Framework for Providing QoS in IP Differentiated Services Networks. In Proceedings of IM2001.
10. L. Lewis. Service Level Management for Enterprise Networks. Artech House, London, 1999.
11. R. Wandresen. Proposta de um Modelo Computacional Baseado em Eventos para Gerência de Níveis de Serviço em Telecomunicações. Dissertação de Mestrado, PPGIA. PUCPR, 2003.