Definition and Evaluation of Penalty Functions in SLA Management Framework

Jacek Kosiński, Dominik Radziszowski,
Krzysztof Zieliński, Sławomir Zieliński

AGH University of Science and Technology
{jgk, radzisz, slawek, kz} @ics.agh.edu.pl

Grzegorz Przybylski,
Paweł Niedziela

Comarch S.A
grzegorz.przybylski@comarch.pl
pawel.niedziela@comarch.com

Abstract

The paper presents an analysis of the possibilities of defining penalty functions to be used in SLA management frameworks. As a basis for the analysis, a brief survey of metrics used in SLA contracts is presented. Next, several definitions of penalty functions are proposed. The definitions are then analyzed in context of embedding them in a working SLA management framework.

1. Introduction

The Service Level Agreements (SLAs)[1] are gaining popularity between telecom service providers and their customers. The SLAs define a basis for guaranteed performance and availability when using services provided by an ISP and thus connect the technical and market perspectives of companies operation [2,3]. The purpose of service level agreement is to formally define the parameters of service the provider guarantees to deliver. After agreeing upon an SLA, the parameters are monitored in order to detect breaches. Commitments a service provider makes in SLAs can be a true service differentiator; however, identifying violations and collecting penalties might prove quite challenging.

Although the evaluation of an SLA may take place at the end of accounting period (which is fine from the service client's point of view), it is expected to be performed more frequently in order to generate alarms for the service provider. Such alarms may indicate possible SLA breaches before they occur. Using the information the ISP can at least minimize losses based on the possible penalties that are assigned to each contract.

The goal of this paper is to analyze the SLAs in terms of penalty functions in order to identify the implementation requirements that result from the complexity of SLA contracts. Section 2 provides a brief overview of the SLA objectives, while section 3 introduces a few possible definitions of penalty functions that reflect the structure of contracts.

As an example of a working system capable to provide estimation of penalties, the Comarch SLA Management Framework is discussed. As shown in section 4, the structure of the framework enables not only for calculating penalties at the end of accounting period, but also to monitor some of the key parameters (that may reflect the penalty) constantly.

2. SLA metrics survey

The metrics ISPs use for service level agreements depend upon the ISP specific situation. In general, the SLA metrics can be categorized as either availability or performance-related [8]. Availability measurement is usually straightforward – it is easy to monitor the service's presence. From the customer perspective, the service availability is meant as the possibility of using the service defined by SLA at the given moment. The availability treats the service as a whole, it does not distinguish the reasons of failures of disks, links or routers. The availability is expressed as a ratio of activity time to measurement time.

The performance-related metrics, although tightly coupled with availability, are much harder to calculate. Determining the actual performance is complicated because it requires either the provider or its client to continuously monitor key performance parameters. The most frequently used performance-related metrics include:

- service response time,
- data transfer rate,
- round-trip time,
- packet loss ratio,
- delay variance.

The metrics must be measured actively during the service operation time. However, active monitoring might introduce additional overhead. In order to reduce that, monitoring systems are appropriately adjusted. As a result, the frequency of data gathering may be too low and the delays of reports from particular devices may be too high to guarantee appropriate precision of measurements. It is not unusual for telecommunications company to have devices configured to report their states only once a day. Such factors are taken into account when designing the SLA templates for clients. The factors also influence the possibilities of evaluating composite metrics, and must be considered when choosing an appropriate penalty function, which can itself be understood as another composite metric.

Another common performance characteristic is volume. Volume may be reflected in the amount of data transferred, the number of transactions processed, users served and so forth. Measurement of volume is often simpler than of performance, therefore a violation of an SLA usually can be noticed just after the end of accounting period.

The metrics defined in an SLA must also reflect the user experience level. Therefore, with some specific exceptions, there is no place in an SLA for error rate, packet loss, percent of frames delivered, network latency, etc. From the user perspective, these parameters are difficult to determine basing only on common symptoms of service quality degradation. It is necessary while defining SLA to go only as deep to accurately reflect the user experience.

3. Predicates and definitions of penalty functions

Different Service Level Agreements are likely to define different kinds of parameter breaches and also different penalties. Therefore, a major task is to represent an agreement in a calculable way - each SLA should be presented as a set of predicates.

In order to monitor and process the SLAs, their parameters must be measurable, preferably in numbers, not descriptions. For numerical parameters, basic predicates based on simple algebraic relations can be defined such as following:

p1. param < threshold

p2. threshold < param

p3. value1 < param < value2 (conjunction of p1 and p2)

and so forth.

The simple predicates can be applied to different parameters and combined to form a logical sentence describing the whole SLA. Such a sentence is quite easy to be evaluated given a snapshot of all the resources. However, such approach disregards the time factor.

What matters in real life is not only if the logical value of a sentence is ‘true’ or ‘false’, but also ‘at what time it was true/false and for how long’. For the reasons mentioned in the previous section, the basic thing to be decided on is how to estimate the value of a parameter between its measurements. For parameters measured frequently, the last value would do. However, if a period between subsequent parameter measurements is high, more sophisticated estimation (inter- or extrapolation) should be considered.

Due to the above matters, the SLA would be presented rather as a logical sentence consisting of predicates such as:

pt1. (param < threshold, t_{start}=x, t_{end}=y)

pt2. (threshold < param, t_{start}=x, t_{end}=y)

and so forth, when t_{start} and t_{end} are the start and end of period, in which the predicate did not change its value. Theoretically, the period can be open, e.g. when the t_{end} is set to infinity, but in real life accounting periods are defined as days, weeks, months, etc. Note that the “param” rarely refers to raw monitoring data. Rather it will be e.g. “availability percentage”.

Another issue is how to represent predicates on parameters that do not have numerical values. They can be quite easily placed in a monitored SLA description as long as their values can be enumerated. Respective predicates would look as follows:

(param in [set of values], t_{min}=x, t_{max}=y)

The SLA can include many sub-agreements (regarding particular services or service parameters) with different policies defined in each of them and possible different penalty policies. A sub-agreement (subcontract) is understood as a part of SLA with a single, well-defined penalty policy.

Such assumptions lead to defining different penalty functions depending mainly on the importance and time constraints defined in a subcontract. The following subsections present appropriate formulas.

3.1. Penalty calculation.

This section attempts to provide an overview of typical formulas for calculating penalty functions in different conditions specified in an SLA. The set of formulas is based on interviews with ISPs which offer SLAs to their customers.

Per-breach charge.

Simplest SLAs can define a constant value not dependent either on the duration of contract breach or the moment it happened. The value of penalty is defined in conditions of SLA agreements. However, even most simple real-life SLAs take into consideration at least one of the factors. More complicated SLAs involve also other factors, such as the moment of time the breach happened, the importance of the broken subcontract and so on. As a result of SLA analysis, a few general formulas can be observed:

1. Charge dependent on number of SLA breaches. In this case the amount of penalty is related with the duration of conditions exceed. Two variants of penalty estimation can be distinguished.

$$P(x) = V(x) * n_x \tag{1}$$

x – belongs to the SLA contracts collection
n_x – number of exceeds of contract x
$V(x)$ – one-time amount of penalty for single exceed of conditions

2. Charge dependent on breach duration. Penalty value is calculated for whole contract (considering all sub-contracts, even those that were not exceeded). In this case the rate of penalty for certain period of time, e.g. the amount of N for each minute of failure is established in SLA definition.

$$P(x) = W(x) * t_x \tag{2}$$

t_x – duration of x contract failure
$W(x)$ – penalty unit price

The value of penalty can be calculated taking into consideration sub-contracts which were exceeded. The final value of penalty is in such case a sum of particle penalties for going back on particular sub-contracts.

$$P(x) = \sum_{i=1}^{k_x} W(x_i) * t_{x_i} \tag{3}$$

k_x – the number of exceeded sub-contracts within the given SLA.
$W(x_i)$ – penalty unit price for i subcontract
t_{x_i} – total time of failure of i service in x contract

3. Charge dependent on the time of occurrence. This case distinguishes the value of penalty in relation to day time when the contract breach occurred. From the client point of view, particular services can have different degree of importance depending on time (e.g. the availability of a web portal is more important at 12am than at 12pm). In case of disregarding subcontracts, appropriate formula looks as follows:

$$P(x) = \sum_{j=1, t_j \in T_x}^{\Delta T_x/\delta t_x} W(x) * I(x, t_j) * \delta t_x \tag{4}$$

δt_x – time between measurements x
t_j – time moment j
$W(x)$ – penalty unit price
$I(x, t_j)$ – the function of service importance
– returning the value between $< 0, 1 >$
T_x – the period of failure occurrence of contract x
ΔT_x – lenth of period of failure occurrence

When taking subcontracts into account, the formula changes to:

$$P(x) = \sum_{i=1}^{k_x} \sum_{j=1, t_j \in T_{x_i}}^{\Delta T_{x_i}/\delta t_i} W(x_i) * I(x_i, t_j) * \delta t_{x_i} \tag{5}$$

k_x – the number of exceeded sub-contracts within the given SLA.
δt_{x_i} – time between measurements x_i
t_j – time moment j
$W(x_i)$ – penalty unit price
$I(x_i, t_j)$ – the function of sub-service importance
T_{x_i} – period of failure occurrence of sub-service x_i
ΔT_{x_i} – lenth of period of failure occurrence of sub-service x_i

Finally the penalty can be calculated considering all combinations of cases from formulas (1), (2), (3), (4) and (5).

$$\begin{aligned} P(x) \;\; = \;\; & V(x) * n_x + W(x) * t_x + \sum_{i=1}^{k_x} W(x_i) * t_{x_i} + \\ & + \sum_{j=1, t_j \in T_x}^{\Delta T_x/\delta t_x} W(x) * I(x, t_j) * \delta t_x + \\ & + \sum_{i=1}^{k_x} \sum_{j=1, t_j \in T_{x_i}}^{\Delta T_{x_i}/\delta t_i} W(x_i) * I(x_i, t_j) * \delta t_{x_i} \end{aligned} \tag{6}$$

The presented formulas cover the most frequently used definitions of SLA contracts penalty functions. Because some of them define penalty in a very complex way, depending on many variables, in order to implement them efficiently, a sophisticated monitoring and processing system should be used. Note that the real value of SLA management system lies not only in evaluating the functions at the end of accounting period, but also in alarming the operator of possible (forthcoming) SLA breaches. The required granularity of raw data and the need for constant monitoring of the SLA contracts impose heavy constraints on the system design and implementation. The next section presents an overview of a system capable of providing the required functionality.

4. The SLA Management Framework

Effective implementation of SLA management framework requires specific infrastructure services to be accessible. This includes data sources together with specific adapters to collect and process the necessary data, graphical user interface, and the communication bus. The framework requires also to be deployed over real network architecture. This process needs access to specific inventory database to retrieve information about physical description, parameters and localization of each item (network device, interface, host) and how it is interconnected. Such inventory is provided by the Comarch InsightNet platform, the SLA management can be integrated with.

4.1. Data available from monitoring

Because the SLAM system has to process different types of information – the modular approach has been proposed as the most suitable for such implementation. Two types of information have been identified to be the most important. The first type it is the Fault Management information, either retrieved from existing Fault Management systems, or gathered directly from network elements. This information should be processed in the Service Monitoring module in the SLAM system. The second type of information it is performance related KPIs (Key Performance Indicators), and heavily time dependent. This information should be processed by the Performance Management System in which the Performance Management engine is the most important.

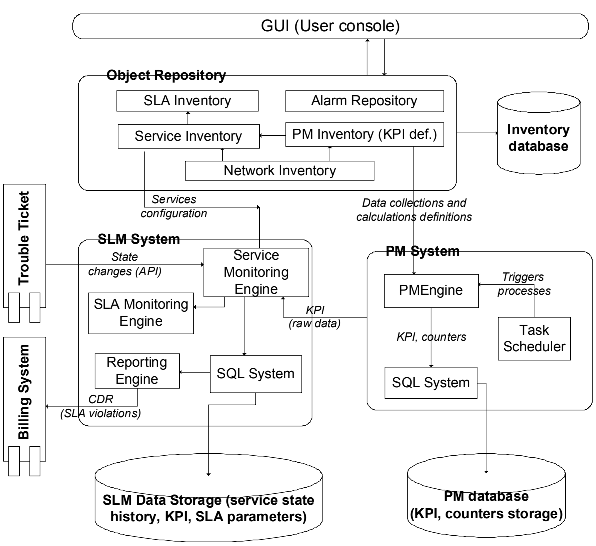

Figure 1. SLAM system architecture

Figure 1 describes the overall architecture schema of the SLAM framework. Modular approach allows, by the use of mediation agents, to collect data from different data sources such as: Trouble Ticketing system, Performance Management database or directly from the network elements. Such approach enables to use the SLAM framework, either as an umbrella system for other vendor specific, or client specific systems, or as a stand alone solution.

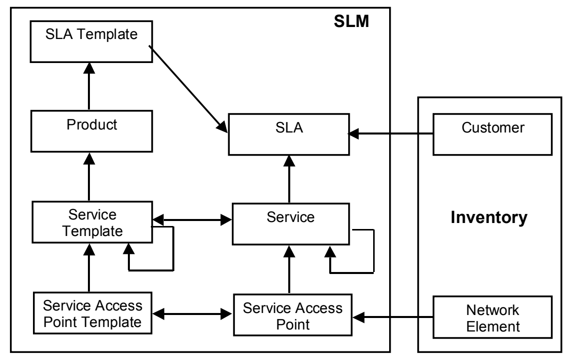

Figure 2. Logical architecture of the SLAM framework

The logical architecture of the system presented in Figure 2 is composed of different levels containing:

- Service Access Points (SAP) – related to network elements (physical devices or components of these devices),
- Services – composed of different service nodes or/and other services,
- Products – understood as groups of services,
- SLAs – linked directly to products.

Each SAP, Service, SLA or Product is defined as a template object first and then instantiated as a working instance (see Fig.3). SAP represents the relation between service and network infrastructure. SAP Template at the same time is a template object for SAP and in addition to it's attributes, SAP Template object consists of set of rules for easy assigning appropriate Network Element objects to SAP objects related to a given SAP Template.

Service Object is a single element of service-tree which is the representation of the real-life service structure and consists of the following attributes:

- current state of the element,
- event propagation formula for given element,
- KPI propagation formula for given element.

Each service object may have one of the two processing modes: Automatic - the state of the Service object is automatically calculated by the engine upon event propagation formula, Manual - the state of the Service is set by operator. Similarly to SAP Template - Service Template object is used to define service hierarchy and processing rules only. This object is not processed by SLAM Engine during Service and SLA monitoring process.Product object is used to group service tree-like structures. The template of the SLA contract can only be created in context of Product.

Finally the SLA Object representing SLA contract consists of the set of common SLA parameters (Availability, Max Time to Restore, Time to Violate) and SLA KPIs, and it is instantiated according SLA Template. Only one Product Object can be attached to the SLA Template.

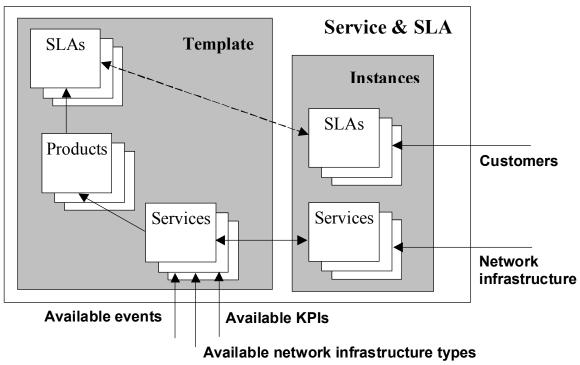

Figure 3. Instances and Templates in the SLAM framework

4.2 The process of evaluating SLAs

The "near real time" computation approach has been identified as the most suitable for updating parameters of the SLA objects. The required granularity of time is the parameter of the system. The performance issues and data delivery schedules determine the frequency of calculations of states for each of the objects.

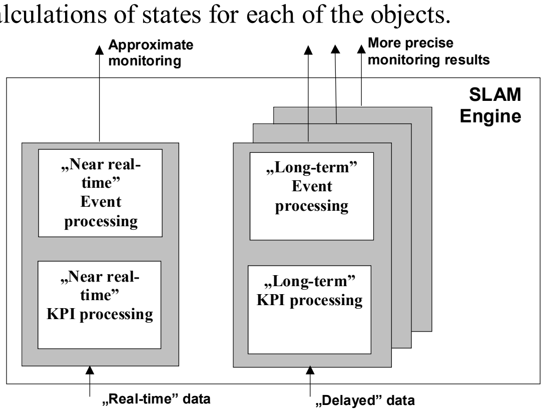

Figure 4. "Near real time" and "Long-term" information processing

The proposed concept is presented in Fig.4. The "near real time" mode processes data incoming from the network infrastructure (events, KPIs) in almost real-time. All parameters of services and SLAs are calculated continuously with configurable time period (depending on number of monitored services and SLAs and hardware specification the „real time" calculation period can vary between 1 minute to 60 minutes). The flexibility in defining the KPI propagation formulas allows the user of the system to calculate estimates of penalties along with other SLA parameters. However, because not all network devices provide required data in short periods of time, it is possible that results of „real time" processing can be inaccurate (for example KPIs can be retrieved form some devices once an hour or even rarely).

In order to process historical or outdated data coming from network infrastructure (events, KPIs) the „Long-term" processing mode has been implemented. It recalculates all parameters of services and SLAs taking care that all the information which was not available during „near real time" processing will be processed tin the "long-term" cycle.

„Long-term" processing helps to improve the results of evaluation of parameters of services and SLAs. It is possible to define more than one „long-term" processes running with different delays and in different schedules, but at least one „long-term" processing must be running to generate reliable data for SLAs after accounting period.

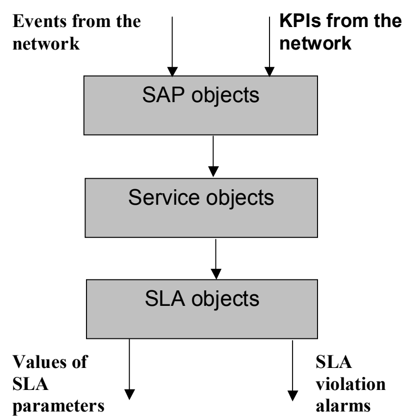

Figure 5. Events and KPIs processing

Events from the network are collected by the Fault Management system and processed through filters classifying them into three categories (CRITICAL, WARNING, CLEAR) – see Figure 5. They are used to define the state of the SAP which is then propagated upwards the service tree structure. For each service object the service propagation rules decide which events should be considered to put the service in a specific state. For SLA object the parameters are calculated and after checking for thresholds the values of SLA KPIs, SLA violation alarms are generated if necessary. The SLA KPIs are specific for SLA and are somewhat different form the performance indicators for services and SAPs.

The KPIs for Service Access Points are aggregated on each level of the structure. It is possible to define separate aggregation formulas for each type of SAP or Service. After aggregation the KPIs are used to define the state of the Service and then propagated upwards to define the state of the SLA.

Three main modes have been implemented to manage the SLA contracts:

- „Real time” reporting with Alarm List,
- „On demand” reporting with Reporting Module,
- Automatic, scheduled reporting with separate Reporting Module.

The Alarm List, shown in Fig.8 enables (with use of specific filters) to show only this information which is at the moment the most relevant and important for the operator. The operator can monitor the states and KPIs of the SLAs, and in case of an emergency can “drill-down” through the whole tree like structure of the services and SAPs exactly to the network element which causes problems. This speeds up the reaction times, and helps to cut down on penalty costs resulting from SLA violations.

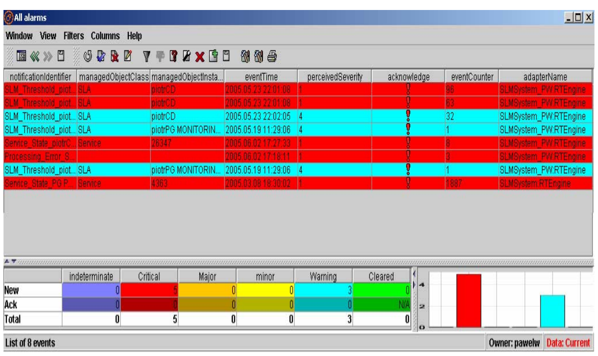

Figure 8. Alarm list service monitoring

The “on demand” reporting helps the operator to see all the aggregated KPIs of the selected service and SLA. It might also be useful before actually signing and implementing the SLA, in order to prepare feasibility study if the expected parameters can possibly be met.

Automatic reporting can be scheduled after the SLA has been signed in order to prepare detailed billing information and SLA report for the customer. Specific rules can be defined in order to calculate the penalties resulting from SLA violations.

SLA management framework can also be integrated with CRM system, and certain access rights to some information can be given to account managers to help them manage specific SLAs of their clients.

5. Summary

The article presents most frequently used definitions of SLA penalty functions. The discussion stresses the need for reflecting the design of an SLA contract in the associated penalty function, especially when the contract consists of a few sub-contracts. As a result of the discussion, a few most frequently used definitions of penalty functions for sub-contracts were proposed.

The presented considerations influenced the design of the Comarch SLA Management Framework, that is presented in the article as a system that is capable of calculating the discussed functions in order to monitor the SLA contracts and signal possible SLA breaches before they occur.

6. Acknowledgement

This research was partially supported by Polish Ministry of Education and Science; grant no. E! 3152 SLAM.

7. References

[1] Carr Jim, *Service Level Agreements,* CMP, Inc., 2001.

[2] *Service-Level Management: Defining and Monitoring Service Levels in the Enterprise*, White Paper, Cisco Systems, Inc., 2001.

[3] John Lee, Ron Ben-Natan, *Integrating Service Level Agreements: Optimizing Your OSS for SLA Delivery*, Wiley, 2002.

[4] Rick Sturm, Wayne Morris, *Foundations of Service Level Management*, Sams, 2000.

[5] *Viewgate's Advisor Product Enables Service Providers to Deliver on VPN SLAs and Build Customer Loyalty* , Viewgate Networks, ISP Business, July 2001.

[6] Aglient, *Agilent's QoS Manager – Firehunter*, www.agilent.com.

[7] Comarch S.A, *InsightNet*, www.comarch.com.

[8] Rick Sturm, SLA Metrics, Network World Fusion, http://www.slminfo.org/articles/slametrics.htm