# Quality of Service Attributes for Software as a Service

***Lukas Burkon***

***Department of Information Technologies***

***University of Economics, Prague, Czech Republic***

burl01@vse.cz

***Abstract:*** *Software as a Service (SaaS) has been developing for over ten years and, is reaching a mature level, where quality and its monitoring and management become significant. Although, SaaS is derived from the ASP model, SaaS background and architecture is different and therefore also SaaS quality management is based on different concepts. This paper is focused on the difference between traditional IT outsourcing and SaaS and proposes a set of quality attributes appropriate for the management of the SaaS quality.*

**Key words**: Quality of Service, Service Level Agreement, Software as a Service

## 1. Problem Area

Service Level Agreement (SLA), a contract between service provider and service customer (Sandholm, T. 2005), is well researched in the IT outsourcing (ITO) context as a way to define the quality of delivered IT service. Now in the era of Software as a Service (SaaS), SLA is also used for the same purpose, but the context has changed – SaaS has different background in comparison to ITO from the business and technological perspective while the SLA structure, content and form should reflect such fact.

There is a lack of research literature covering SLA topic in SaaS environment and this is the motivation for this paper. Firstly, we will summarize important literature and discuss the key differences between SaaS and ITO from the several perspectives to distinguish both terms. Then we will focus on the concept of Quality of Service (QoS), important for the structure and content of SLA. The final part of this paper will be dedicated to quality attributes of SaaS service with the objective to formalize the fundamental set of important SaaS quality attributes to be used while creating SLA for SaaS service.

## 2. Related Work

In this section we will summarize important literature concerning SLA in ITO and Cloud Computing context.

### 2.1 IT Service Management

The background for the IT Service Management (ITSM), which is focused on the management of IT services within the organizations (Thiadens, T. 2005), can be found in IT Infrastructure Library (ITIL) (OGC 2007). ITIL has become de facto standard in regard to IT service management, governance, quality and operational issues by defining the key processes and activities. One of ITIL's key processes is Service Level Management (SLM) aiming to "maintain *and improve IT service quality, through a constant cycle of agreeing, monitoring and reporting upon IT service achievements and instigation of actions to eradicate poor service, in line with business or cost justification"* (OGC 2003).

SLM emphasizes a quantitative monitoring to be applied on the customer requirements' fulfillment within the services provided by the IT organization. Therefore the system of relevant metrics has to be established aligned with the customer requirements and business objectives, using of several frameworks such as Six Sigma (Persse, J.R. 2006).

SLM distinguishes three types of SLAs (Bon J.v. 2007):

- **Service-based SLA** – an SLA covers a service for all clients.
- **Customer-based SLA** – an agreement with a client containing all services he wants to use.
- **Multi-level SLA** – a layered combination of mentioned SLA types.

SLM as a process comprises inputs, outputs, and metrics. Metrics consist of three KPIs (Key Performance Indicators): descending percentage of non-attained goals, client satisfaction percentage, and reduction percentage of SLA infractions. These metrics should be bound to QoS attributes, and SLA metrics.

SLM also proposes the minimum SLA content that includes (Bon J.v. 2007): description of service, service targets, communications and reporting; authorization details and validity period; financial management details; service provider liability and obligations; customer responsibilities; supporting and related services; impact, urgency and priority guidelines; service hours, date exceptions, critical business periods and out-of-hours cover; workload limits; contact details of people authorized to act in case of emergencies; actions to be taken in the event of a service interruption; escalation and notification process; scheduled and agreed interruptions; complaints procedures; housekeeping procedures; exceptions to the terms given in the SLA; and glossary of terms.

## 2.2 Quality of Service

### 2.2.1 QML Fundamental Concepts

Frølund and Koistinen from HP Laboratories developed in late nineties Quality of service Modeling Language (QML) (Koistinen, J., Frolund, S. 1998). Predecessor of the above mentioned languages concentrates on Quality of Service (QoS) of software systems in the form of technology-independent abstract language. QML uses several fundamental concepts:

*Dimensions* are service quality attributes (e.g., latency, throughput, failure semantics, encryption level etc.). Dimensions are related into higher categories.

*Contract Type* represents a QoS category such as performance, availability, security, or timing.

*Contract* is an instance of a contract type and represents a particular QoS specification within a given category.

*Profile* describes associations between contracts and interface elements for a particular interface.

*Refinement* is statically defined relationship between two profiles, much like inheritance in most object-oriented languages.

### 2.2.2 QoS Attributes

Based on SOA concepts O'Brien, Merson and Bass (O'Brien, L., Merson, P., Bass, L. 2007) designed a set of attributes concerning quality in service-oriented environment. Attributes are technically oriented, mostly covering web service topics as follows:

*Interoperability* is an ability of a collection of communicating entities to share specific information and operate on it according to an agreed-upon operational semantics.

*Performance* consists of response time (how long does it take to process a request), throughput (how many requests overall can be processed per unit of time), or timeliness (ability to meet deadlines, i.e., to process a request in a deterministic and acceptable amount of time).

*Security* – composite topic consisting of confidentiality, authenticity, integrity, availability.

*Reliability* is an ability of a system to keep operating over time without failure. Authors distinguish between Message Reliability and Service Reliability referencing standard SOA implementations.

*Availability* is a proportion of time a system or component is operational and accessible when required for use.

*Modifiability* is an ability to make changes to a system quickly and cost-effectively. These changes include adding new services and extending existing services with or without changing the interfaces.

*Testability* is a degree to which a system or service facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met. Testability is primarily used by service provider, but it also could be used by service customer as a degree how far could be other QoS attributes measured.

*Usability* is a measure of the quality of a user's experience in interacting with information or services.

*Scalability* is an ability of SOA to function well (without degradation of other quality attributes) when the system's size and volume increase in order to meet user needs.

## 2.3 Service Level Agreement

Marilly, Martinot, Betgé-Brezetz and Delègue describe main drivers and basic concepts for SLA management in Telco market (Marilly, E., Martinot, O., Betgé-Brezetz, S., Delègue, G. 2002). Authors define *Service Level Specification* (SLS) as a technical part of SLA consisting of a set of technical parameters and their associated semantics that describe the service (network availability, throughput, and latency). Authors distinguish between 2 SLA types – *horizontal* (SLA between two providers being at the same OSI layer) and *vertical* (SLA between two providers at two different OSI layers). According to requirements, authors consider:

- *technical requirements on SLA* – a customer and service provider responsibilities, service provider procedures to be invoked in case of violation of SLS guarantees, service pricing and discounting policies, service description and the QoS commitments, and reporting to the customer.
- *technical requirements on SLS* – the QoS metrics and corresponding thresholds that must be guaranteed by the service provider, service performance measurement method, measurement period, provided reports, and service schedule.

Lamanna, Skene and Emmerich introduce basic principles of SLAng (Lamanna, D.D., Skene, J., Emmerich, W. 2003) – comprehensive language designed for the purpose of SLA definition, focused on electronic services. The SLAng syntax is defined using XML Schema allowing combination with WSDL and Business Process Execution Language (BPEL) (OASIS 2007) to obtain a complete e-business automation solution. The content of an SLA varies depending on the service offered and incorporates the elements and attributes required for the particular negotiation.

In general, SLAng SLA includes an end-point description of the contractors (e.g., information on customer/provider location and facilities), contractual statements (e.g., start date, duration of the agreement, charging clauses), and Service Level Specification (SLS) – the technical QoS description and the associated metrics.

SLAng recognizes 2 kinds of SLA: Vertical (Application, Hosting, Persistence, Communication), and Horizontal (Service, Container, Networking).

Patel, Ranabahu and Sheth focus on XML standard from the web services world – Web Service Level Agreement (WSLA) – covering semantic specification of service level agreements. Authors propose architecture (Patel, P., Ranabahu, A., Sheth, A. 2009) for practical use of WSLA containing also 3 types of services:

- *Measurement Service* – this service is responsible for measuring the runtime parameters of cloud providers' resources.
- *Condition Evaluation Service* – this service is responsible of getting the results from measurement services and evaluating the Service Level Objectives.
- *Management Service* – this service is responsible for taking corrective actions on violation of the Service Level Objectives.

Naomi Karten proposes that SLA should include two fundamental elements (Karten, N. 2008):

- *Service elements* – clarify services such as the services provided, conditions of service availability, service standards, and responsibilities of parties, cost, and escalation procedures.
- *Management elements* – focus on things as tracking service, reporting service information, resolving service-related disagreements, and revising the agreement.

## 2.4 Service Level Agreement

Notwithstanding the availability of several frameworks concentrating on the composition of the SLA's abstract model, there is lack of fundamental set of concrete attributes to be used to monitor and manage quality of SaaS services. The QoS attributes for ITO contracts are available, but SaaS services are not able to fully use them due to different characteristics of SaaS and because these QoS attributes are too general and incomplete for SaaS services. The basic differences between ITO and SaaS will be described in the next section in greater detail.

# 3. Key Differences between SaaS and ITO

ITO and SaaS are two specific terms but with different QoS attributes. Will summarize key differences between SaaS and ITO in this section to explain the reasons to define specific QoS attributes for SaaS services.

## 3.1 ITO Characteristics

Gartner defines ITO as follows (Gartner IT Glossary):

*Outsourcing, which also includes utility services, software as a service and cloud-enabled outsourcing, helps clients to develop the right sourcing strategies and vision, select the right IT service providers, structure the best possible contracts, and govern deals for sustainable win-win relationships with external providers.*

From this perspective, Gartner considers ITO as an umbrella term for the general delivery of IT services by external providers to service customers and therefore cloud services are part of their ITO definition.

Nevertheless, we will focus on the traditional form of ITO where multi-year contracts based on developing and maintaining large amounts of custom code and on-site systems integration work take place (Overby, S. CIO.com). From the broader point of view, we can categorize ITO under the term of Business Process Outsourcing (BPO) (Overby, S. CIO.com) concerning the outsourcing of a specific business process task, such as payroll, online marketing, or IT.

Traditional ITO contracts are usually focused on the large-scale outsourcing concerning desktop and network maintenance, infrastructure and application hosting, etc. These contracts are typically of multi-year scale, and their negotiation and implementation takes several months, the price is usually high, and their elasticity is limited. Even then customer can benefit from such relationships, especially when customer outsource most or all of their IT including specific processes and focus on their core business.

## 3.2 SaaS Characteristics

There are many definitions of Cloud Computing, SaaS, and other related terms. The most widely used definition of Cloud Computing was proposed by NIST (Mell, P., Granc, T. 2011) defining 5 essential characteristics of cloud computing and dividing cloud services into three delivery models and four deployment models. The SaaS delivery models is defined by NIST as:

*The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure and accessible from various client devices through a thin client interface such as a Web browser (e.g., web-based email). The consumer does not manage or control the underlying cloud infrastructure, network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.*

From the perspective of the above definition the difference between standard ITO and SaaS may not be that obvious. The explanation of NIST cloud characteristics makes this difference clearer. NIST proposes following key cloud attributes: on-demand self-service, ubiquitous network access, location independent resource pooling, rapid elasticity, measured service. We will look closer at each attribute to clarify the main differences between SaaS and ITO.

On-demand self-service allows SaaS customer to provision service capabilities automatically without requiring human interaction with service provider using of preprogrammed user interface. ITO contracts are usually based on human interactions between service customer and service provider.

Ubiquitous network access allows SaaS customer to use service through the computer network, basically Internet in terms of public clouds. ITO is more frequently provided in the form internal application and IT maintenance outsourcing so that users are usually able to use outsourced applications only through the company's internal network that often limits them to use the applications physically outside of the company.

Location independent resource pooling is the most important attribute differentiating SaaS and ITO. SaaS applications are built on multi-tenant architecture that is implicitly prepared for the huge size of customer base, the largest SaaS services serve millions of customers. ITO contracts are usually extensively customized for each customer needs and based on the important human interaction that

significantly limits service provider in terms of the extent of its customer base. SaaS aims to automatically serve all customers based on the technical architecture of the SaaS application, so that the costs of adding each new customer are minimal.

Multi-tenant architecture especially in case of large SaaS services is settled on cloud infrastructure, often on another IaaS (Infrastructure as a Service) or PaaS (Platform as a Service) service allowing SaaS to be easily and automatically scalable based on the actual customers' needs. ITO contracts are not that elastic as SaaS are. Basic barriers are formalized contracts, inelastic infrastructure, and human interaction while changing the contract terms.

Both SaaS and ITO contracts are measured based on the amount of consumption. SaaS services due to their technical scalability are also scalable in terms of the price policy. While SaaS customers can dynamically change their expenses by changing the consumption behavior, ITO contracts are usually based on the fixed long-term price and changes in the expenses has to be made by updating the contract.

# 4. SaaS QoS Attributes

Once the differences between SaaS and ITO are clear we can move to compose basic set of SaaS QoS attributes derived from the specifics of SaaS services.

## 4.1 QoS Structure

Before we start forming the basic set of SaaS QoS attributes we should design fundamental structure that will be applied to organize the set of attributes. Firstly, we will design the abstract taxonomy defining essential terms for the QoS structure and then we will populate it with concrete QoS structure content.

Previously mentioned QML framework (Koistinen, J., Frolund, S. 1998) will help us to define the basic QoS taxonomy. QML uses concepts such as *Dimension*, *Contract*, *Contract Type*, *Profile*, and *Refinement*. We will use concepts *Dimension* and *Contract Type* and rename them into the *QoS Dimension* and *QoS Category*.

*QoS Category* represents an organized set of related *QoS Dimensions*. QoS dimension represents concrete QoS attribute such as uptime, or response time.

Now that we have essential terms apparatus we will design the fundamental QoS organization content. QoS dimensions will be designed further, now we will fill the set of QoS categories. For this purpose we can use the above mentioned research paper concentrating on QoS attributes for SOA environments. O'Brien, Merson and Bass (O'Brien, L., Merson, P., Bass, L. 2007) propose following nine QoS categories that we will use to organize our set of QoS dimensions: Availability, Performance, Reliability, Scalability, Security, Interoperability, Modifiability, Usability, and Testability. And finally we will add one more QoS category – Support – that is crucial for most of the IT services.

## 4.2 QoS Dimensions

In this key section we will consider each of the selected QoS categories and based on the literature analysis (O'Brien, L., Merson, P., Bass, L. 2007) and our own considerations we will propose essential set of QoS dimensions for each category.

### 4.2.1 Availability

Prevailing quality attribute used in practically all SaaS SLAs is *Uptime*, specifying proportion of time SaaS service is operational and accessible when required for use. In case of large SaaS services *Uptime* can be measured not for the overall service but for the each of the SaaS service components separately.

### 4.2.2 Performance

Performance dimensions proposed by O'Brien, Merson and Bass (O'Brien, L., Merson, P., Bass, L. 2007) consist of response time, throughput, and timeliness.

*Response time* is second most frequently used QoS attribute for SaaS services specifying how long does it take to process a request. *Response time* is usually measured in milliseconds.

*Throughput* represents the number of requests that can be processed per unit of time. *Throughput* is tightly connected to scalability of service and it can be dynamically adjusted based on the service customer needs and followed by the change of resultant price.

*Timeliness* is the ability to meet deadlines that is quite subjective and significantly different for each service customer and therefore we will not include this dimension into the final list.

### 4.2.3 Reliability

Reliability is an ability of SaaS service to keep operating over time without failure. We can generalize reliability as tracking of availability and performance dimensions over time.

Nevertheless, the key QoS dimension for reliability is *Uptime* that indicates what portion of time has service operated. Therefore key reliability QoS dimension will be *Uptime history* consisting of the historical view of the *Uptime*.

### 4.2.4 Scalability

Scalability is an ability of SaaS service to function well (without degradation of other QoS dimensions) while service customer changes the size or volume of the consumed service resources.

There are two basic characteristics of service scalability – *Granularity* and *Elasticity* – that can be used as scalability QoS dimensions.

*Granularity* represents the level of detail of service resources to which consumption can be scaled up or down to. Higher granularity means deeper detail, i.e. the consumption of more detailed components of service can be applied.

*Elasticity* represents the size of consumption of SaaS service resources that can be changed by the unit of time in both up or down.

### 4.2.5 Security

Security consists of four fundamental topics – confidentiality, authenticity, integrity, and availability.

Cloud Security Alliance (Cloud Security Alliance 2009) defines two categories of domains used for cloud security guidance – governance and operational domains. Governance domains consist of Governance and Enterprise Risk Management, Legal and Electronic Discovery, Compliance and Audit, Information Lifecycle Management, and Portability and Interoperability. Operational domains consist of Traditional Security, Business Continuity, and Disaster Recovery, Data Center Operations, Incident Response, Notification, and Remediation, Application Security, Encryption and Key Management, Identity and Access Management, and Virtualization.

Based on the Cloud Security Alliance domains we will extract fundamental QoS dimensions in the essential form to simplify the comprehensive security domain and fit the resultant dimensions to the SaaS environment.

From the high-level service customer point of view SaaS service should be certified with the appropriate quality and security certifications ensuring the appropriate level of quality and security guaranteed by the third-party auditor. *Compliance certifications* dimension is the checklist containing appropriate certifications such as SAS 70, SSAE 16, ISAE 3402, ISO/IEC 9126, ISO/IEC 27001, or ISO/IEC 27002.

On the application level we should primarily focus on the access management and application security coverage. Access management in the form of *Authentication* and *Authorization* dimensions specifies the identity verification techniques and the character of privilege management apparatus. Application security coverage is represented by *Attack prevention* specifying applied techniques to prevent widespread and specific application attacks such as session hijacking, cross-site scripting (XSS), SQL injection, cross-site request forgery (CSRF), etc.

On the infrastructure level we should primarily focus on application and data backup and recovery, and applied infrastructure virtualization. *Backup coverage* specifies the percentage of application and data that is covered by backup processes. *Backup periodicity* specifies how often are backups created, if it is in continuous manner, or in regular intervals. *Recovery velocity* specifies how fast can be application and data recovered from backup in case of application or infrastructure issues. *Infrastructure architecture* represents the character how the underlying infrastructure is designed, if it consists of only a one physical server, infrastructure cluster, or if it is based on IaaS.

### 4.2.6 Support

Once SaaS services driven on multitenancy principle serve to millions of customers, customization of the service has its limits. Service customers are not able to fully move their applications on different infrastructure or make whichever modifications of the application. They are fully dependent on the SaaS customization capabilities and service provider support.

From the support point of view we can monitor three types activities – support requests and their fulfilling, requests for change (RFC) fulfilling, and activity of community over the service.

QoS dimensions applied on standard support requests (i.e. incident management) consist of two metrics – *Resolution rate* and *Resolution time*. *Resolution rate* measures the percentage of incidents that have been successfully resolved. *Resolution time* measures mean time taking incident to be resolved.

Every customer is in specific situation resulting in specific needs for application features. SaaS services are continually evolving during the time and they should reflect majority opinion of their customers in context of application features. *RFC implementation rate* should measure the ratio of the RFCs that has been implemented. This QoS dimension because of the SaaS nature should also incorporate the weight accounting for the power of the RFC (i.e. how many customers request this change).

In Web 2.0 era many problems concerning technology and IT services could be resolved by the interaction with community around particular technology and services. The existence and extent of such community has positive effect on both service provider and service customer. *Community* QoS dimension should measure which of the community channels and their support tools (e.g. discussion boards, tutorials, screencasts, webinars, events, etc.) are present and their extent (e.g. number of community members, number of discussion threads, number of screencasts, periodicity of webinars, etc.).

### 4.2.7 Interoperability

Interoperability is the ability of a collection of communicating entities to share specific information and operate on it according to an agreed-upon operational semantics. In context of SaaS services, interoperability is usually connected to Application Programming Interface (API) allowing integration of service with other services and systems.

From the context of QoS dimensions we could measure two essential characteristics covering API – *API coverage* and *API form*. *API coverage* represents the percentage of SaaS resources and data that is accessible through API. *API form* consists of the technical nature how the API is implemented, e.g. Representational State Transfer (REST), Simple Object Access Protocol (SOAP), XML-RPC, etc.

### 4.2.8 Modifiability

Modifiability is the ability to make changes to SaaS service quickly and cost-effectively. These changes include modification of three basic application layers – data, logic, and presentation – while taking into account multitenant nature of SaaS services.

Based on the mentioned layers we can define three QoS dimensions – *Data layer modifiability*, *Logic layer modifiability*, and *Presentation layer modifiability* – each dimension consists of list of options representing modifiability capabilities of SaaS service.

*Data layer modifiability* specifies modification capabilities in application data model – creation of new data objects and modification of existing data objects' structure and relations.

*Logic layer modifiability* specifies modification capabilities in application logic model – changes in application workflow, validation rules and filters over the data objects, development of new application features.

*Presentation layer modifiability* specifies modification capabilities in application presentation model – modification of user interface, and changes in application graphic design.

Enabling of maximal SaaS service modifiability requires application to be fully driven on meta-model basis and appropriate engine, usually in combination with PaaS platform beneath the SaaS service, as Salesforce.com has with their Force.com platform.

### 4.2.9 Usability

Usability is a measure of the quality of a user's experience in interacting with SaaS service. Because usability is a subjective matter it should be measured by soft metrics. In context of application usability we will define three important metrics – *Satisfaction*, *Accessibility*, and *Efficiency*.

*Satisfaction* is based on customer survey and it measures aggregated subjective opinion over the service customer base.

*Accessibility* refers to how well is the application usable by people with different and disabilities. In context of SaaS services suitable guidelines are defined by W3C organization – Web Accessibility Initiative (WAI) (W3.org).

*Efficiency* is the implication of good usability. SaaS services should support productivity and amplify work efficiency. *Efficiency* can be measured by the amount of time saved by the use of application, or by the length of the transaction processing.

### 4.2.10 Testability

Testability is a degree to which a SaaS service facilitates the establishment of test criteria and the performance of tests to determine whether those criteria have been met. Testability in context of SaaS services has two perspectives – *QoS testability* and *Application testability*.

*QoS testability* is the percentage of QoS dimensions that are accessible to be tested.

*Application testability* consists of the list of standard application test methods (i.e. unit tests, behavioral tests, performance tests, etc.) that are applied during the application development and their results.

## 4.3 QoS Dimensions Summary

We will summarize previously defined QoS dimensions clearly in Tab. 1.

***Tab. 1: SaaS QoS Dimensions***

| QoS Category | QoS Dimensions |
|---|---|
| *Availability* | Uptime |
| *Performance* | Response Time, Throughput, Timeliness |
| *Reliability* | Uptime History |
| *Scalability* | Granularity, Elasticity |
| *Security* | Compliance Certifications, Authentication, Authorization, Attack Prevention, Backup Coverage, Backup Periodicity, Recovery Velocity, Infrastructure Architecture |
| *Support* | Resolution Rate, Resolution Time, RFC Implementation Rate, Community |
| *Interoperability* | API Coverage, API Form |
| *Modifiability* | Data Layer Modifiability, Logic Layer Modifiability, Presentation Layer Modifiability |
| *Usability* | Satisfaction, Accessibility, Efficiency |
| *Testability* | QoS Testability, Application Testability |

Some of the QoS dimensions seem to be important mostly for service provider (e.g. Application Testability), but from our point of view service customer should consider all of the defined QoS dimensions, even the more technical ones that allows him to fully recognize SaaS service potential and weaknesses and appropriately adapt his business processes.

## 5. Summary and Further Work

In this paper we focused on the specification of key SaaS QoS metrics. Specified metrics that we call QoS dimensions are structured into the ten QoS categories - Availability, Performance, Reliability, Scalability, Security, Support, Interoperability, Modifiability, Usability, and Testability.

We have identified 28 QoS dimensions (such as Uptime, Response time, Compliance certifications, Resolution time, etc.), but the list is not necessarily final. Each user of our QoS system can adjust it according to his situation, i.e. add new QoS dimensions, or remove existing ones to better reflect the needs of his business. Usually, it is the trade-off between large amount of administration (to setup and maintain QoS system) and added business value.

QoS dimensions should enhance impact and improve quality and business value of existing SaaS SLAs and SaaS QoS systems.

Further progress based on mentioned conclusions can be oriented in two basic directions.

Firstly, each of the SaaS QoS dimensions can be specified in greater detail including extent description, units of measure, measurement process and tools, and comparison of service provider and service customer view.

Secondly, resultant SaaS SLA does not have to include all of mentioned QoS dimensions. There is a need for a rigorous process how to choose appropriate QoS dimensions, how to incorporate them into the SLA, and how to setup measurement system on both service provider and service customer side to monitor and control SaaS SLA compliance.

## Acknowledgments

This research was supported by GAČR (Grant Agency, Czech Republic) grant No. P403/11/0574.

## Resources

Bon J.v., and co-authors team, 2007: *IT Service Management – And Introduction*, Van Haren Publishing, ISBN: 978 90 8753 051 8

Cloud Security Alliance, 2009: *Security Guidance for Critical Areas of Focus in Cloud Computing V2.1C*

Gartner IT Glossary. *IT Outsourcing*, http://www.gartner.com/it-glossary/it-outsourcing/

Karten, N., 2008: *Establishing Service Level Agreements*, http://www.nkarten.com/sla.html

Koistinen, J., Frolund, S., 1998: *QML: A Language for Quality of Service Specification*, HP Software Technology Laboratory

Lamanna, D.D., Skene, J., Emmerich, W., 2003: *SLAng - A Language for Defining Service Level Agreements*, University College London

Marilly, E., Martinot, O., Betgé-Brezetz, S., Delègue, G., 2002: Requirements for Service Level Agreement Management, IP Operations and Management, *IEEE Workshop*, ISBN: 0-7803-7658-7

Mell, P., Granc, T., 2011: *The NIST Definition of Cloud Computing*, National Institute of Standards and Technology, http://csrc.nist.gov/publications/nistpubs/800-145/SP800-145.pdf

O'Brien, L., Merson, P., Bass, L., 2007: Quality Attributes for Service-Oriented Architectures, *SDSOA '07 Proceedings of the International Workshop on Systems Development in SOA Environments*, ISBN:0-7695-2960-7

OASIS, 2007: *Web Services Business Process Execution Language (WSBPEL) TC*, https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel

OGC, Office of Government Commerce, 2007: *Service Strategy, ITIL Version 3*, ITSMF- IT Service Management Forum

OGC, Office of Government Commerce, 2003: *ITIL Service Delivery Version 2.0*

Overby, S., CIO.com. *Outsourcing Definition and Solutions*, http://www.cio.com/article/40380/Outsourcing_Definition_and_Solutions

Patel, P., Ranabahu, A., Sheth, A., 2009: Service Level Agreement in Cloud Computing, *Cloud Workshops at OOPSLA*

Persse, J.R., 2006: *Process Improvement Essentials*, O'Reilly Media, Inc.

Sandholm, T., 2005: *Service Level Agreement Requirements of an Accounting – Driven Computational Grid*, http://www.hpl.hp.com/personal/Thomas_Sandholm/sandholm2005b.pdf

Thiadens, T., 2005: *Manage IT! - Organizing IT Demand and IT Supply*, Springer

W3.org. *Web Accessibility Initiative (WAI)*, http://www.w3.org/WAI/

**JEL Classification: L15, M10**