slalom

LEGAL & OPEN MODEL TERMS FOR CLOUD SLA AND CONTRACTS

SLA specification and reference model - a

D3.2

Dissemination level: Public

Work Package	**WP3, Technical Track**
Due Date:	M6 (30/06/2015)
Submission Date:	10/07/2015
Version:	1.0
Status	Final for submission
Author(s):	Nikos Bakalos (ICCS), Dimosthenis Kyriazis (ICCS), Emmanuel Protonotarios (ICCS), Theodora Varvarigou (ICCS), Oliver Barreto (ATOS), Ana Juan (ATOS), Aimilia Bantouna (UPRC), Panagiotis Demestichas (UPRC), Andreas Georgakopoulos (UPRC), Teta Stamati (UPRC), Kostas Tsagkaris (UPRC), Panagiotis Vlacheas (UPRC)
Reviewer(s)	Daniel Field (ATOS), Panagiotis Demestichas (UPRC)

The SLALOM Project is co-funded by the European Commission through the H2020 Programme under Grant Agreement 644720

CONTENTS

Tables

1 Introduction

The current document is the first one in the series of three deliverables of the SLALOM project that aims at proposing a specification for cloud Service Level Agreements (SLAs) as well as a set of management mechanisms that aim at addressing the SLA lifecycle. Legal and privacy aspects are not dealt in this initial version of the SLA specification and management mechanisms but will be addressed in the next versions of this report.

The proposed SLA specification refers to the core SLA document that incorporates metrics (as specific objectives or quality attributes), parameters, rules as well as potential dependencies between rules. An XML schema of the initial proposed SLA specification is also included in the document (aiming to provide a machine-readable format of the SLALOM proposition) along with practical examples of the proposed approach. There should be noted that given the wide set of potential metrics, an initial prioritization is proposed.

Furthermore, the current document captures an initial set of SLA management mechanisms that aim at enhancing the SLA lifecycle through innovative added-value outcomes emerging from research. The proposed set is an initial one which will be further validated and enriched following the planned project activities to engage and interact with various stakeholders, cloud providers and customers / adopters.

The report is structured as follows: Section 2 briefly introduces the process followed to develop the initial SLA specification and identify the first set of management mechanisms. Thereafter an overview of SLA specifications and management mechanisms exploited in today's market (section 3), emerging from research (section 4) or discussed in standardisation bodies (section 5) is presented. The stakeholders (cloud providers and adopters views) are captured in Section 6, while the SLA specification and the management mechanisms are presented in Sections 7 and 8 respectively. Conclusions are drawn in section 9.

2 Process

The process followed to define the initial SLA specification and the first set of SLA management mechanisms included:

1. Current market and research status analysis: Identification of existing commercial and research SLA specifications and management mechanisms.
2. Standardisation approaches analysis: Identification and analysis of the main work items and outcomes of various initiatives and standardisation working groups (e.g. C-SIG, ISO SLA working group, etc).
3. Stakeholders input collection and analysis: Analysis of the stakeholders views based on the activities performed in the project (e.g. questionnaire published to collect the views).
4. Compilation of core SLA specification: Based on the above, the initial SLA specification has been proposed.

5. Proposition of the main mechanisms for SLA management: Based on the conducted analysis, SLALOM proposes an initial set of management mechanisms enhancing the SLA lifecycle.

3 Current market status on SLA specifications

This chapter summarizes the existing findings of the current commercial SLA specifications, which derive includes the survey of SMART. Additionally, it presents the SLAs of some of the most well-known Cloud Service Providers (CSPs) that are considered as key players in the field of cloud computing (i.e., Amazon, Google, Microsoft, etc.)

3.1 Summary of SMART findings

The study presents the findings of a survey that categorizes the rules and practical approaches used with respect to SLAs for Europe's Member States (28 EU Member States and 4 EFTA countries). The study contains a summary of the collected information from all of the countries, packaged around four central themes, and provides a number of initial recommendations for each aspect to be addressed in future and to be implemented in the form of model SLA provisions:

- **Legislative landscape**: It describes whether there are specific rules in relation to cloud computing and/or SLAs in the surveyed countries.
- **Rules and policies in regulated sectors**: They describe multiple initiatives (laws, guidelines from regulators, or policies) in the financial, health and public sector that affect the usage of cloud computing and/or SLAs in the surveyed countries.
- **SLA terms and models**: They assess specific types of provisions which commonly occur in SLAs in the surveyed countries.
- **Policy and standardization**: They indicate whether any global policy or standard is being used to support or promote the use of specific SLAs or standards in the surveyed countries.

Among the recommendations that are given, we must cite the following:

- SLAs should be *clear and unambiguous*, given their function of clarifying obligations and specifying when liabilities may be triggered.
- *Liability* should generally be *addressed* in the service agreements, not in the SLA as such.
- Generally, sector *specific norms* emanate from specialized self-supervision, not from legislation, while health sector appears to be mostly self-regulated and exposed to external auditing to assure compliance**,** with strong requirements in terms of security, confidentiality, privacy (mostly referencing international standards, particularly the ISO 27000 family), availability/uptime, continuity, and support.
- UK, Germany, and Ireland seem to be more advanced than the other countries in terms of the policy framework. There is not a detailed and coherent policy relating to cloud computing or SLAs, referring to the rest of Europe.

- *Standard models* are rare and are not widely used. The same applies to the development of standards, where initiatives are being taken by the public sector, standardization bodies and industry.
- *Approaches to data migration* lack of consistency, and are usually seen as an "extra" rather than a key area of SLAs.
- Instead of defining a model term focusing on e.g., the availability percentage, it might be more beneficial for the cloud adopters to have *model terms for defining issues* such as uptime and downtime, since the Service Level Objectives (SLOs) and the way that they can be measured might be different for the various types of customers.
- Since exclusion of *liability* is common, it should be also taken into consideration.

3.2 Summary of standardisation guidelines

In the context of the European Cloud Computing Strategy, the European Commission issued a document with guidelines on the standardization of SLAs for CSPs in June 2014. They are based on the understanding that standardization will improve the clarity of SLAs for cloud services in the EU, which is a key pillar under this strategy.

The work identifies a set of principles to assist organizations in providing valuable information to help business and technical stakeholders understand the non-legal concepts and vocabulary used in cloud SLAs. These principles are not intended to be limiting nor to even set the final model terms, but establish an initial framework to develop future standardized SLAs in Europe. They include aspects such as technology and business model neutrality; the need for world-wide applicability; completeness and unambiguous definitions; and the fact that the recommendations should be based on specifying the structure of the SLA, rather than illustrating and specifying the concepts that should be addressed. It also establishes a clear line of the agreement between the CSP and the cloud adopter, but the agreement itself, must be done by qualified lawyers.

The work specially emphasizes on the provision of the definitions for different categories of Service Level Objectives (SLOs) designed to leverage standardization and comparison for several aspects of SLAs. To be comparable, SLOs do not need to be determined by identical means but sufficient information about the SLO needs to be provided by CSPs. Moreover, since cloud services offer very heterogeneous services, standardized terminology, concepts, metrics, vocabulary and templates should be used to describe how particular SLOs are determined. SLOs are often associated with metrics, defining a measurement method and scales and therefore, they establish the need to take into account state-of-the-art capabilities of the cloud industry. The Guidelines provide specific and common SLOs for the following categories in the Cloud Computing context:

- **Performance SLOs:** Availability, Response Time, Capacity, Capability Indicators, Support, Reversibility and the Termination Process

- **Security SLOs:** Service Reliability, Authentication & Authorization, Cryptography, Security Incident management and reporting, Logging and Monitoring, Auditing and security verification, Vulnerability Management, Governance
- **Data Management SLOs:** Data classification, Cloud Service Customer Data Mirroring, Backup & Restore, Data Lifecycle, and Data Portability
- **Personal Data Protection SLOs:** Codes of conduct, standards and certification mechanisms, Purpose specification, Data minimization, Use, retention and disclosure limitation, Openness, transparency and notice, Accountability, Geographical location of cloud adopter data, and Intervenability

The document also describes a number of documents and their relation to define technical and legal aspects, especially highlighting the fact that a cloud SLA can be a part of an overall Master Service Agreement (MSA), and that the SLA describes and sets SLOs for the cloud service. These documents may include, but are not limited to: Master Service Agreement (MSA); Service Level Agreement (SLA); Service Agreement; Acceptable Use Policy; Privacy Policy; Security Policy; Business Continuity Policy; and Service Description.

3.3 Cloud providers representative SLAs

The following subsections provide an analysis of the existing SLAs which are offered by commercial public clouds. More specifically, the analysis focuses on IaaS and PaaS offerings from major cloud vendors. The analysis is not very exhaustive, as it just provides a study of five (5) commercial offerings and nine (9) services, but it aims to provide an indication on what the cloud adopters are able to find today in terms of SLA in the market.

3.3.1 Amazon Web Services SLA

Amazon Web Services (AWS) use different SLA documents for different services. This document will first analyse the SLA for Amazon Elastic Compute Cloud (Amazon EC2) and Elastic Block Store (Amazon EBS) which are described by terms provided in [1]; afterwards it presents the SLA for Amazon Simple Storage Service (Amazon S3) terms in [3]. Both SLA definitions refer to the AWS Customer Agreement, available at [3]. This deliverable has decided to limit the analysis to these AWS services due to their predominance and their wide usage.

Amazon EC2 and EBS SLA states a service commitment of a Monthly Uptime Percentage of at least 99.95% per monthly billing cycle. This percentage is calculated as 100% minus the “percentage of minutes during the month” in which applicable region was in state “Region Unavailable”. A region is considered as Unavailable if more than one Availability Zones in which an instance is running is “Unavailable”. In AWS EC2 each region is a separate geographic area. Each region has multiple, isolated locations defined as Availability Zones [3]. At instance deployment time the cloud adopter can select the Availability Zone for the instance or let AWS to decide. The concept of unavailability differs among services. For Amazon EC2 it is defined as “when all of your running instances have no external connectivity” while for Amazon EBS it means that “all of your attached volumes perform zero read write IO, with pending IO in the queue”.

AWS commits to use "commercially reasonable efforts" so as to achieve this uptime percentage, although definition for this concept is not available in the document.

The Monthly Uptime Percentage excludes of the Monthly Uptime Percentage measurements downtime resulting directly or indirectly from the following causes: a) that result from a cloud adopter's suspension described in AWS Agreement [3]; b) caused by factors outside of Amazon's reasonable control, including any force majeure event or Internet access or related problems beyond the demarcation point of Amazon EC2 or Amazon EBS; c) that result from any actions or inactions of the cloud adopter or any third party, including failure to acknowledge a recovery volume; d) that result from cloud adopter's equipment, software or other technology and/or third party equipment, software or other technology (other than third party equipment within AWS direct control); e) that result from failures of individual instances or volumes not attributable to Region Unavailability; f) that result from any maintenance as provided for pursuant to the AWS Agreement; or g) arising from Amazon's suspension and termination of the cloud adopter's right to use Amazon EC2 or Amazon EBS in accordance with the AWS Agreement (collectively, the "Amazon EC2 SLA Exclusions") [1].

In case of Monthly Uptime Percentage less than Service Commitment (99.95%) and only in the case that AWS confirms the unavailability and after a cloud adopter's submission of a claim, AWS issues Service Credits for future service Amazon EC2 or EBS services. Cloud adopter's claim has to be received by the end of second billing cycle after the incident. These service credits are calculated the in following way:

- If Monthly Uptime Percentage is between 99.95% and 99,0%, Service Credit Percentage = 10%
- If Monthly Uptime Percentage < 99,0% , Service Credit Percentage = 30%

Service Credits are calculated as a percentage of total charges paid by the cloud adopter for the services that in the Region affected for the monthly billing cycle in which the Region Unavailability occurred. A Service Credit is a dollar credit and is exclusively for the cloud adopter's account.

Table 1 Amazon EC2 and EBS SLA Terms Summary

Type of Term	Availability
Term	Monthly Uptime Percentage
Value	99,5%
Calculation	100% - percentage of minutes during the month in which applicable region was in state "Region Unavailable"
Guarantee	Not clearly defined, stated as "commercially reasonable efforts"
Penalties	Yes (if confirmed by AWS, not one of excluded cases, after cloud adopter's claim and claim received by the end of second billing cycle after the incident)
Penalties Form	Future Service Credits
Penalties Calculation	If Monthly Uptime Percentage between 99.95% and 99,0%, Service Credit Percentage = 10% If Monthly Uptime Percentage < 99,0% , Service Credit Percentage = 30%

Amazon S3 SLA [2] states a service commitment of a Monthly Uptime Percentage of at least 99.9% per monthly billing cycle. In this case Monthly Uptime Percentage is calculated by "subtracting from 100% the average of the Error Rates from each five minute period in the monthly billing cycle". Error Rates are defined as follows: "the total number of internal server errors returned by Amazon S3 as error status "InternalError" or "ServiceUnavailable" divided by the total number of requests during that five minute period". This is calculated as a percentage for each five minutes period in the concrete cloud adopter account in the monthly billing cycle.

In case of Monthly Uptime Percentage less than Service Commitment (99.9%) Service credits are calculated in the following way:

- If Monthly Uptime Percentage >= 99% and Monthly Uptime Percentage < 99,9%, Service Credit Percentage = 10%
- If Monthly Uptime Percentage < 99% , Service Credit Percentage = 25%

For this service both commitments, calculation of the exclusions and procedures for Service Credit's claims and payments follow a similar approach to the ones exposed for Amazon EC2 and EBS.

Table 2 Amazon S3 Terms Summary

Type of Term	Availability
Term	Monthly Uptime Percentage
Value	99,9%
Calculation	100% - average of the Error Rates from each five minute period in the monthly billing cycle
Guarantee	Not clearly defined, stated as "commercially reasonable efforts"
Penalties	Yes (if confirmed by AWS, not one of excluded cases, after cloud adopter's claim and claim received by the end of second billing cycle after the incident)
Penalties Form	Future Service Credits
Penalties Calculation	If Monthly Uptime Percentage >= 99% and Monthly Uptime Percentage < 99,9%, Service Credit Percentage = 10% If Monthly Uptime Percentage < 99% , Service Credit Percentage = 25%

3.3.2 Google Cloud SLA

Similarly to AWS case, Google presents different SLAs for different services. Two of these services are analysed in this document: Google Compute Engine and App Engine Service, which correspond to one IaaS and one PaaS service offered by the provider.

Google for its Google Compute Engine Service offers a Monthly Uptime Percentage of 99,95%. In this case the Monthly Uptime Percentage Term is defined as "total number of minutes in a month, minus the number of minutes of Downtime suffered from all Downtime Periods in a month, divided by the total number of minutes in a month". Downtime periods for instances consider the "loss of external connectivity and/or persistent disk access for all running Instances that are hosted across two or more

zones combined with the inability to launch replacement Instances in any zone". Meaning that, if the cloud adopter decides not to follow Google's recommendation of deploying applications across multiple zones in a region, Downtime definition is not applicable. A Downtime period is only accounted as such if it is five consecutive minutes of Downtime. The Downtime periods that last less than five consecutive minutes are therefore not accounted in the calculation of Monthly Uptime Percentage. Downtime periods do not account scheduled Downtimes in maintenance windows consultable for the cloud adopters through Admin Console.

In the case that Google does not meet the SLO defined as Monthly update percentage of a 99,95%, cloud adopter is eligible to receive Financial credit only in the case that the cloud adopter requests it. Requests for Financial Credits have to be notified within thirty days and need to provide proofs.

The Financial credits obtained by the cloud adopter are calculated according to the following.

- If Monthly Uptime Percentage < 95%, Financial Credit Percentage = 50%
- If Monthly Uptime Percentage is between 95,00% and 99,00% , Service Credit Percentage = 25%
- If Monthly Uptime Percentage is between 99,00% and 99,95% , Service Credit Percentage = 10%

Financial Credit Percentage is defined as "Percentage of monthly bill for the respective Covered Service which does not meet SLO that will be credited to future monthly bills of Customer". The financial credit is provided in the form of monetary credit for future service usage, and is paid within 60 days. It is established a maximum of 50% of the amount due by the Customer for the service in the applicable month for which the cloud adopter would get Service Financial credit, independently of all Downtime periods on the month.

The SLA does not apply to any: "(a) features designated Alpha or Beta (b) features excluded from the SLA (in the associated Documentation), or (c) errors: (i) caused by factors outside of Google's reasonable control; (ii) that resulted from Customer's software or hardware or third party software or hardware, or both; (iii) that resulted from abuses or other behaviours that violate the Agreement; or (iv) that resulted from quotas applied by the system and/or listed in the Admin Console."

Table 3 Google Compute Terms Summary

Type of Term	Availability
Term	Monthly Uptime Percentage
Value	99,95%.
Calculation	total number of minutes in a month, minus the number of minutes of Downtime suffered from all Downtime Periods in a month[larger than 5 consecutive minutes], divided by the total number of minutes in a month
Guarantee	None provided
Penalties	Yes (not one of excluded cases, after cloud adopter's claim and claim received within 30 days and with an established limit of 50% maximum of customer service's bill)
Penalties Form	Future Services

Penalties Calculation	If Monthly Uptime Percentage < 95%, Financial Credit Percentage = 50% If Monthly Uptime Percentage is between 95,00% and 99,00% , Service Credit Percentage = 25% If Monthly Uptime Percentage is between 99,00% and 99,95% , Service Credit Percentage = 10%

Similarly, for App Engine, Google offers a Monthly Uptime Percentage of a 99,95%. Eligible applications are solely those created by the cloud adopter using High Replication Data Store setting. Downtime in this case is a 10% Error Rate for any Eligible application in periods lasting more than five consecutive minutes. Error Rates are defined in [7] as:

- Serving Infrastructure: Error: HTTP Request sent to App Engine that results in an INTERNAL_SERVING_ERROR; Total calls: The total number of HTTP requests sent to App Engine
- Datastore, Error: Datastore API call returning one of the following errors: INTERNAL_ERROR, TIMEOUT, BIGTABLE_ERROR, COMMITTED_BUT_STILL_APPLYING, TRY_ALTERNATE_BACKEND; Total Calls:The total number of Datastore API calls
- Search, Error:Search API search call returning one of the following errors: TRANSIENT_ERROR, INTERNAL_ERROR; Total calls, The total number of Search API calls

All the rest of considerations are identical to the ones provided for Google Compute Engine Service.

Table 4 Google App Engine Terms Summary

Type of Term	Availability
Term	Monthly Uptime Percentage
Value	99,95%.
Calculation	total number of minutes in a month, minus the number of minutes of Downtime suffered from all Downtime Periods in a month[larger than 5 consecutive minutes], divided by the total number of minutes in a month
Guarantee	None provided
Penalties	Only for Eligible applications[High Replication Data Store] and if not one of excluded cases, after cloud adopter's claim and claim received within 30 days and with an established limit of 50% maximum of customer service's bill
Penalties Form	Future Services
Penalties Calculation	If Monthly Uptime Percentage < 95%, Financial Credit Percentage = 50% If Monthly Uptime Percentage is between 95,00% and 99,00% , Service Credit Percentage = 25% If Monthly Uptime Percentage is between 99,00% and 99,95% , Service Credit Percentage = 10%

3.3.3 Microsoft Azure SLA

Analysis of Microsoft Azure SLA is focused on the Cloud Services and Virtual Machines offerings. Cloud Services is a PaaS environment while Virtual Machines are IaaS Microsoft Azure product.

Cloud Services provide its SLA in terms of "Monthly Uptime Percentage" of a 99,95% [9]. Monthly Uptime is solely defined in terms of External connectivity, and as it is commonly understood, it is defined as the ability of having bi-directional network traffic HTTP/HTTPS sent and received from public IP addresses. According to [8], guarantee is provided in the case of deploying "two or more role instances in different fault and upgrade domains". Monthly Uptime percentage is calculated by the formula $\frac{\text{Maximum Available Minutes}-\text{Downtime}}{\text{Maximum Available Minutes}}$, having Downtime defined in terms of External connectivity only [8].

Similarly to previous public cloud providers analysed, Microsoft Azure offers future Service Credits in the case the SLA levels are not fulfilled. The credits obtained by the cloud adopter are calculated according to the following:

- If Monthly Uptime percentage < 99.95%, Service Credit Percentage = 10%
- If Monthly Uptime percentage < 99%, Service Credit Percentage = 25%

For a cloud adopter to get these service credits, claim has to be submitted within two months after the end of the billing period, indicating information about the incident. Microsoft validates and determines whether it is applicable to get service claims.

SLA document excludes (SLA Exclusions) any performance or availability issues: i. due to factors outside Microsoft's reasonable control; ii. that resulted from Customer's use of hardware, software, or services not provided by Microsoft as part of the Services; iii. due to Customer's use of the Service in a manner inconsistent with the features and functionality of the Service or inconsistent with Microsoft's published documentation or guidance; iv. that resulted from faulty input, instructions, or arguments (for example, requests to access files that do not exist); v. caused by Customer's use of the Service after Microsoft advised Customer to modify its use of the Service, and the latter did not comply with the advice; vi. during or with respect to Previews (as determined by Microsoft) or to purchases made using Microsoft subscription credits; vii. that resulted from Customer's attempts to perform operations that exceed prescribed quotas or that resulted from Microsoft's throttling of suspected abusive behaviour; viii. due to Customer's use of Service features that are outside of associated Support Windows; or ix. attributable to acts by persons gaining unauthorized access to Microsoft's Service by means of Customer's passwords or equipment or otherwise resulting from Customer's failure to follow appropriate security practices."

Table 5 Microsoft Azure Cloud services Terms Summary

Type of Term	Accessibility / External Connectivity
Term	Monthly Uptime percentage
Value	99,95%
Calculation	$\frac{\text{Maximum Available Minutes}-\text{Downtime}}{\text{Maximum Available Minutes}}$ having Downtime defined in terms of External connectivity only
Guarantee	-

Penalties	Only for Eligible applications [two or more role instances in different fault and upgrade domains] after cloud adopter's claim and claim received within 2 months and after claim accepted by Microsoft
Penalties Form	Future Credit Services
Penalties Calculation	If Monthly Uptime percentage < 99.95%, Service Credit Percentage = 10% If Monthly Uptime percentage < 99%, Service Credit Percentage = 25%

For Virtual machine Services, the only definition that changes comparing to the previous case is SLA applicability, which is described as "all Internet facing Virtual Machines that have two or more instances deployed in the same Availability Set" [9]. Recommendations for availability management provided by Microsoft state to group two or more VMs in an Availability set. Each VM in an availability set in Azure platform is assigned to a different update and fault domain, helping to support availability in planned maintenance events, and migration of VMs across physical machines in case of unplanned maintenance events ("local network failures, local disk failures, or other rack level failure"). Fault Domains express a group of virtual machines sharing a common power source and network switch. Virtual machines configured within an Availability Set are by default split across Fault Domains.

Table 6 Microsoft Azure Virtual Machines Terms Summary

Type of Term	Accessibility / External Connectivity
Term	Monthly Uptime percentage
Value	99,95%
Calculation	$\frac{\text{Maximum Available Minutes}-\text{Downtime}}{\text{Maximum Available Minutes}}$ having Downtime defined in terms of External connectivity only
Guarantee	-
Penalties	Only for Eligible applications [have two or more instances deployed in the same Availability Set] after cloud adopter's claim and claim received within 2 months days and after claim accepted by Microsoft
Penalties Form	Future Credit Services
Penalties Calculation	If Monthly Uptime percentage < 99.95%, Service Credit Percentage = 10% If Monthly Uptime percentage < 99%, Service Credit Percentage = 25%

3.3.4 Pivotal Web Services SLA

Pivotal Web Service (PWS) offers an instance of Pivotal Cloud Foundry hosted on AWS. There is no PWS SLA but its "Terms of Service" are described in [11]. These Terms of Service do not refer to specific characteristics of the provided service, such as availability, but to the characteristics of the service itself. An aspect that it is important to remark from this analysis is that terms referring to Data and Content disclosure are already refereeing to European Commission Standard Contractual Clauses [12]. However, as it can be observed in the quoted text below, not concrete commitments or terms are used: "In case of transfer of your personal data as defined by applicable data privacy laws to outside of the European

Economic Area ("EEA") or access of such data from outside the EEA we will apply reasonable legal protection. Such protection may for example include the implementation of European Commission ("EU") Standard Contractual Clauses as published by the EU Commission or our membership to Safe Harbor, an agreement between the U.S. Department of Commerce and the EU Commission for transfers from the EEA to the United States."

3.3.5 Heroku SLA

Similarly to PWS, Heroku does not specify any SLA in any manner than the cloud adopter can get specific assurance for the Service Terms. Instead, Heroku provides a "Customer Promises" page [13] that states a set of good intentions referred to the Service Provision. "You own your code, not us; You own your data, not us; We won't lock you in (your business is our privilege, not our right); We'll do everything we can to achieve 100% uptime; We will never achieve 100% uptime, but when we fall short, we'll explain why and how we'll do better next time"

3.4 Cloud market status snapshot

The analysis provided in sections above has studied nine (9) different cloud services from five (5) commercial cloud providers that can be structured as follows:

Table 7 Cloud market status snapshot

Service Type	Service	Vendor	SLA
IaaS	Amazon EC2 and EBS	AWS	Yes
	Amazon S3	AWS	Yes
	Google Compute	Google	Yes
	Azure Virtual Machines	Microsoft	Yes
PaaS	Google App Engine	Google	Yes
	Azure Virtual Cloud Services	Microsoft	Yes
	Pivotal Web Services	Pivotal	No
	Heroku	Heroku	No

Of these 100% of IaaS offerings presented some kind of SLA while for PaaS offerings only a 50% of them. Notably, these corresponded to market leaders in the field.

Independently of the typology of Cloud offering, all services considering SLAs focus on Availability terms and concretely on some kind of Monthly Uptime percentage. However, definitions of what is Uptime or Downtime significantly differs among providers, and often impose configurations and conditions to the deployment, such as the use of more than one availability zone, which often are specified in the technical documentation as "recommendations" but not giving the clear understanding that not following them implies not to get SLA guarantees.

In a similar manner, all the existing SLAs promise Service credits in case of the provider not fulfilling the agreed Monthly Uptime percentage, but again, mechanisms and procedures to get them, require cloud adopters' claims with different conditions, timeframes, and documentations among providers. In some

specific cases, these require approval or validation by the provider in which the cloud adopter has to trust blindly, given that monitoring services, if any, are provided by the same actor.

Tables below summarize SLA terms and conditions find out in this analysis across different provides per cloud type of service.

Table 8 Comparison of SLA Terms among IaaS Services

Vendor	**AWS**	**AWS**	**Google**	**Microsoft**
Service	**Amazon EC2 and EBS**	**Amazon S3**	**Google Compute**	**Azure Virtual Machines**
Type of Term	Availability	Availability	Availability	Accessibility / External Connectivity
Term	Monthly Uptime Percentage	Monthly Uptime Percentage	Monthly Uptime Percentage	Monthly Uptime %
Value	99,5%	99,9%	99,95%	99,95%
Calculation	100% - percentage of minutes during the month in which applicable region was in state “Region Unavailable”	100% - average of the Error Rates from each five minute period in the monthly billing cycle	total number of minutes in a month, minus the number of minutes of Downtime suffered from all Downtime Periods in a month[larger than 5 consecutive minutes], divided by the total number of minutes in a month	(Maximum Available Minutes – Downtime)/ (Maximum Available Minutes) having Downtime defined in terms of External connectivity only
Guarantee	Not clearly defined, stated as “commercially reasonable efforts”	Not clearly defined, stated as “commercially reasonable efforts”	None provided	-
Penalties	Yes (if confirmed by AWS, not one of excluded cases, after cloud adopter’s	Yes (if confirmed by AWS, not one of excluded cases, after cloud adopter’s claim and claim received by the end of second	Yes (not one of excluded cases, after cloud adopter’s claim and claim received within	Only for Eligible applications [have two or more instances deployed in the same Availability Set]

	claim and claim received by the end of second billing cycle after the incident)	billing cycle after the incident)	30 days and with a established limit of 50% maximum of customer service's bill)	after cloud adopter's claim and claim received within 2 months days and after claim accepted by Microsoft
Penalties Form	Future Service Credits	Future Service Credits	Future Services	Future Credit Services
Penalties Calculation	If Monthly Uptime Percentage between 99.95% and 99,0%, Service Credit Percentage =10% If Monthly Uptime Percentage<99, 0% , Service Credit Percentage =30%	If Monthly Uptime Percentage >=99% and Monthly Uptime Percentage<99,9%, Service Credit Percentage =10% If Monthly Uptime Percentage<99% , Service Credit Percentage =25%	If Monthly Uptime Percentage<95%, Financial Credit Percentage =50% If Monthly Uptime Percentage between 95,00% and 99,00% , Service Credit Percentage =25% If Monthly Uptime Percentage between 99,00% and 99,95 , Service Credit Percentage =10%	If Monthly Uptime % <99.95%, Service Credit Percentage =10% If Monthly Uptime % <99%, Service Credit Percentage =25%

Table 9 Comparison of SLA Terms among PaaS Services

Vendor	**Google**	**Microsoft**	**Pivotal**	**Heroku**
Service	**Google App Engine**	**Azure Cloud Services**	**Pivotal Web Services**	**Heroku**
Type of Term	Availability	Accessibility / External Connectivity	-	-
Term	Monthly Uptime Percentage	Monthly Uptime %		
Value	99,95%	99,95%		
Calculation	total number of minutes in a month, minus the number of minutes of Downtime suffered from all Downtime Periods in a month[larger than 5 consecutive minutes], divided by the total	$\frac{\text{Maximum Available Minutes}-\text{Downtime}}{\text{Maximum Available Minutes}}$ having Downtime defined in terms of External connectivity only		

	number of minutes in a month			
Guarantee	None provided	-		
Penalties	Only for Eligible applications[High Replication Data Store] and if not one of excluded cases, after cloud adopter's claim and claim received within 30 days and with an established limit of 50% maximum of customer service's bill	Only for Eligible applications [two or more role instances in different fault and upgrade domains] after cloud adopter's claim and claim received within 2 months days and after claim accepted by Microsoft		
Penalties Form	Future Services	Future Credit Services		
Penalties Calculation	If Monthly Uptime Percentage<95%, Financial Credit Percentage =50% If Monthly Uptime Percentage between 95,00% and 99,00% , Service Credit Percentage =25% If Monthly Uptime Percentage between 99,00% and 99,95 , Service Credit Percentage =10%	If Monthly Uptime % <99.95%, Service Credit Percentage =10% If Monthly Uptime % <99%, Service Credit Percentage =25%		

4 SLA specifications and management mechanisms research outcomes

This chapter provides an overview of the SLA specifications that have been developed in various research projects and initiatives. Firstly, the most innovative metrics that have been proposed by these research projects are explained, as for how they should be included in the SLA and their different formats. Afterwards, for the various phases of the lifecycle of a SLA, the multiple developed innovative approaches and the SLA management mechanisms are described for each of these research projects, in deep details.

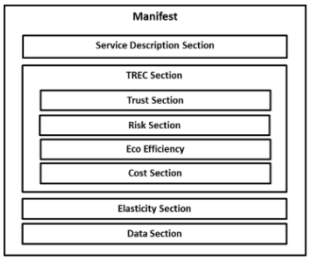
4.1 SLA specifications

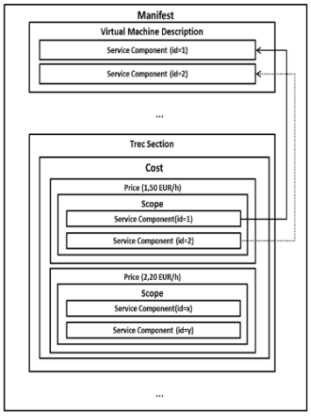
In the area of SLA specifications and term languages, various innovative approaches have been developed such as the service manifest and the TREC factors in OPTIMIS, the blueprint in 4CaaSt, the quality model in CONTRAIL, the QoS-oriented specification in Q-ImPrESS, the interoperability in Helix Nebula, the QoE and QoB policies of SeaClouds, the service description in SLA@SOI, and the virtualised service network in IRMOS. Following, the aforementioned innovative approaches are described thoroughly.

4.1.1 OPTIMIS

The European project OPTIMIS [14], [15] is providing a toolkit enabling service and infrastructure providers to achieve a defined QoS by optimising their respective services in a multi cloud environment. One aspect of the project's research and development is defining the parameters of the services' quality that are the targets for optimisation. Another one is referring to a framework that specifies the parameters in a service manifest, representing the service requirements and evaluating the state of these parameters during the services' lifetime through a SLA.

Therefore, ***service manifest*** [16] was developed as an innovative approach in the area of SLA specifications, considering to be a *term language*, enabling the description of the requirements of the service provider for an infrastructure service provisioning process. It captures both the *functional* and *non-functional parameters* of the service and allows the specification of the (atomic) components of an application (e.g. a web-application may require a web server, an application server and a database server to run). For each component, the corresponding parameters are captured along with the constraints for each one, and the KPIs that will be monitored, while affinity and anti-affinity rules can also be specified per component. The developed service manifest by OPTIMIS consists of





different elements: the *common core service manifest*, the *service provider extensions*, and the *infrastructure provider extensions*. With respect to infrastructure services, the manifest may describe VM images (OVF) and may also include OVF definitions of data location constraints, data protection, and elasticity. Furthermore, the manifest may include *legal terms* such as *Intellectual Property Rights (IPRs)*, standard *contractual clauses* or *binding corporate rules* [15], [17]. To this end, OPTIMIS has highlighted how IPR categories can be exploited during automated SLA negotiation [18]. Besides the manifest, OPTIMIS has developed an API to develop, import and export the service manifest, refine service and infrastructure providers' extensions, and split it if needed since multiple services may be described in a single document. The included figure represents the service manifest structure in a high level overview, accompanied with an example of a service component reference [16].

Additionally, OPTIMIS pursued the optimization of the management of relationships in a secure cloud ecosystem based on the TREC factors (i.e. Trust, Risk, Eco, and Cost). OPTIMIS envisaged infrastructure providers and service providers to be able to self-evaluate their own status, controlling and maximizing reliability, cost and energy expenditures, by providing a holistic approach to perform all management actions (SLA enforcement, elasticity enactment, services consolidation, data placement, replication, etc.) harmonized by overarching policies that will consider trust management and risk assessment to comply with legal, economic and ecological objectives without compromising operational efficiencies. In order to do so, different parameters were considered for each TREC factor and legal constraints that were expressed in a standard WS-Agreement template, which are described below:

- **Trust**: OPTIMIS assessed trust by using reputation mechanisms, according to which the rank of infrastructure providers was based on how well they accomplish promised level of service, while service providers will be ranked according to e.g., the fluctuations in their capacity requirements and their willingness to commit to long term relationships. For that reason the *metrics* that were used were *trust level*, *robustness*, *reliability*, *performance*, *latency/response time*, *network throughput* and *security level*.
- **Risk**: As for the risk factor that corresponded to unsafe events which potentially had negative impact on the provision of functionality, OPTIMIS defined the *metrics* of *probability of failure (PoF), risk impact level* and *risk level*.
- **Eco**: For the part of Eco-efficiency, OPTIMIS enabled to specify and enforce power consumption limits in SLAs, so to decide where services were to execute based on electricity prices, as well as to monitor and assess ecological factors in running services. More specifically, the *metrics* that were used to support this factor were the e*nergy* used for task or resource, *CO2* per task or resource, *annualized average PUE, compliance* to EU code of conduct, *energy star* for datacenter rating, and *LEED* for data center.
- **Cost**: As for the cost part, OPTIMIS tried to reduce the cost in SLAs, by adopting the *Unified Service Description Language (USDL) Pricing Module*, referring to the PricePlan, the PriceComponent, the PriceLevel, and the PriceMetric.

Finally, OPTIMIS [95] in order to ensure confidentiality, integrity, authenticity and availability of the data in accordance with the applicable European data protection directive, conducted an SLA, giving

significant importance to the parts of data destruction, data loss, data alteration, data disclosure, and data access.

4.1.2 4CaaSt

The 4CaaSt project [19] aims to create a PaaS Cloud platform [20], [21] which supports the optimized and elastic hosting of Internet-scale multi-tier applications. It enables flexible definition, marketing, deployment and management of Cloud-based services and applications creating a true business ecosystem, where applications from different providers can be tailored to different cloud adopters, mashed up and traded together [22], [23]. The approaches in 4CaaSt are based on the introduced concept of "products", which refer to service offerings - atomic or composite ones - of any type (i.e. "X-as-a-Service"). In the case of composite services, what is of major importance is the *definition of the dependencies* between the atomic services. To this end, 4CaaSt has developed a description language capturing the service dependencies within and across the cloud layers, resulting to a descriptive document - the so called "blueprint" [24].

The innovative concept of ***blueprint*** is an abstract description of an application or service that decouples what is offered from the resources required from the various layers of the Cloud stack [25]. It enables the definition of *provisioning* and *management rules* with respect to elasticity and multi-tenancy, as well as the inclusion of "hints" from the application developer in order to map high-level application terms into low-level resource parameters. What is more, the blueprint encompasses information with respect both to the technical requirements of a product and to the *business aspects / terms* of such a service offering. The latter is a unique contribution from 4CaaSt, since the use of the eMartkerplace (described in section 4.2.2.1) allows for the optimum identification and selection of the technical terms that should be attached to a service offering through an SLA. Taking into consideration that there is a great degree of flexibility in the application and technical terms, as defined by the application developers (e.g. range of values in a specific parameter), business criteria and simulation aim at *identifying the optimum terms and the corresponding values* for these terms.

4.1.3 CONTRAIL

The CONTRAIL project [26] aims at defining and implementing a framework for Cloud Federations, to relieve the cloud adopter from managing the access to individual CSPs. Thus, its main objective is to offer elastic PaaS services over a federation of IaaS Clouds, while dealing with pertinent issues related to QoS, SLA management, security, interoperability and scalability [27], [28].

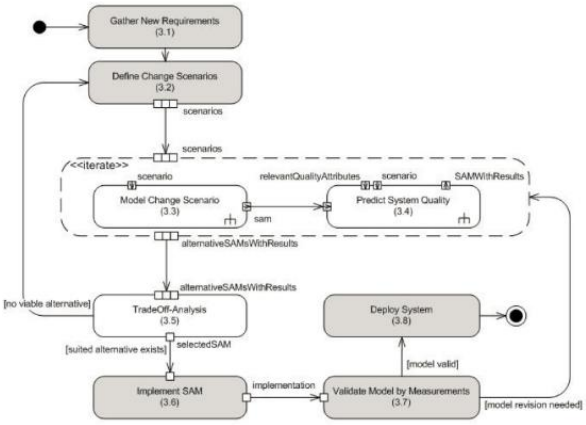
The project has developed an innovative ***quality model*** for capturing different parameters of interest both for customers and providers. Within the quality model, *terms* have been classified to: *unobservable*, *observable*, *enforceable* as well as to: *static* or *dynamic* (regarding their evolution in time). The quality model has been used to develop an SLA specification that reflects either *generic SLAs* (i.e. parameters applicable to any resource) or *specific SLAs* (i.e. parameters applicable to specific OVF resources). Besides QoS terms and advance reservation, the SLA specification includes the so called

Quality of Protection (QoP) terms, such as data locality, protection, replication, etc. and it may also be linked with different pricing models for generating automatic quotations.

4.1.4 Q-ImPrESS

The Q-ImPrESS project [29] has developed a method for quality-driven software development and evolution, where the consequences of design decisions and system resource changes on performance, reliability and maintainability, can be foreseen through quality impact analysis and simulation. It aims at bringing service orientation to critical application domains, such as industrial production control, telecommunication and critical enterprise applications, where guaranteed end-to-end quality of service is particularly important [30].

One of the innovative approaches that was developed in the area of SLA specifications and term languages was the approach of ***QoS-oriented SLA Specification***. To be more specific, focusing on how different QoS / SLA parameters can be captured, Q-ImPrESS has developed a *metamodel* (namely Service Architecture Meta Model – SAMM [31]) that allows the *definition of service attributes,* while it has to fulfil a set of requirements from the perspective of the Q-ImPrESS consortium (i.e. prediction methods, trade-off analysis, multiple views, graphical editors, IDE integration, types of evolution scenarios, model size). *Parameterised definitions* (e.g. data volume, configuration, execution environment) are feasible in SAMM. *Parametric dependencies* can be included in the model, since service architectures may include more than one services (as composite services) and face varying usage contexts. The dependencies are exploited to link different SLAs and incorporate the corresponding relationship structure among service SLAs. The included figure presents a coarse-grained overview on the recommended workflow for using the Q-ImPrESS approach [30].



4.1.5 SLA@SOI

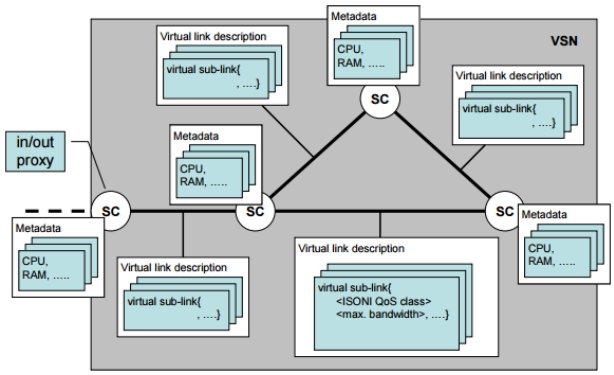
The main objective of the SLA@SOI project has been dependable cloud computing through SLAs [32]. The developed open-source SLA@SOI framework addresses the complete service lifecycle through autonomous negotiation, provisioning, monitoring and adaptation of SLAs, while also dealing with the entire service stack, from business aspects through to the physical infrastructure.

One of the innovative concepts that was developed by the project was the ***Service Description***, in which a *model, namely SLA(T)*, is used for the description of both functional and non-functional characteristics of a service. The aforementioned model, serves both the means by which a provider describes the services they offer, and the means by which customers can describe their requirements and discover

"matching" offers [33]. The model is based on *vocabularies* (e.g. for QoS metrics or constraints) and implemented as an *abstract syntax* as it still leaves various implementation details unspecified. Moreover, it is open to instantiation by various concrete syntactic forms that can be instantiated, in whole or in part, by an appropriate concrete syntactic format (e.g. XML, OWL, or human-readable formats), thus being *language and technology independent*. The developed model follows a *hierarchical approach*, being applicable to SLA templates (forming a generic customisable base) and SLAs (having the same basic structure but being non-customisable).

4.1.6 IRMOS

The IRMOS project [34] developed cloud solutions that allow the *adoption* of interactive *real-time applications*, enabling their rich set of attributes (from time-constrained operation to dynamic service control and adaptation) and their efficient integration into cloud infrastructures [35], [36].



IRMOS incorporates ISONI, an Intelligent Service Oriented Network Infrastructure that provides ***Virtual Service Networks*** by virtualizing computing, networking and storage resources [37]. It is considered as an IaaS environment consisting of a network of resources (e.g. CPU, storage, software) managed and controlled by a middleware which allows resource sharing among multiple services. The main concept is to provide *QoS capable infrastructure resources* on demand for dynamically deployed services [38]. The basic purpose of the ISONI is to reduce the complexity for service providers/developers to roll out new network based services. It does this by undertaking the automatic deployment of the services on best fitting resources distributed in a network. The solution strives to reduce global costs by introducing a resource provider in the value chain that optimizes costs by means of *virtualization techniques*, whereby tailored resources can be provided for the deployment of services. Additionally, the ISONI provides means to isolate different deployed services from each other in order to prevent unwanted crosstalk between them. For that reason, the ISONI has to carry out several tasks (i.e. manage of all the hardware resources distributed in a network from that of deployed services and their associated service components, deploy and instantiate the service developers' service on the ISONI, monitor running services and their resource usage). In order to deploy a service, the service developer has to attach an abstract description of the service's requirements when transferring his service to the ISONI. The included figure describes the Virtual Service Network which can be seen as a graph whose vertices represent the Service Components and whose edges represent the Virtual Links [39].

4.1.7 Helix Nebula

The Helix Nebula project [91] aims at establishing a multi-tenant, multi-provider cloud infrastructure, while identifying and adopting policies for trust, security and privacy, and introducing a governance structure and the related potential funding schemes.

It considers interoperability aspects in five (5) levels: a) political context, b) legal interoperability, c) organizational interoperability, d) semantic interoperability and e) technical interoperability. In order to achieve integration and interoperation of emerging and mature e-Infrastructures, the project analyses the technology and policy interfaces related to each infrastructure. Political context may refer to fragmentation due to different national legal frameworks, contract issues related to data access, portability, control and ownership; and the uncertainty coming from different standards with respect to the interoperability system, application and data formats to permit portability. Legal interoperability refers to a common set of terms and conditions among the contracts (both for terms and conditions and SLAs) and a common policy that protects cloud adopters' IPR. Organizational interoperability deals with shared services among federated environments including a minimal set of requirements for IT management, common set of service management structures and an agreed business model. Semantic interoperability involves a compatible scheme to describe elements of a service catalogue, which eases service selection across different providers and compatible accounting and billing parameters, cross settlements between providers and single integrated billing towards the customer. Finally, technical interoperability refers to the APIs and the Web Portal among the federated environments (e.g. a Single Sign-on authentication mechanism).

4.1.8 CELAR

CELAR project [92] deals with the "Automatic, multi-grained elasticity-provisioning for the Cloud". To this end, the project considers use cases that require the automatic reservation of the right type and amount of cloud resources that meet the needs of the application based on the cloud adopter's policies. The elastic resource allocation needs to be in a completely transparent manner, i.e., the cloud adopters and owners of the applications should be able to perceive the performance and corresponding costs of their applications to vary, at all times, within the limits they defined when submitting the application and the CELAR system to adaptively add and remove resources in real time, so that the perceived behavior range is always bound by the cloud adopter's requirements. To this purpose, the "Decide elastic operation" transforms a higher-level "add resource" or "remove resource" command into the reservation and deployment of the requested resources, along with the installation or configuration of the required software libraries, according to the software type.

Moreover, the application cloud adopters need to be able to run their application on another Cloud provider, because e.g., that provider became cheaper. Or they can choose to re-deploy their application on the same provider but with different structure details in case cloud adopters' demands have changed. Thus, they are also able to change the elasticity requirements and capabilities governing their application behavior, and to dynamically alter the important metrics that will be reported to them.

In particular, an application cloud adopter can either specify just the elasticity requirements that must be met by the application deployment, or he can also specify detailed elasticity capabilities. The elasticity capabilities for an application include the elasticity actions (scaling/reconfiguration actions) and the application level at which they must be applied for elastically adapting the application at runtime based on the specified elasticity requirements. The two types of elasticity requirements are:

- **Cost-related** elasticity requirements: An Application Owner may specify that when the total cost is higher than 800 Euro for a number of clients, there should be a scale-in action for keeping costs within acceptable limits.
- **Quality-related** elasticity requirements: An Application cloud adopter may need to monitor different quality parameters (e.g., response time, data accuracy, etc.), which should be within acceptable limits.

4.1.9 PaaSage

PaaSage project [93] uses SLAs in the life-cycle phases of configuration and deployment for finding the most suitable deployment model for multi-cloud applications to maintain the QoS throughout their execution. In particular, the configuration phase, apart from modelling the deployment of applications and specifying requirements and goals, it also involves the specification of the SLAs particularly focusing on non-functional characteristics such as performance and security policy. The deployment phase is concerned with matching the configuration models of applications with the profile of cloud providers, and selecting the most suitable deployment models based on requirements and goals, SLAs, and historical data about the executions of applications. This may involve a) specification of constraints on the location of the Cloud provider, e.g., to respect legal constraints on the location of data; b) the specification of requirements on the security and privacy of the Cloud provider infrastructure; and c) independent - from the specifics of each Cloud provider - specification of the required resource types, such as requesting Cloud storage resources in the form of a file system or a database.

For this purpose, the project adopts Web Services Agreement (WS-Agreement) standard [94] from the Open Grid Forum (OGF).

4.1.10 SeaClouds

SeaClouds [96], [97] project manages two levels of SLAs for management of business oriented policies depending on the involved parties in the agreement: a) customer and application provider SLA and b) application and cloud provider SLA. While the first one describes the Service Offered by Application provider to its cloud adopters, the latter describes the service offered by the cloud provider to the application provider.

Guarantee *terms* in customer and application provider SLA considers only observable metrics by the customer including *Quality of Experience (QoE)* in application's usage, which are availability and response time. In addition, the application provider offers the capability to add business values to

guarantee terms, so to describe the penalties that the application provider will have to deal with if the term is violated, commonly expressed in terms of service credits. As for the *terms* of the application and cloud Provider, the SLA describe *QoS* characteristics of the service offered. What is more, in SeaClouds the cloud adopter requirements are expressed in the TOSCA AAM, which is translated into a WS-Agreement template. In SeaClouds, the agreements contain *Quality of Business (QoB)* policies specified by the application designer, but not specified in a cloud provider template. It has to be mentioned that a QoB rule is defined as a constraint over a metric provided by monitoring sensors, where when a non-fulfilled constraint raises a QoS violation, business action has to take place for a specific amount of time.

4.2 SLA management mechanisms

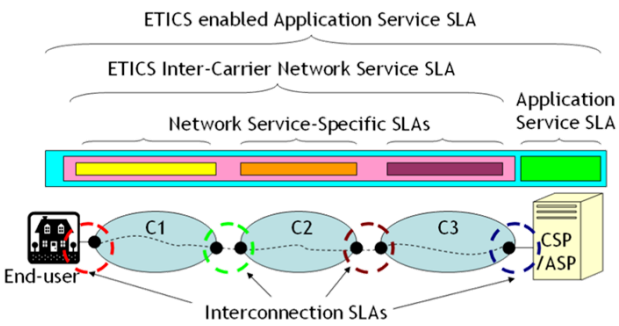
In the area of SLA management mechanisms, various components have been introduced to enhance the SLA management process and aid the corresponding individual phases of its lifecycle (service modelling, SLA template definition, SLA instantiation and management, SLA enforcement [40]) such as interesting works regarding performance estimation and workload prediction, enhancements for trade-off analysis, scalable and efficient monitoring tools, and novel negotiation approaches enabling dynamicity, automation, scalability and re-negotiation during runtime. Following, all the innovative components that have been developed among the various research projects, are described thoroughly.

4.2.1 Service Modelling

The service modelling process aims at *providing additional information* with respect to the service that will be deployed in a cloud infrastructure. As the only actor having the required knowledge for the service, the developer is using a set of frameworks in order to design, model and analyse the service. Service design may be extended to include potential dependencies between service components of an application, elasticity rules for the application or / and performance and behaviour hints that are required to guarantee the offered level of quality.

4.2.1.1 ETICS

ETICS [41] has delivered new network control, management and service plane technologies for the automated end-to-end QoS-enabled service delivery across network service providers allowing for a fair distribution of revenue shares among all the actors of the service delivery value-chain.



More specifically, the project has considered the case of e*nd-to-end, QoS-enabled application services* resulting from the *composition of atomic services* being offered by different providers (e.g. application / content and network providers) in application and network domains. To support the provision of such composite services, a *hierarchy of SLAs* has been defined in reference to the

different composition layers. At the atomic service layer, the interconnections between the network providers, between application and network provider and between end-user and network provider are characterized by *static SLAs*, while the intra-domain network services offered by the each network provider follow a *dynamic and per-service* paradigm. The composition of the SLAs related to the network intra-domain services and interconnections, results in the SLA for the end-to-end, inter-carrier network service that, in turn, can be further aggregated with the SLA for the atomic application service [42]. The final resulting SLA on top of this hierarchy will deal with the end-to-end, QoS-enabled and network-guaranteed application service. Depending on the service chain, the SLAs for composite services consider as providers, either network providers or application providers and as customers, either application providers or end-users. The latter highlights the fact that composition always follows a provider – customer scheme but the customer in some cases may be another provider.

Furthermore, the project has contributed towards the identification and realization of different *SLA composition paradigms*. SLA composition may be *centralized* (i.e. a unique entity such as an independent broker or origin domain acts as mediator and manages the SLA with all the domains) or *distributed* (i.e. consecutive SLA establishments on each provider-customer pair following either a cascade model - from origin to destination, or a reverse cascade model - from destination to origin [43]).

4.2.1.2 Cloud-TM

Cloud-TM [44] develops a data-centric PaaS layered on top of a self-optimizing, highly scalable distributed Transactional Memory platform. Cloud-TM allows the reducing of the development and operational costs of cloud-based applications in a twofold way: i) hiding complexity by providing programmes with intuitive abstractions that encapsulate innovative data management protocols designed from scratch to meet the requirements of large-scale elastic cloud platforms; ii) via pervasive self-tuning strategies that automate the resource provisioning process [45], [46] and transparently reconfigure the data management mechanisms (e.g. consistency protocols [47], [48], [49], data placement [50], replication degree [51], [52]) based on user-specified QoS/cost constraints [53].

The provision of both the initially required resources to services (during deployment) and the additional resources (during runtime), require for mechanisms that deliver *performance estimates* [54], [46], [50], [52] and *workload predictions* [53] in order to identify the optimum resources and deployment patterns. Such mechanisms have been developed by Cloud-TM, enabling the prediction of applications' performance when deployed over transactional platforms of different scale as well as the workload prediction of the transactional application independently from the scale of the system, the capacity of the platform (e.g. CPU speed), the data management scheme and the algorithm used by the transactional data platform on which the application is deployed.

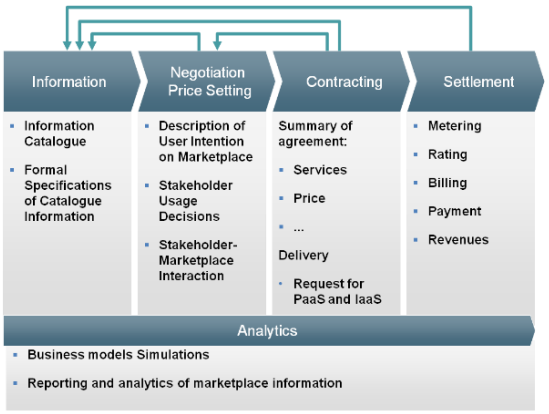
4.2.2 SLA Template Definition

The SLA template definition process aims at *generating and refining the SLA templates*. All providers (i.e. service, platform and infrastructure) analyse their business objectives through a business modelling process in order to optimize their offerings. Furthermore, the service provider uses as a basis the

produced blueprint / manifest of the service and refines the SLA templates (in terms of attributes values) following business modelling outcomes, while the service provider may also include additional attributes in the SLA templates reflecting for example the use of licenses. Thus, the created SLA template may include the outcomes of one or more service blueprints / manifests.

4.2.2.1 4CaaSt

The 4CaaSt project [19] aims to create a PaaS Cloud platform [20], [21] which supports the optimized and elastic hosting of Internet-scale multi-tier applications, as mentioned in subsection 4.1.2.



The 4CaaSt project has implemented an *eMarketplace* framework [55] that deals with the *business and pricing aspects* of service offerings [56]. It enables trading of any type of cloud services, including *composite services* that consist of atomic services offered by different providers. Furthermore, the eMarketplace is enriched with a business model simulation tool supporting the service providers during the identification and definition of complex pricing and business models. Through its *business resolution* feature [57], it exploits the experience of end-users and customers and proposes business offering which effectively cover the needs of each particular request from a pool of technically valid solutions. Based on the above, the eMarketplace could be considered as a supporting environment during the definition of SLA templates.

4.2.2.2 ETICS

As mentioned in section 4.2.1.1, ETICS [41] has delivered new network control, management and service plane technologies for the automated end-to-end QoS-enabled service delivery across network service providers.

The project has proposed an approach for the flexible *integration of business aspects* in the SLA lifecycle [58]. To this end, a SLA template has been developed which is flexible in terms of different *business or charging models*, while meeting general requirements on domain confidentiality and technology heterogeneity. The main components of the SLA template refer to the entities (i.e. customer, providers, or brokers) identification, the service description (i.e. technology agnostic description of service attributes), the business aspects (i.e. price, administrative / legal details, and procedures for handling service modification / violation / termination cases) and the technical aspects (i.e. QoS parameters).

4.2.2.3 Q-ImPrESS

As mentioned in section 4.1.4, the Q-ImPrESS project [59] has developed a method for quality-driven software development and evolution, where the consequences of design decisions and system resource

changes on performance, reliability and maintainability can be foreseen through quality impact analysis and simulation.

One of the challenges in cloud environments refers to the way different application characteristics (in the case of Q-ImPrESS expressed in SAMM) and providers' policies affect the QoS level of the service and the resource provisioning decisions [60]. To address this challenge, the project has developed a trade-off analysis framework that concludes on the *effect of different QoS attributes*, while considering *different states* (i.e. target and as-is) during the service lifecycle [61]. Analysis also aims at *estimating performance metrics*, *accounting for propagation effects* across systems, and *assessing the risk for potential SLA violations* in order to propose design alternatives [62]. The outcome of these frameworks is a SLA prediction in terms of balanced SLA quality dimensions (as reflected in different attributes / parameters).

4.2.3 SLA Instantiation and Management

The goal of this phase is to instantiate a SLA (i.e. electronically signed agreement). The main process refers to the *SLA negotiation*, which may be extended with mechanisms for dynamic negotiation between different entities as well as with mechanisms for automatic re-negotiation during runtime. Moreover, discovery is used to identify providers for specific services (based on the service parameters captured in the service blueprint / manifest), having finally as a result a signed *SLA between the participating entities*.

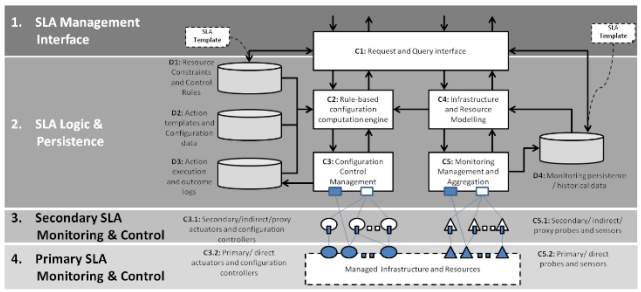
4.2.3.1 CONTRAIL

As mentioned in section 4.1.3, the main objective of CONTRAIL is to offer elastic PaaS services over a federation of IaaS Clouds, while dealing with pertinent issues related to QoS, SLA management, security, interoperability and scalability. In the CONTRAIL vision, small cloud providers can join forces into a cloud federation to stand the competition of bigger players and raise at a worldwide level the competitiveness of the European Cloud market [28].

For that reason, enhanced mechanisms in different phases of the SLA lifecycle have been developed by CONTRAIL to support SLA management for cloud federations. Regarding negotiation, a system has been implemented (based on the SLA@SOI framework) to realize the *federated negotiation with multiple providers* and the *selection of the optimum SLA offer* according to cloud adopter criteria. In this case the CONTRAIL system acts as a cloud broker, realizing the Service Arbitrage model described by NIST in its cloud computing reference architecture [63]. During service execution / usage, CONTRAIL will allow application distribution over multiple providers (thus enabling the execution of composite applications), while cross-*provider enforcement strategies* will be exploited to minimize SLA violations.

4.2.3.2 GEYSERS

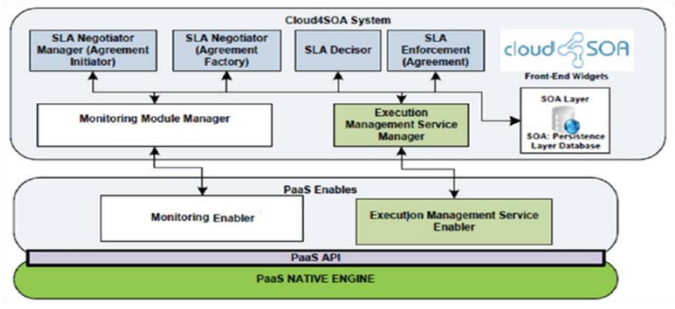
The project [64] as delivered mechanisms for seamless and coordinated provisioning of networking and IT resources, end-to-end service delivery to overcome limitations of network domain segmentation, business models analysis through a business framework and composition of logical infrastructures following the partitioning of infrastructure resources.



More specifically, composed virtual infrastructures [65] aim at enabling dynamic *service provisioning on top of network and IT resources*, encompassing several layers and resource types while dealing with the constraints and dependencies of the resources and the services as well as the dependencies between application deployment and usage [66]. GEYSERS considers the *autonomy of physical and virtual providers* as well as virtual operators, and their respective management domains (independent control of policies and operational objectives). The converged SLA management framework proposed and implemented in GEYSERS [67] aims to *handle the dependencies between physical and virtual resources* in both network and IT domains, and allow for cross-layer handling of events and alerts that may affect the service provision on top of the virtual infrastructure and their SLA lifecycle. The converged SLA management framework [68], [69] implements *different strategies*: *bottom-up* (i.e. initiated by the lower-layer physical infrastructure providers), *top-down or "truly on demand"* (i.e. initiated by customers and service consumers), *mixed* (i.e. combined message exchanges to reach mutually-agreed SLA).

4.2.3.3 Cloud4SOA

The project [70] empowers a multi-cloud paradigm at PaaS level, providing an interoperable framework for PaaS developers. The system supports Cloud-based application developers with multiplatform matchmaking, management, unified application and cloud monitoring and migration. It interconnects heterogeneous PaaS offerings across different providers that share the same technology through the concept of adapter that provides a REST-based API for any-platform access.



Support for on-demand based business models is amongst the requirements of cloud providers. To this end, dynamicity needs to be embedded in the SLA lifecycle in order to support business dynamics and changing customer needs (e.g. redefine specific parameters). Cloud4SOA has developed a framework enabling *dynamic SLA negotiation* and tools that enable PaaS providers to analyse their offerings and performance and adapt the SLAs accordingly. The framework allows providers and customers to *negotiate flexibly between standard and*

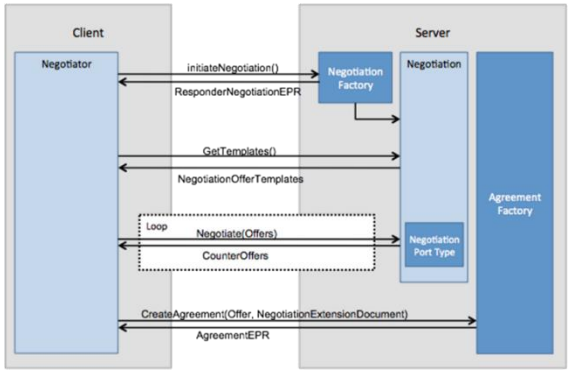
customized SLAs, while supporting business dynamics through *business-performance related SLA metrics* being monitored and analysed.

Cloud4SOA provides a RESTful implementation of the WS-Agreement standard. On top of the implementation the Cloud4SOA governance layer offers three main functionalities that enable cloud adopters *negotiate and enforce SLA*, as well as *recover from SLA violations*, through (i) Agreement Negotiation, which allows the automatic negotiations on behalf of PaaS providers, based on the semantic description of offerings and the QoS requirements specified by application developers; (ii) Agreement Enforcement, to supervise that all the agreements reached in a SLA are respected (i.e. measurements are within the thresholds established in SLA for QoS metrics); and (iii) Violation recovery. Whenever the execution of the business application does not satisfy the SLA (i.e. breaches of the agreement occurs), the most appropriate recovery action (e.g. warning messages, stop or migration of the application) is suggested based on the policies defined by the software developer.

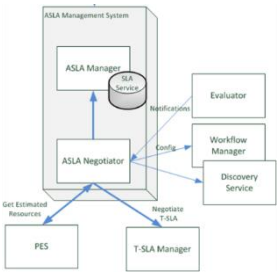
4.2.3.4 OPTIMIS

As mentioned in section 4.1.1, the project [14] aims at enabling organizations to automatically externalize services and applications to trustworthy and auditable cloud providers, while optimizing the complete lifecycle of service engineering, provision, operation, delivery and use.

SLA negotiation is a process that may be undertaken by various roles and includes a number of specialized terms. In OPTIMIS the different deployment and runtime configuration scenarios include different cases such as *private, bursting, federated and multi-cloud deployment*. During a bursting the internal provider negotiates with a public cloud provider in order to acquire resources for a load peak. In the case of the multi-cloud, an intermediate entity, the cloud broker, undertakes the role to find resources possibly expanding to different clouds in order to meet the specific requirements posed by the service customer/owner. These requirements may span across a variety of factors, such as service or *provider risk, trust, ecological or cost levels* (TREC) [71], [72], *legal requirements* (when dealing with personal data) or simple *non-admission of the entire service* by a provider due to lack of resources. Finally, in the federated scenario, the infrastructure provider may (during deployment or runtime) *split the service manifest* (e.g. in deployment if an admission controller specifies that the internal resources are not sufficient for the current or future needs of the service) in order to keep one part internally and use a different provider (transparently to the service provider level) for the non-admitted part. The negotiation framework extends WS-Agreement for multi-round negotiations and enables the interplay between legal and flexible service provisioning based on the legal terms included in the SLAs.



4.2.3.5 IRMOS



As mentioned in 4.1.6, the project [34] has developed specific cloud solutions. Most existing SLA management procedures consider the negotiation to be a process that takes place before the execution phase. Once the negotiation has been produced the service is monitored against the corresponding SLA established and in case there is a violation (or potential violation), then several actions (reflected in the SLA in most cases) are performed. However, sometimes the causes and origin of the violations could be addressed by establishing again a process of negotiation. This is called *SLA re-negotiation* and may be triggered either by the cloud adopter (e.g. change in application parameters that affect the QoS), by one of the providers (e.g. detection of potential SLA violation) or by the application (e.g. scalability rules) [73]. SLAs can be *updated during runtime* to reserve additional resources following a cloud adopter request or a corrective decision from the platform in order to maintain the requested QoS [74]. A prerequisite in cases of renegotiation is always the availability of the additional resources in the infrastructure layer; otherwise the re-negotiation of the SLA may be rejected [75].

Another important aspect in the SLA management within IRMOS is the *automatic way of negotiation and re-negotiation*, thus performed without human intervention but based on policies defined by the actors involved in the negotiation.

4.2.3.6 SLA@SOI

As mentioned in section 4.1.5, the developed open-source SLA@SOI framework mainly addresses the complete service lifecycle through autonomous negotiation, provisioning, monitoring and adaptation of SLAs.

The aim of the developed SLA negotiations framework is to enable *negotiations across multiple tiers: business, software, and infrastructure*. To this end, SLA@SOI implemented a framework that enables *different protocols to be injected* so as to facilitate the interaction between the different layers and entities. The framework consists of a *domain-agnostic protocol engine* and a negotiation protocol. The engine executes the negotiation protocol, providing stateful interaction between the customer and the provider. The negotiation protocol enables the implementation of *custom interaction behaviours* and has been encoded as declarative styled rules in order to make it maintainable, readable and machine interpretable. Since the protocol will be used for specific interaction, it may include domain specific content. The SLA model is described in [76].

4.2.3.7 Helix Nebula

As mentioned in section 4.1.7, Helix Nebula aims at establishing a multi-tenant, multi-provider cloud infrastructure, by integrating the idea of “Blue Box”, which is a complex component of the project’s architecture related to interoperability aspects. It provides API services and a Web Portal that enables

cloud adopters to interact in a central and transparent manner, while federating the infrastructures of multiple cloud providers.

4.2.4 SLA Enforcement

The SLA enforcement phase aims at *ensuring* that the *quality* parameters (agreed in signed SLAs) are retained. All providers exploit monitoring mechanisms to obtain both infrastructure and application monitoring data, while adaptable approaches focus on adjusting the monitoring time intervals or the monitoring metrics based on the collected information during runtime. Evaluation tools are exploited to analyse the monitoring data and trigger corrective actions using SLA violation detection mechanisms, some of which enable proactive violation detection.

4.2.4.1 Cloud4SOA

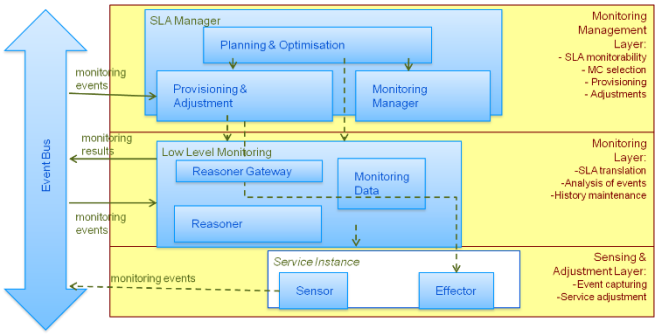
As mentioned in 4.2.3.3 subsection, the project [70] empowers a multi-cloud paradigm at PaaS level, providing an interoperable framework for PaaS developers.

Cloud4SOA has identified the challenge that exists with respect to provide a *unified platform-independent mechanism to monitor* the health and performance of business-critical applications hosted on multiple cloud environments in order to ensure that their performance consistently meets expectations defined by the SLA. In fact, different providers use different metrics and deliver the data by implementing specific APIs. To address this challenge, the project has developed *unified interfaces* that overlook all customers' deployments at once, thus allowing customers to compare and evaluate different deployments. This could be performed externally through the REST API or internally by Platform Components.

Furthermore, a *set of unified metrics* (across PaaS providers) has been selected to monitor the application execution and usage. These are both *application-level metrics* (defined through a library embedded in the source code of the application) and *infrastructure-level metrics* (using the interface of the provider). Currently the following set of metrics and the corresponding APIs have been developed: application / database response time, cloud response time, web container response time, application status, memory usage and CPU usage.

4.2.4.2 SLA@SOI

As mentioned in section 4.1.5, the developed open-source SLA@SOI framework mainly addresses the complete service lifecycle through autonomous negotiation, provisioning, monitoring and adaptation of SLAs.



As a fundamental mechanism for the provision of QoS guarantees (i.e. SLA enforcement), which may also require service (re)provisioning during runtime, SLA@SOI [32] has developed a *three-layered dynamically configurable*

monitoring framework. The upper layer manages the overall monitoring operation by *identifying the monitorable metrics* of the SLAs (i.e. what can be monitored), *selecting monitoring components as sensing elements*, and configuring them in order to identify the optimum sources of events and monitoring data. The middle layer (namely low-level monitoring layer) performs *translation of high level SLAs to operational monitoring specifications* acceptable by specific reasoners (aka monitors), passes operational monitoring specifications to reasoners and receives data from them, while it also maintains the monitoring data. The lower layer (namely sensing and adjustment layer) *captures the events through reasoners* – may be either intrusive (i.e. instrumented into services) or non-intrusive (i.e. run in parallel with the system checking if the events captured from it satisfy the SLA). Reasoners are also able to "understand" the terms included in the SLAs and implement monitoring rules based on abstract syntax trees. The key project results including the SLA Architecture and SLA model are summarised in [77].

4.2.4.3 IRMOS

As a basis for real-time application execution, a *monitoring and evaluation* framework has been developed that collects information from both *application* (high-level performance metrics) and *infrastructure levels* (low-level resource utilization metrics) and evaluates the monitoring data against expected QoS to support runtime decision making [78], [79]. The monitoring framework follows a *hierarchical architectural approach* (i.e. monitoring instances also reside in the VMs hosting the deployed services in order to obtain application-related monitoring data), while being *adaptable in terms of monitoring time intervals* (based on the collected monitoring information and the corresponding SLA terms) to minimize the footprint on the system and network overheads when propagating monitoring information.

4.2.4.4 Stream

Stream [80] architected and developed a system able to process data / event streams in a distributed fashion. By enabling query parallelization and scalability of query operators, thousands of cores can be aggregated to correlate and aggregate millions of events per second. One of the issues with SLA monitoring in large systems (e.g. large data centers) is scalability, given that the amount of monitoring data and the processing needs follow an exponential growth. The project has developed an approach to *parallelize queries for continuous data streams*, such as monitoring data streams, enabling its scalability to large data stream volumes [81]. The latter is of major importance for cloud environments, since reports during runtime in large scale deployments require the collection and analysis of big amounts of monitoring data. The developed mechanisms are *applicable to cloud monitoring frameworks* since the non-intrusive elasticity will enables them to adapt based on the incoming load [82].

4.2.4.5 mPlane

The project aims at developing an intelligent measurement plane for the Internet in order to collect and analyse measurements in large scale networks. mPlane has developed an approach for *defining SLAs according to the OSI layer* (i.e. layer 1-2 to verify the SLA between the ISP and the cloud adopter, layer 4

to capture the user requirements, and layer 7 to capture the user experience) and monitoring the delivery of network services according to these SLAs. The SLA measurement definition has been performed according to different accesses (i.e. xDSL, FTTx, and 3G-4G) and aims at collecting monitoring data for network resources (i.e. data transmission speed, packet delay, loss ratio and unsuccessful transmission ratio). Moreover, analysis of the users' experience (Quality of Experience - QoE) is performed through a Mean Opinion Score (MOS) for objective and subjective evaluation.

4.2.4.6 CloudScale

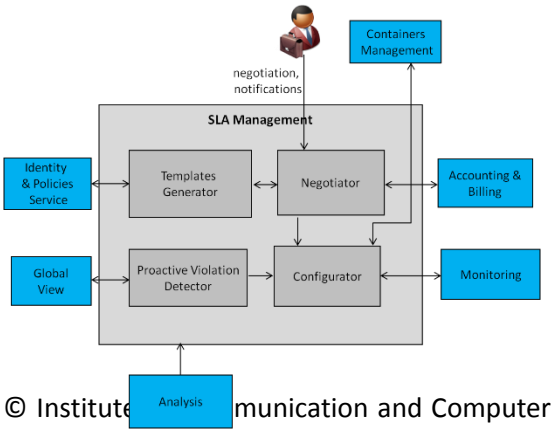
The project [83] aims at supporting scalable service engineering. In this context, mechanisms are developed to support service providers in analysing, predicting and resolving scalability issues in cloud environments [84]. CloudScale among other things focus on scalability aspects (i.e. changing needs for infrastructure resources needed during runtime) and their incorporation in SLAs (i.e. quality requirements / attributes for scalability).

CloudScale is developing mechanisms to *identify causes of potential SLA violations*. When the services do not scale as expected, root causes of the scalability problems based on sources are identified. This analysis is done *based on the source code* for the service. To find out what to do with this scalability problem, a scalability model may be extracted from the same source code. Based on this scalability model a what-if analysis can be performed to find good ways of resolving the scalability issue, for example by using a different cloud provider, or by changing the implementation of the source code. If no viable solution is found, the scalability requirement specified in the SLA may have to be relaxed.

4.2.4.7 VISION Cloud

The goal of VISION Cloud [85] is to introduce a powerful ICT infrastructure for reliable and effective delivery of data-intensive storage services, facilitating the convergence of ICT, media and telecommunications. This infrastructure will support the setup and deployment of data and storage services on demand, at competitive costs, across disparate administrative domains, while providing QoS and security guarantees.

VISION Cloud has developed an efficient and scalable monitoring framework that is *adaptable to the number of clusters and the nodes per cluster* [86], [87]. It follows a *hierarchical architecture* in order to aggregate monitoring information both at cloud and at cluster levels. The information is being propagated to an event management component that generates events in order to detect and handle error conditions or performance degradations and trigger corrective actions.



What is more, the aforementioned monitoring and event management framework focuses on *proactive SLA violation detection* through the enhanced analysis of monitoring data. This analysis aims at the identification of potential *relationships between the different metrics* being monitored in order to conclude to dependencies that may affect the evolution of the metrics during runtime. Moreover, the analysis aims at *discovering repetitive patterns in the monitoring information* that

may provide indications with respect to SLA violations based on the historical data and their evolution in time.

4.2.4.8 CumuloNimbo

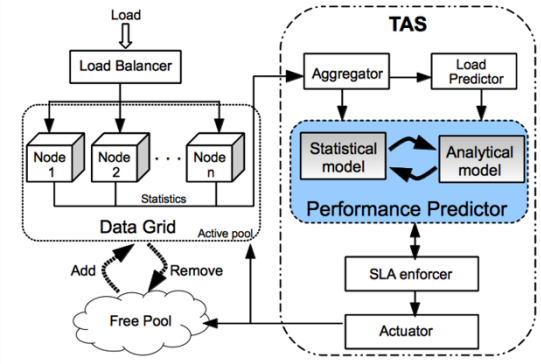
CumuloNimbo [88] has developed a PaaS solution that provides high scalability without sacrificing data consistency and ease of programming. The transactional management system can be integrated with any data management system (databases, NoSQL data stores, SQL engines) and software stack (e.g. Java EE, LAMP, etc.).

Given that one of the hardest questions in SLA enforcement is how to deal with an increase of the load in the system, CumuloNimbo has developed a solution for facilitating scalability and elasticity for transactional systems [89], [90]. Emphasis is put on how cloud data stores can accommodate full coherence, which requires *ACID* (Atomicity, Consistency, Isolation, Durability) transactions. The latter is of major importance for *SLA enforcement* since specific parameters related to the load of the system can only be addressed through scalability and elasticity but the goal is to retain coherence. To this direction, the project has developed mechanisms to deal with the *scalability* (and thus SLA enforcement) of *transactional systems* when exploiting cloud infrastructures.

4.2.4.9 Cloud-TM

As mentioned in 4.2.1.2 subsection, Cloud-TM [44] develops a data-centric PaaS layered on top of a self-optimizing, highly scalable distributed Transactional Memory platform.

In that case, the project has developed an innovative approach for managing and enforcing SLAs when dealing with transactional cloud data stores. The approach is realized through a framework enabling *self-optimization and self-tuning of the infrastructure resources* based on different QoS metrics [53].



It triggers in an automated way elastic scaling while ensuring consistency through adaptive data placement schemes [50]. A unique aspect of the Cloud-TM platform consists in its ability to continuously *self-tune its data management protocols* [46], [49] during service usage in order to enforce SLAs. The overall approach allows for overcoming issues related to contention (due to the inclusion of additional resources) both on the logical layer (through the corresponding data management) and on the physical layer (through resource management based on dynamic resource provisioning).

Cloud-TM leverages on the SLA@SOI framework to define SLOs between the Cloud-TM platform's provider and service developer. In particular, Cloud-TM defines custom SLA templates that allow for *negotiating domain-specific QoS levels*, such as constraints on the response time and abort rate of different transaction profiles.

4.2.4.10 SeaClouds

SeaClouds SLA Service permits the SLA management, which consists of the actions of a) generating and storing WS-Agreement templates and agreements, and b) assessing that all the agreements (SLA guarantees) are respected by being evaluated by the business rules.

Given that SeaClouds SLA considers that the cloud provider enforces its own SLA, it does not impose, any penalty to this actor. In this case, enforcement actions are focused on the migration and the re-planning of the modules of the affected cloud provider, to another one.

4.2.4.11 CELAR

Elasticity platform of CELAR architecture consists of all the algorithms and modules that are necessary in order to provide automatic resource allocation based on the application characteristics, the user-defined optimization and the incoming load.

Moreover, the elasticity platform is responsible for maintaining all necessary information for past and current application deployments, the orchestration of added or removed resources as well as methods for ensuring robustness and availability of the elastic operations. The modules that compose its functionalities are listed below:

- Decision module: It takes real-time, informed decisions on the type and quantity of resources that need to be added or removed from a running application.
- Resource Provisioner: It undertakes the task of automated, on-demand creation of multi-machine runtime environment, by encapsulating both the interaction with the underlying cloud infrastructure for resource request and release, and the task of updating the application configuration in order to employ the newly committed resources or detach those freed (Application Orchestration).
- DataBase: As a central storage module, it maintains information useful for other CELAR components.
- Manager: It handles action orchestration in the elasticity platform, by operating on top of the CELAR DataBase, while providing mechanisms of fault-tolerance.
- Application Profiler: It allows the monitoring and measurement characterization of the application's behavior, over a number of representative resource provisioning and load scenarios.

5 SLA standardisation propositions

5.1 SLA specification

5.1.1 ISO

Defining specific metrics is essential to the cloud SLA and the measuring of accomplishment of objectives. The metrics are used to set boundaries and the margin of errors that the provider must align when providing a cloud service. These metrics can be used in the monitoring of the service as well as determining when a failure took place, and therefore when the customer has the right for a remedy. The use of a standardized set of metrics makes it easier to determine SLOs, define an SLA and compare one cloud SLA to another. The metrics should address the following:

- Determine if SLOs are triggered
- Categorize service capabilities
- Define a purpose for measures and measurements
- A consistent representation of measure and measurement information
- Link properties, measurements and metrics
- Comparison of monitoring between services
- Determine if metrics are effective for business objectives [103].

The use of metrics for Cloud Services is threefold. It addresses the service selection, agreement and verification. In practice, the metric is applied within a given scenario that determines specific conditions, such as a specific resource being measured, at a given time for a specific objective [104].

The ISO team has defined the following terms as a part of their Metrics System.

- **Abstract Metric:** An abstract standard of measurement used to assess a property. The standard of measurement describes what the result of the measurement means, but not how the measurement was performed. The Abstract Metric is not used by itself, but is instantiated using a Metric.
- **Cloud Service Property:** The property of a cloud service entity to be observed. A property may be qualitative or quantitative.
- **Context:** The circumstances that form the setting for an event, statement, or idea, and in terms of which it can be fully understood and assessed.
- **Measurement:** Set of operations having the object of determining a Measurement Result.
- **Measurement Result:** Value that expresses a qualitative or quantitative assessment of a property of an entity.
- **Metric:** A standard of measurement that defines the conditions and the rules for performing the measurement and for understanding the results of a measurement. The metric therefore is a standard set of procedures that generate values for its corresponding abstract metric
- **Observation:** Measurement based on a metric, at a point in time, on a measurement target.
- **Unit of Measurement:** Real scalar quantity defined and adopted by convention, with which any other quantity of the same kind can be compared to express the ratio of the two quantities as a number [104].

The **ISO Metrics Mode**l is analogous to a reference model that uses a standardized meta-model and its taxonomy/terminology. The model expresses the following set of characteristics:

Consistent Representation of Information related to metrics should be represented in a consistent, repeatable way in order to efficiently organize it, share it and use it.

Explicit Relationships Concepts like metrics should be represented in such a way that the relationships among them, if any, are explicit. This clarifies the effects these concepts have on one another and their importance.

Repository of Definitions There should be a way to organize metrics so they are reusable, searchable and derivable.

Comparablity The properties of the different concepts should allow cloud adopters to have enough information to efficiently compare them to find and understands either similarities or differences.

Flexiblity and Adaptablity The concept model should be sufficiently flexible and adaptable to allow for easy integration with other metric models. These models could be complementary to the concept model (e.g. represent measurement methods and process).

Composability The metrics should allow metrics definitions and instances to be reusable. Thus one should be able to use one or more metrics to build a composite metric. This metric that is composed of underlying metrics builds upon the information they contain. This results in metrics that could possibly be composed of underlying metrics of different kind (e.g. qualitative and quantitative). A key aspect of the CSM Core diagram is to allow metric definitions to be composed with other metric definitions. This is an effort to limit the duplication of information without too much of an increase on complexity. CSM Core allows metrics of different kind - qualitative or quantitative - to be defined, which in turns means that metrics of different kind can potentially be composed with one another. This can affect the estimation – measurement results – of a particular property in several ways like uncertainty, precision, accuracy etc [104].

The whole process of metrics definition follows a specific Metrics Template. The template provides standardization for defining both Abstract and Concrete Metrics. The Abstract Metrics Template defines a general notion for a metric as well as extra definitions. The Concrete Metrics Template allows for variations upon a specific metric from one provider to another.

5.1.2 C-SIG

Cloud Selected Industry group (C-SIG) set-up by DG CONNECT produced as a final result a set of SLA standardisation guidelines aiming to “improve the clarity and to increase the understanding of SLAs for cloud services in the market” [98] In order to maximise the internalisation and therefore, impact, of these guidelines, this work was contributed to ISO/IEC 19086 project [99].

C-SIG's cloud standardisation guidelines are developed following next set of principles:

- Technology neutral, so to consider the diversity in the different technology foundations in which Cloud provider's base their offerings today, and possible evolutions over time.
- Business model neutral, in order to not to restrict to a particular business model but to consider existing a future variety of these such as pay-per-use, long term contracts, advertising, public funds, etc.
- World-wide applicability.
- Unambiguous definitions.
- Comparable SLOs, considering that although these SLOs are not determined by identical means in different provider's; associated terminology, metrics, and templates are useful mechanism so to enable comparison in relation to quantitative SLOs.
- Conformance through disclosure, while considering diversity in the SLO measures mechanisms, method to achieve them have to be documented for the specificity of the service.
- Standards and guidelines which span multiple customer types, in order to take into account all typologies of cloud customers, from the smallest to the largest.
- Cloud Essential characteristics, as defined in ISO/IEC 17788 [100]. These are: Broad Network Access, Measured Service, Multi-tenancy, On-demand self-service, Rapid elasticity and scalability, as well as, Resource pooling.
- Proof points, those ensure that cloud essential characteristics are viable from both technical and business perspectives for provider organisations.
- Information rather than structure, this is that SLOs defined describe concepts rather than a specific structure for the SLA.
- Leave legal agreements to Attorneys, meaning that these guidelines refer to concepts and definitions, however legal agreements are not considered in the scope.

The next table summarizes the SLOs defined and recommended by these guidelines. Terms and vocabulary of these are aligned with ISO/IEC 17788 [100]. SLOs are structured in four (4) main typologies:

- Performance: SLOs that relate to the performance of the cloud service and the performance of related aspects of the interface between the cloud service customer and the CSP.
- Security: SLOs that aim to improve assurance and transparency. These are defined using well-known frameworks such ISO/IEC 27001 [101] and ISO/IEC 27002 [102].
- Data Management: SLOs defining data life cycle management quantitative and qualitative indicators.
- Personal Data Protection: SLOs applicable to cases where the CSP acts as a data processor.

For each of these types, different categories of SLOs are provided. Categories definition quote definitions in [98], where more detailed definitions can be found both for these categories and the specific SLOs defined.

Table 10 Summary of C-SIG SLA Guidelines Service Level Objectives

SLO type	Category	Applicable SLO
Performance	**Availability** "Availability is the property of being accessible and usable upon demand by an authorized entity."	Level of uptime (Often termed "availability")
		Percentage of successful requests
		Percentage of timely service provisioning requests
	Response Time "Interval between a cloud service customer initiated event (stimulus) and a cloud service provider initiated event in response to that stimulus."	Average response time
		Maximum response time
	Capacity "Capacity is the maximum amount of some property of a cloud service. It is often an important value for cloud service customers to know when using a cloud service."	Number of simultaneous connections
		Number of simultaneous cloud adopters
		Maximum resource capacity
		Service Throughput
	Capability Indicators "Capability indicators are service level objectives which promise functionality relating to the cloud service."	External connectivity
	Support **"**Support is an interface made available by the cloud service provider to handle issues and queries raised by the cloud service customer."	Support hours
		Support responsiveness
		Resolution time
	Reversibility and the Termination Process **"**Termination process takes place when a cloud service customer or a cloud service provider elect to terminate the agreement"	Data retrieval period
		Data retention period
		Residual data retention
Security	**Service Reliability** "Service reliability is the property of a cloud service to perform its function correctly and without failure, typically over some period of time"	Level of redundancy
		Service reliability
	Authentication & Authorization "Authentication is the verification of the claimed identity of an entity (typically for cloud computing the entity is a cloud service user). Authorization is the process of verifying that an entity has permission to access and use a particular resource based on predefined user privileges."	Cloud adopter authentication and identity assurance level
		Authentication
		Mean time required to revoke cloud adopter access
		Cloud adopter access storage protection
		Third party authentication support
	Cryptography	Cryptographic brute force

SLO type	Category	Applicable SLO
	"Cryptography is a discipline which embodies principles, means and methods for the transformation of data in order to hide its information content, prevent its undetected modification and/or prevent its unauthorized use**"**	resistance
		Key access control policy
		Cryptographic hardware module protection level
	Security Incident management and reporting **"**An information security incident is a single or a series of unwanted or unexpected information security events that have a significant probability of compromising business operations and threatening information security. Information security incident management are the processes for detecting, reporting, assessing, responding to, dealing with, and learning from information security incidents**"**	Percentage of timely incident reports
		Percentage of timely incident responses
		Percentage of timely incident resolutions
	Logging and Monitoring **"**Logging is the recording of data related to the operation and use of a cloud service. Monitoring means determining the status of one or more parameters of a cloud service.**"**	Logging parameters
		Log access availability
		Logs retention period
	Auditing and security verification **"**Auditing is the systematic, independent and documented process for obtaining audit evidence about a cloud service and evaluating it objectively to determine the extent to which the audit criteria are fulfilled"	Certifications applicable
	Vulnerability Management **"**Management of vulnerabilities means that information about technical vulnerabilities of information systems being used should be obtained in a timely fashion, the organization's exposure to such vulnerabilities evaluated and appropriate measures taken to address the associated risk"	Percentage of timely vulnerability corrections
		Percentage of timely vulnerability reports
		Reports of vulnerability corrections
	Governance "Governance is system by which cloud service is directed and controlled." **Service changes**	Cloud service change reporting notifications
		Percentage of timely cloud service change notifications
Data Management	**Data classification** **"**description of the classes of data which are associated with the cloud service"	Cloud service customer data use by the provider
		Cloud service derived data use
	Cloud Service Customer Data Mirroring, Backup & Restore	Data Mirroring Latency
		Data Backup Method

SLO type	Category	Applicable SLO
	"mechanisms used to guarantee that the customers' data is available (online or offline) in case of failures forbidding access to it"	Data Backup Frequency
		Backup Retention Time
		Backup Generations
		Maximum Data Restoration time
		Percentage of Successful Data Restorations
	Data Lifecycle "efficiency and effectiveness of the provider's data-life cycle practices"	Data deletion type
		Percentage of timely effective deletions
		Percentage of tested storage retrievability
	Data Portability "CSP capabilities to export data"	Data portability format
		Data portability interface
		Data transfer rate
Personal Data Protection	**Codes of conduct, standards and certification mechanisms** "data protection codes of conduct, standards or certification schemes that the service complies with"	Applicable data protection codes of conduct, standards, certifications
	Purpose specification "purposes of the processing must be determined"	Processing purposes
	Data minimization "cloud service customer is responsible for ensuring that personal data are erased (by the provider and any subcontractors) from wherever they are stored as soon as they are no longer necessary for the specific purposes"	Temporary data retention period
		Cloud service customer data retention period
	Use, retention and disclosure limitation "any legally binding request for which the provider is compelled to disclose the personal data by a law enforcement or governmental authority"	Number of customer data law enforcement disclosures
		Number of personal data disclosure notifications
	Openness, transparency and notice	List of tier 1 subcontractors
		Special categories of data
	Accountability "ability of parties to demonstrate that they took appropriate steps to ensure that data protection principles have been implemented"	Personal data breach policy
		Documentation
	Geographical location of cloud service customer data "location of data processed in the cloud"	Data geolocation list
		Data geolocation selection
	Intervenability "rights of access, rectification, erasure, blocking and objection"	Access request response time

5.2 SLA management mechanisms

5.2.1 ISO

The ISO proposed Cloud SLA Management mechanism covers the SLA Design, Evaluation & Acceptance, Implementation, Execution and Changes to the cloud SLA. It is the obligation of the Cloud Customer to ensure the alignment of the SLA with the companies overall strategy.

The **Design** stage outlines the needed steps that the Cloud Customer and Cloud Provider should undertake in the case of a jointly designed SLA. The purpose of this process is to ensure the capabilities of the Provider to cover the Customer's needs and align with the capabilities of the covered service.

The design process should outline the appropriate actors and roles, using **Rec. ITU-T Y.3502 | ISO/IEC 17789** as a reference. **Rec. ITU-T Y.3502 | ISO/IEC 17789** states the following entities as key actors in the SLA Design Process:

- **Cloud Service Customer (CSC)** is the party that enters a business relationship in order to use a Cloud Service
- **Cloud Service Provider (CSP)** is the party that makes the cloud service available
- **Cloud Service Partner** is the party that is engaged in support of, or auxiliary to activities of the CSP, CSC or both

The choices of concepts that a cloud SLA contains depend on the business model and/or the provided service. The design of the SLA should also outline the process to change the Cloud SLA. This process should be a part of the SLA or other governing document. The Design Stage should also outline the monitoring mechanisms of the service level, as well as the reporting of failures to meet SLOs.

The **Evaluation & Acceptance** is performed using the ISO/IEC JTC 1/SC 38 as a reference. The overall process is a catalog that a Customer should review before accepting a Cloud SLA.

- The CSC should review and pick the concepts critical to their business objective, as well as that the SLOs of the provider are sufficient and the remedies are analogous to the business impact.
- The business policies and governance of the customer the context which should be used to review the SLA.
- Data management, performance of the cloud service, incident management, disaster recovery, termination, security, privacy and intellectual property rights are also critical areas that should be checked
- Other standards that are important to the business objectives should be a part of the cloud SLA. Some CSPs are certified by the conformance to specific industry standards
- Methods for monitoring the cloud service levels and report failures, such as web portals, email, text messages, telephone, posts to social media sites or management systems should be determined.

Clicking a check box usually performs the acceptance of a Cloud SLA. In some cases, the usage of the Cloud Service constitutes the acceptance of the SLA. In the instance of an official signing both the CSP and CSC should be certain that they are ready to implement and execute the SLA.

The **Implementation** stage of the cloud SLA covers the monitoring and managing the cloud service levels, the failures reporting and the claiming of remedies. The CSP and CSC are in some cases collaborating for the implementation of the SLA. The choice of concepts and terms to be included in the SLA depends on the provided service.

The **Execution** stage covers the operation of the cloud service as well as the monitoring and reporting mechanisms. In case the customer believes the SLOs have not been met, the CSC should follow the procedures, report the failure and claim the remedy.

In the **Changes** stage the CSP should use the processes established at the Design stage to inform the CSC for adjustments in the Cloud SLA. Moreover the CSP should include request mechanisms that allow the customer to ask for changes in the SLA. The CSP should provide timely notifications that allow the CSC to react (provide feedback, negotiate) to changes in the SLA. Both the CSC and the CSP should review the changes and ensure that they cover their business objectives.

5.2.2 C-SIG

Cloud Selected Industry group (C-SIG) did not consider SLA management processes in its work, solely the definition of standard SLO terms.

6 Stakeholders views

This chapter provides a short summary as for the requirements and needs from both the CSPs and the adopters (customers), which derive from questionnaires, about the “key” metrics that each one of them assume and believe that are necessary to be generally included in a SLA.

6.1 Cloud providers

Most of the cloud providers (CSP) agreed with the idea of SLALOM consortium to align the Master Service Agreement (MSA) with the ISO specifications but also recommended some additions and/or enhancements. The additions referred to aspects such as warranty and respective penalties, payment, cancellation/termination rights while the proposed enhancements mainly focused in security, privacy, response time, availability and auditing aspects.

Moreover, the participating CSPs were also asked to provide feedback with respect to the ISO components and metrics, and their importance. The quantitative analysis of their answers reveals the following prioritization (higher importance to lower importance) of the first (most important) ten (10) considered components:

- Information Security Component
- Availability Component
- Protection of personally identifiable information component
- Service reliability component – customer data and back up component
- Service reliability component – disaster recovery component
- SLA definitions component
- Data management component – data location component
- Cloud service performance component – cloud service response time component
- Service reliability component [/cloud service provider disaster recovery plan]
- Data management component [/Data location]

The revealed prioritization of the ten (10) most important (according to CSPs) metrics is as follows:

- Availability component [/Total downtime]
- Availability component [/Availability]
- Availability component [/Uptime]
- Availability component [/Allowable downtime]
- Service reliability component [/Recovery time objective]
- Availability component [/Availability percentage]
- Service reliability component [/Maximum time to service recovery (MTTSR)]
- Service reliability component [/Recovery point objective (RPO)]
- Service reliability component [/Number of service failures]
- Cloud service performance component [/response time observation]

Information with respect to the prioritization of the remaining components and metrics, as well as CSPs' comment with respect to them, can be found in SLALOM Deliverable D4.1 and D5.1, "Initial Position Paper Reflecting Cloud Service Provider and Cloud Adopter Requirements".

Last, but not least, the CSPs provided their feedback regarding the research scenarios that should be taken into account in the SLALOM model terms and specifications. Among the 5 discussed scenarios, "SLAs at different levels", "Multi-level SLA interaction model" and "Proactive SLA violation detection" were characterized as "somewhat important" by the majority of the CSPs but not considered as "a core requirement" while the scenarios "SLA Negotiation across multiple layers" and "Automated SLA re-negotiation" were mainly considered as lower importance scenarios ("somewhat unimportant"). More details (e.g., percentages) about these outcomes can be found in SLALOM Deliverable D3.1, "Initial position paper (technical)".

6.2 Cloud adopters

Cloud adopters also found good the idea of SLALOM consortium to align the MSA with the ISO specifications and suggested a few more additions and/or enhancements. In this case, the suggestions reflected the concerns of the adopters with respect to the cloud subcontracting and the minimum security controls required.

The twelve (12) most important ISO components for the cloud adopters differ from those identified from the CSPs either in the component or in their importance:

- Information security component
- Data management component [/cloud service customer data component]
- Service reliability component [/Network redundancy]
- Service reliability component [/cloud service provider disaster recovery plan]
- Availability Component
- SLA definitions component
- Protection of personally identifiable information component
- Service reliability component – disaster recovery component
- Data management component
- Data management component – data deletion component
- Governance component [/regulation adherence]
- Service reliability component [/retention period for backup data]

The respective most important metrics for the cloud adopters are:

- Cloud service performance component [/cloud service throughput]
- Availability component [/total downtime]
- Service reliability component [/Recovery time objective]
- Service reliability component [/Maximum time to service recovery (MTTSR)]
- Service reliability component [/Recovery point objective (RPO)]

- Service reliability component [/Time to Service recovery (TTSR)]
- Cloud service performance component [/response time observation]
- Availability component [/Downtime]
- Service reliability component [/number of service failures]
- Cloud service performance component [/cloud service bandwidth]

Finally, most cloud adopters agree that the scenarios coming from the research initiatives are almost all ("SLAs at different levels", "Multi-level SLA interaction model", "SLA Negotiation across multiple layers" and "Automated SLA re-negotiation") of high importance and thus characterise them as "core requirements" while "Proactive SLA violation detection" is considered as "somewhat important".

7 SLA Specification

A SLA is nothing more than a type of contract between two parties where SLAs dictate the quality and type of service that will be provided to the client in exchange for a fee. In general, SLAs are important due to their setting of the tone for the relationship between the parties and will govern if and when things break down. A "good" SLA is a balance between being thorough and clear on the one side, while not being overly onerous on the service provider on the other side.

Having in mind the importance of a SLA, the problem that occurs is that each different CSP is publishing its own SLA, containing different types of metrics, rules, and/or parameters. To this end, it can be considered very difficult for the cloud service customer to be able to compare the different SLAs that the different providers offer to him, in order to sign the most promising agreement for him. For that reason, the main problem that occurs is the fact that the SLAs do not have a specific comparable structure, and they do not obey to common standardized rules.

Several approaches have been made in order to make the life of both the providers and the customers easier and overcome today's great variability in SLA specifications (e.g., ISO, C-SIG), but they are still immature. The SLALOM SLA specification, tries to bridge these gaps by providing a "core" SLA specification, giving for every single CSP, a way to write down his SLA with a specific structure, accompanied with certain rules and dependencies. Thus, both the customer and the provider will be able to compare, negotiate and sign the different SLAs, in a quicker, easier and more effective way, giving the chance for better monitoring and verification.

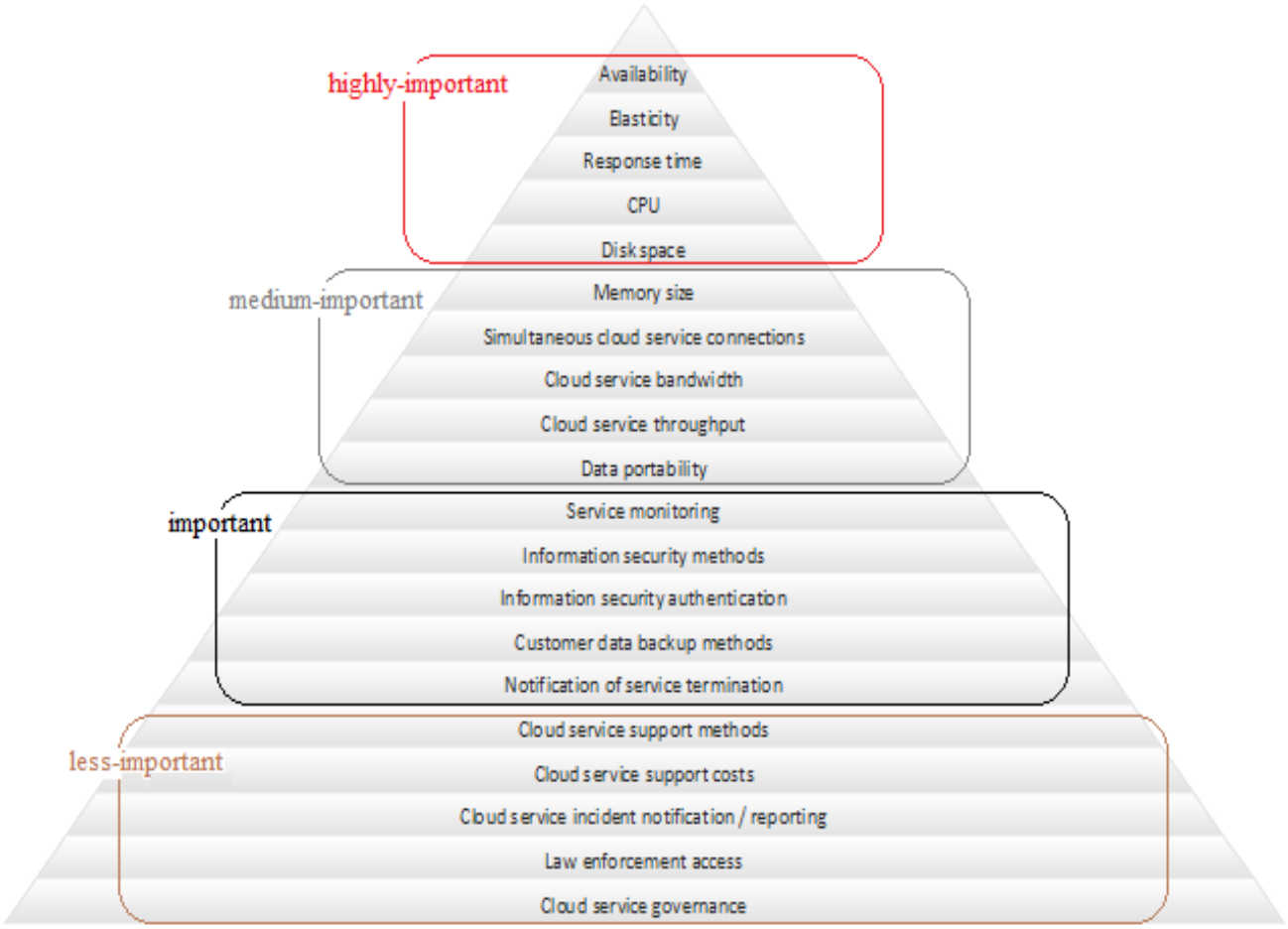
The SLALOM SLA specification will allow the definition of parameters / terms through a well-defined set of metrics. It contains specific key metrics/terms (used to assess a property of the SLA), parameters (used for the expression of a metric), rules (used for further possible constraints of a metric), and dependencies (used for specifying the dependencies between the different metrics) that have to be followed by the CSPs, which will be described in the following subsections.

One should keep in mind that the suggested "core" SLA specification may contain some changes as for its content according to each provider's requirements, but it is mandatory for both the providers and the customers to obey and follow the provided structure, definitions and rules.

7.1 Key metrics

At first stage, a thorough research was made about the different SLA metrics that are used by the key CSPs (e.g., Amazon, Microsoft, SAP), specified into the projects that have been discussed in section 4, and referred by organizations of standardizations (e.g., ISO).
In combination with the aforementioned and the idea that the metrics have to consistently represent the information, be comparable, flexible and adaptable, and the fact that the SLAs must be enforceable and state specific remedies that apply when they are not met, the SLALOM SLA project concludes that the metrics that have to be provided into the aforementioned "core" SLA specification concept, are stated below in the figures in descending order (according to their priority).

It must be mentioned that as it is difficult to sort the metrics and find out which are the most and less important, (as most of them can switch their places without affecting the efficiency of the others) the proposed "core" metrics of the SLALOM SLA specification, are separated into four (4) different types of group *(highly-important, medium-important, important, less-important*). In that case, each of the metrics that belong to the same group are equivalent to each other, meaning that they all have the same amount of importance.



Following, it is given the explanation of the twenty (20) chosen metrics, accompanied with their *units of measurement*, the *reason* of why they have been chosen in that exact order, the *category* that they belong to (e.g., Availability, Performance, etc.), their corresponding *priority group* (e.g., highly-important, medium-important, etc.), as well as their *scale*. As for the scale characteristic, it has to be mentioned that the metrics must belong either to the category of the SLOs, where the value of the metric follows the interval or ratio scale (ISO 3534-2), or to the category of the service-qualitative objective (SQO), where the value of the metric follows the nominal or ordinal scale (ISO 3534-2).

- **Availability**: Availability is the property of the cloud service being successfully accessed via its interfaces and usable upon demand by an authorized entity [ISO/IEC17788]. It provides the cloud service customers with a *percentage* which corresponds to a high level indicator that the services respond to the requests and fulfil the functions of the service description, at a specific point of time.
  Reason: Availability is the most vital metric for a SLA due to the fact that when the cloud service customer needs it in order to have access to the cloud services, it is the leading characteristic of the service's excellence. Thus, what makes this characteristic particularly important is the fact that no other elements or services that are provided matter, if the available services are not accessible.

Category	*Priority Group*	*Scale*
Availability	Highly-important	Ratio (SLO)

- **Elasticity**: Elasticity describes the ability of a cloud service to quickly provision or de-provision resources as needed, and dynamically adjust the amount of resources that are allocated to an instance of a service. What has greater importance is the speed (seconds) characteristic of elasticity which describes how fast a cloud system is able to react on the request of a cloud service customer, for resource re-allocation (in the case of manual elasticity) or changing workload (in the case of automatic elasticity).
  Reason: Elasticity is a key metric in a SLA, as the cloud service customers do not have to allocate a sufficient amount of resources for the cloud service during service configuration, but can rely on the ability of the cloud system to scale up resources over time as needed, and scale down resources that are no longer needed. As a result, customers do not have to pay for any extra resources that they may have been (wrongly) peaked during the service configuration, as long as all the resources will be adjusted automatically by the system.

Category	*Priority Group*	*Scale*
Performance	Highly-important	Ratio (SLO)

- **Response time**: Response time is a key metric for characterizing the performance of a service, indicating the exact *time (seconds)* between a stimulus to the cloud service and the service's response to this stimulus.
  Reason: Response time is one of the metrics with high priority in a SLA, as it is of mayor importance for the cloud service customers to be able to calculate the total period of time of their requests and understand the performance of the service. Without the response time of the service, the customer would not be able to keep track on how fast and effective the provided cloud service responds, and as a consequence he will not be able to compare its time performance with corresponding services of other providers.

Category	*Priority Group*	*Scale*
Performance	Highly-important	Ratio (SLO)

- **CPU**: CPU is an individual metric of capacity referring to the maximum *number of the processing power (Hertz)* that is provided by the CSP to the customer for processing his data.
  Reason: CPU is a key metric in a SLA, as the cloud service customer gains the ability to find out how fast the requested cloud service is able to process his data. For that reason, the service becomes more reliable and desirable for its users, due to the fact that everyone demands high processing speed.

Category	*Priority Group*	*Scale*
Performance	Highly-important	Ratio (SLO)

- **Disk space**: Disk space is another individual metric of capacity referring to the maximum *number of the available space (Petabytes)* that is provided by the CSP to the customer for storing his data.
  Reason: Disk space is one of the metrics with high priority in a SLA, as all the cloud service customers always want to have the ability to keep track on the services' available space that is provided for their data, giving them the opportunity to know and manage the way they want to store their information.

Category	*Priority Group*	*Scale*
Performance	Highly-important	Ratio (SLO)

- **Memory size**: Memory size, another individual metric of capacity, refers to the maximum *number of information (Gigabytes)* that the service is able to hold for the cloud service customer for reading and writing his data.
  Reason: Accompanied with the metrics of CPU and disk space, memory size can be considered to be another very important key metric of a SLA, because it gives the ability to the cloud service customers to keep track on how they will be able to apply multiple access procedures to their stored data (e.g. multiple reads, multiple writes). Thus, it will be more convenient for them to decide how they will serve and manage their multiple users of the service, as they will know exactly the offered memory size of their used cloud services.

Category	*Priority Group*	*Scale*
Performance	Medium-important	Ratio (SLO)

- **Number of simultaneous cloud service connections**: The number of simultaneous cloud service connections, another metric of capacity, is the maximum *number of simultaneous connections* that can be supported by the system at the same time.
  Reason: The maximum number of simultaneous connections has significant importance as a SLA metric, due to the fact that the customer is able to have a clear view on how the service that is used will be served and used concurrently by multiple users.

Category	*Priority Group*	*Scale*
Performance	Medium-important	Ratio (SLO)

- **Cloud service bandwidth**: Cloud service bandwidth, a capacity metric, refers to the *amount of outbound data transfers (Gigabyte/month)* that can be transferred through the network over a certain period of time.
  Reason: Keeping track on the amount of outbound data transfers can help the cloud service customers to identify whether and how their data is transferred per month, giving them the ability to customize the cloud services that they use and/or their corresponding SLAs, according to the demand and the usage of the services.

Category	*Priority Group*	*Scale*
Performance	Medium-important	Ratio (SLO)

- **Cloud service throughput**: Cloud service throughput, another metric of capacity, indicates the *number of data inputs (Gigabytes/sec)* or the amount of sets of inter-dependent data inputs that can be processed in every unit of time by a system. In other words, it refers to the minimum number of specified requests that can be processed by the cloud service in a stated time period (e.g. requests per minute).
  Reason: Cloud service throughput can be considered as a pretty important metric for the cloud service customers, as they can easily track the amount of data that is being accepted per second by the cloud service, giving them the ability to customize the cloud services that they use, according to their requirements and needs.

Category	*Priority Group*	*Scale*
Performance	Medium-important	Ratio (SLO)

- **Data portability**: Data portability is the *ability* to easily transfer data from one system to another, without being required to re-enter data and its corresponding properties from one cloud service to another. It includes the movement of data between the cloud services to support distributed processing or to enable movement of data to another cloud service.
  Reason: Using portability as one of the key metrics of a SLA is of mayor importance, as cloud service customers are able to easily and securely transfer and use their data in different cloud services, being ensured that this action will be successfully completed.

Category	*Priority Group*	*Scale*
Data management	Medium-important	Nominal (SQO)

- **Service monitoring**: Service monitoring lists the *parameters* (e.g. performance, availability) for the covered services that are monitored by the CSP and for which data is provided to the cloud service customer. These parameters may assist the provider and the customer to determine that the metrics provided in the SLA are being met.
  Reason: Without service monitoring, both CSPs and cloud service customers would not be able to observe the efficiency and various important information of the services. Thus, the reason why service monitoring is considered as a vital metric and has to be included in a SLA, is that the customers are able to keep track on the parameters of the agreed cloud service and contact with the provider's support system, if any problem occurs.

Category	*Priority Group*	*Scale*
Availability	Important	Nominal (SQO)

- **Information security methods**: Information security *methods* that are supported by the CSPs are described in order to ensure that the transmitted and stored data which belong to a customer are encrypted and kept safe from mainly malicious third parties.
  Reason: It is mandatory for the cloud service customer to be aware of the information security methods that are being used from the CSP for securing the data transferred and stored to the cloud services. As a result, cloud service customers will always be ensured that their data is being kept safe.

Category	*Priority Group*	*Scale*
Information security	Important	Nominal (SQO)

- **Information security authentication**: The authentication of information security has to do with all the *possible ways* (e.g. simple login with password) that are provided by the CSPs to verify the claimed identity of an entity (cloud service customer).
  Reason: Information security authentication seems to be another very important metric, as the cloud service customer, depending on how important considers the access to the cloud service, will be able to choose among some authentication methods, those that he thinks is the most convenient and safe not only for him and his data, but also for his different users.

Category	*Priority Group*	*Scale*
Information security	Important	Nominal (SQO)

- **Customer data backup methods**: Customer data backup methods refer to all the *methods* (e.g. verification, replication) provided by the CSPs to the cloud service customers, referring on how and when the stored data is being backed up, in order for the backup and restore functionality to be always available.
  Reason: Customer data backup methods are considered as a vital metric for a SLA, as most of the cloud service customers want to be always aware of the ways that their data is available whenever and wherever they want. They feel more confident when they know how their data is backed up and recovered, in case of an emergency or a fatal error.

Category	*Priority Group*	*Scale*
Reliability	Important	Nominal (SQO)

- **Notification of service termination**: The notification of service termination refers to the process being used for notifying the cloud service customers that their cloud service agreements are being terminated, sending the corresponding *notification* including the *notification period*.
  Reason: All the cloud service customers want to know exactly when their signed SLAs will terminate. For that reason, the notification of service termination must be included indisputably in the SLA, in order for each customer either to renew his agreement or change the signed SLA with a different one.

Category	*Priority Group*	*Scale*
Termination of service	Important	Nominal (SQO)

- **Cloud service support methods**: Cloud service support methods refer to the *methods* (e.g. telephone, email, online chat, community forums) that the cloud service customer can use in order to obtain support by the CSP.
  Reason: The existence of the cloud service support metric that indicates the available supported methods by each CSP is reasonably very important. Using cloud services, cloud service customers do not have any physical access to the computing resources of their used services, making them extremely dependent on the providers so as to resolve any hard-to-solve incidents.

Category	*Priority Group*	*Scale*
Cloud service support	Less-important	Nominal (SQO)

- **Cloud service support costs**: Cloud service support costs refer to the *costs* that the customers have to pay, for having access to the available support methods.
  Reason: Cloud service customers according to the cloud service support methods, are always aware of the available support methods that exist, but the cost of each method varies, so they should be informed about how much each support method costs, as different support methods have different costing impacts and results.

Category	*Priority Group*	*Scale*
Cloud service support	Less-important	Ratio (SLO)

- **Cloud service incident notification / reporting**: Cloud service incident notification / reporting refer to a list of *options* that the cloud service customer may use to report and notify about service incidents to the CSP. Additionally, it refers to a list of *terms and conditions* of which the CSP must disclose the details of a service outage or condition that affects the operation of the service.
  Reason: Cloud service incident notification and reporting, must be included in the provided SLA specification, as most of the customers of the cloud service want to be informed about the ways and the different methods they have to report and notify the provider, about a problematic situation as for the service that is provided to them. As a result of that, the customers is able to choose among various options, according to his needs and the importance of the incident.

Category	*Priority Group*	*Scale*
Cloud service support	Less-important	Nominal (SQO)

- **Law enforcement access**: Law enforcement access refers to the CSP's *plan* for notifying cloud service customers of any *law enforcement requests* or *violations*. Customers and providers may also be required to preserve data from deletion in anticipation of a request either by an existing regulation or practice or a specific request to retain specific data.
  Reason: Cloud service customers must be informed about the penalties they will have to deal with when they violate the terms of a law as for a cloud service, as it is common that these violations are mostly taking part accidentally without being realized by the customer. For that reason, the customer must know the plan that suits him best according to the application that he has deployed to the cloud service.

Category	*Priority Group*	*Scale*
Data management	Less-important	Nominal (SQO)

- **Cloud service governance**: Cloud service governance defines the *metrics, processes* and *standards* that enable the CSP, the cloud service customer and other roles defined in ISO/IEC 17789 to support and/or comply with their governance requirements (ISO/IEC17998 SOA Governance Framework, ISO/IEC 38500 IT Governance). The main area of concern is the way in which changes and updates to a cloud service are managed, whether the change request originates with the cloud service customer or with the CSP.
  Reason: Cloud service governance is a quite important metric which must be included in the SLALOM SLA specification, as the cloud service customer is not aware (most of the times) of the governance requirements for each cloud service, and in order to avoid any additional problems or violations, the CSP has to notify and consult the customer about the aforementioned requirements.

Category	*Priority Group*	*Scale*
Governance	Less-important	Nominal (SQO)

Metric Definitions

Each metric, in order to be explicitly consistent and defined, has to provide the customer (in our case the CSPs) with specific standardized metric definitions, keeping the basic information that is necessary to understand the measurement of a property to be observed. In other words, for every metric that was defined in the previous section, as for the metric definition, the CSP has to analyze several properties (e.g. name, category) concerning the displayed metric, by clarifying some "generalized" definitions, in order to cover all the different kinds of metrics.

Taking into consideration the ISO 19086-2 SLA specification, in combination with the SLA specification of the cloud service key providers and the projects that have been discussed above, the SLALOM SLA specification concludes that as for the metric definitions, there should be added and/or replaced the following attributes:

- **metricId**: The metricId should replace the referenceId of the ISO specification, as in the metric's parameter and rule definitions, the unique metric identifier is having again the same name (i.e. referenceId), creating a conflict between these unique keys.

- **category**: The category attribute should be added in the SLALOM SLA specification in order for the CSP to define in a more clear way, the "core" groups that each of the metrics belong to, making it easier for the cloud service customer to classify the metrics that seem more or less important to him.

- **gradeOfImportance (GoI)**: The grade of importance attribute should be added in the SLALOM SLA specification in order for the CSP to define the metrics that he is giving more importance in comparison with other metrics, making it easier for the cloud service customer to find out the most important metrics that each provider supports, and decide the one that fulfills his needs.

- **dependency**: The dependency attribute should be added in the SLALOM SLA specification in order for the CSP to define the metrics that directly or indirectly depend on other metrics, as for their usage, efficiency and performance, making it clearer for the customer to understand the effects of one metric, in combination with another.

- **definition**: The definition attribute should be removed from the SLALOM SLA specification as it is redundant, taking into consideration the name and the note attributes where the CSP is able to both name and define (including notes) the metric.

To sum up, the final metric's definition structure that the SLALOM SLA specification proposes and each metric has to follow, is the following:

metricId: A unique identifier for the metric.

name: The name of the metric.

category: The general category that the metric belongs to. *Allowable values*: Accessibility, Availability, Performance, Information security, Termination of service, Cloud service support, Governance, Data management, Reliability
(note: For the proposed "core" metrics, their category has already been defined in subsection 7.1)

unit: The unit that will be used for expressing the metric (e.g. seconds, bytes).

scale: Information on how the measurement value can be interpreted and what sort of operations can be performed on it. *Allowable values*: Nominal, Ordinal, Interval, Ratio
(note: For the proposed "core" metrics, their scale has already been defined in subsection 7.1)

gradeOfImportance (GoI): The grade of importance that the provider gives to the metric, in comparison with its covered services. *Allowable values*: 1 (very high importance) - 10 (very low

importance)

dependency: A variable indicating whether a metric depends on another metric or not. *Allowable values*: Yes, No

note: Formal and/or additional information related to the metric.

7.2 Parameters

Having defined the metric definitions, a CSP has to define the parameter definitions to effectively express each one of them. More specifically, for every metric that was defined in the previous section, the CSP has to describe the core parameters that need to be accompanied with the metrics. To this end, the CSPs achieve to express how the metric has to be counted (e.g. float, integer), what the customer should expect to observe from the specific metric of the SLA, and cover different aspects to quantify the corresponding metrics.

Taking into consideration the ISO 19086-2 SLA specification, in combination with the SLA specification of the cloud service key providers and the projects that have been discussed above, the SLALOM SLA specification concludes that as for the parameter definitions, there should be added and/or replaced the following attributes:

- **parameterId**: The parameterId should replace the referenceId of the ISO specification, for the same reason that was explained above (subsection 7.1.1).
- **definition**: The definition attribute should be removed from the SLALOM SLA specification, for the same reason that was explained above (subsection 7.1.1).

To sum up, the final parameter's definition structure that the SLALOM SLA specification proposes and each metric's parameter has to follow, is the following:

parameterId: A unique identifier for the parameter.

name: The name of the parameter.

type: The type of the parameter definition, as for the way that it should be interpreted. *Allowable values*: Integer, Decimal, String, Boolean, Byte

note: Formal and/or additional information related to the parameter.

7.3 Rules

In combination with the definition of the parameters, each metric has to be followed by its corresponding rule definitions, referring to these elements that are used to further constrain some parts of each metric and indicate possible methods for measurement. Thus, for every metric there should be

described its proposed generalized rules, including all the potential cases through, such as if/while statements, exponential increases in values, etc. As a result, the customer will be able to observe the expressions and the rules under which the metrics of the SLA should obey, so as to keep in mind how they should react in different scenarios (e.g., when the number of allowable users riches its maximum value).

Taking into consideration the ISO 19086-2 SLA specification, in combination with the SLA specification of the cloud service key providers and the projects that have been discussed above, the SLALOM SLA specification concludes that as for the rule definitions, there should be added and/or replaced the following attributes:

- **ruleId**: The ruleId should replace the referenceId of the ISO specification, for the same reason that was explained above (subsection 7.1.1).
- **ruleExpression**: The ruleExpression should be added in the SLALOM SLA specification in order for the CSP to define an exact or more general expression of the rule and the circumstances under which the metric has proper functionality. **However, to this end, it is difficult to specify a standard format for defining the aforementioned expression, as the providers serve with different priorities and capabilities, whereas the metrics have their own characteristics and requirements.**
- **consequenceOfViolation (CoV):** The consequence of violation should be added in the SLALOM SLA specification in order for the CSP to specify the consequences and/or the penalties of a potential violation of the rule that was expressed in the ruleExpression attribute, for the customer.
- **definition**: The definition attribute should be removed from the SLALOM SLA specification, for the same reason that was explained above (subsection 7.1.1).

To sum up, the final rule's definition structure that the SLALOM SLA specification proposes and each metric's rule has to follow, is the following:

ruleId: A unique identifier for the rule.

name: The name of the rule.

ruleExpression: The expression / function under which the specific metric of the SLA must obey.

consequenceOfViolation (CoV): The consequence of violation of the rule / expression.

note: Formal and/or additional information related to the rule.

7.4 Dependencies

Concepts like metrics should be represented in such a way that the relationships among them, if any, are explicit. The understanding of the relationships between different data elements of cloud service metrology is very important to create meaningful and traceable metrics. This clarifies the effects that these concepts have on one another and their importance. For that reason, in comparison with the ISO 19086-2 SLA specification, an additional extension to the aforementioned definitions addresses the need for capturing potential dependencies between different metrics. In that scenario, the customer will be able to trace and understand the relationships and dependencies among the different metrics, making it easier to compare and understand the usage and the need of each metric.

Thus, the SLALOM SLA specification concludes that as for the dependency definitions, the attributes that should be included are the following:

dependencyId: A unique identifier for the dependency.

name: The name of the dependency.

dependentMetricId: The unique identifier of the metric that the current metric has dependency with (in case of many dependencies, they should be split with a semicolon (;)).

dependencyExpression: The expression under which the metric depends on another metric.

note: Formal and/or additional information related to the dependency.

7.5 XML Schema

As XSD (i.e., XML Schemas) are used to validate the different types of XML documents, in the case of the SLALOM SLA specification, after having fully described what it should be included and specified, a XML Schema was created in order to illustrate exactly the rules that the different definitions of each different abstract metric should obey.

Following, the XML schema that was created for the SLALOM SLA specification is cited:

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">

<xs:element name="abstractMetric">
 <xs:complexType>
  <xs:sequence>

            <xs:element name="metricDefinitions">
```

```
<xs:complexType>
            <xs:annotation>
  <xs:documentation>These are the metric definitions for the Abstract Metric</xs:documentation>
</xs:annotation>
 <xs:sequence>
  <xs:element name="metricId" type="xs:string"/>
  <xs:element name="name" type="xs:string"/>
  <xs:element name="category">
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                               <xs:enumeration value="Accessibility"/>
                               <xs:enumeration value="Availability"/>
                               <xs:enumeration value="Performance"/>
                               <xs:enumeration value="Information security"/>
                               <xs:enumeration value="Termination of service"/>
                               <xs:enumeration value="Cloud service support"/>
                               <xs:enumeration value="Governance"/>
                               <xs:enumeration value="Reliability"/>
                               <xs:enumeration value="Data management"/>
                              </xs:restriction>
                     </xs:simpleType>
               </xs:element>
  <xs:element name="unit" type="xs:string"/>
                     <xs:element name="scale">
                        <xs:simpleType>
                                    <xs:restriction base="xs:string">
                                     <xs:enumeration value="Nominal"/>
                                     <xs:enumeration value="Ordinal"/>
                                     <xs:enumeration value="Interval"/>
                                     <xs:enumeration value="Ratio"/>
                                    </xs:restriction>
                        </xs:simpleType>
               </xs:element>
                     <xs:element name="gradeOfImportance">
                        <xs:simpleType>
                              <xs:restriction base="xs:integer">
                               <xs:pattern value="[1-10]"/>
                              </xs:restriction>
                        </xs:simpleType>
                     </xs:element>
                     <xs:element name="dependency">
                        <xs:simpleType>
                              <xs:restriction base="xs:string">
                               <xs:pattern value="Yes|No"/>
                              </xs:restriction>
                        </xs:simpleType>
                     </xs:element>
                     <xs:element name="note" type="xs:string" minOccurs="0"/>
 </xs:sequence>
</xs:complexType>
</xs:element>

      <xs:element name="parameterDefinitions" minOccurs="0" maxOccurs="unbounded">
```

```
    <xs:complexType>
                <xs:annotation>
      <xs:documentation>These are the parameter definitions for the Abstract Metric</xs:documentation>
    </xs:annotation>
     <xs:sequence>
      <xs:element name="parameterId" type="xs:string"/>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="type">
                                  <xs:simpleType>
                                                <xs:restriction base="xs:string">
                                                 <xs:enumeration value="Integer"/>
                                                 <xs:enumeration value="Decimal"/>
                                                 <xs:enumeration value="String"/>
                                                 <xs:enumeration value="Boolean"/>
                                                 <xs:enumeration value="Byte"/>
                                                </xs:restriction>
                                  </xs:simpleType>
                  </xs:element>
                        <xs:element name="note" type="xs:string" minOccurs="0"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>

         <xs:element name="ruleDefinitions" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
                <xs:annotation>
      <xs:documentation>These are the rule definitions for the Abstract Metric</xs:documentation>
    </xs:annotation>
     <xs:sequence>
      <xs:element name="ruleId" type="xs:string"/>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="ruleExpression" type="xs:string"/>
      <xs:element name="consequenceOfViolation" type="xs:string" minOccurs="0"/>
                        <xs:element name="note" type="xs:string" minOccurs="0"/>
     </xs:sequence>
    </xs:complexType>
   </xs:element>

   <xs:element name="dependencyDefinitions" minOccurs="0" maxOccurs="unbounded">
    <xs:complexType>
                <xs:annotation>
      <xs:documentation>These are the dependency definitions for the Abstract Metric</xs:documentation>
    </xs:annotation>
     <xs:sequence>
      <xs:element name="dependencyId" type="xs:string"/>
      <xs:element name="name" type="xs:string"/>
      <xs:element name="dependentMetricId">
                                  <xs:simpleType>
                                          <xs:restriction base="xs:string">
                                                  <xs:pattern value="^\w+(;\w+)*$"/>
                                          </xs:restriction>
                                  </xs:simpleType>
                  </xs:element>
```

```
                        <xs:element name="dependencyExpression" type="xs:string"/>
                        <xs:element name="note" type="xs:string" minOccurs="0"/>
      </xs:sequence>
     </xs:complexType>
    </xs:element>

   </xs:sequence>
  </xs:complexType>
</xs:element>

</xs:schema>
```

7.6 Practical examples

The following tables capture three (3) randomly selected examples of different metrics (*Availability*, *Elasticity*, *Simultaneous cloud service connections*), in order to depict the proposed elements of the XML Schema of the "core" SLALOM SLA metrics.

Table 11 Example 1: Availability

Metric: **Availability**	
Metric Definitions	
metricId	AVL1
name	Availability
category	Availability
unit	Percentage (%)
scale	Interval
gradeOfImportance	1
dependency	Yes
note	Indicator that the services are up and running, responding to users' requests
Parameter Definitions	
parameterId	AVL1.PR1
name	requestResponding
type	Decimal
note	Maximum number of the services' response to requests
Rule Definitions	
ruleId	AVL1.RL1
name	lackOfAvailability
ruleExpression	while(SimultaneousCons>maxCons)
consequenceOfViolation	Unavailable cloud service
note	Service not running due to exceedance of the supported concurrent

	connections
Dependency Definitions	
dependencyId	AVL1.DP1
name	ConnectionDep
dependentMetricId	SML1
dependencyExpression	AVL1 = totalErrors/SML1
note	Higher availability is achieved with more simultaneous connections

Table 12 Example 2: Elasticity

Metric: **Elasticity**	
Metric Definitions	
metricId	EL1
name	Elasticity
category	Performance
unit	milliseconds
scale	Ratio
gradeOfImportance	1
dependency	Yes
note	Dynamic adjustment of the service resources
Parameter Definitions	
parameterId	EL1.PR1
name	adjustementTime
type	Decimal
note	Minimum required time for resources' adjustment
Rule Definitions	
ruleId	EL1.RL1
name	unsuccesfulAdjustement
ruleExpression	while(EL1.PR1<tolleranceTime && availability=false)
consequenceOfViolation	Unavailable cloud service
note	Service not running due to lack of resources
Dependency Definitions	
dependencyId	EL1.DP1
name	availabilityDep
dependentMetricId	AVL1
dependencyExpression	EL1 = a * (1/AVL1), *where a = random constant*
note	Elasticity depends on the availability of the cloud service

Table 13 Example 3: Simultaneous cloud service connections

Metric: **Simultaneous cloud service connections**
Metric Definitions

metricId	SML1
name	simultaneousConnections
category	Performance
unit	Number
scale	Ratio
gradeOfImportance	1
dependency	No
note	Maximum number of simultaneous connections
Parameter Definitions	
parameterId	SML1.PR1
name	concurrentConnections
type	Integer
note	Maximum number of concurrent cloud service connections
Rule Definitions	
ruleId	SML1.RL1
name	lackOfConnections
ruleExpression	-
consequenceOfViolation	Service becomes unavailable to users
note	Service is not running until adjustment of resources

7.7 Conclusions

7.7.1 Top key metrics

Having fully defined the "core" metrics that should be included in the SLALOM SLA specification, in combination with how they should be specified (metric definition, parameter definition, rule definition, dependency definition), it is easy to conclude that the metrics that are the most important and should be definitely included in the SLA are the following:

- Availability
- Response time
- Elasticity
- CPU
- Disk space

These metrics for both the ISO SLA specification (ISO 19086-2) and the key providers/projects that have been discussed in the previous subsections, are considered to be the most important and consist the top priority for each CSP and customer. Observing the groups that these metrics are characterized that they belong to, someone is able to see that the objectives of the cloud service availability and performance, are of major importance.

7.7.2 SLALOM SLA specifications for customers

Another important part that should be mentioned is the fact that the SLALOM SLA specification could not refer only to the CSPs, in order to form their SLAs for the cloud services that they provide, but it could also be optimized to be displayed to the cloud service customers, giving them the ability to choose the preferred SLA.

In more details, during the phase of the SLA management instantiation and negotiation, the customer could observe and interact with the following attributes of the four (4) different definitions (metric, rule, parameter, dependency) that were proposed on subsection 7.1.

Table 14 Customers' SLALOM SLA specifications

Metric Definition	Rule Definition	Dependency Definition	Parameter Definition
metricId: Visible **name**: Visible **category**: Visible **unit**: Visible **scale**: Non-visible **GoI**: Visible **dependency:** Visible **note**: Visible	**ruleId**: Non-visible **name**: Visible **ruleExpression:** Visible **CoV**: Visible **note**: Visible	**dependencyId**: Non-visible **name**: Visible **dependentMetricId**: Visible **dependencyExpression**: Visible **note**: Visible	**parameterId**: Non-visible **name**: Non-visible **type**: Non-visible **note**: Non-visible

7.7.3 SLALOM SLA additional attributes for customers

One additional part that should be mentioned, is the fact that the SLALOM SLA specification would not only be able to be editable by the CSPs, but also the customers should have the ability to edit certain attributes of the aforementioned definitions, in order to manage and negotiate according to their preferences, the SLA that will finally fit their needs and requirements.

As a consequence, the customers will be able to change the SLAs anytime they want, according to their needs of availability, performance, as well as costing, as they will be able to reduce the total amount of money that they have to pay to the CSP. So, in that scenario, the customer could change the SLA when it should be necessary, to serve in the best and more economical way his applications' users. For

example, if the SLA provides the customer with response time equal to 10msec, in the case that the customer prefers less response time for serving faster his concurrent applications' users, he could add the following ruleExpression in the corresponding field:
if users>10,000 then ResponseTime <10 msec.
To this end, it is difficult to specify a standard format for the editable attributes for the customer, as the customers with different priorities and capabilities will complete different rules and expressions, for the same characteristic.

In more details, during the SLA negotiation phase, the customer could edit and parameterize according to his specific needs, the following attributes of the four (4) different definitions that were proposed on subsection 7.1.

Table 15 Additional attributes for customers

Metric Definition	Rule Definition	Dependency Definition	Parameter Definition
metricId: Non-editable **name**: Non-editable **category**: Non-editable **unit**: Non-editable **scale**: Non-editable **GoI**: Editable **dependency:** Editable **note**: Non-editable	**ruleId**: Non-editable **name**: Non-editable **ruleExpression:** Editable **CoV**: Editable **note**: Non-editable	**dependencyId**: Non-editable **name**: Non-editable **dependentMetricId**: Editable **dependencyExpression**: Editable **note**: Non-editable	**parameterId**: Non-editable **name**: Non-editable **type**: Non-editable **note**: Non-editable

8 SLA Management mechanisms

This chapter highlights the initial set of mechanisms as the main required ones that will enhance the SLA management process. A short description of these research areas / outcomes along with potential uses (through scenarios) are summarized in the following paragraphs.

8.1 SLAs at different levels

Anastasia - focused on the media domain - would like to obtain a cloud service but as a non-technical cloud adopter, she can only specify terms at a “high-level” (i.e. not related to resource attributes such as CPU or memory). She wants to specify terms at her own “understandable language” level (e.g. frames per second) so that these terms will be translated to the corresponding technical ones.

The IRMOS EU project has proposed two types of SLAs, namely application and technical along with a mechanism for performing the required translation between these agreements.

8.2 Multi-level SLA interaction model

Irene is a data analyst in a financial trading company. She aims at utilizing resources / services offered from two different providers, as her service is both computational- and data-intensive and different providers offer the corresponding “best” services. To this end, a model for cloud federations is required. The model is based on automated SLA offer generation and enables the cloud adopter to negotiate an SLA with the federation and the federation looks for the best way to satisfy it by negotiating SLAs with one or more providers (on behalf of the cloud adopter).

The CONTRAIL EU project has proposed an approach for SLAs in multi-provider environments, including a scheme for SLA splitting, which allows for service-, resource-, or performance-based SLA splitting and revenue sharing / compensation provision.

8.3 SLA Negotiation across multiple layers

Yannis is a doctor and would like to deploy his eHealth application in a cloud environment. The application requires workflow management that highlights the need for software-layer negotiation on top of the infrastructure-layer negotiation. What is more, the negotiation process across these two layers should be transparent, while domain-specific (i.e. medical) knowledge should also be incorporated in the SLA lifecycle without additional human intervention.

The SLA@SOI EU project implemented a framework that enables different protocols to be injected through an engine, so as to facilitate the interaction between the different layers and entities. Since the protocol will be used for specific interaction, it may include domain specific content.

8.4 Automated SLA re-negotiation

Michael is an entrepreneur who has developed a new tourist service that mixes virtual and augmented reality. As an interactive real-time application, there are specific requirements (towards the cloud

infrastructure that hosts it) which however change during runtime based on the number of end-users. His goal is to sign a contract with an SLA provider, which will allow automated re-negotiation during runtime without human intervention. The re-negotiation should also address cases where the causes and origin of an agreement violation could be addressed by establishing again a process of negotiation.

The IRMOS EU project has developed an SLA re-negotiation mechanism that can be triggered either by the cloud adopter (e.g. change in application parameters), by one of the providers (e.g. detection of potential SLA violation) or by the application (e.g. elasticity rules). The mechanism allows for automated re-negotiation during runtime without human intervention.

8.5 Proactive SLA violation detection

Gabriel is a computer engineer at the operation center of the publication transportation company. The operation center software has been deployed and is running on a cloud infrastructure. There are specific requirements for downtime (near zero) and availability of these mission critical services. Therefore, he sets explicit values for the aforementioned terms and requires “proactive” mechanisms that will protect his application from any performance degradation or failure.

The VISION Cloud EU project has proposed a proactive SLA violation detection mechanism that bases estimations on the analysis of monitoring data. The analysis enables discovering of repetitive patterns and identification of potential relationships between the different parameters to identify dependencies that may affect the evolution of the parameter values during runtime.

9 Conclusions

The current document presents an analysis of the cloud SLA landscape with respect to research outcomes, initiatives’ and standardisation bodies’ efforts, as well as the views of the stakeholders (i.e. cloud providers and adopters). The aforementioned analysis covers both the SLA specifications and the mechanisms used during the SLA lifecycle. Based on this analysis, the report proposes an initial SLA specification covering different aspects (metrics, parameters, rules and dependencies) as well as practical examples of this specification. What is more, a set of SLA management mechanisms that provide added value and enhance the current management processes are proposed.

Future work includes refinement of the initial SLA specification, alignment with other initiatives and standardisation bodies’ activities, and inclusion of legal and privacy aspects both in the SLA specification and in the SLA lifecycle process through the corresponding mechanisms that will be proposed.

10 References

[1] Amazon EC2 Service Level Agreement, http://aws.amazon.com/es/ec2/sla/
[2] Amazon S3 SLA, http://aws.amazon.com/es/s3/sla/
[3] AWS Customer Agreement, http://aws.amazon.com/es/agreement/
[4] Regions and Availability Zones, http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html
[5] Google Compute Engine Service Level Agreement (SLA), https://cloud.google.com/compute/sla
[6] Google App Engine SLA, https://cloud.google.com/appengine/sla
[7] Google App Engine Error Rates, https://cloud.google.com/appengine/sla_error_rate
[8] Microsoft Azure Service Level Agreements, http://azure.microsoft.com/en-us/support/legal/sla/
[9] Microsoft Azure SLA, http://www.microsoft.com/en-us/download/details.aspx?id=44584
[10] Manage the availability of virtual machines, https://azure.microsoft.com/en-us/documentation/articles/virtual-machines-manage-availability/
[11] Pivotal Web Services Terms of Service, http://run.pivotal.io/policies/terms-of-service/
[12] Model Contracts for the transfer of personal data to third countries, http://ec.europa.eu/justice/data-protection/document/international-transfers/transfer/index_en.htm
[13] Customer Promises, https://www.heroku.com/policy/promise
[14] OPTIMIS Project, http://www.optimis-project.eu/
[15] H. Rasheed, A. Rumpl, O. Wäldrich, W. Ziegler, “A standards-based approach for negotiating service QoS with cloud infrastructure providers” eChallenges2012, Lisbon, Portugal, 2012
[16] OPTIMIS Service Manifest, Scientific Report, http://www.optimis-project.eu/sites/default/files/content-files/document/service-manifest-scientific-report.pdf
[17] J. Tordsson, K. Djemame, D. Espling, G. Katsaros, W. Ziegler, O. Wäldrich, K. Konstanteli, A. Sajjad, M. Rajarajan, G. Gallizo, S. Nair, “Towards holistic cloud management”, European research activities in cloud computing. Newcastle: Cambridge Scholars Publishing, 2012
[18] G. Kousiouris, G. Vafiadis, M. Corrales, “A Cloud Provider Description Schema for Meeting Legal Requirements in Cloud Federation Scenarios”, Conference on e-Business, e-Services, and e-Society (I3E), Athens, Greece, 2013
[19] 4CaaSt Project, http://4caast.morfeo-project.org/
[20] S. Garcia-Gomez, M. Jimenez-Ganan, Y. Taher, C. Momm, F. Junker, J. Biro, A. Menychtas, V. Andrikopoulos, S. Strauch. "Challenges for the comprehensive management of Cloud Services in a PaaS framework." Scalable Computing: Practice and Experience, 2012
[21] J. L. Vazquez-Poletti, R. Moreno-Vozmediano, I. M. Llorente, E. Oliveros, S. Ortega, M. Jimenez, J. Soriano, A. Menychtas, “Reducing Time to Market with the Platform as a Service Cloud of the Future”, in European Research Activities in Cloud Computing, Cambridge Publishing, 2011
[22] S. Garcia-Gomez, M. Escriche-Vicente, P. Arozarena-Llopis, M. Jimenez-Ganan, F. Lelli, Y. Taher, J. Biro, C. Momm, A. Spriestersbach, J. Vogel, G. Le Jeune, A. Giessmann, F. Junker, M. Dao, S. P. Carrie, J. Niemoller, D. Mazmanov, “4CaaSt: Comprehensive management of Cloud services through a PaaS”
[23] J. F. Moore, “Predators and Prey: A New Ecology of Competition”, Harvard Business Review, May-June 1993
[24] M. Papazoglou, W. van den Heuvel, "Blueprinting the Cloud," IEEE Internet Computing, 2011
[25] Sergio García Gómez, “4CaaSt Technical Value Proposition”, 4CaaSt Whitepaper, February 2012
[26] CONTRAL Project, http://contrail-project.eu/

[27]M. Coppola, P. Dazzi, A. Lazouski, F. Martinelli, P. Mori, J. Jensen , I. Johnson, P. Kershaw, “The CONTRAIL approach to Cloud Federations”, The International Symposium on Grids and Clouds (ISGC) 2012 Academia Sinica, Taipei, Taiwan February 26 - March 2, 2012
[28]R. Cascella, L. Blasi, Y. Jegou, M. Coppola, C. Morin, “Contrail: Distributed Application Deployment under SLA in Federated Heterogeneous Clouds”, Springer, Lecture Notes in Computer Science, 2013
[29]Q-ImPrESS Project, http://www.prestoprime.org/
[30]Q-ImPrESS Overview, http://www.q-impress.eu/wordpress/welcome-to-the-q-impress-project-site/overview/index.html
[31]S. Becker, L. Bulej, T. Bureš, P. Hnetynka, L. Kapová, J. Kofron, H. Koziolek, J. Kraft, R. Mirandola, J. Stammel, G. Tamburrelli, M. Trifu, “Service Architecture Meta-Model (SAMM)”, http://www.q-impress.eu/wordpress/wp-content/uploads/2009/05/d21-service_architecture_meta-model.pdf
[32]SLA@SOI Project, http://sla-at-soi.eu/
[33]SLA@SOI Final Report, http://sla-at-soi.eu/wp-content/uploads/2008/12/SLA@SOI-FinalReport.pdf
[34]IRMOS Project, http://www.irmosproject.eu/
[35]D. Kyriazis, A. Menychtas, G. Kousiouris, K. Oberle, T. Voith, M. Boniface, E. Oliveros, T. Cucinotta, S. Berger, “A Real-time Service Oriented Infrastructure”, International Conference on Real-Time and Embedded Systems (RTES), Singapore, 2010
[36]A. Menychtas, D. Kyriazis, S. Gogouvitis, K. Oberle, T. Voith, G. Galizo, S. Berger, E. Oliveros, M. Boniface, “A cloud platform for real-time interactive applications “, 1st International Conference on Cloud Computing and Services Science (CLOSER), Noordwijkerhout, The Netherlands, 2011
[37]The IRMOS Cloud Solution, http://irmosproject.eu/Files/IRMOS_Flyer_V3.pdf
[38]Infrastructure as a Service bundle: ISONI, http://www.irmosproject.eu/isoni.aspx
[39]Interactive Realtime Multimedia Applications on Service Oriented Infrastructures, http://irmosproject.eu/Files/IRMOS_WP6_7_ISONI_White_Paper_ALUD_USTUTT_v2_0.pdf
[40]D. Kyriazis, “Cloud Computing Service Level Agreements: Exploitation of Research Results”, Brussels, June 2013
[41]ETICS Project, https://www.ict-etics.eu/
[42]ETICS project, Deliverable D4.1, “End-to-end service specification template”, https://bscw.ict-etics.eu/pub/bscw.cgi/d19910/D4.1%20End-to-End%20service%20specification%20template.pdf
[43]H. Pouyllau, G. Carofiglio, “Inter-carrier SLA negotiation using Q-Learning”, Telecommunication Systems Journal Special issue on “Socio-economic Issues of Next Generation Networks”, 2011
[44]Cloud-TM Project, http://www.cloudtm.eu/
[45]D. Didona, P. Felber, D. Harmanci, P. Romano, J. Schenker, “Identifying the Optimal Level of Parallelism in Transactional Memory Systems”, The International Conference on Networked Systems, Best Paper Award, 2013
[46]D. Didona, Paolo Romano, S. Peluso, F. Quaglia, “Transactional Auto Scaler: Elastic Scaling of In-Memory Transactional Data Grids”, 9th International Conference on Autonomic Computing (ICAC 2012), San Jose, CA, USA, 2012
[47]M. Couceiro, P. Ruivo, Paolo Romano, L. Rodrigues, “Chasing the Optimum in Replicated In-memory Transactional Platforms via Protocol Adaptation”, 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2013
[48]P. Romano, M. Leonetti, “Self-tuning Batching in Total Order Broadcast Protocols via Analytical Modelling and Reinforcement Learning”, IEEE International Conference on Computing, Networking and Communications, Network Algorithm & Performance Evaluation Symposium (ICNC'12), 2012
[49]M. Couceiro, P. Romano, L. Rodrigues, “PolyCert: Polymorphic Self-Optimizing Replication for In-Memory Transactional Grids”, ACM/IFIP/USENIX 12th International Middleware Conference (Middleware), 2011

[50]J. Paiva, P. Ruivo, P. Romano, L. Rodrigues, “AutoPlacer: scalable self-tuning data placement in distributed key-value stores”, The 10th International Conference on Autonomic Computing (ICAC 2013), San Jose, CA, USA, 2013
[51]P. Di Sanzo, F. Antonacci, B. Ciciani, R. Palmieri, A. Pellegrini, S. Peluso, F. Quaglia, D. Rughetti, R. Vitali, “A Framework for High Performance Simulation of Transactional Data Grid Platforms”, 6th International ICST Conference on Simulation Tools and Techniques (SIMUTools), Cannes, French Riviera, 2013
[52]P. Di Sanzo, D. Rughetti, B. Ciciani, F.Quaglia, “Auto-tuning of Cloud-based In-memory Transactional Data Grids via Machine Learning”, 2nd IEEE International Symposium on Network Cloud Computing and Applications (NCCA), London, UK, IEEE Computer Society Press, 2012
[53]D. Didona, P. Di Sanzo, R. Palmieri, S. Peluso, F. Quaglia, P. Romano, “Automated Workload Characterization in Cloud-based Transactional Data Grids”, 17th IEEE Workshop on Dependable Parallel, Distributed and Network-Centric Systems (DPDNS), 2012
[54]Modeling and Analysis of Real-time and Embedded Systems - MARTE, http://www.omgmarte.org/
[55]A. Menychtas, S. Garcia Gomez, A. Giessmann, A. Gatzioura, K. Stanoevska, J. Vogel, V. Moulos, “A Marketplace Framework for Trading Cloud-based Services”, 8th International Workshop on the Economics and Business of Grids, Clouds, Systems, and Services (GECON), Paphos, Cyprus, 2011
[56]A. Gatzioura, A. Menychtas, V. Moulos and T. Varvarigou, “Incorporating Business Intelligence in Cloud Marketplaces”, International Workshop on Clouds for Business and Business for Clouds (C4BB4C), Madrid, Spain, 2012
[57]A. Menychtas, A. Gatzioura, T. Varvarigou, “A Business Resolution Engine for Cloud Marketplaces”, 3rd IEEE International Conference on Cloud Computing (CloudCom), Athens, Greece, 2011
[58]G. Carrozzo, N. Ciulli, P. Donadio, A. Cimmino, “The Path Computation Element for the Network Service and Business Plane – Computation of route offers and price modelling for inter-carrier services”, 17th European Conference on Network and Optical Communications (NOC 2012)
[59]Q-ImPrESS Project, http://www.prestoprime.org/
[60]R. Calinescu, L. Grunske, M. Kwiatkowska, R. Mirandola, G. Tamburrelli, “Dynamic QoS Management and Optimisation in Service-Based Systems”, IEEE Transactions on Software Engineering 37(3), 2011
[61]S. Becker, M. Trifu, R. Reussner, “Towards Supporting Evolution of Service-Oriented Architectures through Quality Impact Prediction”, Proceedings of the First International ARAMIS Workshop, L'Aquila, Italy, July, 2008
[62]H. Koziolek, B. Schlich, C. Bilich, R. Weiss, S. Becker, K. Krogmann, M. Trifu, R. Mirandola, A. Martens, “An Industrial Case Study on Quality Impact Prediction for Evolving Service-Oriented Software”, 33rd ACM/IEEE International Conference on Software Engineering (ICSE), Software Engineering in Practice Track, ACM, 2011
[63]NIST Cloud Computing Reference Architecture, http://www.nist.gov/customcf/get_pdf.cfm?pub_id=909505
[64]GEYSERS Project, http://www.geysers.eu/
[65]A. Jamakovic, T.M. Bohnert, G. Karagiannis, “Mobile Cloud Networking: Mobile Network, Compute, and Storage as One Service On-Demand”, in “The Future Internet – Future Internet Assembly 2013: Validated Results and New Horizons” – Lecture Notes in Computer Science Volume 7858, 2013
[66]GEYSERS project, Deliverable D2.6, “Refined GEYSERS architecture, interface specification and service provisioning workflow”, http://www.geysers.eu/images/stories/D2.6-final.pdf
[67]P.Robinson, A.F. Antonescu, F. Anhalt, J. Aznar, E. Escalona, J.A. Garcia Espin, L.M. Contreras Murillo, P. Vicat Blanc, “SLA management for composite infrastructure as a Service”, whitepaper, http://www.geysers.eu/images/stories/GEYSERS_White_Paper_-_SLA_Management_For_Composite_Infrastructure_As_A_Service.pdf

[68]A.F. Antonescu, P. Robinson, L.M. Contreras-Murillo, J. Aznar, S. Soudan, F. Anhalt, et al (2012). "Towards Cross Stratum SLA Management with the GEYSERS Architecture". In ISPA 2012
[69]A.F. Antonescu, M. Thoma, P. Robinson, "Service Level Management Convergence for Future Network Enterprise Platforms", FNMS 2012
[70]Cloud4SOA Project, http://www.cloud4soa.eu/
[71]W.Ziegler, "SLAs for energy-efficient data centres: The standards-based approach of the OPTIMIS project", International Workshop on Energy-Efficient Data Centers (E2DC), Madrid 2012
[72]A. Lawrence, K. Djemame, O. Wäldrich, W. Ziegler, C. Zsigri, "Using service level agreements for optimising cloud infrastructure", International Conference ServiceWave , Ghent, 2010
[73]R. Kübert, G. Gallizo, K. Oberle, E. Oliveros , "Enhancing the SLA Framework of a Virtualized Service Platform by dynamic re-negotiation", eChallenges2010, Warsaw, Poland, 2010
[74]T. Voith, K. Oberle, M. Stein, "Quality of Service provisioning for distributed data center inter-connectivity enabled by network virtualization", Future Generation Computer Systems, Elsevier , 2011
[75]T. Cucinotta, F. Checconi, G. Kousiouris, D. Kyriazis, T. Varvarigou, A. Mazzetti, Z. Zlatev, J. Papay, M. Boniface, S. Berger, D. Lamp, T. Voith, M. Stein, "Virtualized e-Learning with Real-Time Guarantees on the IRMOS Platform", IEEE International Conference on Service-Oriented Computing and Applications, SOCA2010, Perth, Australia, 2010
[76]K. Kearney, F. Torelli, C. Kotsokalis, "SLA*: An Abstract Syntax for Service Level Agreements", Grid Computing (GRID) 2010, 11th IEEE/ACM International Conference on Grid Computing, Brussels, Belgium, 2010.
[77]W. Theilmann, J. Lambea, F. Brosch, S. Guinea, P. Chronz, F. Torelli, J. Kennedy, M. Nolan, G. Zacco, G. Spanoudakis, M. Stopar, G. Armellin, "SLA@SOI Final Report", September 2011.
[78]G. Katsaros, G. Kousiouris, S. Gogouvitis, D. Kyriazis, A. Menychtas, T. Varvarigou, "A Self-adaptive hierarchical monitoring mechanism for Clouds", Elsevier Journal of Systems and Software, 2012
[79]E. Oliveros, T. Cucinotta, S. C Phillips, X. Yang, S. Middleton, T. Voith, "Chapter: Monitoring and Metering on the Cloud", IGI Global Book: Achieving Real-Time in Distributed Computing: From Grids to Clouds, 2012
[80]Stream Project, http://www.streamproject.eu/
[81]V. Gulisano, R. Jimenez-Peris, M. Patiño-Martínez, P. Valduriez, "A Large Scale Data Streaming System", 30th IEEE Int. Conf. on Distributed Systems (ICDCS), Genoa, Italy, 2010
[82]L. Coppolino, D. De Mari, L. Romano, V. Vianello, "SLA compliance monitoring through semantic processing", Grid Computing (GRID), 2010
[83]CloudScale Project, http://www.cloudscale-project.eu/
[84]G. Brataas, E. Stav, S. Lehrig, S. Becker, G. Kopcak, D. Huljenic, "CloudScale: scalability management for cloud systems", 4th ACM/SPEC International Conference on Performance Engineering, New York, USA, 2013
[85]VISION Cloud Project, http://www.visioncloud.eu/
[86]S. Gogouvitis, V. Alexandrou, N. Mavrogeorgi, S. Koutsoutos, D. Kyriazis, T. Varvarigou, "A Monitoring Mechanism for Storage Clouds", 2nd International Conference on Cloud and Green Computing (CGC), 2012
[87]A. Voulodimos, D. Kyriazis, S. Gogouvitis, A. Doulamis, D. Kosmopoulos, T. Varvarigou, "QoS-oriented Service Management in clouds for large scale industrial activity recognition", IEEE International Conference of Soft Computing and Pattern Recognition (SoCPaR), 2011
[88]CumuloNimbo Project, http://www.cumulonimbo.eu/
[89]R. Jimenez-Peris, M. Patiño-Martinez, K. Magoutis, A. Bilas, I. Brondino, "CumuloNimbo: A Highly-Scalable Transaction Processing Platform as a Service", ERCIM News 89, Special Issue on Big Data, 2012

[90]F. Perez-Sorrosal, R. Jimenez-Peris, M. Patiño-Martinez, B. Kemme, "Elastic SI-Cache: Consistent and Scalable Caching in Multi-Tier Architectures", VLDB Journal, 2011
[91]Helix Nebula Project, http://www.helix-nebula.eu/
[92]Celar Project, http://www.celarcloud.eu/
[93]Paasage Project, http://www.paasage.eu/
[94]A. Andrieux, K. Czajkowski, A. Dan, K. Keahey, H. Ludwig, T. Nakata, J. Pruyne, J. Rofrano, S. Tuecke, M. Xu, Web Services Agreement Specification (WS-Agreement). Tech. rep. OpenGrid Forum, Mar. 2007
[95]OPTIMIS project, Cloud legal guidelines final report, http://www.optimis-project.eu/content/cloud-legal-guidelines-final-report
[96]SeaClouds Project, D4.4 Dynamic QoS verification and SLA management approach, http://www.seaclouds-project.eu/deliverables/SEACLOUDS-D4.4-Dynamic_QoS_verification_and_SLA_management_approach.pdf
[97]SeaClouds Project D2.4 Final SeaClouds Architecture, http://www.seaclouds-project.eu/deliverables/SEACLOUDS-D2.4-Final_SeaClouds_Architecture.pdf
[98]C-SIG subgroup, Cloud Service Agreement Standarisation Guidelines, http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?action=display&doc_id=6138
[99]ISO/IEC JTC 1/SC 38 Cloud Computing and Distributed Platforms, http://www.iso.org/iso/home/standards_development/list_of_iso_technical_committees/iso_technical_committee.htm?commid=601355
[100] ISO/IEC 17788:2014 , Information technology -- Cloud computing -- Overview and vocabulary, http://www.iso.org/iso/catalogue_detail?csnumber=60544
[101] ISO/IEC 27001 - Information security management, http://www.iso.org/iso/home/standards/management-standards/iso27001.htm
[102] ISO/IEC 27002:2013, Information technology -- Security techniques -- Code of practice for information security controls, http://www.iso.org/iso/catalogue_detail?csnumber=54533
[103] ISO/IEC CD 19086-1 Information technology — Cloud computing – Service Level Agreement (SLA) framework — Part 1: Overview and concepts
[104] ISO/IEC WD 19086-2, Information Technology - Cloud Computing – Service Level Agreement (SLA) Framework and Terminology — Part 2: Metrics

11 Glossary of Acronyms

Acronym	*Definition*
Amazon EC2	Amazon Elastic Compute Cloud
Amazon EBS	Amazon Elastic Block Size
Amazon S3	Amazon Simple Storage Service
AWS	Amazon Web Service
C-SIG	Cloud Select Industry Group
CSP	Cloud Service Provider
EEA	European Economic Area
EFTA	European Free Trade Association
EU	European Union
IaaS	Infrastructure as a Service
MSA	Master Service Agreement
PaaS	Platform as a Service
PWS	Pivotal Web Service
SaaS	Software as a Service
SLA	Service level Agreement
SLO	Service level Objective
SQO	Service Qualitative Objective