SLA specification and reference model - c

D3.6

Dissemination level: Public

Work Package	**WP3, Technical Track**
Due Date:	M18
Submission Date:	27/06/2016
Version:	1.0
Status	Final for submission
Author(s):	Efstathios Karanastasis (ICCS), Vassiliki Andronikou (ICCS), George Kousiouris (ICCS), Theodora Varvarigou (ICCS), Nikolaos Bakalos (ICCS), Anastasios Dalias (ICCS), Antonis Litke (ICCS), Dimitrios Zografos (ICCS), Ersi Zevgoli (ICCS), Oliver Barreto (ATOS), Ana Juan (ATOS), Aimilia Bantouna (UPRC), Panagiotis Vlacheas (UPRC), Andreas Georgakopoulos (UPRC), Kostas Tsagkaris (UPRC), Yiouli Kritikou (UPRC), Aggelos Rouskas (UPRC), Nikolaos Protonotarios (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

Figures

1 Introduction

The current document is the third and final one in the series of three deliverables of the SLALOM project that aim at proposing a specification for Cloud Service Level Agreements (SLAs). 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. Examples of metrics along with their JSON implementation are also included in the document.

Comparing to the previous (second) report, this document highlights and provides a concrete SLA specification proposition addressing the following:

- *SLA specification*: Following the analysis and assessment (through concrete SLA examples) of the SLALOM SLA specification and reference model, which was performed and presented in the previous report [2], this document describes the interaction with ISO regarding the evolving ISO 19086-2 standard in terms of blocks and definitions for different metrics, parameters and rules as well as its final outcome.

- *SLA metrics definition and examples*: Based on the SLALOM specification and reference model, specific proposals for cloud SLA metrics are provided, which are intended to be immediately usable especially by adopters, with or without modifications. For each metric the standard metric provisions used in the market, the provider's and adopter's perspective and the position proposed by SLALOM are presented. Additionally, an indicative SLO definition for the metric is provided by using the SLALOM model.

- *SLA comparability:* Even when SLA metrics descriptions are aligned, in most of the cases they are still not directly comparable. Comparability is of particular importance when it is needed to assess the SLAs of different providers of cloud services for adoption in a given application or domain of interest. The SLALOM model enables the usage of metrics for the comparative evaluation of SLA clauses.

- *SLALOM model applicability in the IoT domain*: With the advent of XaaS and the emergence of IoT, SLAs may refer to services external to the data centre. In this context, a survey was designed and conducted in cooperation with the COSMOS project (which focuses on the domain of IoT) in order to gather more information on popular IoT metrics and test the applicability of the SLALOM model for describing metrics from the IoT domain.

The report is structured as follows: Section 2 summarises the SLALOM proposed SLA specification and reference model. Section 3 presents the outcome of the efforts for alignment with ISO and Section 4 demonstrates specific examples of metrics and their JSON representations. In Section 5 work upon open issues regarding SLA terms comparability are presented along with applicability of the SLALOM reference model and specifications to the IoT domain. Conclusions are drawn in section 6.

2 SLALOM SLA Specification and Reference Model

The SLALOM specification and reference model has been built on top of standardisation approaches and working groups outcomes, current SLAs offered by commercial cloud providers, expressed views by cloud providers and adopters, and research outcomes. This analysis was documented in the first version of this report [1] and the model and specifications has been thoroughly presented in [2].

Moreover, the SLALOM specification and reference model was created with the aim to standardise the definition of SLA clauses in a manner that serves the whole lifecycle of SLAs for cloud services and overcomes the shortcomings of the few existing approaches, by eliminating ambiguities in the definition and calculation of SLA clauses and facilitating the measurement, monitoring and enforcement of SLAs to achieve non-repudiability, so that these measurements cannot be contested. Another objective was to abstract the SLA clause definitions as much as possible so as to enable the application of metrics that allow for direct comparability of SLA clauses among providers. The SLALOM reference model is ISO-compliant, utilising the classes and parameters of the ISO 19086-2 metric model, but further allows for the instantiation of a Sampling class for concretely defining the sampling process of the SLA clause. What is more all SLA clauses defined via the SLALOM model are machine understandable.

Following the ISO 19086-2 SLA specification [4] and SLALOM works, the proposed building blocks of the SLALOM SLA specification / reference model include the ones below:

- ***Metric****:* The metric block corresponds to the service metric / objective (e.g. availability). Each metric is defined through standardized metric definitions, including the basic information that is necessary to understand the measurement of a property to be observed.

- ***Parameter****:* The parameter block links the metric with a set of parameters that need to be accompanied with the metrics (expressing in detail each metric). Parameters include how the metric has to be expressed (e.g. float, integer), what the customer should expect to observe from the specific metric of the SLA, and how different aspects quantify the corresponding metrics.

- ***Rule***: The rule block refers to metric “constraints” (e.g. number of concurrent connections for a number of users metric), as 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.

- ***Dependency***: The dependency block aims at capturing the dependencies between expressed metrics (e.g. response time and bandwidth).

The SLA Components of each one of the building blocks are described in detail in the second version of the current report [2].

The SLALOM model consists of three basic layers as appears in the following figure (Figure 1).

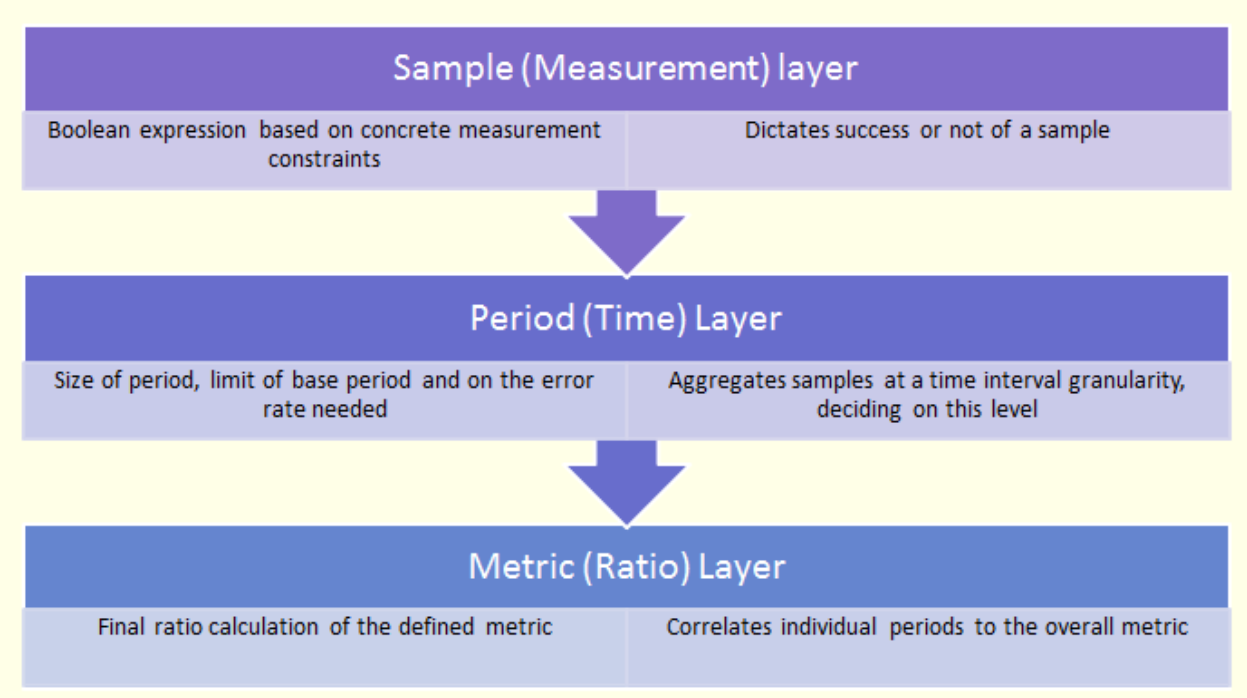

Figure 1: SLALOM proposed layer approach

The applicability of the different layers and the SLALOM model on representative commercial examples of SLAs has been showed in [2].

3 Alignment with ISO and machine understandable SLA definitions

Slalom engaged with the standards organisations ISO at a very early stage. The project achieved liaison status specifically with JTC1 SC38 WG3, which is the body producing the 19086 family of standards on cloud computing SLAs. The project has attended both physical and remote meetings, had access to publically-restricted documentation and submitted documentation to the group. As a consequence, work produced in SLALOM and especially by the technical track, reflecting on the SLALOM SLA specification and reference model, has influenced and in some cases directly appeared in the current drafts of what will be an international standard.

Initially, the SLALOM 3-layer approach was mapped to the ISO baseline model. SLALOM further demonstrated and suggested the extendibility of the ISO model for fully defining the way an SLO can be audited. In specific, SLALOM suggested the inclusion of an extension class in the ISO model, which would be instantiated as the base Sample class the of SLALOM model. This suggestion was discussed and accepted by the ISO working group, hence being able to introduce the SLALOM Sample layer for concretely defining the sampling process of an SLO or metric. Following our accepted suggestion, ISO decided in the latest revision of its draft model to make all classes extensible.

SLALOM further demonstrated the usage of the ISO model for creating directly machine understandable SLA definitions. Machine understandable SLAs can be consumed by legacy or new systems and mechanisms and consist the main vehicle for enabling automation of relevant processes throughout the lifecycle of SLAs, also aiding the composition of advanced composite cloud services and the emergence of new business opportunities covering the current needs of stakeholders and society, as described in more detail in [3].

A number of Objectives from SLAs of real world commercial providers ([5], [5], [7]) were reformulated and mapped to the SLALOM model according to the updated joint approach, including the SLALOM based extensions that are necessary in order to unambiguously declare the SLA parameters in a machine understandable case, which were provided in [2].

With relation to the Rules field, we propose the strict definition of the Rules class to be concerning the necessary preconditions to apply for a given deployment to be eligible for an SLA. Example rules of this case may include, based on a given SLA:

- Deployment in different Availability Zones
- Enablement of specific features like replication options
- Throttling of requests in case of unavailability
- Scheduled Maintenance Downtime
- etc.

Given that all concepts are depicted without the need of text, we may use the Note field as an informative placeholder of the relevant SLA text that dictated the specific section creation.

In summary, the main contribution of SLALOM in the ISO model was two-fold. SLALOM successfully demonstrated that the ISO model classes and parameters could be used for the creation of machine

understandable descriptions of SLA metrics and Objectives. SLALOM further proposed and introduced the extendibility of the ISO model, which was exploited for defining the sampling process of a metric. An SLA is ISO-compliant when the fields (classes, parameters) of the ISO model are used for the description of its Objectives and metrics. But the SLA is not necessarily fully defined. However, when an SLA is SLALOM-compliant it also is ISO-compliant and at the same time clear, well-defined and non-repudiable, i.e. the involved parties are not able to contest its measurement.

The liaison and cooperation with ISO will continue beyond the official end of SLALOM project funding period. Hence, any advances of the models and specifications will be communicated to and aligned with ISO allowing for well-targeted, unified evolution of the SLALOM results and effective sustainability of the SLALOM model beyond the project's lifetime.

4 Cloud SLA Metrics based on the SLALOM model

This section provides specific proposals for cloud SLA metrics based on the SLALOM specification and reference model described in the previous sections. These metrics are intended to be more immediately usable especially by Adopters than the generic SLALOM technical model.

For each metric, the following are provided:

- General description of the metric
- Standard metric provisions used in the market
- Provider's perspective
- Adopter's perspective
- Position proposed by SLALOM
- SLALOM proposed metric parameters
- Indicative SLO definition for the metric, based on the SLALOM reference model, where available

Parameters are discussed using the following categories:

- Measurement. This corresponds to the 'Sample (Measurement)' layer in the reference model (see section 2), also described in the text as the 'sample definition'.
- Qualification. This corresponds to the 'Period (Time)' layer in the reference model, also described in the text as the 'boundary period and error definition'.
- Result. This corresponds to the 'Metric (Ratio)' layer in the reference model, also described in the text as the 'abstract metric definition'.

4.1 Metrics: General

Provider's perspective	Adopter's perspective
Providers generally prefer availability metrics which show the Provider in the most positive way possible. This means that the following tend to be priorities for the Provider: • **Controllability.** Providers will want to avoid metrics which can be impacted by factors beyond their direct control, such as network availability when they cannot control it. Instead the emphasis is on metrics which can be measured entirely within the CSP's facilities. Defining metrics by component (e.g. storage, compute) is another way of making the metrics more controllable and	Adopters generally prefer metrics which have the following characteristics: • **End-point measurement.** Adopters will generally wish to measure performance at the point where they consume the service, without breakdowns by component which could imply satisfactory performance when overall it does not exist. • **Provider reporting responsibility.** Adopters will generally wish to have metrics and exceptions automatically reported by the Provider, with penalties automatically processed.

predictable. • **Measurability.** Providers will typically want to report availability against the criteria which are easiest for them to measure, and possibly also which provide for the least comparability with other Providers, for competitive or lock-in reasons. • **Significant impact.** Minor service exceptions generally do not have a significant impact on the customer, and therefore a threshold is needed to determine whether a service exception causes significant impact. Typically this is accomplished by requiring that the service exception persists for a designated period. Potentially, the definition can require a continuous service exception during this period, which may be impossible to demonstrate because of the periodic nature of measurements. • **Impact recognized by the customer.** The Provider should not be penalized for minor service exceptions which occur when the customer is not actually using the system. The easiest way of achieving this objective is to place the onus on the customer of identifying service exceptions.	

Position proposed by SLALOM

Responsibility for and location of monitoring. There is an inherent conflict between the principles of monitoring performance at the end-point (where the end-user experiences it), and of giving the responsibility for monitoring to the Provider (who is most capable of performing the monitoring efficiently and effectively). SLALOM's view is that it is preferable to give the contractual responsibility for performance monitoring to the Provider, with the requirement that monitoring is performed at the boundary of the Provider's infrastructure. However, the Adopter should also have the right to perform its own monitoring remotely, or via a third party. Furthermore, the Adopter should have the right to audit the Provider's own monitoring, in accordance with general audit provisions which the Adopter has vis-à-vis the Provider.

Reporting of service exceptions and determination of penalties. SLALOM's view is that it should be the responsibility of the Provider to provide detailed reports of service exceptions and metric calculations to Adopters on a regular basis, at a minimum of once each billing cycle, or monthly, whichever is less. For certain types of service exceptions, such as data breaches, there may be more demanding requirements for such reporting, whether regulatory or as agreed between the Provider and Adopter. The Provider should also automatically process any penalty consequences of the service exceptions, at least on a monthly basis.

Prohibition of special treatment of monitoring transactions. SLALOM considers that the Provider should be prohibited from implementing measures which result in monitoring transactions having better results than non-monitoring transactions. The Adopter should have the right to audit the Provider's systems for this type of issue, in accordance with general audit provisions which the Adopter has vis-à-vis the Provider.

4.2 Availability (Accessibility) Metric

General description of the metric

The most important metric for most cloud service Adopters is the availability of the cloud service, i.e. if the service is accessible for use by the end-user. This metric may be used for any layer of the cloud stack, e.g. IaaS, PaaS, and SaaS. There is an alternative availability metric defined in terms of functional performance, which is described in

Availability (Functionality) Metric.

There is comparatively limited agreement on how to define this availability metric, with differences commonly found between how it is measured, and what is excluded from the calculation, often depending on the Provider and on the level at which the service resides (VM, storage, platform, database etc.). For example, there can be availability metrics defined in terms of response times, or in terms of specific error responses when attempting to use a cloud service (e.g. obtaining a specific error response when trying to access a cloud database).

It is possible to define availability for specific components of availability, e.g. for compute service availability, storage service availability, and network availability. However, overall availability as seen by the end-user is typically the most important metric.

<table>
<tr><th colspan="2">Standard metric provisions used in the market.</th></tr>
<tr><td colspan="2">Measurement: [what things are measured, how, and where?]

Network access of Virtual Machines in running mode from locations external to the datacenter in which they are hosted.

Qualification: [what exceptions are excluded; what qualifying conditions are there for exceptions, e.g. how long does an interruption need to continue]

The main exception is for scheduled downtime. There is typically a minimum duration of the fault. If the fault is under a specific limit (5 minutes for Google, 1 minute for AWS etc.), then it does not count in the measurement level. Furthermore, for a fault to be considered, all running VM instances must be inaccessible.

Also a number of deployment preconditions must exist, such as having launched VMs in more than one availability zones (AZ). (AZs are areas of the datacenter that share the same network and power infrastructures. A cloud Adopter can select in which AZ they will launch a given instance).

Result:

[typical availability percentages offered for different levels of mission-criticality]

Typically the percentage of time in which the service is accessible divided by the overall time of the billing cycle (1 month) is used for the calculation.

Typical availability percentages are in the range of 99.95%, however the existence of different qualification levels implies that for a standard fault scenario, the calculated percentages would be different for each Provider</td></tr>
<tr><th>Provider’s perspective</th><th>Adopter’s perspective</th></tr>
<tr><td>See also the general discussion of the Provider perspective on metrics in section 2 above. This section considers only issue specific to availability metrics.

• Exclusion of planned downtime. Anything which is understood in advance to be necessary downtime should be excluded from calculations. The question is how far in advance it must be planned, and if there is a limit on how much planned downtime is permissible.</td><td>See also the general discussion of the Provider perspective on metrics in section 2 above. This section considers only issue specific to availability metrics.

• Excessive scheduled downtime. Providers may schedule downtime with little notice, resulting in no availability but with the claimed ‘scheduled’ downtime not affecting the metric.

• Who monitors the SLA? Providers often mention that the responsibility of monitoring the SLA belongs to the Adopter

• Is the monitoring process repudiable (i.e. contestible between the Provider and Adopter)? In many cases, SLAs are complex, with many hidden or ambiguous factors (e.g. how is network accessibility evaluated?</td></tr>
</table>

	Based on which protocol among many options for example?)

Position proposed by SLALOM

Acceptable downtime. SLALOM considers that acceptable downtime must comply with two conditions:

- It must be scheduled and notified to the Adopter reasonably in advance. Scheduling in advance by one week should be reasonable to expect.
- It must be reasonably limited. Limiting scheduled downtime to a maximum of 5% of contractually expected total time should be reasonable to expect.

Clear specification of the measurement process. SLALOM considers that necessary information for the Adopter or a 3rd party to provide monitoring of SLAs should be completely defined in the SLA in a non-ambiguous manner:

- Details of protocols used, timeouts used, minimum sampling rates etc. are typical examples

Billing cycle. Providers typically consider that the overall month should be considered for the calculation of the available time, regardless if the services are used by the Adopters. SLALOM's approach assesses as more reasonable the actual usage time of the services should be considered instead.

SLALOM proposed metric parameters

Measurement:

- SLALOM suggests that measurement should be based on an agreed transaction using an agreed protocol. Different options exist but with different pros and cons per case (e.g. ICMP might be a security threat, HTTP might include application server faults also that are not the IaaS Providers responsibility etc.).
- There should be an agreed interval between measurements. This is dictated by the Provider (and potentially by the latter's ability to respond to monitoring requests). In some service levels (e.g. storage) throttling considerations are used, but not in the case of the IaaS level metrics. The SLALOM proposed measurement interval depends on the minimum continuous fault time set by the Provider. However, 1 sample per minute could be considered as fine-grained enough.
- Determination of a measurement as being successful should be based on an agreed outcome within an agreed time limit.

  The agreed time limit may also depend on the interval between measurements. Thus:

  Time_limit = max (interval between measurements)

Qualification:

- Determination of a valid interruption to availability should be based on an agreed period of

continuing measurements indicating unavailability.
The SLALOM proposed value is 60 seconds, a value used by AWS.

- There should be agreed parameters to determine what constitutes scheduled downtime. The SLALOM proposed value for required advance notification is 7 calendar days.

  The SLALOM proposed value for the maximum downtime is 5% of contractually expected total time per billing period, or per calendar month, whichever is less.

Result:

- The reporting period should be agreed. Actual service running time is preferable over overall calendar billing cycle.
- The availability target should be agreed. This depends on the specific type of application and its requirements and from the Provider formula used for the calculation, thus SLALOM cannot propose a specific percentage. However SLALOM proposes the use of standardized fault scenarios that typically represent different application categories requirements. Benchmarking Provider formulas against these scenarios could be an indication of limit, as well as a directly comparable feature of Provider guaranteed availability.
- Allowed downtime = min (actual scheduled downtime, maximum permitted downtime)
- Available time = Total time expected contractually in the reporting period – allowed downtime
- Availability = [Available time – (total downtime – allowed downtime)] / (Available time)
- Simplest metric would be Availability = Available samples / Overall Samples, provided that samples follow a minimum period

Indicative SLO definition for the above metric based on the SLALOM reference model

The Indicative SLO example below is based on the above SLALOM proposed metric parameters and the Amazon EC2 Service Level Agreement.

```
{
  "name": "SLALOM Indicative Availability (Accessibility) SLO",
  "referenceId": "ASV_001",
  "scale": "NOMINAL",
  "expression": {
    "expression": "CFA_002<PARAM_002",
  },
  "parameters": [
    {
      "name": "availability_limit",
      "referenceId": "PARAM_002",
      "unit": "%",
      "parameter": "99.95"
    }
  ],
  "underlyingMetrics": [
    {
      "name": "CloudServiceAvailability",
      "referenceId": "CFA_002",
```



```
      "unit": "%",
      "scale": "RATIO",
      "expression": {
        "expression": "CFA_002 = ((BP_001 - UAP_001) / BP_001)",
      },
      "parameters": [
        {
          "name": "billing cycle",
          "referenceId": "BP_001",
          "unit": "month",
          "parameter": "1"
        }
      ],
      "underlyingMetrics": [
        {
          "name": "CloudServiceUnavailability",
          "referenceId": "UAP_001",
          "unit": "second",
          "scale": "INTERVAL",
          "expression": {
            "expression": "UAP_001 = SUM(QDT_001)",
          },
          "underlyingMetrics": [
            {
              "name": "CloudServiceUnavailability_INTERVAL",
              "referenceId": "QDT_001",
              "unit": "second",
              "scale": "INTERVAL",
              "expression": {
                "expression": "IF (QDT_001_TEMP > PARAM_001) THEN QDT_001 =
QDT_001_TEMP",
                "subExpressions": [
                  {
                    "expression": "IF (SAMPLE_001 = PARAM_002) THEN QDT_001_TEMP
= delta(SAMPLE_001.timestamp)",
                  }
                ]
              },
              "parameters": [
                {
                  "name": "boundary_period",
                  "parameter": "60",
                  "unit": "seconds",
                  "scale": "INTERVAL",
                  "referenceId": "PARAM_001"
                },
                {
                  "name": "service_ping_sample_unreachable",
                  "parameter": "unreachable",
                  "scale": "NOMINAL",
                  "referenceId": "PARAM_002"
                },
                {
                  "name": "service_ping_sample_responses",
```

```
                "referenceId": "PARAM_003",
                "parameter": [
                  "reachable",
                  "unreachable"
                ],
                "scale": "ordinal"
              }
            ],
            "rules":                                                    [
              {
                "rule": "Services deployed in at least two availability zones",
                "note": "Region Unavailable and Region Unavailability mean that
more than one Availability Zone in which you are running an instance, within the
same          Region,          is          Unavailable          to          you.",
                "referenceId":                                        "QDT_R001"
              }
            ],
            "samples":                                                  [
              {
                "name":                                  "service_ping_sample",
                "referenceId":                                      "SAMPLE_001",
                "timestamp":     "the     timestamp     of     the     sample",
                "scale":                                              "NOMINAL",
                "value":                                            "PARAM_003",
                "protocol":                                               "ICMP",
                "operation":                                              "ping",
                "note": "example sample to identify if a service is reachable
or                                                                       not"
              }
            ]
          }
        ]
      }
    ]
  }
}
```


4.3 Availability (Functionality) Metric

General description of the metric
In many cases (especially for platform level services) availability is not measured by accessibility (e.g. in terms of response times), but by the availability of specific functionality, which can also be described as 'correctness of operation'. For this type of metric, a successful return response vs. a specific range of error responses are counted as part of a ratio, which over a given period indicates whether a violation has occurred. For this metric, the same concerns as in the availability (accessibility) metric apply, but with some adaptations as to how success or failure is determined. A representative example is included based

on Google App Engine Datastore SLA.

Standard metric provisions used in the market.

Measurement: [what things are measured, how, and where?]

API calls as performed from within the framework offered by the PaaS Provider. Typically an enumerated list of specific responses identifies the ones which indicate failure.

Qualification: [what exceptions are excluded; what qualifying conditions are there for exceptions, e.g. how long does an interruption need to continue]

Adopters need to be aware that there might be preconditions from where the call is made. Calls from within the framework proposed by the Provider are typically accepted, external calls not. Also in some cases specific options need to be enabled e.g. replication options offered by the Provider. Fault periods may again be subject to a minimum interval.

Furthermore the error rate limit (number of error calls divided by overall calls) for an interval higher than the minimum qualifying one is also dictated by the Provider.

Result:

Typically again in the range of 99.95%, but with the same concerns as availability as measured by accessibility.

Provider's perspective	**Adopter's perspective**
See also the general discussion of the Provider perspective on metrics in section 2 above. The specific issue is usually with the framework used to perform the call. This should be the one dictated by the Provider.	See also the general discussion of the Provider perspective on metrics in section 2 above. In this case the specific aspect is that completely abiding to a framework specified by a single Provider may lead to vendor lock-in cases.

Position proposed by SLALOM

Same as in availability as measured by accessibility, but with the only difference that in this case the protocol is usually well defined.

SLALOM proposed metric parameters

The proposed metric parameters here are the same as the Availability (Accessibility) metric for the cases of reporting period, simplest metric used (Availability= successful samples/overall samples) as well as indicative fault scenarios.

Indicative SLO definition for the above metric based on the SLALOM reference model

The indicative example below is based on the above SLALOM proposed metric parameters. The SLA violations API responses examples stem from the Google App Engine specification.


```
{
  "name": "SLALOM Indicative Availability (Functionality) SLO",
  "referenceId": "ASV_001",
  "unit": "",
  "scale": "NOMINAL",
  "expression": {
    "expression": "CFA_002<PARAM_002",
  },
  "parameters": [
    {
      "name": "availability_limit",
      "referenceId": "PARAM_002",
      "unit": "%",
      "scale": "RATIO",
      "parameter": "99.95"
    }
  ],
  "underlyingMetrics": [
    {
      "name": "CloudServiceAvailability",
      "referenceId": "CFA_002",
      "unit": "%",
      "scale": "RATIO",
      "expression": {
        "expression": "CFA_002 = ((BP_001 - UAP_001) / BP_001)",
      },
      "parameters": [
        {
          "name": "billing cycle",
          "referenceId": "BP_001",
          "unit": "month",
          "scale": "INTERVAL",
          "parameter": "1"
        }
      ],
      "underlyingMetrics": [
        {
          "name": "CloudServiceUnavailability",
          "referenceId": "UAP_001",
          "unit": "second",
          "scale": "INTERVAL",
          "expression": {
            "expression": "UAP_001 = SUM(QDT_001)",
          },
          "underlyingMetrics": [
            {
              "name": "CloudServiceUnavailability_INTERVAL",
              "referenceId": "QDT_001",
```

```
              "unit": "second",
              "scale": "INTERVAL",
              "expression": {
                "expression": "QDT_001 = IF (DUR_001 > PARAM_001 AND ER_001 >
PARAM_002) THEN QDT_001 = DUR_001",
                "subExpressions": [
                  {
                    "expression": "DUR_001 = delta(SAMPLE_001.timestamp)",
                  },
                  {
                    "expression": "ER_001=SUM(SAMPLE_001.value belonging to
PARAM_003)/SUM(SAMPLE_001)",
                  }
                ]
              },
              "parameters": [
                {
                  "name": "boundary_period",
                  "parameter": "300",
                  "unit": "seconds",
                  "scale" : "INTERVAL",
                  "referenceId": "PARAM_001"
                },
                {
                  "name": "error_rate",
                  "parameter": "10",
                  "unit": "%",
                  "scale" : "RATIO",
                  "referenceId": "PARAM_002"
                },
                {
                  "name": "SLA VIOLATION API RESPONSES",
                  "parameter": [
                    "INTERNAL_ERROR",
                    "TIMEOUT",
                    "BIGTABLE_ERROR",
                    "COMMITTED_BUT_STILL_APPLYING",
                    "TRY_ALTERNATE_BACKEND"
                  ],
                  "scale": "NOMINAL",
                  "referenceId": "PARAM_003"
                }
              ],
              "samples": [
                {
                  "name": "datastore_API_CALL",
                  "referenceId": "SAMPLE_001",
                  "timestamp": "the time stamp of the sample",
                  "scale": "NOMINAL",
                  "value": "the response value string",
                  "protocol": "REST",
                  "operation": "API CALL",
                  "note": "example sample to identify the service response
status"
```



```
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}
```

4.4 Response Time (Transactional) Metric

General description of the metric
According to Cloud Service Measurement Index Consortium (CSMIC) framework, service response time is an attribute of the performance category. The original draft of ISO/IEC 19086-1 identified 6 metrics related to response time closely related to the service performance component while C-SIG on SLA's guidelines refer to the maximum response time SLO.
Standard metric provisions used in the market.
Measurement: [what things are measured, how, and where?] The measurement of the response time may start when the cloud Adopter initiates the stimulus on their device, or it may start when the request from the cloud Adopter arrives at the cloud service Provider's endpoint – the difference being the network transit time, which may be outside the control of the cloud service Provider. Similarly, the point at which the response is measured can vary. ***Qualification: [what exceptions are excluded; what qualifying conditions are there for exceptions, e.g. how long does an interruption need to continue]*** Many cloud services support multiple operations and thus it is likely that the response time will differ for different operations. The respective SLOs need to clearly state which operation(s) are concerned so as to avoid misunderstanding of the SLA terms. ***Result:*** Clauses and metrics well written and unambiguous to express the measurement and the qualification level.

Provider's perspective	**Adopter's perspective**
See also the general discussion of the Provider perspective on metrics in section 2 above. This section considers only issue specific to response time.	See the general discussion of the Provider perspective on metrics in section 2 above.

• 8th most important component of an SLA	

Position proposed by SLALOM

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. It refers to the performance of a service and it is rated as a highly important term in an SLA, as cloud service customers need 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. Accepting the fact that the network transit time is probably outside the control of the cloud service Providers, the measurement should start when the request from the cloud Adopter arrives at the cloud service Provider's endpoint and end at the cloud service Provider's endpoint as well. The above measurement process should be explicitly stated within the SLA.

SLALOM proposed metric parameters

Measurement:

sc <= 1 sec

Samples regarding response time obtained through different requests (e.g. sequential, parallel, from different locations, etc). Either one or more than one sample conditions can be defined.

Qualification:

bp < 30 sec

Boundary period of e.g., 30 secs reflecting for example the HTTP timeout period, within which requests not accommodated, will not be counted as actual non-responsiveness.

ec < 7%

Error condition (response) reflecting the number of cases for which the response time cannot exceed the specified value of the sample definition (Measurement).

Result:

response time < 97.77 %

Metric definition with respect to availability given the boundary period and error condition (to be considered for the validation of the given availability constraint).

Indicative SLO definition for the above metric based on the SLALOM reference model

The Indicative SLO example below is based on the above SLALOM proposed metric parameters and

the Mircosoft Azure SLA for storage.


```
{
  "name": "SLALOM Indicative Transactional Response Time SLO",
  "referenceId": "MAS_001",
  "scale": "NOMINAL",
  "expression": {
    "expression": "CFA_002 < PARAM_002",
  },
  "parameters": [
    {
      "name": "availability_limit",
      "referenceId": "PARAM_002",
      "unit": "%",
      "parameter": "99.9"
    }
  ],
  "underlyingMetrics": [
    {
      "name": "Monthly Uptime Percentage",
      "referenceId": "CFA_002",
      "unit": "%",
      "scale": "RATIO",
      "expression": {
        "expression": "CFA_002 = 100 - AER_001",
      },
      "underlyingMetrics": [
        {
          "name": "Average Error Rate",
          "referenceId": "AER_001",
          "unit": "%",
          "scale": "RATIO",
          "expression": {
            "expression": "AER_001 = AVG(HER_001) AND HER_001 belonging to
BP_001",
          },
          "parameters": [
            {
              "name": "billing cycle",
              "referenceId": "BP_001",
              "unit": "month",
              "parameter": "1"
            }
          ],
          "underlyingMetrics": [
            {
              "name": "Hourly Error Rate",
              "referenceId": "HER_001",
              "unit": "%",
              "scale": "RATIO",
              "expression": {
                "expression": "HER_001=HER_003/HER_002",
                "subExpressions": [
                  {
                    "expression": "HER_002=SUM(SAMPLE_001 belonging to
```



```
PARAM_001)",
                    "note": "Number of samples within the boundary period"
                  },
                  {
                    "expression": "HER_003=SUM(SAMPLE_001.value > PARAM_003
belonging to PARAM_001)",
                    "note": "Number of error samples within the boundary period"
                  }
                ]
              },
              "parameters": [
                {
                  "name": "boundary_period",
                  "parameter": "3600",
                  "unit": "seconds",
                  "referenceId": "PARAM_001"
                },
                {
                  "name": "GET BLOCK LIST LIMIT",
                  "value": "60",
                  "unit": "seconds",
                  "referenceId": "PARAM_003"
                },
                {
                  "name": "billing cycle",
                  "referenceId": "BP_001",
                  "unit": "month",
                  "parameter": "1"
                }
              ],
              "samples": [
                {
                  "name": "STORAGE GET BLOCK LIST API CALL response time",
                  "referenceId": "SAMPLE_001",
                  "timestamp": "the time stamp of the sample",
                  "scale": "interval",
                  "value": "the time needed to perform the operation",
                  "unit": "seconds",
                  "protocol": "REST",
                  "operation": "GetBlockList",
                  "note": "example sample to measure the response time of the
service"
                }
              ]
            }
          ]
        }
      ]
    }
  ]
}
```

4.5 Response Time (Incident) Metric

General description of the metric	
Cloud Selected Industry group (C-SIG) set-up by DG CONNECT describes response time as the "interval between a cloud service customer initiated event (stimulus) and a cloud service Provider initiated event in response to that stimulus". The DG Justice expert group expands the term of service availability to everything related to the actual functioning of the cloud service, including the quality of the service in terms of response time in case of interruption. However, not all cloud contracts contain clauses regarding response time in case of incidents while in contracts where such clauses do appear, they are often insufficiently clear or non-committal.	
Standard metric provisions used in the market.	
Measurement: [what things are measured, how, and where?] The measurement of the response time (incident) metric starts when the cloud Adopter reports an incident (which includes leaving a phone message, sending an email, or using an online ticketing system) and ends when the provider actually responds (automated responses don't count) and lets the client know they've currently working on it. When included in an SLA, it is typically expressed in terms of minutes or hours ***Qualification: [what exceptions are excluded; what qualifying conditions are there for exceptions, e.g. how long does an interruption need to continue]*** Anything outside of normal service support hours will need to be treated as an exception in some way. The Cloud Standards Customer Council (CSCC)[1] and UK Ministry of Justice[2] highlight the need for clarity with respect to time zone used when stating the service support hours. This is particularly important in cases where the cloud Adopter may expand their activity in multiple locations. Clarity is also required with respect to the definition of "weekends" and/or "holidays" and the variance of their meaning among different countries. Response time may also vary depending on the severity level or the user's prioritization. ***Result:*** Clauses and metrics well written and unambiguous to express the measurement and the qualification level.	
Provider's perspective	**Adopter's perspective**

[1] Practical Guide to Cloud Service Agreements Version 2.0 CSCC, April 2015, available at http://cloud-council.org/CSCC_Practical_Guide_to_Cloud_Service_Agreements_Version_2.0.pdf [last accessed: June 2016]

[2] Ministry of Justice guidance on Cloud Computing and CJSM, October 2012, available at http://www.lawcloud.co.uk/security/law-society-cloud-guidance [last accessed: June 2016]

See also the general discussion of the Provider perspective on metrics in section 2 above. This section considers only issue specific to response time. • Additional terms are needed with respect to the Service desk response time and Change management response time (where applicable) so as to address locality issues (e.g., timezones, bank holidays, etc.)	See the general discussion of the Provider perspective on metrics in section 2 above.

Position proposed by SLALOM

Response time (incident) metric characterizes the customer support that the cloud Provider offers. This term should be explicitly included in an SLA and it is important to clearly define working hours/days and ensure clients know that only these working hours are included in a response time. Moreover, different response time values should/may apply among different severity levels (in terms of the impact of the failure to the cloud Adopter.

SLALOM proposed metric parameters

Measurement:

Samples regarding response time obtained through different requests (e.g., type of failure, severity levels, etc.).

Qualification:

Boundary period of e.g., 0.5 (business) hour.

Error condition (response) reflecting the locality vs. (bank) holidays or non-working hours.

Result:

A table of 3-4 severity levels (e.g., Critical, High, Medium, Low) versus the response time in hours. Clarity is required with respect to the definition of the "week-ends" and/or "holidays" and the variance of their meaning among different countries.

Indicative SLO definition for the above metric based on the SLALOM reference model

The Indicative SLO example below for a medium severity incident is based on the above SLALOM proposed metric parameters.

```
{
  "name": "SLALOM Indicative Incident Response Time SLO",
  "referenceId": "IRespT_001",
  "scale": "NOMINAL",
  "expression": {
    "expression": "MIRespT < MIRespL",
  },
```



```
  "parameters": [
    {
      "name": "MediumIncidentResponseLimit",
      "referenceId": "MIRespL",
      "unit": "business hours",
      "scale": "NOMINAL",
      "parameter": "4"
      }
    ],
    "underlyingMetrics": [
      {
        "name": "MediumIncidentResponseTime",
        "referenceId": "MIRespT",
        "unit": "business hours",
        "scale": "INTERVAL",
        "expression": {
          "expression":    "MIRespT    =    ((SAMPLE_001.incident_response_time    -
SAMPLE_001.incident_report_time)/3600) - 24*PBH",
        },
        "underlyingMetrics": [
          {
            "name": "ProviderBankHolidays",
            "referenceId": "PBH",
            "unit": "days",
            "scale": "NOMINAL",
            "expression": {
              "expression": "PBH = PBH + 1 for each day belonging to PBH_List",
            },
            "parameters": [
              {
                "name": "ProviderBankHolidays_List",
                "referenceId": "PBH_List",
                "scale": "NOMINAL",
                "parameters": [
                  "2016-03-25",
                  "2016-10-28",
                  "2016-03-20",
                  "2016-03-13"
                ]
              }
            ],
            "samples": [
              {
                "name": "An incident, reported by the customer",
                "referenceId": "SAMPLE_001",
                "scale": "NOMINAL",
                "unit": "date/time",
                "incident_report_time":  "the  date/time  the  incident  was  first
reported by the customer",
                "incident_response_time":  "the  date/time  the  provider  first
responded to the incident",
                "incident_resolution_time": "the date/time the provider resolved
the incident",
                "note": "example of a sample to measure the response time for an
```

```
incident"
                }
              ]
            }
          ]
        }
    ]
}
```

4.6 Incident Resolution Time Metric

General description of the metric
ISO specified “Maximum incident resolution time” as a metric for the cloud service support component while C-SIG refers to the “resolution time” as an applicable SLO for support, i.e., the interface made available by the cloud service Provider to handle issues and queries raised by the cloud service customer and the “Percentage of timely incident resolutions” SLO in security incidents. In particular, resolution time SLO refers to the target resolution time for customer requests – in other words, the time taken to complete any necessary actions as a result of the request. This target time can vary depending on the severity level of the customer request, with shorter times attached to requests of higher severity. Percentage of timely incident resolutions SLO describes the percentage of defined incidents against the cloud service that are resolved within a predefined time limit after discovery.
Standard metric provisions used in the market.
Measurement: [what things are measured, how, and where?] Maximum incident resolution time metric reflects the maximum time within which the service Provider guarantees to have fixed an incident reported by the Adopter. When included in an SLA, it is typically expressed in terms of hours or business days. ***Qualification: [what exceptions are excluded; what qualifying conditions are there for exceptions, e.g. how long does an interruption need to continue]*** Providers usually avoid committing to the resolution time due to the diversity of the nature of errors, e.g., an error may simply need a server reboot (~5 mins) or the replacement of a hard disk (including setting up its functionality and recovering its files/data). Escalation procedures may complement the SLA when the resolution time is not met. ***Result:*** A table of 3-4 severity levels (e.g., Critical, High, Medium, Low) versus the resolution time in hours and/ or business days. Similarly to response time metric, clarity is required with respect to the definition of the "weekends" and/or "holidays" and the variance of their meaning among different countries.

Provider’s perspective	**Adopter’s perspective**

See the general discussion of the Provider perspective on metrics in section 2 above.	See also the general discussion of the Provider perspective on metrics in section 2 above. This section considers only issue specific to incident resolution time. ▪ Poor resolution of incidents is one of the 3 key problems of MSAs based on Adopter's feedback

Position proposed by SLALOM

The main issue with respect to this metric is rather that it is rarely mentioned in cloud SLAs used in the market. SLALOM's position is that this metric should be commonly used.

SLALOM proposed metric parameters

Measurement:

Maximum incident resolution time = [(Timestamp when the problem is fixed – timestamp when the incident was initially reported)/3600] hours

Maximum incident resolution time = [(Timestamp when the problem is fixed – timestamp when the incident was initially reported)/86400] days

Qualification:

# of bank holidays (on the Providers side) when the metric is expressed in business days

Result:

Max. Incident Resol. Time < x hours or

Max. Incident Resol. Time - # of bank holidays included during resolution < x business days

Indicative SLO definition for the above metric based on the SLALOM reference model

The Indicative SLO example below for a high severity incident is based on the above SLALOM proposed metric parameters.


```
{
  "name": "SLALOM Indicative Incident Resolution Time SLO",
  "referenceId": "IRT_001",
  "scale": "NOMINAL",
  "expression": {
    "expression": "SIRT < SIRL",
  },
  "parameters": [
    {
      "name": "SevereIncidentResolutionLimit",
      "referenceId": "SIRL",
      "unit": "business days",
```



```
      "scale": "NOMINAL",
      "parameter": "2"
      }
    ],
    "underlyingMetrics": [
      {
        "name": "SevereIncidentResolutionTime",
        "referenceId": "SIRT",
        "unit": "business days",
        "scale": "INTERVAL",
        "expression": {
          "expression":    "SIRT    =    ((SAMPLE_001.incident_resolution_time    -
SAMPLE_001.incident_report_time)/86400) - PBH",
        },
        "underlyingMetrics": [
          {
            "name": "ProviderBankHolidays",
            "referenceId": "PBH",
            "unit": "days",
            "scale": "NOMINAL",
            "expression": {
              "expression": "PBH = PBH + 1 for each day belonging to PBH_List",
            },
            "parameters": [
              {
                "name": "ProviderBankHolidays_List",
                "referenceId": "PBH_List",
                "scale": "NOMINAL",
                "parameters": [
                  "2016-03-25",
                  "2016-10-28",
                  "2016-03-20",
                  "2016-03-13"
                ]
              }
            ],
            "samples": [
              {
                "name": "An incident reported by the customer",
                "referenceId": "SAMPLE_001",
                "scale": "NOMINAL",
                "unit": "date/time",
                "incident_report_time":  "the  date/time  the  incident  was  first
reported by the customer",
                "incident_response_time":  "the  date/time  the  provider  first
responded to the incident",
                "incident_resolution_time": "the date/time the provider resolved
the incident",
                "note": "example of a sample to measure the resolution time for
an incident "
              }
            ]
          }
        ]
```

```
      }
    ]
}
```

4.7 Performance of Virtual Cores Metric

General description of the metric
Performance of virtual cores indicates the ability of the virtualized resource (e.g. VM) to handle a computational task. This cannot be based on any one given metric given that it is a complex process that depends on aspects such as clock frequency, RAM and cache sizes and technology, how the application may utilize the resources (e.g. leading to many cache misses for example). Thus typically performance of (virtual or otherwise) cores relies on the use of benchmark tests, that are associated with a specific KPI indicative of the resource's ability to serve the respective workload.
Standard metric provisions used in the market.
To the best of our knowledge there are no guarantees in the market for this aspect from commercial cloud service Providers. In some cases a core capacity is provided, however based on Provider specific metrics that are vague and not comparable with external services (e.g. AWS Compute Units). Typically Providers guarantee the allocation of the number of cores and RAM of a given virtual resource (e.g. VM). However due to workload consolidation management, more virtual cores may have been assigned on a physical node than the available physical ones, leading to overlap. Even if this does not happen, the issue of VM interference **Error! Reference source not found.** even when using separate cores is also a factor that affects performance and Adopter Quality of Experience. In some cases dedicated hosts may be provided as an option by Providers ***Measurement: [what things are measured, how, and where?]*** Core number, size of RAM (different options may apply or able to be set by the Adopter) ***Qualification: [what exceptions are excluded; what qualifying conditions are there for exceptions, e.g. how long does an interruption need to continue]*** N/A ***Result:*** VM is allocated with the agreed size

Provider's perspective	**Adopter's perspective**
See also the general discussion of the Provider perspective on metrics in section 2 above. This section considers only issue specific to actual performance of a given VM.	See also the general discussion of the Adopter perspective on metrics in section 2 above. This section considers only issue specific to experienced performance of virtual cores.

• **User based selection of VM sizes**. The user may select the size of their VMs, typically from a preselection of types or in some cases by defining their own size. • **Indicative capability of VM size**. Providers indicate the expected computational capability of the VMs in some way (mostly static, e.g. compute units) and do not guarantee the stability of the runtime performance. In some cases they also indicate the fitness for a purpose of a specific offering (e.g. GPU enhanced for graphics, SSD-enhanced for storage I/O etc)	• **Stability of experienced performance.** In many cases the Adopters are more interested not in absolute performance values but in the stability of the experienced performance. This is especially needed for giving their end users a stable environment for the services, as well as to be able to calculate accurately the pricing of the services residing in virtual resources (if the Adopters are e.g. SaaS Providers that rent IaaS level services). • **Mapping of performance and cost to application level metrics.** Adopters need an abstracted way with which they can understand the ability of a specific virtual resource to handle a specific type of application, and how would this be translated to a KPI level for their end users.

Position proposed by SLALOM

Defined benchmarks based on application categories. Benchmarking should use tests that are indicative of specific application categories and directly understood by the users. Thus metrics such as FLOPS, MB/sec etc. should be replaced by application level metrics that are typical in such benchmarks. An indicative categorization appears in **Error! Reference source not found.**.

Defined benchmarking process iterated periodically. Given the cloud's dynamic environment, any benchmarking process should be repeated periodically, and in a manner that covers different time zones or usages of cloud services (e.g. business hours, entertainment hours etc.). The execution of the benchmarks should be agnostic to the Provider, if performed by the Adopter or a 3rd party on his behalf.

Limits on deviation of benchmark values. Limits should exist in the SLA for which the tolerance in deviation is acceptable.

SLALOM proposed metric parameters

Measurement:

Execute agreed benchmarks on an agreed time period/schedule, no other workload (e.g. Adopter-side generated) should be present concurrently.

Indicative schedule: 3 days per week (including week ends), 3 times per measurement day covering business hours, afternoon to midnight and late night). Indicative duration of each test set: 1 hour

Qualification:

1st Case: Average percentage deviation of results from the mean value for the same benchmark, the same workload and the same size of VM should be less than a limit across all measurements, at least

for the worst case side.

2nd Case: Another more static case could be that the deviation of the minimum and maximum value from the mean value for the same benchmark, the same workload and the same size of VM should not be larger than a limit.

Agreed mean values should also be present for a given benchmark, workload and VM size.

Indicative values cannot be given since this is heavily dependent on the type of benchmarks used, workloads etc.

Result:

1st Case:

100 * average[(abs(measurement - average(all measurements)) / average(all measurements)] < X%

2nd Case:

100 * max(measurement) - average(all measurements) / average(all measurements) < X%

(in the 2nd case max and/or min can be used, depending on if we want constraints from both sides and if the benchmark value is ascending or descending

Indicative SLO definition for the above metric based on the SLALOM reference model

The example presented here assumes that the imaginary provider issues guarantees on two levels, the average value of the metric used in the specific benchmark test and the deviation of this metric across the measurements (generic, not dependent on the specific benchmark).

The limits on the average value can be higher or lower than the value limit, depending on if the metric of the specific benchmark is ascending or descending. Only one benchmark test has been incorporated (Avrora from the DaCapo Suite)


```
{
  "name": "SLALOM Indicative Provider X vCore guarantee for Micro VM Size Offering SLO",
  "referenceId": "MAS_001",
  "scale": "NOMINAL",
  "expression": {
    "expression": "STD_001 < PARAM_002 & AVG_001><PARAM_003",
  },
  "parameters": [
    {
      "name": "deviation_limit",
      "referenceId": "PARAM_002",
      "unit": "%",
      "parameter": "10"
    },
    {
      "name": "average_value_limit",
      "referenceId": "PARAM_003",
      "unit": "operations per second",
      "parameter": "100*10^9"
    }
```



```
  ],
  "underlyingMetrics": [
    {
      "name": "Average Standard Deviation of Benchmarked Values as % of mean
value",
      "referenceId": "STD_001",
      "unit": "%",
      "scale": "RATIO",
      "expression": {
        "expression": "STD_001= 100*average[(abs(SAMPLE_001- AVG_001)/AVG_001]",
      },
      "parameters": [
        {
          "name": "billing cycle",
          "referenceId": "BP_001",
          "unit": "month",
          "parameter": "1"
        }
      ],
      "underlyingMetrics": [
        {
          "name": "Average Value of Benchmark Execution",
          "referenceId": "AVG_001",
          "unit": "",
          "scale": "interval",
          "expression": {
            "expression": "AVG_001= average(SAMPLE_001) belonging in BP_001",
         },
       "parameters": [
         {
           "name": "workload_size",
           "referenceId": "PARAM_004",
           "parameter": [
             "small",
             "default",
             "large"
           ],
           "scale": "ordinal"
         },
         {
           "name": "measurement_frequency",
           "referenceId": "PARAM_005",
           "unit": "perday",
           "value": "3"
          }
         ],
         "samples": [
          {
            "name": "DaCapo Benchmark",
            "referenceId": "SAMPLE_001",
            "scale": "interval",
            "value": "throughput",
            "unit": "operations/sec",
            "operation": "Avrora",
```

```
            "workload_type": "PARAM_004",
            "workload_value": "default",
            "frequency": "PARAM_005",
            "note": "example definition of a benchmark test"
          }
        ]
      }
      ]
    }
  ]
}
```


5 SLA comparability and applicability in the IoT domain

Two of the main open issues that the SLALOM technical team is currently working and aims to continue working beyond the end of the project's funding, are the comparability of SLA terms and the applicability of the SLALOM reference model and specifications to the IoT domain.

5.1 SLA comparability

Despite the fact that through the SLALOM / ISO model the SLA metrics descriptions may be aligned, this does not mean that they will be directly comparable. Comparability is of particular importance when there is a need for assessing the SLAs of different providers of cloud services for adoption in a given application or domain of interest. In order to be able to make direct comparisons there is the need for more abstract metrics [8], such as SLA success ratio and SLA strictness levels or for the usage of standardised datasets.

The SLA success ratio metric is based on the experience of usage of a service or provider. In the course of time, the successful or violated SLAs and total SLAs are kept track of, and their numbers are recorded. These data are used to calculate the ratio: (successful SLAs/Total SLAs).

The SLA strictness levels metric is based on the extraction of static SLO parameters of importance for a given domain or application. Then, weights are assigned to these parameters and they are normalized. The parameters along with their normalized weights are mapped to an arbitrary function, which allows for the comparative ranking of SLOs from different providers.

Standardised datasets can be used for the definition of failure scenarios which pertain to the specific characteristics of a given domain or application. Then, the SLA definition of each provider is benchmarked against these predefined scenarios which allows for comparative assessment of their behaviour for the specific application domain needs.

The SLALOM model enables the application of such metrics that allow for direct comparability because of the abstraction level that it offers. The definition of an SLA clause via the SLALOM model is abstracted as much as possible. The clause is built up gradually as a summation of internal building elements (samples, metrics and sub-metrics, thresholds and conditions), each of which are clearly and well defined and identifiable. This way, the parameters of importance can be easily identified within a metric, as well as how they affect the metric's behaviour. At the same time, these SLA clauses expressed via the SLALOM model are directly machine understandable. This significantly aids the application of metrics for the comparable evaluation of SLAs among providers as well as the automation of relevant tasks.

5.2 Applicability in the IoT domain

With the advent of XaaS (Anything as a Service) and the emergence of Internet of Things (IoT), SLAs may refer to services external to the data center. Network, IoT, big data and HPC services are increasingly becoming part of the cloud ecosystem. SLA and legal research should take a step back from the simple cloud situation and consider what requirements non-human or non-cloud service consumers may require. What clauses may be floated up from data producers? What level of quality, performance,

service guarantees, security and disaster recovery might be need for end-users building critical systems with smart cities, wearables, driverless cars and the such like? How can new services comply with the market status quo and still permit innovation?

To this end, SLALOM project in collaboration with COSMOS project (focusing on the IoT domain) designed and conducted a survey and circulated it in the IERC mailing list, the purpose of which was to investigate aspects of Cloud (or in general service oriented) SLA metrics that would be more appropriate for the IoT domain, select the highest ranking of these metrics and create example metric descriptions following the SLALOM reference model. This aims to provide more information on proving the applicability of the SLALOM model in the IoT domain or to make recommendations for improvements.

Figure 2: SLALOM-COSMOS-IERC collaboration survey

In order to select a few indicative examples from the IoT domain, the results from the survey were analyzed. Furthermore, the goal was to select more “exotic” features and not ones typically found in all Cloud services such as availability.

For the type of services, *sensing*, *data availability* and *prediction services* were the most prominent ones. From these, the features that were mostly interesting were:

Sensing services

Quality of information: this is a feature that is a combination of the sensor base capabilities and the data transfer quality, which primarily depends on the transmission medium and can be enhanced with error identification and correction techniques. While in typical service uses QoI is considered as a must-have (no one dares think of a corrupt disk as an option in Cloud computing, it should not happen in any case), in the sensor domain variations are considered reasonable due to the inherent measurement process. Thus it is a metric that was selected for description. This metric can also include other aspects (submetrics) such as e.g. maximum missing values in a data flow (e.g. % of overall values).

Data Delivery

Latency: is a measure of time delay that describes how long it takes for a packet of data to move from one designated point to another in a given system. This was indicated as specifically important by users and it is understandable since many applications depe nd on low latency for effective operation. Thus it is one of the selected metrics.

Prediction Services

Prediction error: this should be defined in terms of common model metrics (e.g. Mean Absolute Error)

Prediction horizon: this is in terms of multi-step ahead prediction in e.g. time series models. This is also heavily tied with the error. It is anticipated that the larger the horizon gets (for the same model) the larger the error will be. Thus any description of this metric should also include how the error increases when the horizon increases.

Based on the first implementations of some of the above metrics that have been described via the SLALOM reference model and specification, our proposed approach seems to be able to cover very well metrics from the IoT domain. Some open issues may exist that will be further clarified via the interaction with COSMOS project, which is still ongoing.

6 Conclusions

Following the analysis documented in the second of this series of deliverables, this (third and final edition) report provides an overview of the conducted work which led to the final proposed SLALOM SLA specification / reference model, following and extending the under-development ISO specification. Furthermore, the report provides proposals for a number of popular cloud SLA metrics, which are intended to be directly usable by cloud adopters and providers. For each of the metrics their detailed descriptions and parameters are provided, and the SLALOM position is presented, while an indicative SLO definition based on the SLALOM specification is given. Finally this report discusses some open issues which have to do with the comparability of SLAs and our cooperation with the COSMOS project for proving the applicability of the proposed SLALOM specification/model in the IoT domain.

7 References

[1] SLALOM SLA Specification and Reference Model – a – (Public Deliverable D3.2), http://slalom-project.eu/content/d32-%E2%80%93-sla-specification-and-reference-model

[2] SLALOM SLA Specification and Reference Model – b – (Public Deliverable D3.3), http://slalom-project.eu/content/slalom-sla-specification-v2-early-2016

[3] Guidance on including SLALOM in research – (Public Deliverable D3.5), http://slalom-project.eu/content/guidance-including-slalom-research

[4] ISO/IEC 19086-2, Information Technology - Cloud Computing - Service Level Agreement (SLA) Framework and Terminology - Part 2: Metrics

[5] Microsoft Azure Storage SLA text, available at: [1] https://azure.microsoft.com/en-us/support/legal/sla/storage/v1_0/

[6] Amazon EC2 Service Level Agreement, available at: https://aws.amazon.com/ec2/sla/

[7] Google App Engine Service Level Agreement, available at: https://cloud.google.com/appengine/sla

[8] Nikolas Herbst, Rouven Krebs, Giorgos Oikonomou, George Kousiouris, Athanasia Evangelinou, Alexandru Iosup, Samuel Kounev: Ready for Rain? A View from SPEC Research on the Future of Cloud Metrics. CoRR abs/1604.03470 (2016), available at: https://research.spec.org/fileadmin/user_upload/documents/rg_cloud/endorsed_publications/SPEC-RG-2016-01_CloudMetrics.pdf

[9] COSMOS-SLALOM-IERC collaboration survey, available at: https://docs.google.com/forms/d/1JmwDXyO_1hT9iR-lm1c3LCQu_zF64nf-uFnxBeGMv3g/viewform [last accessed: May 2016]

8 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
EU	European Union
IaaS	Infrastructure as a Service
IoT	Internet of Things
MSA	Master Service Agreement
PaaS	Platform as a Service
SaaS	Software as a Service
SLA	Service level Agreement
SLO	Service level Objective
XaaS	Anything as a Service

