TECHNICAL REPORT

ISO/IEC TR 23951

First edition
2020-06

Information technology — Cloud computing — Guidance for using the cloud SLA metric model

Technologies de l'information — Informatique en nuage — Recommandations pour l'utilisation du modèle métrique d'accord de niveau de service (SLA) dans le Cloud

Reference number
ISO/IEC TR 23951:2020(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see http://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 38, *Cloud Computing and Distributed Platforms.*

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

In most cases, cloud service providers (CSPs) and cloud service customers (CSCs) negotiate service level agreements (SLAs) which include service level objectives (SLOs) and service qualitative objectives (SQOs) for which CSPs make commitments. The commitments described in SLAs are expected to be measured against actual performance of the service to ensure compliance with the SLA. How actual performance compares against commitments in SLAs is explained in ISO/IEC 19086-2. Cloud SLAs are covered in ISO/IEC 19086-1 and in ISO/IEC 19086-4.

The metric model in ISO/IEC 19086-2 establishes common terminology, defines a model for specifying metrics for cloud SLAs, and includes applications of the model with examples. This document provides guidance and examples on using the metric model to compose the calculation of a cloud service performance measure in order to compare against an SLA commitment. A few examples from the SLOs listed in ISO/IEC 19086-1:2016, Clause 10 are given in the document, such as Cloud Service Mean Response Time and Simple Cloud Service Availability. As specific, measurable characteristics of a cloud service, SLOs are the basis for defining the metrics used to evaluate and compare agreements between parties.

In Clauses 8, 9 and 10 of this document, a basic explanation of these examples is provided using a practical method based on a tabular format that is a refinement of the informative tables provided in ISO/IEC 19086-2:2018, Annex B. The tabular representation described in this document serves as templates for designing metrics. Guidance in using the metric model with these templates is provided while developing metric examples.

Information technology — Cloud computing — Guidance for using the cloud SLA metric model

1 Scope

The scope of this document is to describe guidance for using the ISO/IEC 19086-2 metric model, illustrated with examples.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 17788, *Information technology — Cloud computing — Overview and vocabulary*

ISO/IEC 17789, *Information technology — Cloud computing — Reference architecture*

ISO/IEC 19086-1, *Information technology — Cloud computing — Service level agreement (SLA) framework — Part 1: Overview and concepts*

ISO/IEC 19086-2, *Cloud computing — Service level agreement (SLA) framework — Part 2: Metric model*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 17788, ISO/IEC 17789, ISO/IEC 19086-1 and ISO/IEC 19086-2 apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

4 Symbols and abbreviated terms

CCRA cloud computing reference architecture

CSC cloud service customer

CSN cloud service partner

CSP cloud service provider

SLA service level agreement

SLO service level objective

SQO service quality objective

5 Structure of this document

In supporting the scope presented in Clause 1, this document develops the rationale for a practical metric representation to complement the metric model in ISO/IEC 19086-2 in the following clauses:

— Clause 6 states the rationale for complementing the metric model as defined in ISO/IEC 19086-2 with a practical representation and for providing related usage guidance as introduced by this document. It identifies some usage patterns and highlights some usage scenarios where metric definitions are shared across various parties. The users who benefit from this document include parties with roles defined in ISO/IEC 17789 (Cloud computing — Reference architecture).

— Clause 7 introduces the tabular metric representation supportive of the metric model and derived from the informative tables listed in ISO/IEC 19086-2:2018, Annex B. This representation is based on tables intended to serve as templates for metric definitions. This clause represents initial guidance in using the metric model, which is then illustrated and discussed throughout the examples developed later in the document.

— Clause 8 introduces a simple case of metric definition that illustrates the use of the table templates introduced in Clause 7. This example starts with the description of a metric as it would appear in the narrative of an existing SLA and illustrates the extraction of this description toward a more structured and distinct representation using the proposed tabular representation. The example shows practical aspects when designing and developing metrics, such as how metric rules relate to expressions, and how to parameterize rules and expressions.

— Clause 9 is a set of guidelines on how to use the metric model with the tabular templates (Clause 7). This guidance is motivated and illustrated by the examples throughout the document. These guidelines are best understood after developing a preliminary example (Clause 8). They explain how to use the metric model with the tabular templates for metric use cases posing similar challenges or using similar features.

— Clause 10 develops a more elaborate metric example for cloud service availability. It describes two variants of the same metric that illustrate two different approaches in using the metric model elements. Since it comes after the guideline items listed in Clause 9, it is easier to relate the development of this second example to these guidelines.

6 Motivation

6.1 Preamble

This clause first identifies the audience of this document and for the tabular metric representation described in this document. This clause then describes some metric usage patterns and then identifies scenarios and roles for these metric usage patterns. Sharing common guidelines and conventions in using the metric model improves the ability to reuse and compare metrics. These common guidelines extend to the aspects of a metric that are part of the metric model but the details of which are out of scope of the metric model in ISO/IEC 19086-2, such as the use of rule and expression languages and how these constructs relate to each other. Supportive of the goal of harmonizing the usage of the metric model across users, this document proposes a tabular representation for metric definitions that is derived from and augments the tables provided in Annex B of the metric model in ISO/IEC 19086-2:2018, as explained in 7.2.2.

6.2 Audience and some user categories

6.2.1 General

The audience for this document is expected to be diverse, as the metric representation proposed in this document is intended for different parties involved in providing or using cloud services. However not every clause is of interest to all. Those who read, negotiate or create SLA content, such as business users and administrators, are expected to be interested in Clauses 1 to 7 and in the initial approach to the first metric example (see 8.1). In addition to these clauses, metric designers and developers are expected to be interested in the remaining clauses including more elaborate examples of metrics (starting from 8.2 and beyond).

The parties interested in this document include representatives of the following roles defined in ISO/IEC 17788.

6.2.2 Cloud service customer (CSC)

This document helps the CSC to understand the metrics used for service quality and other assurances described in SLAs. When blended into the narrative of the SLA, metrics are often ambiguous or incomplete. A structured definition as described in the metric model and made practical with a tabular representation helps to avoid or at least detect such issues.

Specific types of customers are interested in understanding how a service is measured without having to read the entire SLA or prior to establishing an SLA. These customers are defined in the CCRA as a cloud service users (who uses a cloud service to fulfil her/his role), a service administrator (who oversees all the operational processes relating to the use of cloud services, serving as intermediary between the user and the provider) and a business manager (who has overall responsibility for the business aspects of using cloud services, including the purchase of the service under appropriate terms and possibly the request of audit reports**)**.

The tabular representation in this document is also an analysis tool for the CSC to identify and extract the metric material found in an SLA in order to get a clearer understanding of how the service is measured, as illustrated in 8.1.

6.2.3 Cloud service provider (CSP)

This document helps the CSP to describe the service metrics that support his or her SLAs, potentially avoiding contentious claims afterward that result from CSCs misunderstanding the terms and conditions of these SLAs. It also helps providers to harmonize their metrics across data-centre operators or world regions. Among activities expected from CSPs as defined in ISO/IEC 17789, the following are facilitated by metric definitions and evaluations: monitoring service, administering service security, providing audit data on request, defining and gathering metrics, managing security and risks, and, finally, handling support requests, reports and incidents from cloud service customers. For these activities, this document helps to establish a common and unambiguous representation of metrics used between parties involved in these activities and distinct from other SLA material.

6.2.4 Cloud service partner (CSN)

The following CSN sub-roles are expected to find value in a metric definition template and guidelines:

— **Cloud service developer**: this user is responsible for designing, developing, integrating, testing, and maintaining cloud services. Developers need to understand the measurements used to evaluate a cloud service. This role includes composing a new cloud service from existing separate cloud services. By having access to precise metric definitions and their rules, such as those illustrated in Clauses 8 and 10, developers understand what features are to be monitored, what is the expected quality of the developed cloud service and its priorities, as well as how to evaluate the quality and risks when integrating third party cloud services.

— **Cloud auditor**: the auditor has the responsibility of conducting an audit of the provision and the use of cloud services. A cloud audit typically covers operations, performance and security and examines whether a specified set of audit criteria are met. By using metrics, the auditor understands or communicates clearly the details of the measurements to perform. Such precision and clarity are provided by a distinct and detailed metric representation, as illustrated in the two examples developed in Clauses 8 and 10.

6.2.5 Regulators and policy makers

Several aspects of policy definition and enforcement concern measurable properties both about the cloud service usage (including cloud service usage duration and times, volume and type of data involved), and the cloud service performance (such as cloud service quality, elasticity and scalability, availability and reliability). Other policies (such as those about trust and transparency, security procedures, privacy) concern the relationship, governance and risk management between parties, especially CSCs and CSPs. Whether these policies involve automated monitoring or some human assessment instead, they rely on some form of measurement for tracking their implementation. See ISO/IEC TR 22678 for more information regarding the development of policies that govern or regulate cloud service providers (CSPs) and cloud services, and those policies and practices that govern the use of cloud services in organizations.

The expression of policies and rules sometimes translates into predefined metric elements that are expected to be used even when defining a customized metric. An example is of a policy that determines the formula (metric expression) to be used when assessing cloud service uptime percentage, while leaving other details unconstrained. As another example, if there is agreement for sharing across CSPs the common definitions of “natural disaster” or “service misuse”, the reuse of such definitions helps to establish a common understanding of what a valid cloud service downtime means. Creating and sharing predefined metric material is a usage pattern described in 6.3.5 as sharing a metric foundation.

6.3 Usage patterns

6.3.1 General

A summary of various usage patterns for the tabular metric representation given in 7.2 and a rationale for doing so are provided in the next subclauses. Some of these usage patterns match usage categories identified in ISO/IEC 19086-2:2018, 6.4.2.

6.3.2 Extract and clarify an existing metric description from an SLA

Often, the metric(s) information in a cloud service SLA is scattered over the SLA narrative. Parts of metric material (such as measurement rules, exceptions, underlying quantities and metrics) is mixed with related information that is not part of the metric definition (such as performance objectives, remediation measures and penalties).

Distinguishing a metric definition apart from its context of usage in an SLA and using for this the metric model and its concrete representation helps detect ambiguities and missing elements. This also promotes the reuse of a metric across SLAs and providers. (See 8.1 for an example of the extraction of a metric definition from an SLA narrative). This pattern of using the tabular metric representation supports the usage categories listed in ISO/IEC 19086-2:2018, 6.4.2.1 (cloud services) and 6.4.2.3 (cloud service agreements).

6.3.3 Create and share a metric description

A metric definition is sometimes intended to be used by various parties including CSP, different CSCs and CSNs including sub-roles such as cloud service developer and cloud auditor. Using a common metric representation and its usage conventions helps these parties to describe and understand metrics that they use and share.

A metric representation separate from an SLA helps different parties to share metrics while leaving aside any other SLA content. Beyond an informal plain text description understandable by all, the tabular representation introduced in 7.2 supports more formal descriptions such as specific languages for the calculation logic (expressions) and its rules, thus serving different users. See 8.1 for using different expression languages of interest to various parties. This pattern of using the tabular metric representation supports usage categories listed in ISO/IEC 19086-2:2018, 6.4.2.1 (cloud services) and 6.4.2.4 (developing performance monitoring tools).

6.3.4 Compare metric descriptions

There are many variants of a seemingly common metric across CSPs. CSCs often want to compare these. Such a comparison is made easier by using the same metric model and elements but also common representation and guidelines. For example, significant variations have been observed between CSPs in a metric as common as “service availability as uptime percentage” due to different definitions of cloud service downtime.

A well-structured metric representation makes it easier to assess comparable metric elements. This pattern of using the tabular metric representation supports usage categories listed in ISO/IEC 19086-2:2018, 6.4.2.2 (comparing cloud services).

6.3.5 Share a common foundation for a set of metrics

In many cases, it is desirable to share the same metric conventions and elements, if not the same metric. These conventions and elements are expressed as a partially developed metric definition, called a metric foundation in this document for convenience. For example, a metric foundation can be defined for cloud service availability that imposes the same general calculation of “availability” as cloud service uptime percentage, leaving the details for each CSP to define (see 6.2.5 about policies requiring to use predefined metric elements).

6.3.6 Build a metrics catalogue

A metrics catalogue collects metrics and their variants in specific areas, along with their association to useful resources such as available implementations. This is of interest to CSPs or communities of CSNs interested in sharing and reusing metric material.

A common metric representation and a set of conventions based on a shared metric model are a step toward building a metrics catalogue. This also helps to create a catalogue of metric implementations of interest to cloud service developers. Such catalogues in turn serve as resources for various parties to search, select and reuse metrics or metric elements.

6.4 Examples of scenarios and roles involved in sharing metric definitions

Metrics may be used in various ways and for different purposes, and therefore may have different measurement definitions. Consider that calculating car rental “availability” is different from calculating “airline seat availability”. CSPs may even have different definitions for the same metric. For example, some CSPs may exclude “planned maintenance outages” when calculating “availability”, while others include “planned maintenance outages”.

Consider an enterprise or government agency that purchases an IaaS service for compute. The role it plays for the IaaS provider is the CSC who consumes a compute service. The enterprise (or government agency) develops customized (PaaS) services to be reused by others in the enterprise or agency. After the PaaS services are deployed, its role is also of a PaaS CSP.

Application developers in the enterprise (or government agency) can use the PaaS to help develop the business applications (SaaS) for the benefit of their own end-users or business customers. After the SaaS services are deployed, the department that provides these services acts in the role of SaaS CSP.

Availability metrics are of interest to several roles and sub-roles with different perspectives and concerns, as illustrated in Table 1.

Table 1 — The use of cloud service availability metrics across roles

Metric: Cloud Service availability		
Role description	**Use case scenario for the metric**	**Potential availability metrics of interest**
CSP – IaaS cloud compute service provider	What level of “cloud infrastructure compute service availability” can the CSP commit to for its customers?	Cloud infrastructure compute availability
CSN – cloud PaaS service developer	The PaaS developer needs to determine what availability “targets” are reasonable when developing and deploying their PaaS services (such as APIs). Does the cloud infrastructure compute service availability commitment from the CSP enable PaaS availability intended targets to be met?	PaaS availability, Cloud infrastructure compute availability
CSC – cloud PaaS service customer, as a SaaS developer	The user needs to understand what availability “targets” they should require to satisfy their customer needs. Does the PaaS availability enable SaaS availability targets to be met? Does the “Cloud infrastructure compute service availability” CSP commitment enable SaaS and/or PaaS availability targets to be met?	SaaS availability, PaaS availability, Cloud infrastructure compute availability
SaaS customer	What application service availability can be expected by the application user?	SaaS availability

Table 1 illustrates the rationale for a metric definition that takes into account a variety of usages and accommodates a range of users. A way to ease the understanding of a metric definition across various parties is to describe it using more than one language, as illustrated in Clause 8.

Table 1 also suggests requirements for metrics to build on each other or to extend each other. An example is of an IaaS CSP who does not include connectivity as part of the availability calculation, while other roles/sub-roles are affected by outages related to network connectivity (such as between the CSNs, CSCs and the CSP) and want it to be part of their notion of availability. A more complete cloud service availability metric could compose a metric already developed for measuring the quality of network connectivity with an availability metric for the service itself.

Designing a metric for reusability makes it more versatile and supports a variety of usages and intents. This concern is briefly discussed in design options for the examples (see 8.2.4.2 and 10.2.3.2) as well as in guideline 6 about parameterization in 9.7.

7 The metric model in practice: templates

7.1 A brief reminder of the metric model

An overview of the metric model is provided in Figure 1 as it appears in ISO/IEC 19086-2 in the form of an UML diagram.

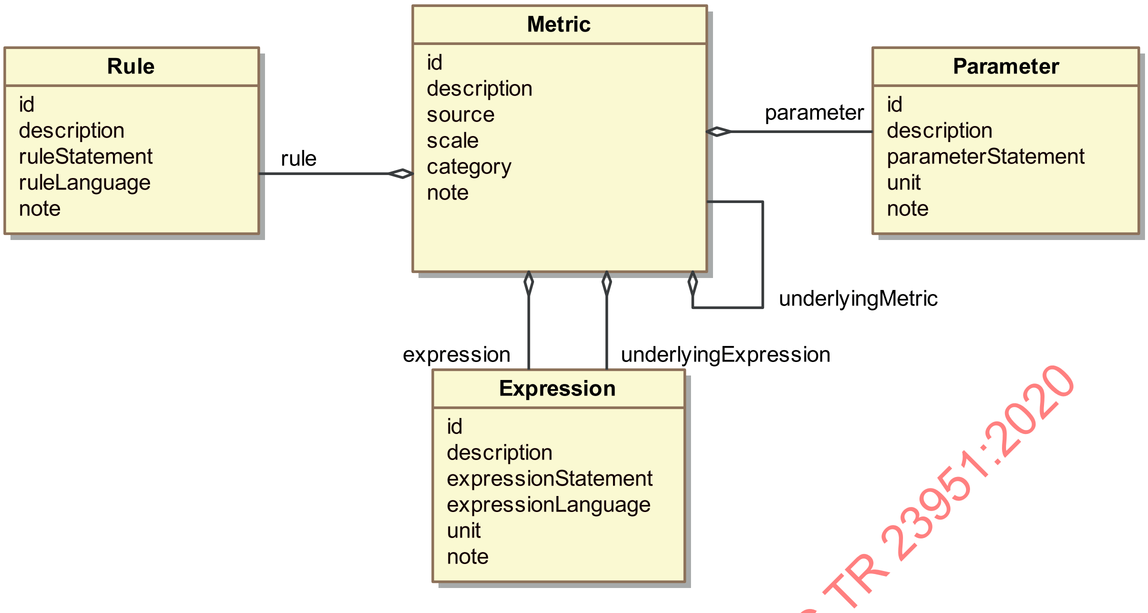

Figure 1 — The metric model in ISO/IEC 19086-2

Each one of the four blocks in Figure 1 (Metric, Rule, Expression, Parameter) is also called a metric element. A more detailed description of the metric elements and their attributes is provided in ISO/IEC 19086-2.

The term Metric element (capitalized) is used to refer specifically to the element named Metric in the UML diagram and the tabular representation of this element.

The term "metric element" is used generically to denote any element that is part of a metric, without necessary reference to a particular element type (i.e., Metric, Expression, Rule, Parameter).

7.2 A tabular representation

7.2.1 General

A tabular rendering of the metric model usable as a set of templates for capturing the definition of a particular metric is provided in Table 2, Table 3, Table 4 and Table 5. There is one table for each type of metric element (Metric, Rule, Expression, Parameter). For example, the Metric table, once filled with values, represents the definition of a Metric element. These templates are described in the next subclauses.

In the Metric table, the associations between the Metric element and its constituents (Rule, Expression and Parameter elements) are represented as references to these constituent elements using the id attribute (see associated element fields in Table 2). Such associations are illustrated as solid lines in the diagram of Figure 1.

There are other relationships between metric elements that are accounted for in the metric model in ISO/IEC 19086-2 although not represented in the diagram of Figure 1. These relationships are instead represented in ways that depend on the metric languages (ruleLanguage, expressionLanguage), the description of which is out of scope of the metric model. These relationships are between metric elements that depend on other metric elements. Examples of such dependencies are:

— An Expression element using another underlying Expression element to support its calculation.

— An Expression element referencing Rule elements that govern its calculation.

— An Expression element referencing another (underlying) Metric element of which it uses the output.

— A Rule element designed on purpose to support a particular Expression element and referencing this element.

— Rule or Expression elements parameterized by some Parameter element.

These cross-element relationships are denoted in this document as “depends on” relationships, as they indicate that an element is somehow depending on another element. Unlike the association relationships shown in the Figure 1 diagram and given explicit representation in the tabular templates (see the associated element fields in Table 1), the “depends on” relationships are indicated inside the value of some fields of the tables (such as ruleStatement, expressionStatement, or note attributes) and have no explicit support in the table templates themselves.

In other words, the “depends on” relationship representation is dictated by user-defined conventions, such as those described in ruleLanguage and expressionLanguage, and becomes apparent in the values given to these fields for a specific metric. In the tables for the metric examples of this document, it is typically represented by a reference to the id of the element that is depended on. The “depends on” relationship is represented by dashed arrows in the figures that give an overview of the structure of each metric example, such as in Figure 2 in 8.1.3.

7.2.2 The tabular representation for the Metric element

A non-normative tabular representation for metrics and their elements is provided in Annex B of the metric model in ISO/IEC 19086-2:2018. The tabular representation proposed in this document and used in the examples is a refinement of this metric model that differs mostly in its layout. Differences and their rationale are pointed out for each table.

The table for the Metric element in this document clearly separates two sets of fields and qualifies explicitly their expected content, to avoid any confusion for users:

— The attributes (content: a value)

— The associated elements (content: a reference)

The right column in the table template of Table 2 contains as an informative reminder a brief account of the field definition (derived from the one originally provided in ISO/IEC 19086-2) and its multiplicity (number of possible occurrences for that field in the table).

When used in an actual metric definition, the attributes are given a value in the right column, while the associated elements contain a reference to another metric element, meaning to another table or table row in this tabular representation.

In several examples, the id attribute is reported in the header of the metric for easier identification (in addition to or instead of appearing in the same way as any other attribute), as shown in Table 2.

Table 2 — Tabular representation for the Metric element

Metric (id)	
attribute	**value**
id	(1..1) A unique identifier for the metric.
description	(0..n) A description of the metric. A case where several instances of this attribute are used is when descriptions are provided in multiple languages.
source	(1..1) The individual or organization that created this metric definition.
scale	(1..1) Classification of the type of measurement result when using the metric. The value of scale is one of: nominal, ordinal, interval, or ratio.
category	(0..1) A grouping of metrics of similar characteristics or intent (e.g. “cloud elasticity” or “cloud service availability”) and sharing comparable expressions, rules or parameters.
note	(0..n) additional information about the metric and how to use it.

Table 2 *(continued)*

Metric (id)	
associated element	**Reference**
expression	(0..1) id of the main expression of the metric (the expression of the calculation of the metric). Expressions can reference other expression ids, metric ids, and parameter ids. This field represents a relationship to the main expression element (identified by its id) described in a separate table (see the "Set of Expressions" table).
parameter	(0..n) id of a parameter used by the metric. This field represents a relationship to a parameter (identified by its id), described in a separate table (see the "Set of Parameters" table).
rule	(0..n) id of a rule used by the metric. This field represents a relationship to a rule (identified by its id), described in a separate table (see the "Set of Rules" table).
underlyingMetric	(0..n) id of another metric used in support of this metric. This field represents a relationship to the underlying metric (identified by its id). To each referenced underlying Metric corresponds a set of tables {Metric, Set of Expressions, Set of Rules, Set of Parameters}.
underlyingExpression	(0..n) id of an expression used in support of the main expression of this metric. This field represents a relationship to an expression (identified by its id), described in a separate table (see the "Set of Expressions" table).

The order in which the fields (attributes and associated elements) are listed in the tabular templates for the Metric, Expression, Rule and Parameter elements is not significant, and is not restricted by the metric model. However, for consistency this document follows the order used in ISO/IEC 19086-2 (UML diagram reported in Figure 1 for attributes, and tables in ISO/IEC 19086-2:2018, 8.1.5 for the listing order of associations in Table 2).

The following tables are for the metric elements (Rule, Expression, and Parameter) associated with and referenced by the Metric element. The layout of these is different from Table 2 for the Metric element and from the Annex B table representation in ISO/IEC 19086-2:2018.

For each kind of metric element of multiplicity greater than one (Expression, Rule and Parameter) there is a single table where the attributes are listed horizontally instead of vertically in contrast with the previous Metric element table. This makes it possible for such a table to hold several instances of the same kind of element (one row for each instance) when associated with the same metric. It allows for a compact representation and the visual grouping of a set of related metric elements. This layout is visually convenient when metric elements refer to each other, as is often the case for expressions, or more generally for grouping elements of a same kind, as for rules. This is supportive of situations where such elements are shared together (as a group) across metrics.

7.2.3 The tabular representation for the Expression elements

The main expression element calculates the final result of a measurement. This calculation is sometimes supported by underlying expressions.

All expressions in a metric including the main expression and the underlying expressions are represented in the same table, using the template in Table 3. Expressions in a metric often share the same language. The first row under the table header indicates the default language for all expressions in the table.

Table 3 — Tabular representation for Expression elements (main or underlying)

<table>
<tr><td colspan="5">Set of Expressions
The first expression listed in this table is the metric main expression as referenced by the expression association in a metric element. It is convenient to make an explicit mention of this in the note attribute.
Expressions listed after the first expression are underlying expressions as referenced by the underlyingExpression associations in the Metric element.</td></tr>
<tr><td colspan="5">Here, the default expression language (expressionLanguage) for all expressions in the table is specified. In case an expression uses a different language, that language is indicated by an addition of the form: “expressionLanguage = <value>” under the expression itself, in the expressionStatement column.</td></tr>
<tr><td>id (1..1)</td><td>description
(0..n)</td><td>expressionStatement (1..1)
(and its expressionLanguage (1..1) if appropriate)</td><td>unit (0..1)</td><td>note
(0..n)</td></tr>
<tr><td>An identifier for the main expression, also used as placeholder for the value of the expression.</td><td>A description of the expression.</td><td>A representation of the expression, according to the language specified in “expressionLanguage” also specified in this column in case it is different from the default.</td><td>A real scalar quantity, defined and adopted by convention.</td><td>Additional information about the expression.</td></tr>
<tr><td>An identifier for an underlying expression.</td><td></td><td></td><td></td><td></td></tr>
</table>

7.2.4 The tabular representation for the Rule elements

Rules are used to give details about measurement operations, as well as to state other constraints and restrictions on the metric and its elements.

All Rule elements in a metric are represented in the same table, using the template shown in Table 4. In case there are several rules for this metric, each one has a row in this Rules table. Table 4 illustrates the table template for a set of metric rules.

Table 4 — Tabular representation for the set of Rules elements

Set of Rules			
Here the default rule language **(ruleLanguage)** for all rules in the table is specified. In case of a different language for a rule, a specific language is indicated by an addition of the form: “ruleLanguage = <value>” under the rule itself, in the ruleStatement column.			
id (1..1)	**Description** (0..n)	**ruleStatement** (1..1) (and its **ruleLanguage** (1..1) if necessary)	**note** (0..n)
A unique identifier for the rule.	A description of the rule.	A representation of the rule, according to the language specified in “ruleLanguage”, also specified in this column in case it is different from the default.	Additional information about the rule.
Identifier of the next rule.			

7.2.5 The tabular representation for the Parameter elements

A parameter is a variable the value of which is set prior to a measurement by the metric user, as opposed to resulting from a calculation, and which remains constant during an actual measurement or execution of the metric.

A single table is used to represent all Parameter elements in a metric, using the similar compact layout (one row for each parameter) used in the Set of Expressions and Set of Rules tables. Table 5 illustrates the table template for a set of metric parameters.

Table 5 — Tabular representation for a set of Parameter elements

Set of Parameters				
id (1..1)	**description** (0..n)	**parameterStatement** (1..1)	**unit** (1..1)	**note** (0..n)
A unique identifier for the parameter.	Plain text narrative that describes this parameter.	Detailed description of the parameter content such as its value type and structure (e.g. for a vector). Includes also any additional restriction on the possible values (e.g. a taxonomy or enumeration).	The unit of the values for this parameter if applicable.	Additional information about the parameter.
Identifier of the next parameter.				

8 An example of metric definition: the cloud service mean response time metric

8.1 The cloud service mean response time metric: informal variant

8.1.1 Extracting metric elements from an SLA narrative

A common usage pattern is to extract metric material from an existing SLA as mentioned in 6.3.2. This example illustrates the extraction of metric material for calculating the mean response time of a cloud service from a fictitious CSP named “ShinyCloud”. This metric named “cloud service mean response time” (informal variant) is related to the cloud service SLO named Cloud Service Response Time Mean (see SO/IEC 19086-1:2016, 10.4.2.3). It is averaging the elapsed time between all cloud service requests and their responses over a period of time, typically a billing period.

Only parts of a simplified cloud SLA are reported in this example. Consider the following excerpts from the narrative of a fictitious SLA for this cloud service:

Excerpt #1: "*We monitor each request-response transaction. We guarantee that our cloud service will return a response within 800 ms of receiving a request, as an average over a billing period. Requests-responses occurring during periods of scheduled maintenance are not counted.* "

Excerpt #2: "*...If we fail to meet this guarantee, you will be eligible to receive a credit to your account. The credit will be calculated as a percentage of your last billed fee for the cloud Files Storage service. Credits will be based on how much the average response time exceeds the guaranteed threshold as defined below....*"

Excerpt #3: "*...The request-response time is measured as the elapsed time between the reception of a service request in the CSP system as recorded in the logging system (syslog call) and the sending of the response also recorded in the logging system.*"

As a first step, the mean response time metric material used for this cloud service is extracted and it is represented using the proposed tabular format (see 7.2). To this end, the SLA narrative is analyzed to identify and isolate the metric elements:

— Excerpt #1 defines some objective for a measurement of mean response time for request-response events: mean response time to be less than or equal to 800 ms. This objective in itself is not part of the metric definition, but this excerpt also defines the overall calculation logic for the metric in use: averaging the response time over a billing period. This is captured by the expression element of the metric. The third sentence also specifies a case where request-response events are not counted (scheduled maintenance). This is represented as a rule element of the metric.

— Excerpt #2 specifies a penalty resulting from failure to meet the objectives for this metric. This contingency, like the metric objectives, is not part of the metric definition but part of the SLA context where the metric definition is used. As such, it is left out of the metric definition. This excerpt shows a case where contextual information for a metric is mixed with metric definition material in an SLA.

— Excerpt #3 specifies measurement details for this metric. These details are captured as another metric rule element.

8.1.2 Using the tabular representation

Based on this analysis, a tabular representation of the cloud service mean response time metric (informal variant) is created using the table templates introduced in 7.2.

Some optional and unused fields have been removed from the tables. The id of the metric is CSMRT-1.

The Metric element table for this metric is shown in Table 6.

Table 6 — The Metric element for the CSMRT-1 metric

Metric (id:CSMRT-1**)**	
attribute	**Value**
description	This metric is averaging response times for a cloud service over a measurement period. Measurement conditions (such as where/when the time measurements are taken) and the exclusion cases are captured by rules. The language used for both expressions and rules is plain text.
source	CSP "ShinyCloud"
scale	Ratio
category	cloud service response time
associated element	**Reference**
expression	**id**: MRT
rule	**id**: SME
rule	**id**: MM

NOTE 1 As the id attribute is listed in the header row of the table for easier identification, it is not repeated as a row in the table to avoid redundancy.

NOTE 2 When an optional attribute or association is not used in a metric, it is simply removed from the table (which is why the Metric table for CSMRT-1 is not as complete as the Metric element table in 7.2.2.) Another option is to leave the unused fields of the table unfilled.

The Set of Expressions table (capturing a single expression in this case) for the CSMRT-1 metric is shown in Table 7.

Table 7 — The Expression elements for the CSMRT-1 metric

Set of Expressions			
expressionLanguage: plain English text (default language common to all)			
Id	**expressionStatement**	**Unit**	**Note**
MRT	Average of elapsed time for request-response events over a billing period, with request-response data complying with metric rules (SME, MM).	millisecond	(The id MRT is an abbreviation for Mean response time.)

The Rule element table (capturing all rules) for the CSMRT-1 metric is shown in Table 8.

Table 8 — The Rule elements for the CSMRT-1 metric

Set of Rules			
ruleLanguage: plain English text (default language common to all)			
id	**Description**	**ruleStatement**	**Note**
SME	The rule disqualifies some of the requests-responses events.	Requests-responses overlapping with scheduled maintenance periods are not counted.	(The id SME is an abbreviation for scheduled maintenance exclusion.)
MM	This rule defines how a request-response event is measured, based on two of its properties: a starting time and an ending time.	The request-response time is measured as the elapsed time between the reception of a service request in the CSP system as recorded in the logging system (syslog call) and the sending of the response also recorded in the logging system.	(The id MM is an abbreviation for measurement method.)

There are differences between the tables used for an actual metric definition and the generic templates for related metric elements:

— Unlike in the templates (see 7.2), the cardinality of each field (attribute or associated element) is not reported in the tables used for actual metrics.

— A missing field in a metric definition means that the field is optional and not used in this metric definition: its cardinality in the template is (0..1) or (0..n).

— An added field indicates an extension to the metric model.

In this example the metric model and its tabular representation are used as an analysis tool for extracting metric material from an existing SLA. While this directly supports the usage pattern identified in 6.3.2 about metric material extraction, it also supports other usage patterns such as metric comparison (see 6.3.4) and metrics catalogues (see 6.3.6) in situations where SLAs already exist with metric data that does not conform to the metric model.

8.1.3 Overall structure of the metric

Figure 2 gives an overview of the overall structure of the CSMRT-1 metric including its elements and their usage relationships. The “depends on” relationship between metric elements (see 7.2.1) is indicated with dashed arrows.

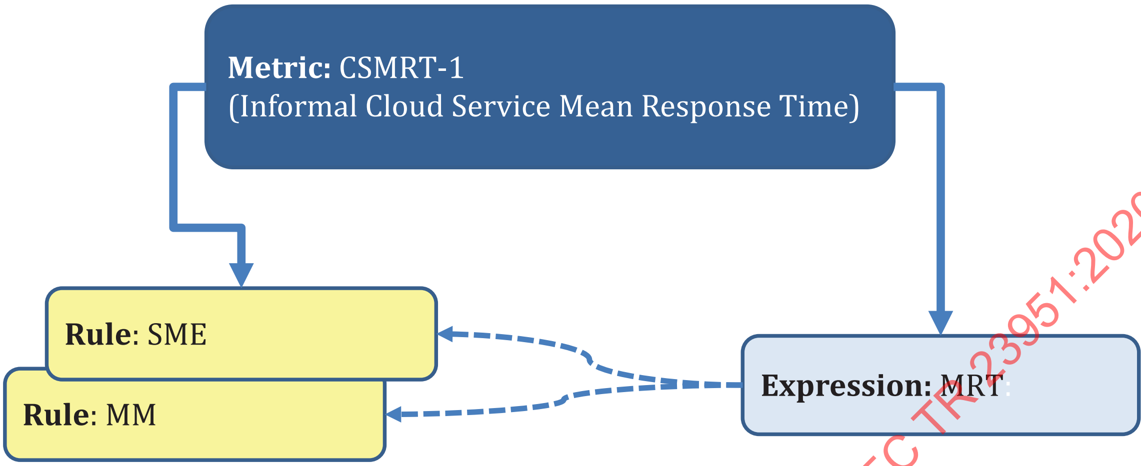

Figure 2 — The elements of the mean response time CSMRT-1 metric

8.2 The cloud service mean response time metric: more formal variant

8.2.1 A more formal variant of the metric

Consider a variant of the previous cloud service mean response time metric, with a more formal expression language. This variant would be more useful to a CSN party such as a cloud service developer (one of the users of metric definitions, as stated in 6.2). The new variant id is CSMRT-2. Besides the introduction of a more formal expression language (attribute expressionLanguage) some of the differences with the previous version of this metric example are:

— Addition of a parameter for the monthly period of measurement.

— Introduction of an underlying expression, as the formal translation of the sole expression in metric CSMRT-1 is broken down in two expressions in CSMRT-2.

The Metric element for the CSMRT-2 metric is shown in Table 9:

Table 9 — The Metric element for the CSMRT-2 metric

Metric (id:CSMRT-2**)**	
attribute	**value**
description	This metric is averaging response times for a cloud service over a measurement period. Measurement conditions (such as where/when the time measurements are taken) and the exclusion cases are captured by rules. The metric is parameterized by its measurement period. The language used for expressions is a formal language, and plain text for rules.
source	CSP “ShinyCloud”
scale	ratio
category	cloud service response time
associated element	**reference**
expression	**id**: MRT_2
parameter	**id**:MPeriod

Table 9 *(continued)*

Metric (id:CSMRT-2**)**	
rule	**id**:SME
rule	**id**: MM
underlyingExpression	**id**: QRT

8.2.2 Adding a parameter

The metric CSMRT-2 is parameterized with the monthly period of measurement. This is done by defining a Parameter element as shown in Table 10.

Table 10 — The Parameter elements for the CSMRT-2 metric

Set of Parameters				
id	**Description**	**parameterStatement**	**Unit**	**Note**
MPeriod	Time boundaries for all request-response events that participate in a measurement.	Month of the year	Label in an enumeration `{January, February, March, April, May, June, July, August, September, October, November, December}`.	A measurement parameter (expected to change at each measurement or execution of the metric.) (The id MPeriod is intended to serve as an abbreviation for monthly period.)

An alternative to this parameter representation would be to define MPeriod as a time interval (start, end) as done for the OPeriod parameter in the next metric example on Cloud Service Availability (Clause 10).

8.2.3 The metric rules

The rules are unchanged from CSMRT-1.

8.2.4 The metric expressions

8.2.4.1 Tabular representation

The expressions of this metric variant are defined as follows using the Set of Expressions table template, as shown in Table 11. The selected expression language in this example is algorithmic and uses a functional notation in the expression statements. The signature and semantics of these functions are briefly described in the expressionLanguage attribute (a more detailed description would more conveniently be given outside this table and be referenced from the table).

Table 11 — The Expression elements for the CSMRT-2 metric

<table>
<tr><td colspan="5">Set of Expressions</td></tr>
<tr><td colspan="5">expressionLanguage: algorithmic, including the following functions: (informal definitions)
— function: duration(<event>) returning a duration quantity (elapsed time between start time, end time of an event).
— function: average(<iterator>) returning an average of the values iteratively produced.
— function: apply_rules (<rule(s)>) to (<occurrence(s)>) [over (<a time period>)] is a generic function that applies rule(s) to some incidents or events that occur during a measurement, possibly bounded by a time window. The function returns an event qualified by these rules, the timing of which (start-time, end-time) is determined. See a more complete definition used in examples in guideline 2 in 9.3</td></tr>
<tr><td>id</td><td>Description</td><td>expressionStatement</td><td>Unit</td><td>Note</td></tr>
<tr><td>MRT_2</td><td>Mean duration for all qualified request-response events.</td><td>= average (QRT)</td><td>millisecond</td><td>Main metric expression, iterating over successive outputs from QRT, used as a generator (as in programming languages).</td></tr>
<tr><td>QRT</td><td>Duration of a single request-response event, qualified according to the rules SME and MM.</td><td>= duration(
apply_rules (SME, MM) to (a request-response event) over (MPeriod))
)</td><td>millisecond</td><td>This expression is iteratively evaluated for each request-response occurrence, with the iteration controlled outside the expression. The QRT symbol is used as a generator in other expressions.
MPeriod is a parameter.
(The id QRT is an abbreviation for qualified response time.)</td></tr>
</table>

8.2.4.2 Design choices

The convention in this document (see guideline 3 for expressions in 9.4) is to use the expression id, for example QRT, as a variable for holding the result of evaluating the expression to be reused in another expression (MRT_2). This intent is indicated by starting the metric expressions with "=".

The expression in the previous metric (CSMRT-1) has been broken up in two expressions (MRT_2 and QRT) in the more formal metric version (CSMRT-2). This is an intentional design choice.

A monolithic single expression is sometimes unwieldly, depending on the expression language. Breaking it up into underlying expressions helps reduce the complexity, as commonly done in algorithmic representations with functions and components.

Splitting an expression in two or more expressions, as done in this example, sometimes promotes expression reusability. In this example, the main expression (MRT_2) is independent from any rule that could be specific to a provider (about measurement details or maintenance policies). Such rules are then only associated with the underlying expression (QRT). The main expression (MRT_2) is now abstracted from the rules (SME, MM), unlike MRT. This makes MRT_2 potentially more reusable across CSPs, in metrics that have to perform similar calculations yet with different measurement rules or exclusion rules for request-response events.

Reusability of expressions is not just about facilitating implementation. It is also about ensuring a common interpretation and calculation of these expressions across metrics and their users. In turn, this promotes the reuse of their implementations and reduces the risk of unwanted variations between metrics supposed to use same calculations.

Finally, the expression QRT is designed as a value "generator" that produces a duration value for each request-response at a time. It is iteratively invoked in the MRT_2 main expression. This is in contrast with the design option consisting of an expression producing all the values at once, passed to the main expression as a list or an array (such an option is developed in the example of Clause 10).

Such a design choice depends on the context of use of the metric and other practical requirements. In this metric the collection of request-responses is potentially large and not suitable for being stored in a data structure (such as a list) to be passed to another expression. However, in this example, the processing of these values (averaging them) is done iteratively without keeping an increasing state of previous values. In other words, this computation has a complexity of O (1) w/r to memory. These are appropriate conditions for an iterative use of underlying expressions as value generators.

8.2.5 Overall structure of the metric

An overview of the overall structure of the CSMRT-2 metric variant including its elements and their relationships is shown in Figure 3. The "depends on" relationship between these elements is represented by dashed arrows.

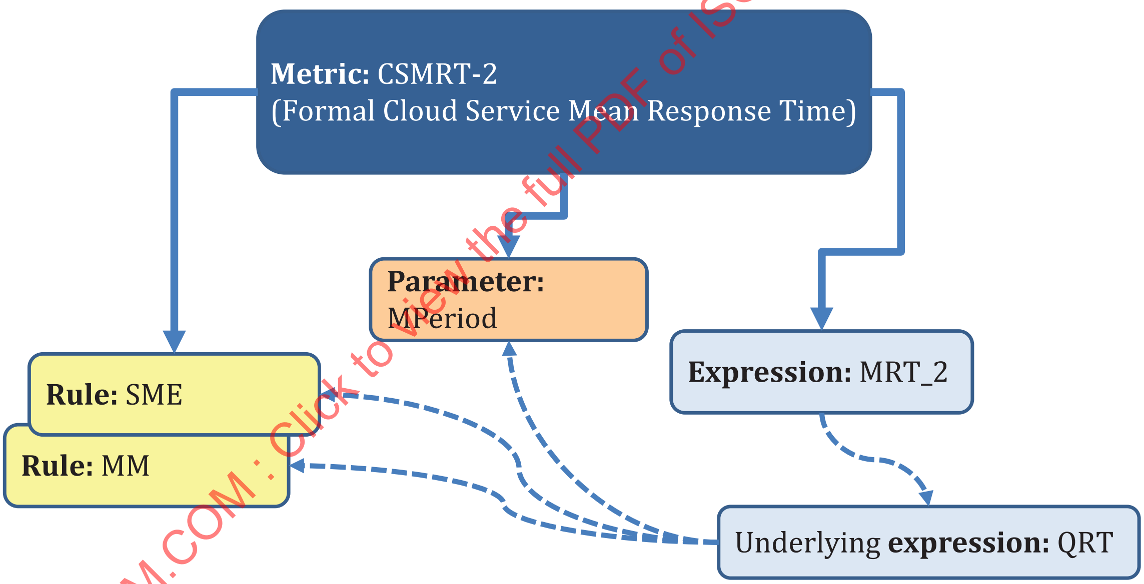

Figure 3 — The elements of the mean response time CSMRT-2 metric

8.2.6 Using constants

8.2.6.1 General

Sometimes the definition of a metric includes values that are constant throughout all measurements done with that metric, but that are expected to be adjustable in a similar way as configuration parameters are for a software application. This is achieved by representing such a constant as a distinct metric element, as opposed to hardcoding its value in every element (expression or rule) where it is used.

Consider for example a refinement of the CSMRT-2 metric that aims at offsetting some inaccuracies in recording a request-response event. The measurement method as described by rule MM (see 8.1.2) is not precisely defining the "reception" time of a request. The monitoring mechanism that implements this rule could vary from one implementation to another. For example, the recording of a request event (as done with a log record) could be delayed if done after some pre-processing of the request takes place on the server side. In case there is consensus that the “request time” to be considered is the time of the initial reception of the request by the server, this delay causes a recording for each request-response transaction time (and for the final average) of a duration shorter than it must be.

Assuming this delay is known by the metric developer and that it is roughly the same for all requests, an option is to represent it by a constant to be systematically added to the average request-response time calculated by the metric. Considering a case where the estimated delay in recording a request is of 500 milliseconds, there are two ways to represent such a constant as a metric element in a metric definition:

(1) As an underlying expression that assigns a value to a variable (the expression id);

(2) As a parameter of known value at definition time.

These options are illustrated in 8.2.6.2 and 8.2.6.3. Adjusting this constant would amount to replacing (or modifying) the metric element used to represent this constant, by substituting to it a new underlying expression (1), or a new parameter (2).

8.2.6.2 Representing a constant as an underlying expression

An underlying expression is added that represents a (static) variable assigned with the estimated delay (500 ms) in recording the request event, so that this delay is compensated for in the final calculation.

delayOffset = 500 millisecond

Following the guidance adopted in these examples for representing expressions (see 8.2.4, also reported as guideline 3 in 9.4), delayOffset would be the expression id, and “= 500” would be the value (expressionStatement). Table 12 shows the updated expressions.

Table 12 — The Expression elements for the CSMRT-1 metric variant with constants

Set of Expressions				
expressionLanguage: algorithmic (using functions as previously defined)				
id	**Description**	**expressionStatement** **(expressionLanguage** if different**)**	**Unit**	**Note**
MRT_2	Mean duration for all qualified request-response events, augmented with the estimated offset of the delay in recording the request time.	= average (QRT) + delayOffset	millisecond	Main metric expression, iterating over successive outputs from the QRT expression and averaging these.
QRT	Duration of a single request-response event, qualified according to the rules.	= duration(apply_rules (SME, MM) to (a request-response event) over (MPeriod)))	millisecond	
delayOffset	Estimated delay in recording a request (to add to each request-response duration.) Represented as a constant.	= 500	millisecond	

The expression delayOffset would then be modified or substituted in case the 500 millisecond constant is no longer appropriate.

8.2.6.3 Representing a constant as a parameter

Alternatively, the delayOffset constant could be represented as a new parameter, the value of which is known at metric definition time. Adding this parameter into the previous parameters (Table 10) results in the extended Table 13.

Table 13 — The Parameter elements for the CSMRT-2 metric variant with constant

Set of Parameters				
id	**Description**	**parameterStatement**	**Unit**	**Note**
MPeriod	Time boundaries for all request-response events that participate in a measurement.	Month of the year	Label in enumeration `{January, February, …}`.	A measurement parameter (expected to change at each measurement.)
delayOffset	Estimated delay in recording a request (to add to each request-response duration to offset this delay in the final output.) Represented as a constant.	Time delay for recording requests	millisecond	A configuration parameter (a constant value, set at configuration time).

Notice that MPeriod and delayOffset are two different kinds of parameters that are distinguished as described in guideline 6 in 9.7. The parameter MPeriod is given a new value for each measurement: it is called a measurement parameter. In contrast, to give the parameter delayOffset a value means to complete the metric definition. Such a parameter acts as a configuration element for the metric: it is called a configuration parameter, described in guideline 6 in 9.7. Its value remains stable for every measurement based on this configuration.

One way to set the value of a configuration parameter is in the main Metric element, at the place where the parameter is referenced. Table 14 indicates how the Metric table given in Table 9 (see 8.2.1) is modified with the parameter line that contains the additional configuration parameter delayOffset and sets its value.

Table 14 — Setting the value of the parameter used as a constant in the CSMRT-2 Metric element

Metric (id:CSMRT-2**)**	
attribute	**Value**
associated element	**Reference**
parameter	**id**: MPeriod (measurement parameter)
parameter	**id**: delayOffset, value=500 (configuration parameter)

Measurement parameters and configuration parameters play different roles in a metric and its usage (see guideline 6 in 9.7). This distinction when annotated in the previous Parameter table (note attribute), and also in the table for the Metric element (parameter fields under associated elements), indicates to a user the role of a parameter.

Guideline 7 in 9.8 compares the merits of different ways to represent a constant (expression or parameter).

9 Guidelines for using the metric model with the tabular representation

9.1 General

As part of the guidance for using the metric model, tabular templates have been introduced in Clause 7. Clause 9 is about guidelines on using these templates in practice. It collects and discusses some guidance illustrated in the two examples of metrics developed in this document: the cloud service mean response time metric in Clause 8, and the simple cloud service availability metric in Clause 10. These guidelines were found useful when using the tabular templates to develop the examples in this document. They are expected to serve as guidance for using the tabular templates when developing other metric cases presenting similarities with these examples.

9.2 Guideline 1 about defining expression and rule languages

The language used for an expression or a rule is indicated in the expressionLanguage or ruleLanguage attribute respectively. The language definition is not always a formally specified language. An ad-hoc language definition or reference is adequate as long as the intended audience understands it. The tabular template allows for extending or detailing a formally defined (or referenced) language with definitions of language elements that are specific to the context of a metric. Subclauses 8.2.4 and 10.3.5 illustrate how to add specific functions used in the context of a metric to a more general language definition in the expressionLanguage content. In all examples, the rules use plain English text as ruleLanguage.

9.3 Guideline 2 about associating rules with expressions

As rules affect the metric calculation, it is sometimes useful to be explicit as to where and how they do it. An association between a Rule element and an Expression element is characterized as a "depends on" relationship as defined in 7.2.1. There are different ways to express a dependency between a rule and an expression; three of them are illustrated by the examples in Clauses 8 and 10.

The first option is to indicate the dependency in the Expression element and does not affect the Rule element. It is a simple annotation in the note attribute of the expression. The advantage of this association is that it is independent from the expression language. The drawback is that using the note attribute does not convey the importance of a dependency that is key to the understanding of the expression.

The second option is to specify the dependency in the Expression element, directly referencing the rule(s) from the expression statement. In this way, the expression statement is made inseparable from the rules that are indispensable to its evaluation. When the expression language is informal, a simple reference to the rule identifier(s) is sufficient (as illustrated in 8.1.2). When using a more formal expression language, a way to formally express this dependency is to use a functional notation added as part of the expression language such as the function apply_rules():

qualified event(s) ← apply_rules (<rules>) to (<incident occurrence(s)>) [over (<time period>)]

In this form, the function applies rules in their listed order to some metric input such as observed incident(s), event occurrence(s) or operation(s), to be defined in the context of use and with the time of occurrence optionally bound by a time period. The result is a set of events qualified by these rules, with these events to be defined in the context of use. Such rules give details on how to produce related event data, how to perform measurements for capturing these occurrences or how to select valid occurrences. Some examples of use of apply_rules() are shown in 8.2 and 10.3. The apply_rules() function is not used to formalize the logic of applying rules. Its main role is to indicate explicitly and precisely which rules participate in the processing of which input in order to produce events used in an expression.

The third option is to specify the dependency in the Rule element and does not affect the Expression statement. The rule explicitly references the expression it supports either from its note attribute, or from its rule statement, as illustrated in the example of 10.2. See guideline 5 in 9.6 for more details.

Each option (or combination of options) is addressing particular constraints or interests, such as keeping some Metric elements independent from others, or on the contrary, associating them tightly and formally in their statement.

9.4 Guideline 3 about relating expressions to each other

A relationship among expressions is another form of "depends on" relationship as defined in 7.2.1. In case expression elements use or reference each other (underlyingExpression elements are typically sub-expressions of the main expression element as referenced via the "expression" association from the Metric element), it is convenient to use the id of an expression as the name of a variable holding the value of this expression, for usage in another expression. This role of the id is indicated in the tabular representation by starting the expressionStatement with "=". This guideline is illustrated in 8.2, 10.2 and 10.3.

9.5 Guideline 4 about the identifiers of metric elements

Globally unique identifiers are sometimes appropriate for the id attributes of metric elements. For example, this would be the case if these elements are expected to be archived, searched or reused on a large scale. One way to achieve this is to use URIs for these ids.

9.6 Guideline 5 about rules specifically designed to support an expression

In some cases, rules are tailored to support the calculation of specific expressions. Such a case is illustrated by the example in 10.2. Given the tight dependency of such rules on the expression of which they support the evaluation, it is useful to explicitly reference this expression by its identifier in the rule itself as indicated in the third option of guideline 2 in 9.3.

The role of such rules is to detail how to evaluate the expression and produce its output. While in general the calculation logic expressed in the rule could also be expressed in the expression statement itself or in its underlying expressions, it is sometimes preferable to use a rule for this. This is the case when the rule is combining or reusing several other rules, as illustrated by the example in 10.2, detailing the logic of combining these rules involving disjunctions (logical or), conjunctions (logical and), rules applying before, used as exceptions to or subsuming other rules. Such combination logic is sometimes better described within a new rule that explicitly mentions the rules that are combined, than within an expression. This is the case of rules QualifiedStartTime and QualifiedStopTime that explicitly reference the expression to which this combination applies. This is in contrast with the translation of this logic into an expression statement where computations sometimes do not clearly show how the rules that govern these have been interpreted and composed.

9.7 Guideline 6 about the role of parameters

Parameters apply to either expressions, or rules, or both. A parameter in an expression appears as a variable name, the value of which is given by the user of the metric, instead of resulting from a calculation. The parameter allows for late substitution of some value (text or number) in the full text of an expressionStatement or ruleStatement attribute. Two ways to parameterize a metric have been identified in the examples of 8.2 and 10.3 that are distinguished by defining the type of a parameter either as a measurement parameter or as a configuration parameter:

1. Measurement parameters: Such a parameter is given a value prior to each measurement. Instantiating such a parameter does not change the metric definition, but only provides a different input to the calculation involved in a measurement. An example is the monthly period for a cloud service evaluation such as expected in an SLA in order to prepare a billing statement. Such parameters are illustrated by the examples in 8.2, 10.2 and 10.3 (MPeriod, MeasurementPeriod and OPeriod).

2. Configuration parameters: The role of such a parameter is to make a metric definition configurable. A configurable metric is a more abstract metric definition from which a fully defined metric is derived by giving a value to its configuration parameters. These values remain constant over several measurements, in contrast with measurement parameters. An example is the MinOutage parameter introduced in the InvalidMinimum rule (see 10.2 and 10.3). Another example is the delayOffset parameter in the example of 8.2.6.3.

In these examples, the type of each parameter is indicated in the note field of the Parameter element. Although the metric model does not establish these types, they help clarify to metric users what the role of a parameter is, when its value is expected to be set and by whom.

9.8 Guideline 7 about representing constants

A constant is a value that is determined at the time a metric definition is completed, and is considered as part of that definition. Such a value does not change during the evaluation of the metric (during a measurement) or from one measurement to the other. In case such a constant is reused at several places and could be modified later, it is better to give it an id and isolate it as a distinct metric element instead of duplicating the constant value in the metric or rule elements where it is to be used.

As illustrated in 8.2.6, constants are represented with either one of two metric elements: (a) as an underlying expression assigning a value to a constant, or (b) as a configuration parameter (in which case the constant is seen as part of the metric configuration, as described in guideline 6). While there is no compelling rationale in preferring either way, there are some convenience aspects to consider in choosing one:

— An underlying expression (illustrated in 8.2.6) is a simple way to represent a constant when its modification or adjustment is not expected or at most very seldom. When the constant is only used by other expressions in a metric, the tabular representation allows for a convenient display in the same Set of Expressions table that describes the other expressions where it is used. However, when the constant is to be used in a rule, this makes the rule dependent on an expression element.

— A configuration parameter (see guideline 6 in 9.7) indicates that the constant might be changed: it is a variability point in the metric definition. See the case of the InvalidMinimum rule in 10.2 and 10.3 with the introduction of the MinOutage parameter. Parameterizing metric elements increases their reusability. An option for setting the value of a configuration parameter is to assign its value in the metric element that references it. This is the case of the Metric element in the cloud service availability example in 10.3, and also of the Configuration extension element (see the example variant in 10.3.7). Changing its value would affect the metric element that references the parameter, but not the parameter definition. Another option is to set the value of a configuration parameter in the parameter definition itself, as illustrated in 10.2.4. Changing its value would conversely affect the parameter definition but not the elements that reference it.

10 The simple cloud service availability metric

10.1 Measuring cloud service availability

10.1.1 General

This clause develops a more elaborate metric example for cloud service availability. It describes two variants of the same metric that illustrate two different approaches in using the metric model elements. This metric example is placed after Clause 9 as it exercises all of the guidelines described in Clause 9.

Consider the simple cloud service availability metric (“Simple_SAM”) which is based on a simple notion of cloud service availability. There are different ways to measure cloud service availability. One approach is to count the service requests that fail compared with those that succeed. Another more common interpretation of cloud service availability is based on the percentage of the time a service has been up (available) over a given period of time.

Many cloud service providers make a commitment in their SLA on the percentage of the time that their cloud services will be available (for example 99 % availability). Although this is one of the most common cloud metrics found in SLAs, it is not a simple one due to its dependency on several elaborate rules that usually vary from one CSP to the other.

Two variants of this metric, Simple_SAM_1 (see 10.2) and Simple_SAM_2 (see 10.3), are developed in the next subclauses. They are based on the same interpretation of availability and same SLA rules. They share the same calculation logic overall and are expected to produce the same output. However, they are designed quite differently on purpose. These two variants illustrate very different usages of the metric model and of the tabular templates toward the same objective, producing equivalent metrics with different metric elements.

10.1.2 Overall design approach

The general design common to both metric variants involves the following:

— Expressions: evaluate availability based on the percentage of cloud service uptime. This percentage is calculated over a measurement period.

— Parameters: although both metric variants illustrate different levels of parameterization, both consider the period of measurement (typically a billing month) as a measurement parameter (see guideline 6 in 9.7).

— Rules: govern various aspects of a measurement, including how a cloud service downtime is defined and validated. When validating an observed downtime, rules also define exclusion cases, for example as reflecting a CSP policy.

As a requirement specific to this metric example, it is not assumed that downtime incidents as defined by the rules and recorded by the metric expressions are always disjoint in time. Overlapping downtime records might result from the use of multiple monitoring sources. In case downtime records overlap, their overlapping period must not be counted twice in the total downtime.

10.1.3 SLA rules and metric rules

The downtime calculation is subject to certain rules, such as those defining what kind of service disruption qualifies as downtime, or defining how to measure the service disruptions. These rules are initially dictated by business decisions and various constraints (legal or technical). Because these rules typically appear in SLAs they are called SLA rules to distinguish them from metric rules as represented by the Rule elements in a metric representation.

The SLA rules as shared by both Simple_SAM_1 and Simple_SAM_2 variants are:

(a) A downtime is a period when the service is unreachable for at least one minute, or when a response to a request takes longer than one minute. (As per this rule, the duration of a valid downtime incident is always greater than or equal to one minute.)

(b) A downtime incident the duration of which is less than MinOutage (a parameter) is not counted as a valid downtime period.

(c) A downtime incident that occurs during a scheduled service maintenance is not counted as a valid downtime period.

The rules that appear in a metric definition (the metric rules) sometimes do not match one-to-one the SLA rules, even if they are derived from them. Variations include cases such as aggregations and splits. Translating in a metric what appears to be expressed as a rule in an SLA is sometimes done more appropriately in a metric expression instead of a metric rule.

In this example some rules are parameterized. This is the case of rule (b) with MinOutage that defines the minimal outage duration for a downtime period to be counted in the availability percentage calculation.

Because these rules interact, the ordering of their application is significant and affects the metric outcome:

— Rules (b) and (c) override rule (a). If rule (a) is applied first, downtime recordings per rule (a) could be disqualified by rules (b) and (c).

— A downtime period as recorded per rule (a) could be reduced by rule (c) if partially overlapping with a maintenance period. In some cases, this shortened downtime is in turn disqualified by rule (b) as expected, while it would not be the case if rule (b) was applied before rule (c).

Both metric variants are controlling the application order of these rules in different ways. The Simple_SAM_1 variant does so by introducing new metric rules to this effect, while the Simple_SAM_2 variant is controlling the application order within its metric expressions.

10.2 The simple cloud service availability metric variant Simple_SAM_1

10.2.1 The Metric element

The Metric element for the Simple_SAM_1 metric is shown in Table 15.

Table 15 — The Metric element for the Simple_SAM_1 metric

Metric (id: Simple_SAM_1**)**	
attribute	**value**
description	The percentage of the time relative to the measurement period when the cloud service was available.
source	SC38_WG3
scale	ratio
associated element	**reference**
parameter	**id:** MeasurementPeriod
parameter	**id:** MinOutage
rule	**id:** InvalidMinimum
rule	**id:** ScheduledMaintenance
rule	**id:** QualifiedStartTime
rule	**id:** QualifiedStopTime
expression	**id:** ServiceUpTimePercentage
underlyingExpression	**id:** TotalDowntimeDuration
underlyingExpression	**id:** QualifiedDowntimePeriod
underlyingExpression	**id:** StartTime
underlyingExpression	**id:** StopTime

10.2.2 The metric rules

In this design variant of the cloud service availability metric, some of the metric rules (described as Rule elements in Table 16) match closely the SLA rules described in 10.1.3:

— The InvalidMinimum metric rule directly matches the SLA rule (b) given in 10.1.3.

— The ScheduledMaintenance metric rule directly matches the SLA rule (c) given in 10.1.3.

Some other metric rules are designed so that they closely support the computation of specific underlying expressions in this metric. In doing so, they do not match exactly one-to-one the informal SLA rules given in 10.1.3 although they contribute in the same way to the metric calculation. These metric rules explicitly reference the expressions they are designed to support, and also depend on other rules:

— The metric rule QualifiedStartTime is directly supportive of the calculation of the start of a downtime period as done by the underlying expression StartTime, and explicitly references this expression. It details how to measure and qualify a valid StartTime value. This rule is derived from the SLA rules (a), (b) and (c) (see 10.1.3) and controls the order in which the SLA rules are applied.

— The metric rule QualifiedStopTime is directly supportive of the calculation of the end of a downtime period as done by the underlying expression StopTime, and explicitly references this expression. It details how to measure and qualify a valid StopTime value. This rule is derived from the informal SLA rules (a) and (c), and it controls the order in which these SLA rules are applied.

The corresponding Rule elements are recorded as shown in Table 16.

Table 16 — The Rule elements for the Simple_SAM_1 metric

Set of Rules		
ruleLanguage: plain English text		
id	**ruleStatement** (and its **ruleLanguage** if departing from the default language)	**Note**
QualifiedStartTime	Record current time as a StartTime value (beginning of a downtime) as soon as all 3 conditions are satisfied: (1) It is not during a scheduled maintenance downtime (based on SLA rule (c)), (2) The service has not been reachable for at least 1 min or the response time takes more than 1 min (based on SLA rule (a)), and (3) The incident has been lasting for at least MinOutage (based on SLA rule (b))	Describes how to measure the beginning of a downtime period. Note that there is a delay of MinOutage in the recording of the StartTime quantity (to be offset in the expressions of the metric). MinOutage is a parameter. StartTime is an underlying expression. This rule is derived from SLA rules (a), (b) and (c).
QualifiedStopTime	Record current time as a StopTime value relative to a recorded StartTime value, as soon as either one of these conditions is satisfied: (1) The service becomes reachable again and a response to a request takes less than one minute; or (2) A scheduled maintenance period begins.	StopTime is an underlying expression. This rule is derived from SLA rules (a) and (c).
InvalidMinimum	A downtime incident the duration of which is less than the value of MinOutage parameter is not counted as a valid downtime period.	MinOutage is a parameter. This rule matches the SLA rule (b).
ScheduledMaintenance	A downtime incident that occurs during a scheduled cloud service maintenance period is not counted as a valid downtime period.	This rule matches the SLA rule (c).

10.2.3 The metric expressions

10.2.3.1 Tabular representation

The metric expressions in this example as shown in Table 17 involve an underlying expression designed as a value generator (QualifiedDowntimePeriod). Such a design already illustrated in the example of 8.2 has the benefit of working without keeping a potentially large state of previous iterations in memory.

Table 17 — The Expression elements for the Simple_SAM_1 metric

Set of Expressions			
expressionLanguage: algorithmic, including functions: — function: SUM(<iterator>) returning the sum of all the durations iteratively produced. — function: CurrentTime, returning the current date and time that also satisfies some rules.			
Id	**expressionStatement**	**Unit**	**Note**
ServiceUpTimePercentage	= ((MeasurementPeriod – TotalDowntimeDuration) / MeasurementPeriod) * 100	percentage	Main expression. Result: service uptime percentage. MeasurementPeriod is a parameter.
TotalDowntimeDuration	= SUM(QualifiedDowntimePeriod)	minute	All QualifiedDownTimePeriods are added together. This expression iterates over successive outputs from QualifiedDowntimePeriod, used as a generator of values.
QualifiedDowntimePeriod	= StopTime – StartTime	minute	Conforms to rules InvalidMinimum and ScheduledMaintenance rules. This expression is iteratively evaluated for each downtime incident (pair of values StopTime and StartTime), with the iteration controlled outside the expression.
StartTime	= CurrentTime – MinOutage	minute	Conforms to rule QualifiedStartTime. Note that the recording of the actual start time of a downtime has a delay of MinOutage per this rule.
StopTime	= CurrentTime	minute	Conforms to rule QualifiedStopTime.

10.2.3.2 Design considerations

As for other examples in this document, the set of metric expressions is not designed to assume a particular execution model. The set of expressions in Table 17 support different evaluation modes including:

(a) Evaluation synchronized and controlled by the occurrence of physical events. The set of expressions is evaluated in a bottom-up manner with the main expression (ServiceUpTimePercentage) evaluation reacting to the evaluation of underlying expressions. (This evaluation mode is akin to the use of variables called “futures” in programming languages.)

(b) Evaluation of expressions imperatively controlled in a top-down manner, with the main expression (ServiceUpTimePercentage) driving the evaluation of its underlying expressions. This mode is appropriate when events are not processed in real time but in batch after having been recorded for example in a log.

In mode (a) the CurrentTime function is not “invoked” in a conventional sense in order to return a value. Instead it returns a time value whenever the conditions for producing this value are satisfied, as defined by the rules QualifiedStartTime and QualifiedStopTime.

10.2.4 The metric parameters

Two parameters are involved in Simple_SAM_1:

— The measurement period MeasurementPeriod is parameterizing a metric expression (ServiceUpTimePercentage);

— The minimum outage MinOutage is parameterizing a metric rule (InvalidMinimum).

The parameters in this variant of the cloud service availability metric illustrate a design approach different from the one described for the parameterization of the example in 8.2. Both parameters are treated as constants (called configuration parameters in guideline 6 in 9.7), with a value set in the parameter definition itself as shown in Table 18 (see guideline 7 in 9.8).

Changing these parameter values would require modifying or replacing these Parameter elements, but would not affect other elements in the metric definition.

Table 18 — The Parameter elements for the Simple_SAM_1 metric

Set of Parameters				
id	**Description**	**parameterStatement**	**Unit**	**Note**
MeasurementPeriod	The time period over which the availability was observed.	43200	minute	Based on a 30 day month.
MinOutage	The minimum duration for an outage to be considered as a downtime.	5	minute	N/A

10.2.5 Overall structure of the metric

An overview of the overall structure of the Simple_SAM_1 metric including its elements and their relationships is shown in Figure 4. The dependency relationship between these elements is indicated as in other figures with dashed arrows, or alternatively by a text annotation in some Rule element boxes, denoting a particular form of dependency such as in this case rules that explicitly support specific expressions.