ETSI TS 103 840 V0.1.6 (2023-06)

SmartM2M;

Model for oneM2M Performance Evaluation

<

**TECHNICAL SPECIFICATION**

Reference

DTS/SmartM2M-103840

Keywords

IoT, oneM2M, Performance Evaluation, Description Model, Applications, CSE, Infrastructure

***ETSI***

650 Route des Lucioles

F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - APE 7112B

Association à but non lucratif enregistrée à la

Sous-préfecture de Grasse (06) N° w061004871

***Important notice***

The present document can be downloaded from:  
<https://www.etsi.org/standards-search>

The present document may be made available in electronic versions and/or in print. The content of any electronic and/or print versions of the present document shall not be modified without the prior written authorization of ETSI. In case of any existing or perceived difference in contents between such versions and/or in print, the prevailing version of an ETSI deliverable is the one made publicly available in PDF format at [www.etsi.org/deliver](http://www.etsi.org/deliver).

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at <https://portal.etsi.org/TB/ETSIDeliverableStatus.aspx>

If you find errors in the present document, please send your comment to one of the following services:  
<https://portal.etsi.org/People/CommiteeSupportStaff.aspx>

If you find a security vulnerability in the present document, please report it through our

Coordinated Vulnerability Disclosure Program:

<https://www.etsi.org/standards/coordinated-vulnerability-disclosure>

***Notice of disclaimer & limitation of liability***

The information provided in the present deliverable is directed solely to professionals who have the appropriate degree of experience to understand and interpret its content in accordance with generally accepted engineering or

other professional standard and applicable regulations.

No recommendation as to products and services or vendors is made or should be implied.

No representation or warranty is made that this deliverable is technically accurate or sufficient or conforms to any law and/or governmental rule and/or regulation and further, no representation or warranty is made of merchantability or fitness for any particular purpose or against infringement of intellectual property rights.

In no event shall ETSI be held liable for loss of profits or any other incidental or consequential damages.

Any software contained in this deliverable is provided "AS IS" with no warranties, express or implied, including but not limited to, the warranties of merchantability, fitness for a particular purpose and non-infringement of intellectual property rights and ETSI shall not be held liable in any event for any damages whatsoever (including, without limitation, damages for loss of profits, business interruption, loss of information, or any other pecuniary loss) arising out of or related to the use of or inability to use the software.

***Copyright Notification***

No part may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm except as authorized by written permission of ETSI.  
The content of the PDF version shall not be modified without the written authorization of ETSI.  
The copyright and the foregoing restriction extend to reproduction in all media.

© ETSI 2023.

All rights reserved.

Contents

Intellectual Property Rights 4

Foreword 4

Modal verbs terminology 4

Executive summary 4

Introduction 5

1 Scope 6

1.1 Context for the present document 6

1.2 Scope of the present document 7

2 References 7

2.1 Normative references 7

2.2 Informative references 7

3 Definition of terms, symbols and abbreviations 8

3.1 Terms 8

3.2 Symbols 8

3.3 Abbreviations 9

4 Multi-models For IoT solutions based on oneM2M 9

4.1 Multi-layer abstraction 9

4.2 Meta Model 10

5 Key Performance Indicators and Key Configuration Parameters 12

5.1 Introduction 12

5.2 Key Performance Indicators 12

5.3 Key Configuration Parameters 12

6 oneM2M Application Scenario Descriptor (OASD) 12

7 oneM2M2 CSE Performance Descriptor (OCPD) 15

8 oneM2M2 Solution Deployment Descriptor (OSDD) 19

9 Example of instantiation 22

10 Conclusions 28

Annex A (informative): Change History 29

History 30

# Intellectual Property Rights

Essential patents

IPRs essential or potentially essential to normative deliverables may have been declared to ETSI. The declarations pertaining to these essential IPRs, if any, are publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server ([https://ipr.etsi.org](https://ipr.etsi.org/)).

Pursuant to the ETSI Directives including the ETSI IPR Policy, no investigation regarding the essentiality of IPRs, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Trademarks

The present document may include trademarks and/or tradenames which are asserted and/or registered by their owners. ETSI claims no ownership of these except for any which are indicated as being the property of ETSI, and conveys no right to use or reproduce any trademark and/or tradename. Mention of those trademarks in the present document does not constitute an endorsement by ETSI of products, services or organizations associated with those trademarks.

**DECT™**, **PLUGTESTS™**, **UMTS™** and the ETSI logo are trademarks of ETSI registered for the benefit of its Members. **3GPP™**and **LTE™** are trademarks of ETSI registered for the benefit of its Members and of the 3GPP Organizational Partners. **oneM2M™** logo is a trademark of ETSI registered for the benefit of its Members and of the oneM2M Partners. **GSM**® and the GSM logo are trademarks registered and owned by the GSM Association.

# Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Smart Machine-to-Machine communications (SmartM2M). It builds on the set of use cases and requirements collected as part of ETSI TR 103 839 [i.1] and which aim at identifying relevant deployment scenarios for systems using the oneM2M technology.

# Modal verbs terminology

In the present document "**shall**", "**shall not**", "**should**", "**should not**", "**may**", "**need not**", "**will**", "**will not**", "**can**" and "**cannot**" are to be interpreted as described in clause 3.2 of the [ETSI Drafting Rules](https://portal.etsi.org/Services/editHelp!/Howtostart/ETSIDraftingRules.aspx) (Verbal forms for the expression of provisions).

"**must**" and "**must not**" are **NOT** allowed in ETSI deliverables except when used in direct citation.

# Executive summary

The present document proposes a Meta-Model (MM), *i.e.,* a general-purpose modeling of oneM2M applications, written using an UML-like general purpose language that is intended to provide a general pattern for representing IoT applications, using all the entities defined in the oneM2M standard, and offering a pragmatic tool for rapidly prototyping the most important characteristic we want to highlight and measure.

The MM will be instantiated to real examples to express some (but not all) application requirements (e.g., response time, level of scalability, hardware requirement) and resource representation (e.g., smart campus, traffic light system). The MM is composed of three subsets, each one representing a part of operational IoT systems and expressing an abstraction with four layers: the Application Scenario layer, the oneM2M resource layer, the oneM2M service platform layer and the infrastructure layer. Such onion-like view proves useful in the work to conceive, model, inject and decompose into the oneM2M standard almost any IoT application in full detail and make it ready to be explored from the performance evaluation standpoint. Those layers are expressed in three model views called, respectively: 1- oneM2M Application Scenario Descriptor (OASD), 2- oneM2M CSE Performance Descriptor (OCPD), and 3- oneM2M Solution Deployment Descriptor (OSDD). For information purposes, a detailed instantiation of the common MM is proposed based on one of the use cases described in [i.1], namely the one described in Clause 9.

# Introduction

The objective of the present document in conjunction with three other ones ETSI TR 103 840 [i.1], ETSI TR 103 841 [i.3], ETSI TR 103 842 [i.4] as well as ETSI TR 103 843 [i.5] is to provide a MM that is suitable to be instantiated to deploy an oneM2M application, based on application description, a oneM2M specific stack over a specific hardware and a network platform. A MM is very useful to capture the key performance evaluation aspects that need to be measured e.g., response time, data transfer volume. Use cases described in ETSI TR 103 840 [i.1] will be able to be smoothly be instantiated within the MM. Based on this formalization, a simulation and performance evaluation can be created and executed, starting from a generic implementation of the MM in a generic or ad hoc simulator, a choice that is clarified in deliverables ETSI TR 103 841 [i.3] and ETSI TR 103 842 [i.4]). This will enable users of the present document to ultimately tune their oneM2M applications based on measurements of the most performance-critical aspects, usually called Key Performance Indicators (KPI).

# 1 Scope

## 1.1 Context for the present document

The oneM2M ETSI standard is now mature: multiple deployments exist all over the world at both experimental and operational levels. The experimental deployments are conducted for multiple reasons:

* to evaluate the capabilities of the standard in terms of expressiveness, usability on specific equipment, connection with specific existing systems or performance evaluation.
* to provide a methodological study, based on performance evaluation (time, space) on a given set of "paradigmatic use cases".
* to measure KPIs defined in this document*.* Different implementations exist that are compliant with the oneM2M standard, available either freely or commercially.

Use cases are evaluated in terms of chosen KPI: e.g., running time, memory space, numerosity of oneM2M entities (e.g., AE, MN-CSE, CSE.), data transfer volume and real-time needs. Using a select set of available oneM2M CSE implementations, a simulation library or an *ad hoc* simulator is to be provided, offering the ability to evaluate and simulate the performance of the use cases and give crucial information/feedback to the general user of the oneM2M to choose and tune their IoT applications based on oneM2M framework. The results of this tool development and evaluations of the use cases will be the basis to generate other deliverables. The present document was developed in the context of ETSI TTF T019, set up to perform work on "Performance Evaluation and Analysis for oneM2M Planning and Deployment". Five elements were addressed sequentially:

1. a collection of **use cases and derived requirements** were formally identified and defined. This work includes identification of relevant deployment scenarios. We adopted the use case style and template from oneM2M with a minor modification to address some performances issues. This phase of the work resulted in deliverable ETSI TR 103 839 [i.1].
2. The definition of **performance evaluation model**, with specification of procedures to assess the performance of oneM2M-based IoT platforms. This includes the identification and definition of a set/list of KPIs necessary to assess the deployment. For those KPIs, provision of a formal description of the test campaign and the test results to be obtained. This phase of the work resulted in deliverable ETSI TS 103 840 [i.2] (the present document).
3. The creation of a **proof of concept** of a performance evaluation tool. This work also relies on a formal description of the identified deployment scenarios (single vertical domain & multiple vertical domains). This phase of the work resulted in deliverable ETSI TR 103 841 [i.3].
4. A practical **demonstration and analysis** exercise putting the proposed tool to use, with a specific oneM2M implementation but aimed at being a blueprint for the adoption and re-use of the results of TR 103 839, TS 103 840, and TR 103 841 with other oneM2M implementations and deployment scenarios. This phase of the work resulted in deliverable ETSI TR 103 842 [i.4].
5. The development of a set of **guidelines and best practices** documenting best practices and lessons learnt as well as providing instructions for IoT solution topology, capacity provisioning, and expected performances that will gives crucial directives and information to designer and implementors. This phase of the work resulted in deliverable provisioning and expected performances. This phase resulted in deliverable ETSI TR 103 843 [i.5].

The present document covers the second of the five items listed above and provides the basis for the related ETSI publications listed below:

* ETSI TR 103 839: Scenarios for evaluation of oneM2M deployments [i.1].
* ETSI TS 103 840: Model for oneM2M Performances Evaluation [i.2] (the present document).
* ETSI TR 103 841: oneM2M Performance Evaluation Tool (Proof of Concept) [i.3].
* ETSI TR 103 842: Demonstration of Performance Evaluation and Analysis for oneM2M Planning and Deployment [i.4].
* ETSI TR 103 843: oneM2M deployment guidelines and best practices [i.5].

## 1.2 Scope of the present document

The present document identifies additional requirements to be potentially submitted to oneM2M in the areas of performance evaluation by means of a MM able to characterize application representation and deployment in the oneM2M standard. The document is structured as follows:

* Clauses 1 to 3 set the scene and provide references as well as definition of terms, symbols and abbreviations, which are used in the present document.
* Clause 4 describes a layered vision of one IoT application, designed using oneM2M and deployed over a distributed and connected platform of execution.
* Clause 5 highlights Key Configuration Parameters that could be manipulated by IoT architect and Key Performance Indicators that they could expect.
* Clause 6 focuses on the oneM2M Application Scenario Descriptor (OASD) able to describe the application in term of behavior, requirements and description as oneM2M resource.
* Clause 7 presents the subset model oneM2M CSE Performance Descriptor (OCPD) that expresses how a specific oneM2M service layer is decomposed and how is used hardware resource to create the oneM2M function.
* Clause 8 describes the oneM2M Solution Deployment Descriptor (OSDD). This subset model expresses processing, storage and network part of the execution platform on which common service entities are running.
* Clause 9 presents how a use case could be instantiated with the previous model templates.
* Clause 10 provides the conclusions of this specification.

# 2 References

## 2.1 Normative references

References are either specific (identified by date of publication and/or edition number or version number) or non‑specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

Referenced documents which are not found to be publicly available in the expected location might be found at <https://docbox.etsi.org/Reference>.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are necessary for the application of the present document.

[1] oneM2M TS-0001 (V4.18.0). "Functional Architecture.

[2] oneM2M-0011 Common Terminology (V1.4.0)

NOTE: Available at <https://onem2m.org/technical/published-specifications/release-1>

## 2.2 Informative references

References are either specific (identified by date of publication and/or edition number or version number) or non‑specific. For specific references, only the cited version applies. For non-specific references, the latest version of the referenced document (including any amendments) applies.

NOTE: While any hyperlinks included in this clause were valid at the time of publication, ETSI cannot guarantee their long-term validity.

The following referenced documents are not necessary for the application of the present document but they assist the user with regard to a particular subject area.

The following referenced documents are not necessary for the application of the present document, but they assist the user with regard to the general scope of the present TTF.

[i.1] ETSI TR 103 839: "Scenarios for evaluation of oneM2M deployments”.

[i.2] ETSI TR 103 840: "Model for oneM2M Performance Evaluation".

[i.3] ETSI TR 103 841: "SmartM2M; oneM2M Performances Evaluation Tool (Proof of Concept)".

[i.4] ETSI TR 103 842: "SmartM2M; Demonstration of Performance Evaluation and Analysis for oneM2M Planning and Deployment".

[i.5] ETSI TR 103 843: "SmartM2M; oneM2M deployment guidelines and best practices".

NOTE: Available at <https://member.onem2m.org/Application/documentapp/downloadLatestRevision/default.aspx?docID=35545>.

[i.7] oneM2M-Template-Use-Case.

NOTE: Available at

<http://ftp.onem2m.org/Templates/oneM2M-Template-Use-Case.doc>.

[i.8] oneM2M TR-0002 (V4.6.0): "Requirements".

NOTE: Available at <http://member.onem2m.org/Application/documentapp/downloadLatestRevision/default.aspx?docID=29274>.

[i.9] oneM2M TR-0001 (V4.3.0): "Use Cases Collection".

NOTE: Available at <https://member.onem2m.org/Application/documentapp/downloadLatestRevision/default.aspx?docID=28153>.

# 3 Definition of terms, symbols and abbreviations

## 3.1 Terms

**Performance Evaluation:** refers to the evaluation of temporal, data transfer volumetry, and scalability aspects of a system.

**oneM2M Numerosity Objects:** refers to the scalability of a oneM2M application.

**Real Time Requirements:** refers to the timing constraints to be fulfilled by a system.

**oneM2M Implementations Standard:** refers to the list of the implementations of the oneM2M standard.

**Platform Evaluation Tool:** refers to the simulation environment that is used to calculate/demonstrate the performance of the oneM2M standard.

**Guidelines and Good Practices:** refers to a methodological document that gives hints to deploy a oneM2M infrastructure.

## 3.2 Symbols

Void.

## 3.3 Abbreviations

For the purposes of the present document, the following terms as well as those listed in oneM2M TS-0011 [2] apply:

ACP Access Control Policy

ADN Application Dedicated Node

AE Application Entity

API Application Program Interface

BER Bit Error Rate

CIN Content INstance

CNT Container

CSE Common Service Entity

CRUD Create, Read, Update, Delete

ETSI European Telecommunications Standards Institute

FGT Formal Graph Topology

GGP Guidelines and Good Practices

IN-CSE Infrastructure Node - Common Services Entity

HW Hardware

KPI Key Performance Indicators

KCP Key Configuration Parameters

IoT Internet of Things

IPE Interworking Proxy Entity

M2M Machine-to-Machine

M2MSP M2M Service Provider

Mca Reference Point for M2M Communication with AE

Mcc Reference Point for M2M Communication with CSE

MM Meta Model

MN-CSE Middle Node - Common Services Entity

OASD oneM2M Application Scenario Descriptor

OCPD oneM2M CSE Performance Descriptor

OIS oneM2M Implementations Standard

OM2M Eclipse OM2M - Open-Source platform for M2M communication

ONO oneM2M Numerosity Objects

OSDD oneM2M Solution Deployment Descriptor

OS Open Source

PE Performance Evaluation

PER Packet Error Rate

PET Platform Evaluation Tool

RDF Resource Description Framework

RDM Requirements and Domain Models

RT Real-Time

RTR Real Time Requirements

SPARQL Simple Protocol and RDF Query Language, see [i.7]

SUB Subscription

SW Software

TC Technical Committee

TR Technical Report

TS Technical Specification

UCT Use Case Template

# 4 Multi-models For IoT solutions based on oneM2M

## 4.1 Multi-layer abstraction

A view of a layered IoT system used in this document is shown in Figure 1. This model makes it possible to break down all the physical, hardware, software and human entities involved. Each layer has interactions with the other layers and finally merges to constitute the IoT system and its environment of use.

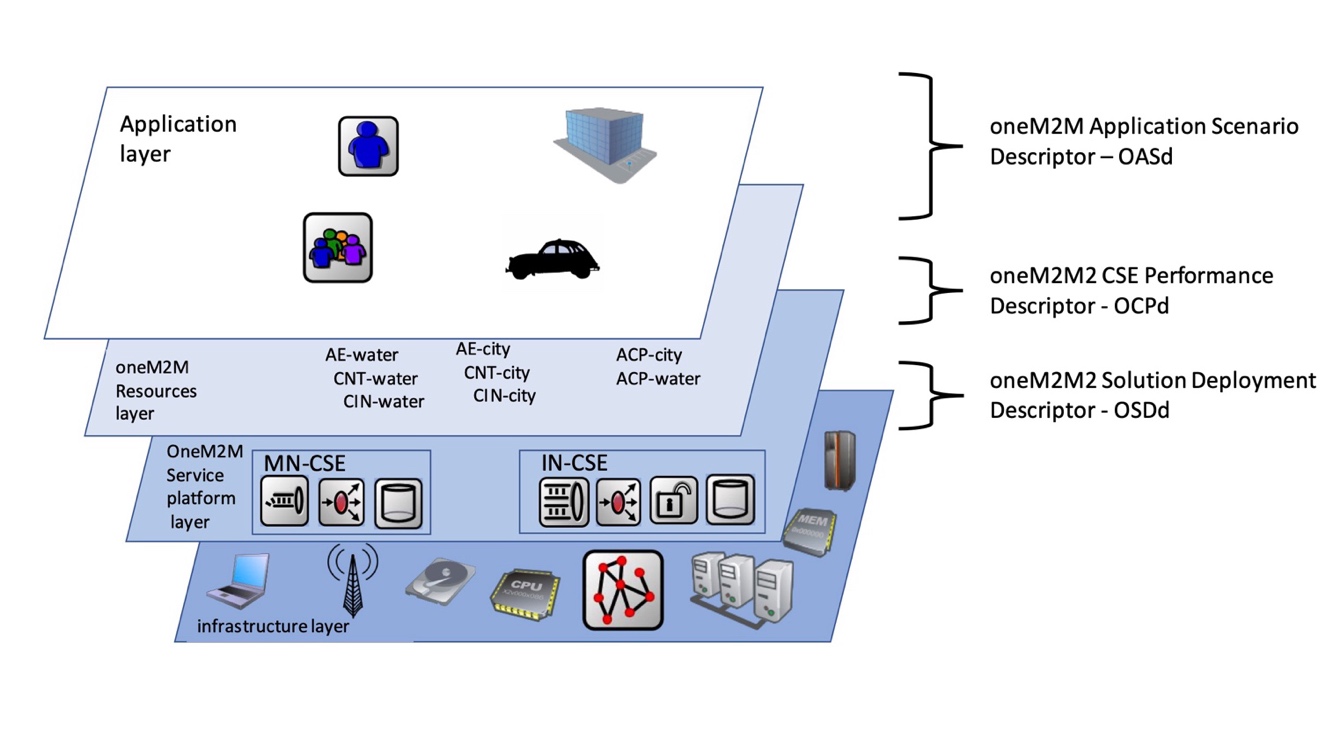


Figure 1: multi-layers vision of oneM2M IoT system

The first layer, called the **application layer** in the broad sense, corresponds to the physical world with which the IoT system interacts. This layer is on the one hand made up of the interaction that humans and the environment have with sensors and actuators and on the other hand of the constraints and needs that application domains can express for an action for example in the physical world.

The second layer, called the **resource layer**, reflects the projection of the IoT system and the specific use case on resources (in the oneM2M sense). This layer models the relationships between its resources as well as their ownership. This layer expresses the business logic on the application side.

The environment and resource layers will be described in Clause 6 through a model called oneM2M Application Scenario Descriptor (OASD).

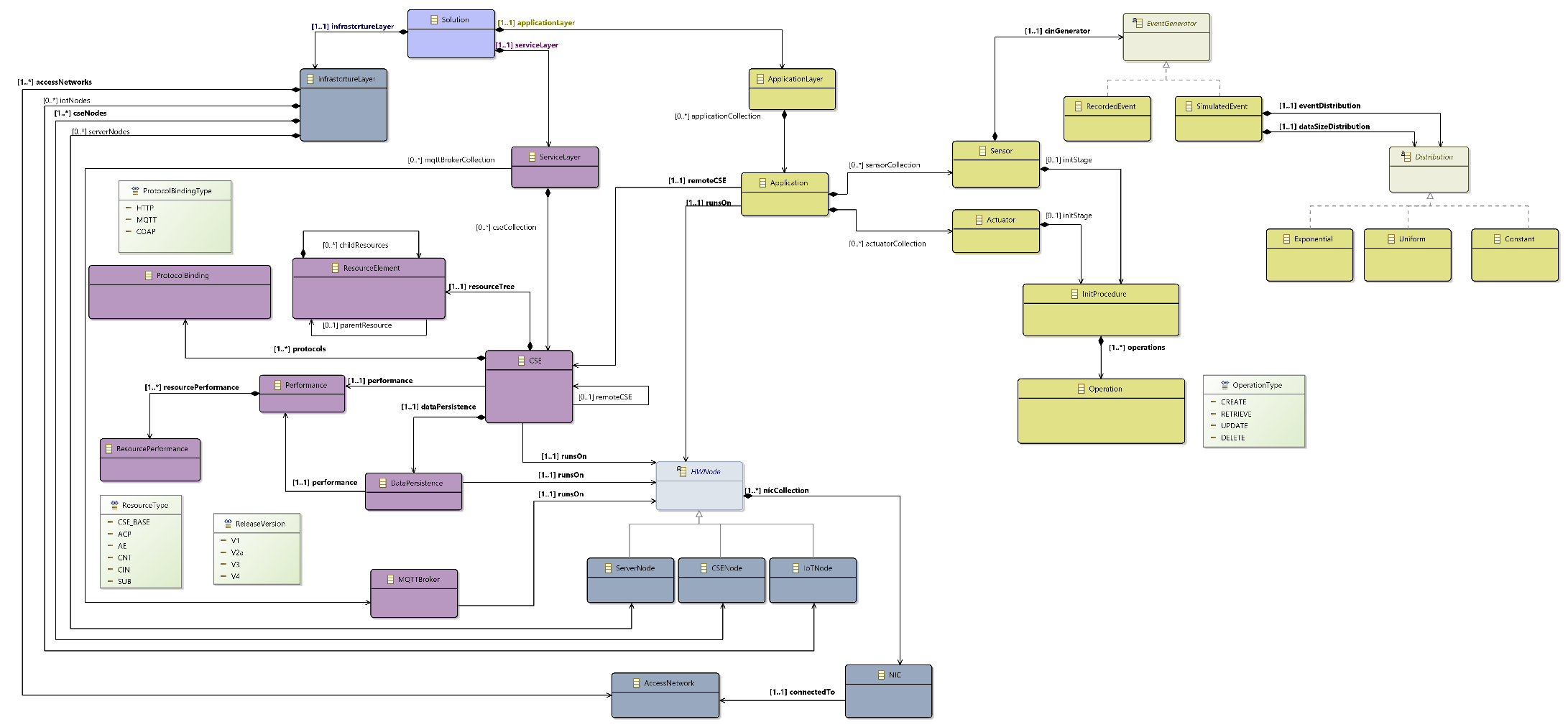
The third layer, called the **oneM2M service platform layer**, represents the software or middleware system to be put in place to support the needs of layers 1 and 2. In the oneM2M world, it represents on the one hand the CSEs and the links between them and on the other hand, it models the behavior and performance of the CSE according to its configuration and the projection of layers 1 and 2 in this service platform layer. This layer will be instantiated in a model called oneM2M2 CSE Performance Descriptor (OCPD) in Clause 7.

The fourth and last layer called **Infrastructure layer** represents the hardware infrastructure that hosts all of the previous layers. It makes it possible to define the equipment, the communication links and any hardware characteristics. The oneM2M2 Solution Deployment Descriptor (OSDD) model will allow to describe this layer.

This vision makes it possible to dissociate and break down the complexity of an IoT system and to call on targeted skills in each layer. Designing and deploying an IoT system means implementing and merging all of these layers. At the performance analysis level, this makes it possible to characterize the needs, parameters and behavior of each layer and to clearly explain the links between these layers through models.

## 4.2 Meta Model

The MM on Figure 2 integrates and links OASD, OCPD, and OSDD descriptions. Each model is represented by a different color. The root item of an IoT system is represented by the Solution class. This one is connected to one or several “IoT application layer” made of one or multiple applications. Each application runs on hardware nodes and is composed of sensors and actuators. Those applications make requests to one CSE. The CSE is running on a hardware node and could use a persistent system to store data. All the hardware nodes are connected to one or several networks to exchange information between CSE, application and MQTT or data persistency system.

**Figure 2: Meta-model for oneM2M IoT system**

# 5 Key Performance Indicators and Key Configuration Parameters

## 5.1 Introduction

Key Performance Indicators (KPI) and Key Configuration Parameters (KCP) have been selected and validated by the oneM2M community to express important input of IoT systems and performance requirements.

## 5.2 Key Performance Indicators

This list is naturally not exhaustive, but it will be suitable to be extended according to usage and extension of use cases.

* Running time
* Memory space
* Data transfer volume
* Per CRUD Operation:
* [Min, Avg, Max, Variance, Std-Dev] Processing Time
* [Min, Avg, Max, Variance, Std-Dev] Persistence Delay
* [Min, Avg, Max, Variance, Std-Dev] Persistence Usage
* Per CSE:
* [Min, Avg, Max, Variance, Std-Dev] CPU Usage
* [Min, Avg, Max, Variance, Std-Dev] Memory Usage
* [Min, Avg, Max, Variance, Std-Dev] Disk Usage
* [Min, Avg, Max, Variance, Std-Dev] Network Usage
* Per oneM2M object:
* [Min, Max] Number of CIN per CSE
* [Min, Avg, Max, Variance, Std-Dev] Event to Notification Time
* Networking:
* [Min, Avg, Max, Variance, Std-Dev] Network Usage per kind of Network employed

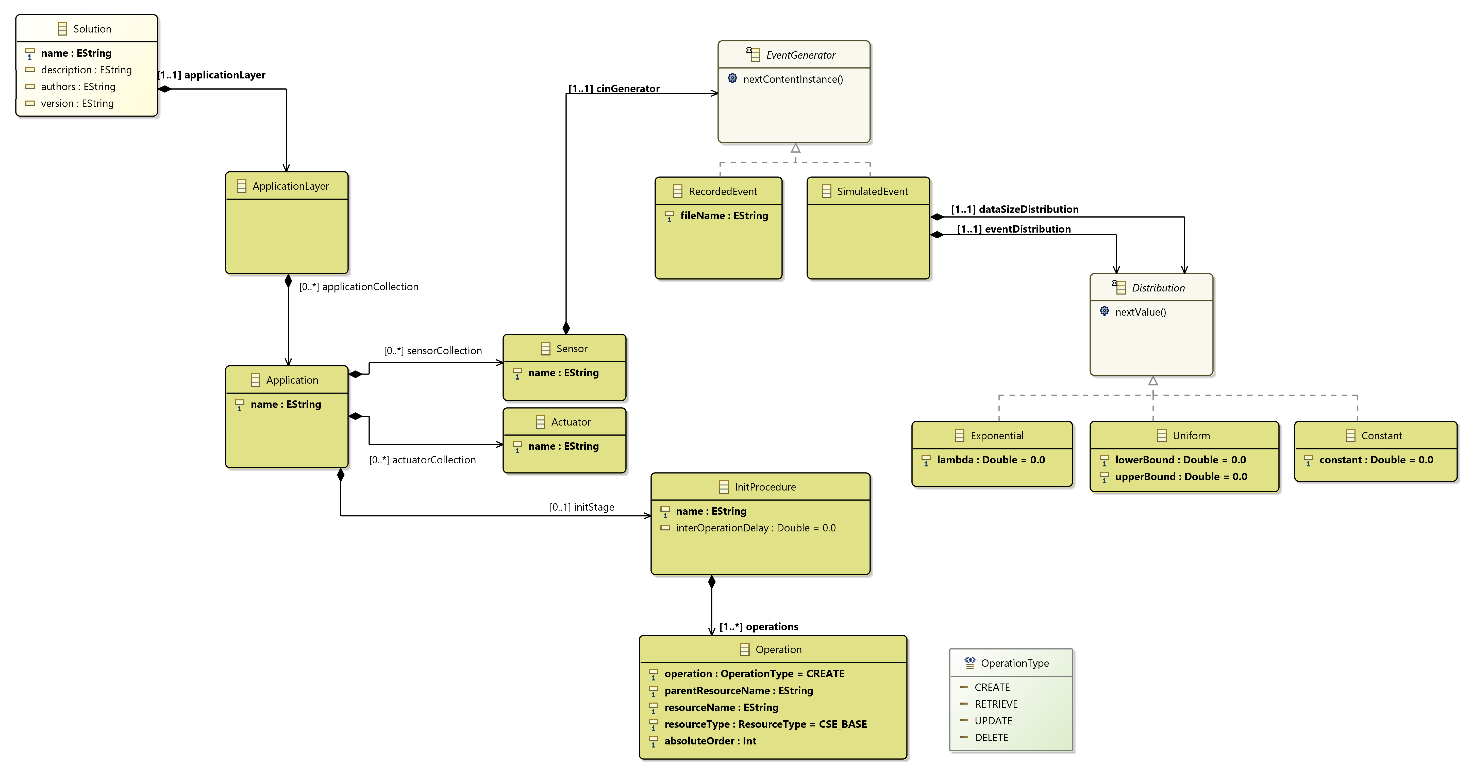
## 5.3 Key Configuration Parameters

These parameters will be able to measure the impact of key parameters of the IoT application on KPI of the simulation. Those key parameters could be:

* Infrastructure Layer aspects:
  + Total Number of IoT devices
  + Number of IoT devices per CSE
  + Type of access networks (Rate, Delay, PER, BER)
* Service Layer Aspects:
  + Total Number of CSEs
  + CSE Topology
  + CSEs’ Hardware Nodes (CPU, RAM, Disk)
  + CSE Software (name, version, oneM2M Release Version)
  + oneM2M resources organization (over CSEs)
* Application Layer Aspects:
  + IoT traffic generation profiles
  + Way of interaction with CSE

# 6 oneM2M Application Scenario Descriptor (OASD)

The objective of the OASD model in Figure 3 is to represent the behavior of the IoT application. To do so, sensors and actuators are endowed with a behavior modelled by event generation (for the sensors). The policy for the generation of sensor data is based on different distribution profiles *i.e.,* constant (equivalent to periodic) or sporadic with different laws. Sensors and actuators have at an initial stage a transient behavior that corresponds to the creation of oneM2M resources.



**Figure 3: OASD model (Application Layer)**

The detailed description of the model attributes, with related type and informal description is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| **ApplicationLayer** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| applicationCollection | 1..n | Application | A collection of all the IoT applications in the IoT solution. The IoT Solution have at least one IoT application. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Application** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the IoT application. |
| remoteCSE | 1..1 | CSE | Reference of the remote CSE this IoT application is connected to. |
| runsOn | 1..1 | HWNode | Reference of the hardware node this IoT application is running on (see Note). |
| sensorCollection | 0..n | Sensor | Collection of sensors managed by the IoT application. |
| actuatorCollection | 0..n | Actuator | Collection of actuators managed by the IoT application. |
| initStage | 0..1 | InitProcedure | Reference to an eventual initialization procedure. |
| NOTE: an IoT application can run on both a dedicated HWNode::IoTNode or on a HWNode::CSENode. It can also run on a HWNode::ServerNode to represent a server side applications. | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Sensor** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the sensor. |
| cinGenerator | 1..1 | EventGenerator | Reference to a CIN generator. |

|  |  |  |  |
| --- | --- | --- | --- |
| **EventGenerator** | | | |
| This is an abstract class used to describe how an event will be generated. It is derived into two classes : RecordedEvent and SimulatedEvent. | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| (void) | (void) | (void) | (void) |

|  |  |  |  |
| --- | --- | --- | --- |
| **RecordedEvent : EventGenerator** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| fileName | 1..1 | String | File name where the sequence of events is stored. Each event is timestamped. The generated message size is also recorded. |

|  |  |  |  |
| --- | --- | --- | --- |
| **SimulatedEvent : EventGenerator** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| eventDistribution | 1..1 | Distribution | The way to compute the next event date. |
| dataSizeDistribution | 1..1 | Distribution | The way to compute the next message size. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Distribution** | | | |
| This is an abstract class used to describe statistical distribution. It is derived into: Exponential and Uniform. For harmonization purposes, the periodic process is also derived from Distribution and is called Constant. | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| (void) | (void) | (void) | (void) |

|  |  |  |  |
| --- | --- | --- | --- |
| **Exponential: Distribution** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| lambda | 1..1 | Decimal | Parameter of the exponential distribution. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Uniform: Distribution** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| lowerBound | 1..1 | Decimal | Lower bound of the uniform distribution. |
| upperBound | 1..1 | Decimal | Upper bound of the uniform distribution. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Constant: Distribution** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| constant | 1..1 | Decimal | The constant value this distribution is always returning. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Actuator** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the actuator. |

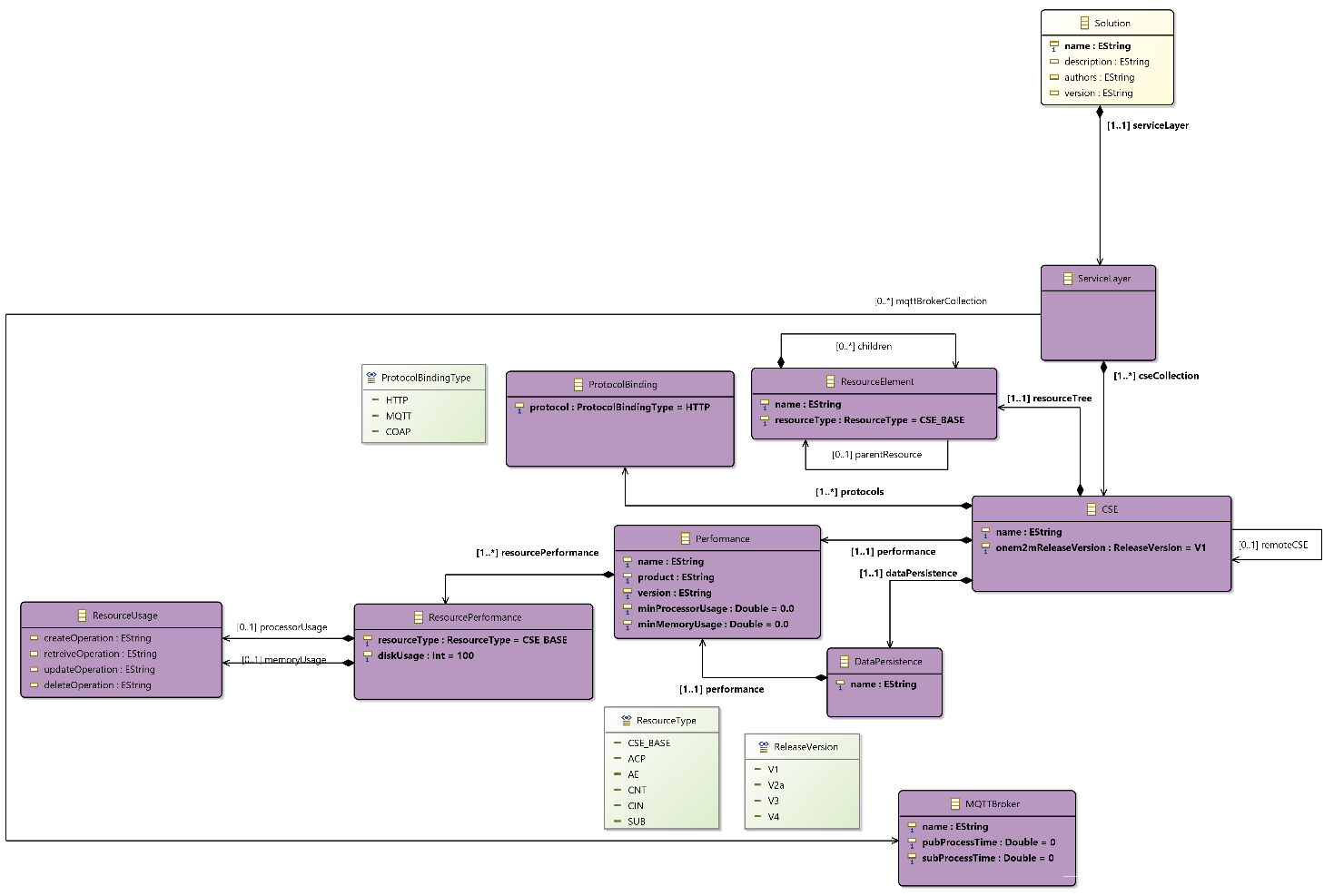
|  |  |  |  |
| --- | --- | --- | --- |
| **InitProcedure** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the initialization procedure. |
| operations | 1..n | Operation | Sequence of elementary operation composing the initialization procedure. |
| interOperationDelay | 0..1 | Decimal | Delay between two consecutive operations in seconds. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Operation** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| operation | 1..1 | OperationType | Type of operation to be performed (CREATE, RETRIEVE, UPDATE, or DELETE). |
| parentResourceName | 1..1 | String | Name of the parent resource. |
| resourceName | 1..1 | String | Name of the resource to be created, retrieved, updated, or deleted. |
| resourceType | 1..1 | ResourceType | oneM2M resource type for this operation. |
| absoluteOrder | 1..1 | Integer | Absolute order of this operation in all the IoT solution. |

|  |  |
| --- | --- |
| **OperationType** | |
| **Short text description** | An enumeration for the CRUD operations. |
| **Values** | * CREATE * RETRIEVE * UPDATE * DELETE |

# 7 oneM2M2 CSE Performance Descriptor (OCPD)

The objective of the OCPD sub-model on Figure 4 is to describe the service layer. To do this, it describes the interconnection between CSEs (IN-CSE, MN-CSE, ASN), the impact of the implementation choices of a CSE by a supplier, the deployment on equipment taking into account their processing capabilities, their memory and network connection.



**Figure 4: OCP model (Service Layer)**

The detailed description of the model attributes, with related type and informal description is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| **ServiceLayer** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| cseCollection | 1..n | CSE | The collection of CSEs in the IoT solution. At least one CSE is present. |
| mqttBrokerCollection | 0..n | MQTTBroker | The collection of MQTT brokers in the IoT solution. |

|  |  |  |  |
| --- | --- | --- | --- |
| **CSE** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the CSE in the IoT Solution. |
| remoteCSE | 0..1 | CSE | Reference to the remote CSE this CSE is connected to. |
| runsOn | 1..1 | HWNode | Reference to the hardware node this CSE is running on.  NOTE: a CSE runs only on a dedicated HWNode::CSENode. |
| dataPersistence | 1..1 | DataPersistence | Reference to the persistency mechanism. |
| performance | 1..1 | Performance | Performance descriptor of CRUD operations supported by the CSE. |
| protocols | 1..n | ProtocolBinding | Collection of the protocol bindings supported by the CSE. |
| resourceTree | 1..1 | ResourceElement | Reference to the oneM2M CSEBase resource (i.e., the root element of the oneM2M resource tree). |
| onem2mReleaseVersion | 1..1 | ReleaseVersion | The oneM2M release version selected for the CSE. |

|  |  |  |  |
| --- | --- | --- | --- |
| **Performance** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Textual description. |
| product | 1..1 | String | Name of the CSE implementation. |
| version | 1..1 | String | Version of the CSE implementation. |
| minProcessorUsage | 1..1 | Decimal | Processor usage at start-up. |
| minMemoryUsage | 1..1 | Decimal | Memory usage at start-up. |
| resourcePerformances | 1..n | ResourcePerformance | The performance of CRUD operation for each resource supported by the CSE. |

|  |  |  |  |
| --- | --- | --- | --- |
| **DataPersistence** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the data persistence service. |
| runsOn | 1..1 | HWNode | Reference of the hardware node this data persistence service is running on.  NOTE: a data persistence service can run on the same HWNode::CSENode along its CSE. It can also run on a dedicated HWNode::ServerNode. |
| performance | 1..1 | Performance | Performance descriptor of CRUD operations for this data persistence service. |

|  |  |  |  |
| --- | --- | --- | --- |
| **ResourceElement** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the oneM2M resource. |
| resourceType | 1..1 | ResourceType | The oneM2M resource type. |
| parentResource | 0..1 | ResourceType | Reference to the parent resource. |
| childResources | 0..n | ResourceType | References to the child resources. |

|  |  |  |  |
| --- | --- | --- | --- |
| **ResourcePerformance** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| resourceType | 1..1 | ResourceType | oneM2M resource type. |
| processorUsage | 1..1 | ResourceUsage | Reference to a resource usage description for the processing resource on the HW node. |
| memoryUsage | 1..1 | ResourceUsage | Reference to a resource usage description for the memory resource on the HW node. |
| diskUsage | 1..1 | Decimal | Memory in bytes occupied by this resource. |

|  |  |  |  |
| --- | --- | --- | --- |
| **ResourceUsage** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| createOperation | 1..1 | String | A string expressing a formula to describe a resource usage (processor or memory) for the CREATE operation. |
| retrieveOperation | 1..1 | String | A string expressing a formula to describe a resource usage (processor or memory) for the RETRIEVE operation. |
| updateOperation | 1..1 | String | A string expressing a formula to describe a resource usage (processor or memory) for the UPDATE operation. |
| deleteOperation | 1..1 | String | A string expressing a formula to describe a resource usage (processor or memory) for the DELETE operation. |
| NOTE: Each formula can depend on parameters such as time, resource type, CPU usage of the host, etc. | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **MQTTBroker** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the MQTT Broker. |
| runsOn | 1..1 | HWNode | Reference of the hardware node this MQTT broker is running on.  NOTE: an MQTT Broker can run on the same HWNode::CSENode along a CSE. It can also run on a dedicated HWNode::ServerNode. |
| pubProcessTime | 1..1 | Decimal | Processing time in seconds for publish operations on this MQTT Broker. |
| subProcessTime | 1..1 | Decimal | Processing time in seconds for subscribe operations on this MQTT Broker. |

|  |  |  |  |
| --- | --- | --- | --- |
| **ProtocolBinding** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| protocol | 1..1 | ProtocolBindingType | The protocol supported by this protocol binding. |

Enumerations:

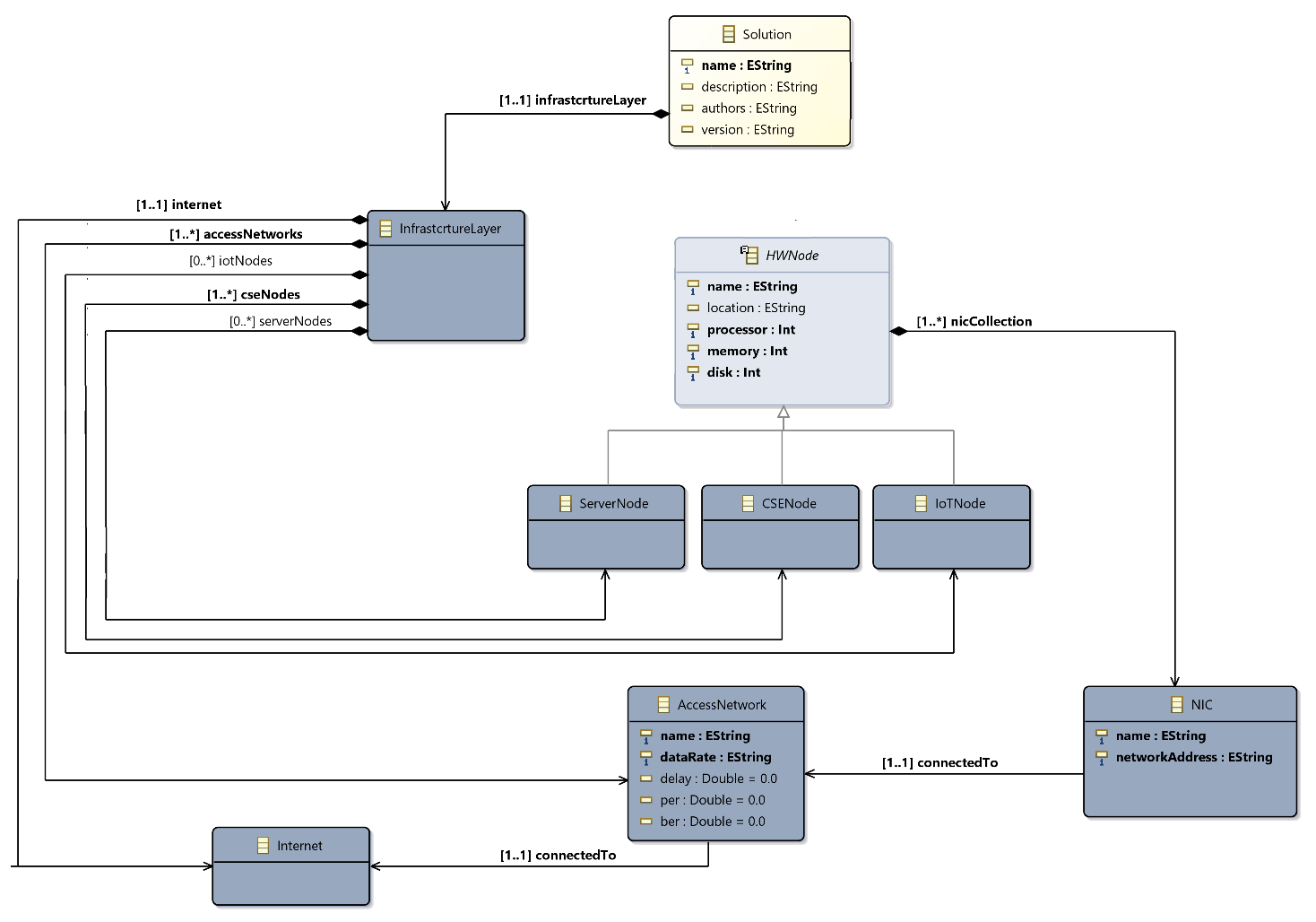
|  |  |
| --- | --- |
| **ReleaseVersion** | |
| **Short text description** | An enumeration for all supported oneM2M release versions. |
| **Values** | * V1 (oneM2M v1) * V2a (oneM2M v2a) * V3 (oneM2M v3) * V4 (oneM2M v4) |

|  |  |
| --- | --- |
| **ProtocolBindingType** | |
| **Short text description** | An enumeration for all supported protocol bindings. |
| **Values** | * HTTP * MQTT * COAP |

|  |  |
| --- | --- |
| **ResourceType** | |
| **Short text description** | An enumeration for all currently supported oneM2M resource types. |
| **Values** | * CSE\_BASE * ACP * AE * CNT * CIN * SUB |

# 8 oneM2M2 Solution Deployment Descriptor (OSDD)

The OSDD model on Figure 5 describes the hardware platform that hosts the IoT application. This infrastructure is made of multiple nodes of different types, interconnected by a network. This model quantifies the capacities (memory storage, processing capabilities, location) of the physical nodes and the underlying communication networks, The deployment refers to the allocation of CSEs on these physical hardware nodes.



**Figure 5: OSDD model (Infrastructure Layer)**

The detailed description of the model attributes, with related type and informal description is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| **InfrastructureLayer** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| iotNodes | 0..n | IoTNode | List of IoT devices in the IoT solution. |
| cseNodes | 1..n | CSENode | List of CSE nodes in the IoT |
| serverNodes | 0..n | ServerNode | List of specific servers if needed for MQTT or persistency |
| accessNetworks | 1..n | AccessNetwork | List of Network connected to this node |

|  |  |  |  |
| --- | --- | --- | --- |
| **HWNode** | | | |
| This is an abstract class that describes a runtime node. It is derived into IoTNode, CSENode, and ServerNode. | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the Hardware node. |
| nicCollection | 1..n | NIC | Collection of the NICs available on this hardware node. |
| processor | 1..1 | Integer | Quantifier of the processing capability of this node in terms of number of instructions per second. |
| memory | 1..1 | Integer | Quantifier of the memory capability of this node in terms of number of bytes. |
| disk | 1..1 | Integer | Quantifier of the storage capability of this node in terms of number of bytes. |
| location | 0..1 | String | Geographical location of the node. |

|  |  |  |  |
| --- | --- | --- | --- |
| **IoTNode : HWNode** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| (void) | (void) | (void) | (void) |

|  |  |  |  |
| --- | --- | --- | --- |
| **CSENode : HWNode** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| (void) | (void) | (void) | (void) |

|  |  |  |  |
| --- | --- | --- | --- |
| **ServerNode : HWNode** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| (void) | (void) | (void) | (void) |

|  |  |  |  |
| --- | --- | --- | --- |
| **NIC** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the NIC. |
| networkAddress | 1..1 | String | Network address of the NIC. |
| connectedTO | 1..1 | AccessNetwork | Reference of the AccessNetwork this NIC is connected to. |

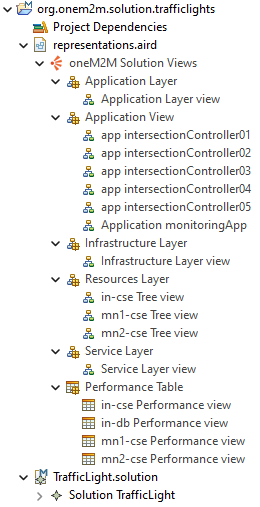
|  |  |  |  |
| --- | --- | --- | --- |
| **AccessNetwork** | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| name | 1..1 | String | Name of the access network. |
| dataRate | 1..1 | String | A string expressing a formula for representing the data rate based on parameters such as distance. |
| delay | 1..1 | Decimal | Network delay in seconds. |
| packetErrorRate | 0..1 | Decimal | The packet error rate in percentage. |
| bitErrorRate | 0..1 | Decimal | The bit error rate in percentage. |
| NOTE: Any network topology in the meta model is composed of a list of access networks connected to the Internet. | | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Internet** | | | |
| This a special class that represents the core network interconnecting all access networks present in the IoT solution. This class is instantiated only once. Its network characteristics are assumed perfect since any network characteristics (delay, data rate, etc.) to be considered are expressed at the level of the access networks. | | | |
| **Attribute** | **Cardinality** | **Type** | **Short text description** |
| (void) | (void) | (void) | (void) |

# 9 Example of instantiation

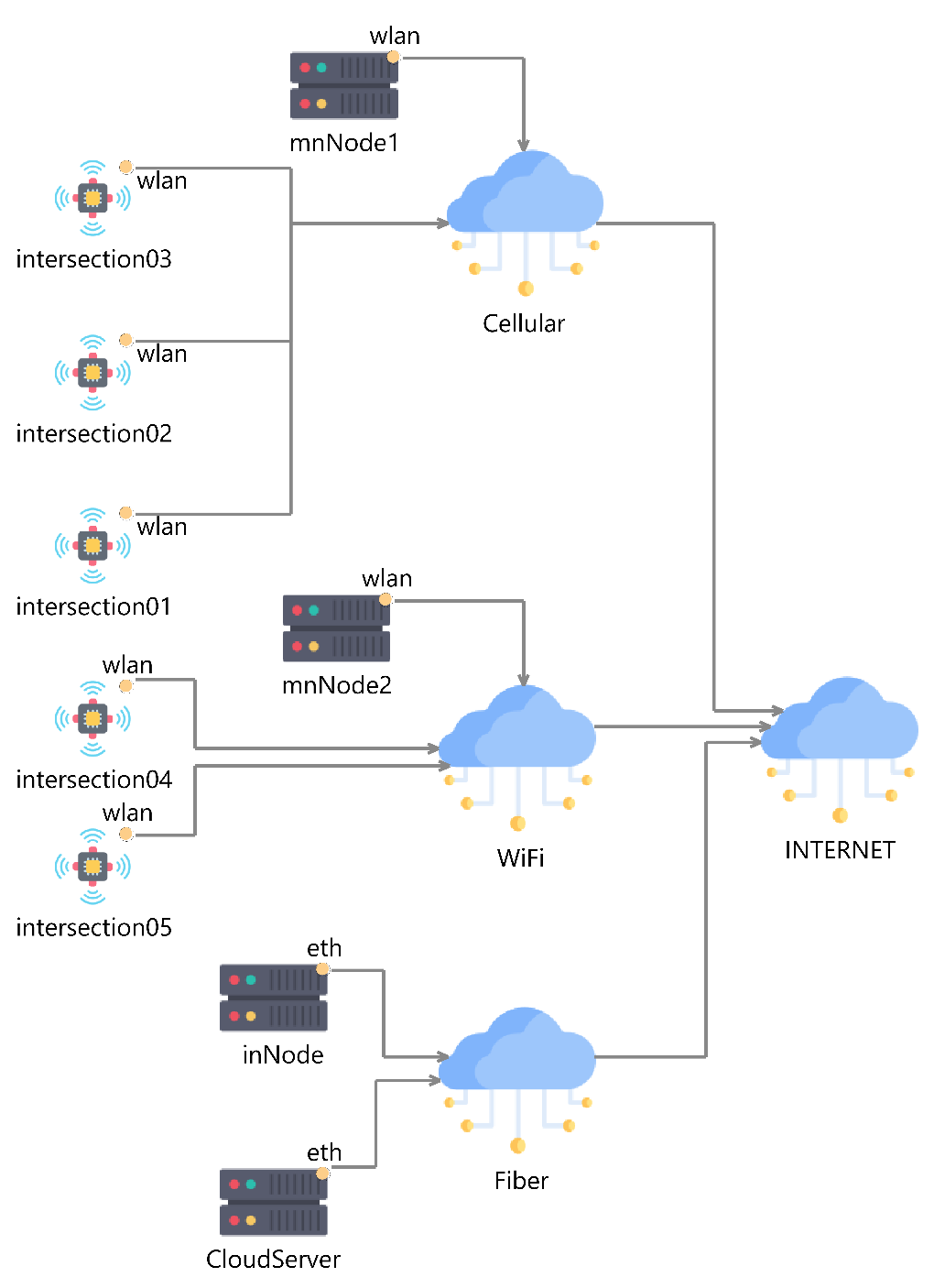
The MM proposed in this document is instantiated to represent the traffic lights use case of document [i.1]. It includes a simple execution infrastructure, the deployment of CSEs and the oneM2M resources necessary to represent this scenario.

The instance of the meta-model can be presented using different views. Figure 6 presents the list of the available views :



**Figure 6: The different representations of the Traffic Light instance of the meta-model**

The infrastructure (figure 7) view of this instance is as follows:

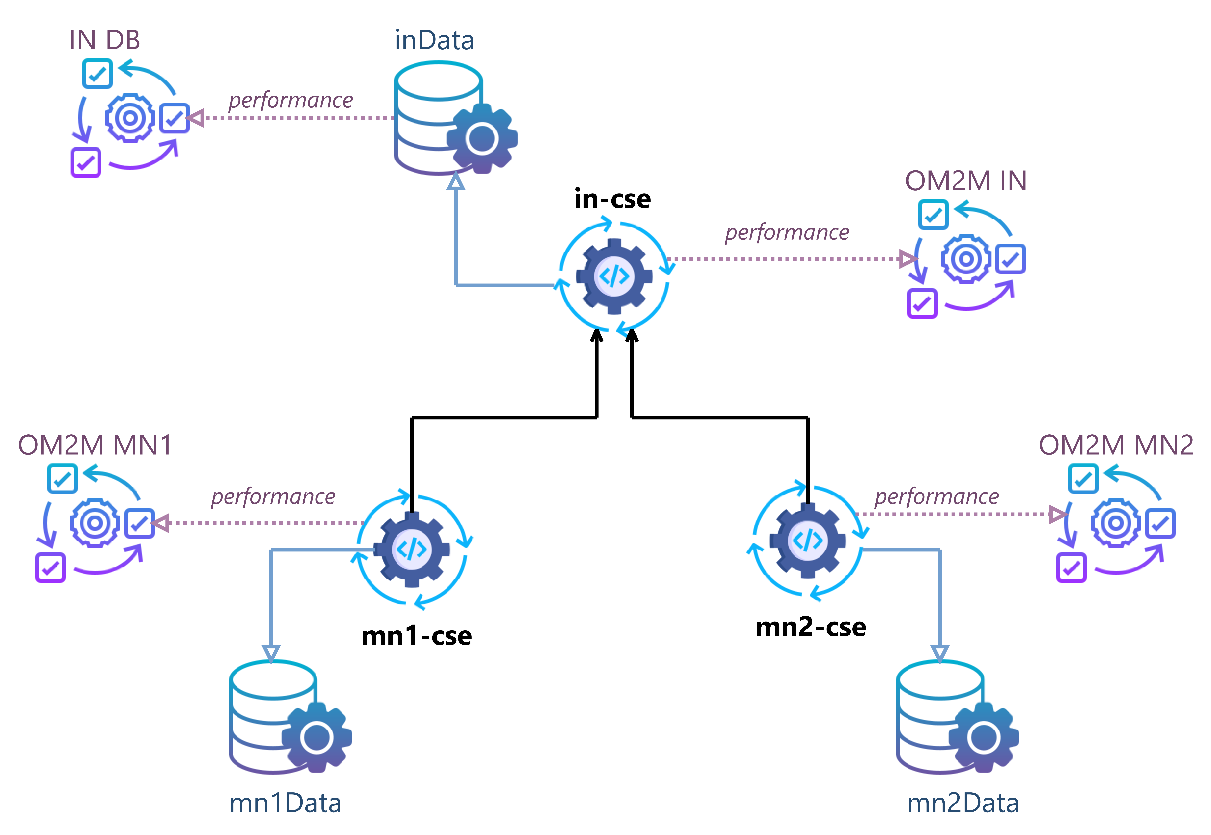


**Figure 7: Infrastructure layer view of the Traffic Light use case**

In this example, 5 intersections have been considered. IoT devices of intersections 01, 02, 03 are connected to a CSE node: **mnNode1** through a cellular network. IoT devices of intersections 04 and 05 are connected to another CSE node: **mnNode2** through a Wi-Fi network. Finally, a third CSE node: **inNode** is also part of this IoT solution and is connected to a fiber network. All the three access networks (Cellular, Wi-Fi, Fiber) are connected to the Internet; thus, a connectivity exists between all the nodes.

Following the meta-model, the nodes have attributes describing their capacities in terms of memory, processing, and storage. The access networks have attributable describing their rate, delay, and error rates.

The oneM2M service layer view is as follows (figure 8):



**Figure 8: Service layer view of the Traffic Lights use case**

In this example, a simple oneM2M topology is considered. The IoT solution is composed of two middle nodes CSEs : **mn1-cse** and **mn2-cse**. They are registered to an infrastructure node CSE: **in-cse**.

It is worth noting that these 3 CSEs are running on different hardware nodes not represented in this view. **mn1-cse** is running on **mnNode1**, **mn2-cse** is running on **mnNode2**, and **in-cse** is running on **inNode** (cf. Figure 7).

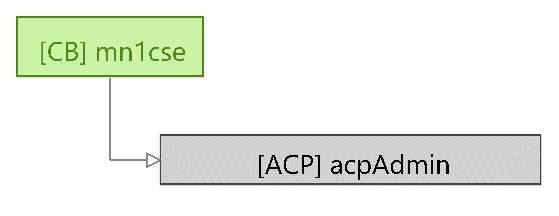
The data persistence service of each CSE is also represented in this view. Both CSE and data persistence service can have a descriptor of their performances. In this example, all CSE are following an Eclipse OM2M implementation performance (described later). The data persistence service of the **in-cse** is also described. However, the performances of the data persistence service on **mn1-cse** and **mn2-cse** are supposed negligeable (thus not shown in this instance).

Following the meta-model, every CSE maintains a resource tree (figure 9). A different view can show this tree structure. The following figures shows these resource trees.

Une image contenant diagramme

Description générée automatiquement

**in-cse oneM2M Resource Tree**



**mn1-cse oneM2M Resource Tree**

Une image contenant diagramme

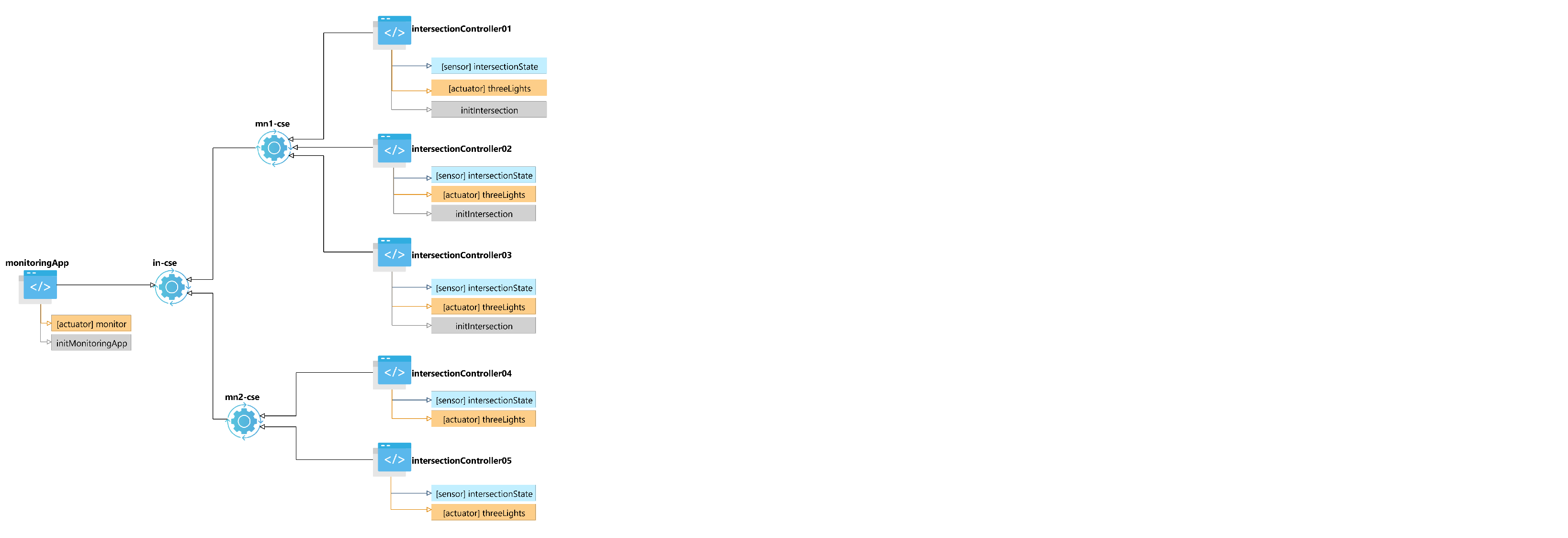
Description générée automatiquement

**mn2-cse oneM2M Resource Tree**

**Figure 9: Resource Tree of MN and IN CSE**

It is worth noting that the resource tree is predetermined on **mn2-cse** but not on **mn1-cse**.

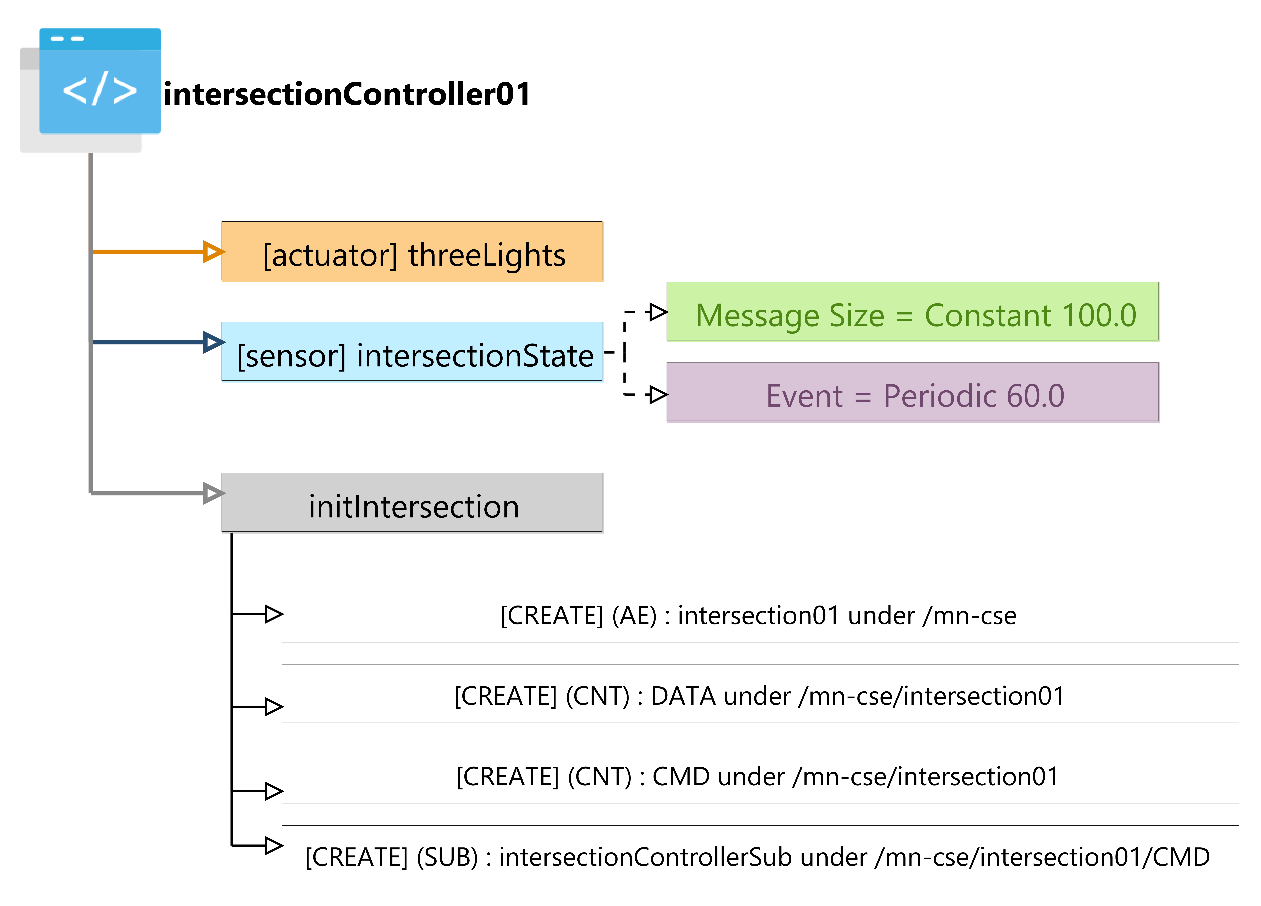
The following figure 10 shows the relationship between the IoT applications and the oneM2M service layer.



**Figure 10: Application layer view of the Traffic Lights use case**

From this view, for intersections 01, 02, and 03 controllers (i.e., IoT applications) initialization procedures have been defined along with one sensor: **intersectionState** and one actuator: **threeLights**. However, intersections 04 and 05 controllers are described with the same sensor and actuator but with no initialization procedures.

A detailed look into one IoT application shows the characteristics of the application behavior in terms of initialization procedure but also in terms of data generation (figure 11) :



**Application view of intersectionController01 application**

Une image contenant diagramme

Description générée automatiquement

**Application view of intersectionController04 application**

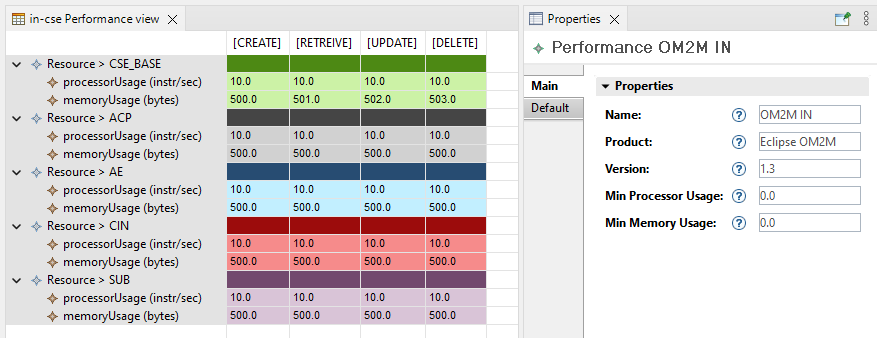
**Figure 11: IntersectionController applications**

In intersectionController01, an initialization procedure has been defined. It is composed of list of the CRUD operation that will be executed at the startup of the simulation. In this example, an ApplicationEntity is first created under the remote CSE’ base. Then two oneM2M containers: **DATA** and **CMD** are created. Finally, a subscription resource is created under the **CMD** container.

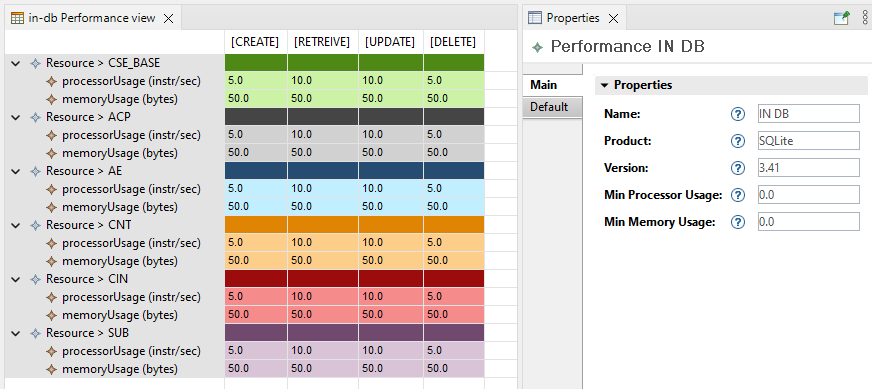
The data generated by the sensor representing the intersection state will be simulated through a periodic process with a period of 60 seconds and generated data messages with a constant size of 100 bytes.

Since the resource tree on the **mn2-cse** (the remote CSE for intersections 04 and 05) is already defined, **intersectionController04** does not have an initialization procedure. The sensor is described the same way as in the previous IoT application.

Finally, all the CSE involved in this IoT solution have performance characteristics that can be visualized in tables. The figure 12 shows an overview of performance of both the **in-cse** and its data persistence service.



**in-cse Performance Table**



**in-db Performance Table**

**Figure 12: Performance tables overview**

As stated in the meta-model, a performance description includes the usage in terms of memory, processor, and disk for every oneM2M resource and for each of the CRUD operations. For example, from the **in-db** performance table, the resource of type **CSE\_BASE** requires 5 instructions of processing for **CREATE** and **DELETE** operations. But it requires 10 instructions of processing for **RETRIEVE** and **UPDATE** operations. In this example, the numbers in the table are chosen arbitrarily.

# 10 Conclusions

The present document gives a detailed view of a model able to describe at a high level, a oneM2M IoT system and to evaluate the performance of such a system.

The meta model developed in Clause 4 gives an overview of the model. The different parts are detailed in Clauses 6, 7 and 8. This meta model makes it possible to represent an overall oneM2M IoT system *i.e.,* the hardware and network infrastructure, the CSEs deployed and the application that will generate exchanges through the oneM2M resources. In order to evaluate the performance of such a system clause 5 lists the set of KPIs and parameters needed to set up a performance analysis of the IoT system and to describe the expected simulation results. All of this information will be used in document [i.3] and [i.4] for the development of a performance analysis tool and for the representation of the system to be analyzed.

Annex A (informative):  
Change History

| Date | Version | Information about changes |
| --- | --- | --- |
| Mars 2023 | 0.0.1 | Bootstrapping Intro and Table of Contents of the document (T. Monteil) |
| Mars 2023 | 0.1.1 | Filling Clauses 4, 6, and 7 (T. Monteil and S. Medjiah) |
| April 2023 | 0.1.2 | Filling Clauses 4, 6, 5 and 7 (T. Monteil, S. Medjiah, L. Liquori, MA Peraldi-Frati) |
| April 2023 | 0.1.3 | Typos, Type and Proof checking in Tables, instantiation on traffic light (L. Liquori, T. Monteil, S. Medjiah) |
| May 2023 | 0.1.4 | Version review with comments to be addressed (P. Guillemin) |
| May 2023 | 0.1.5 | Revision with cleanup (X. Piednoir) |

# History

|  |  |  |
| --- | --- | --- |
| **Document history** | | |
| <Version> | <Date> | <Milestone> |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

*Latest changes made on 2023-05-23*