An overview about the oneM2M Open Source Ecosystem

Andreas Kraft
(EXACTA GSS)

Andreas Kraft

+25 years of experience as senior researcher and principal enterprise architect for IoT @ Deutsche Telekom

Since 2024 consulting on oneM2M and IoT

Maintainer of the ACME oneM2M CSE Open Source implementation and other oneM2M-related projects

Active contributor to oneM2M

The Adventures First! Explanations Take Such a Dreadful Time

Open Source in the oneM2M Partnership Project

Why Open Source for Standards Development?

Implementation first

- ➡ Validation of new features and test cases
- ➡ Ensure the standard is implementable and interoperable

Make the standard available to a broader audience

- ➡ Not only code: tutorials, tools, art-ware …
- ➡ Education, training, and marketing

Backend Implementations

- ➡ [ACME] CSE
- ➡ KETI Mobius
- ➡ Sejong U tinyIoT
- ➡ Eclipse OM2M

Application and Community Support

- ➡ ATIS Open Source - IoT library
- ➡ Arduino oneM2M connectivity libraries on GitHub
- ➡ Jupyter Notebooks
- ➡ Tutorials & Articles
- ➡ Hackster.io Projects
- ➡ …

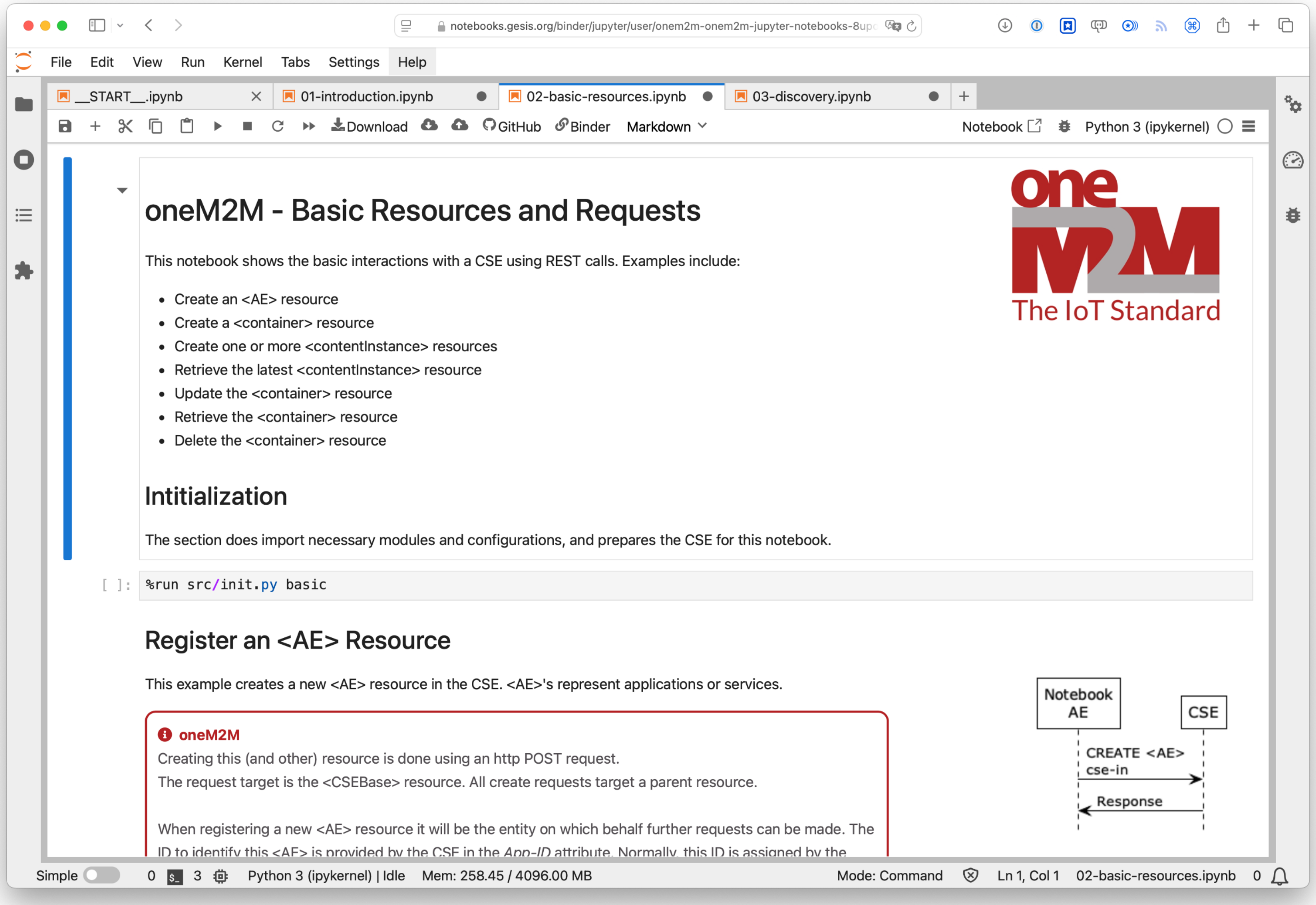
Specification Tools and more

- ➡ Data Modelling Language
- ➡ Specification Converters:
  .docx → .md
  .md → *
- ➡ Architecture Icons
- ➡ …

And what is the use of a book without pictures or conversation?

Examples

Jupyter Notebooks: Learn oneM2M in an evening



oneM2M - Basic Resources and Requests

This notebook shows the basic interactions with a CSE using REST calls. Examples include:

- Create an <AE> resource
- Create a <container> resource
- Create one or more <contentInstance> resources
- Retrieve the latest <contentInstance> resource
- Update the <container> resource
- Retrieve the <container> resource
- Delete the <container> resource

Intitialization

The section does import necessary modules and configurations, and prepares the CSE for this notebook.

```
[ ]: %run src/init.py basic
```

Register an <AE> Resource

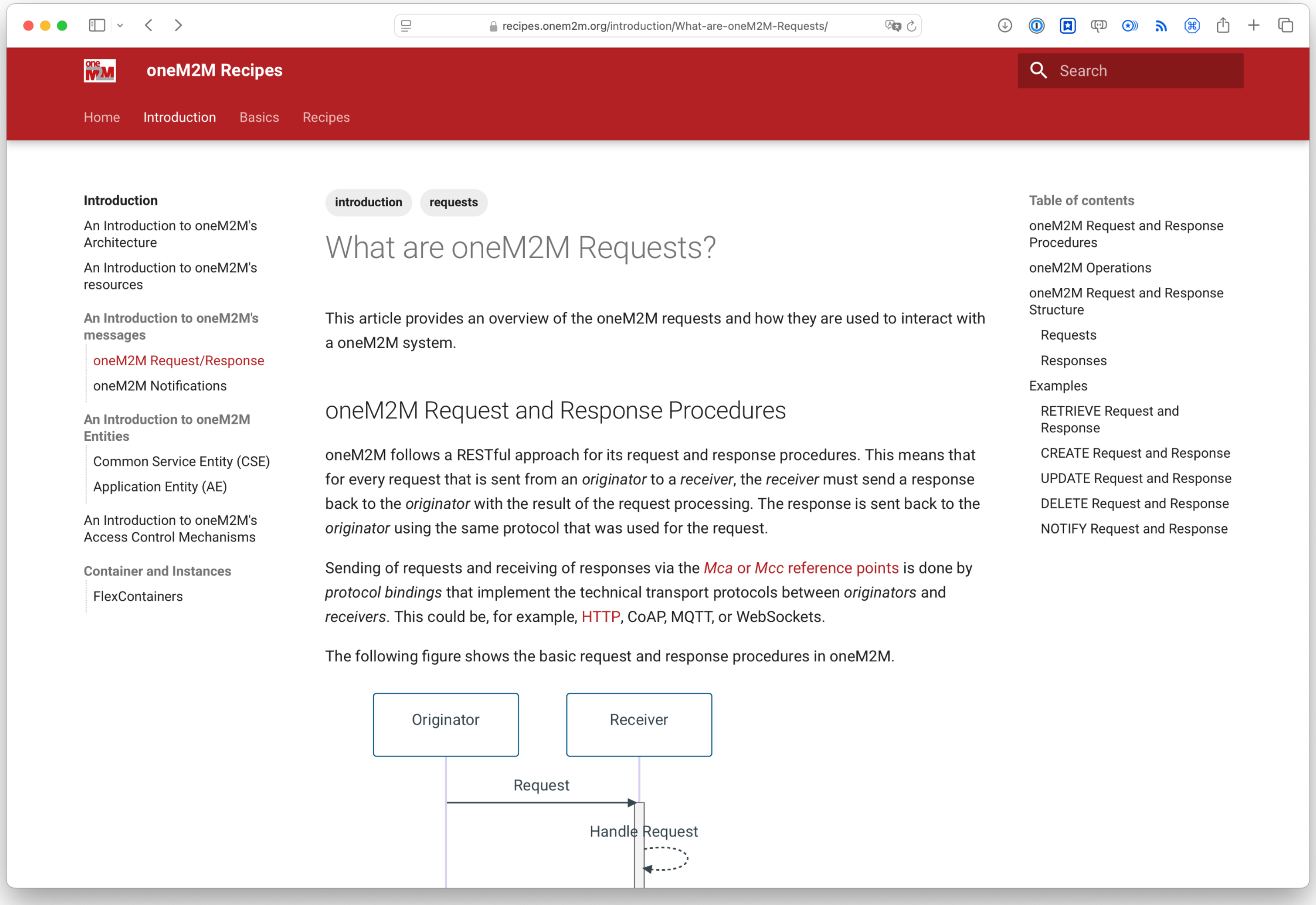
This example creates a new <AE> resource in the CSE. <AE>'s represent applications or services.

oneM2M
Creating this (and other) resource is done using an http POST request.
The request target is the <CSEBase> resource. All create requests target a parent resource.

When registering a new <AE> resource it will be the entity on which behalf further requests can be made. The ID to identify this <AE> is provided by the CSE in the *App-ID* attribute. Normally, this ID is assigned by the

Simple 0 3 Python 3 (ipykernel) | Idle Mem: 258.45 / 4096.00 MB Mode: Command Ln 1, Col 1 02-basic-resources.ipynb 0

oneM2M Recipes: A Cookbook for oneM2M



recipes.onem2m.org/introduction/What-are-oneM2M-Requests/

oneM2M Recipes

Search

Home Introduction Basics Recipes

Introduction

An Introduction to oneM2M's Architecture

An Introduction to oneM2M's resources

An Introduction to oneM2M's messages

- oneM2M Request/Response
- oneM2M Notifications

An Introduction to oneM2M Entities

- Common Service Entity (CSE)
- Application Entity (AE)

An Introduction to oneM2M's Access Control Mechanisms

Container and Instances

- FlexContainers

introduction requests

What are oneM2M Requests?

This article provides an overview of the oneM2M requests and how they are used to interact with a oneM2M system.

oneM2M Request and Response Procedures

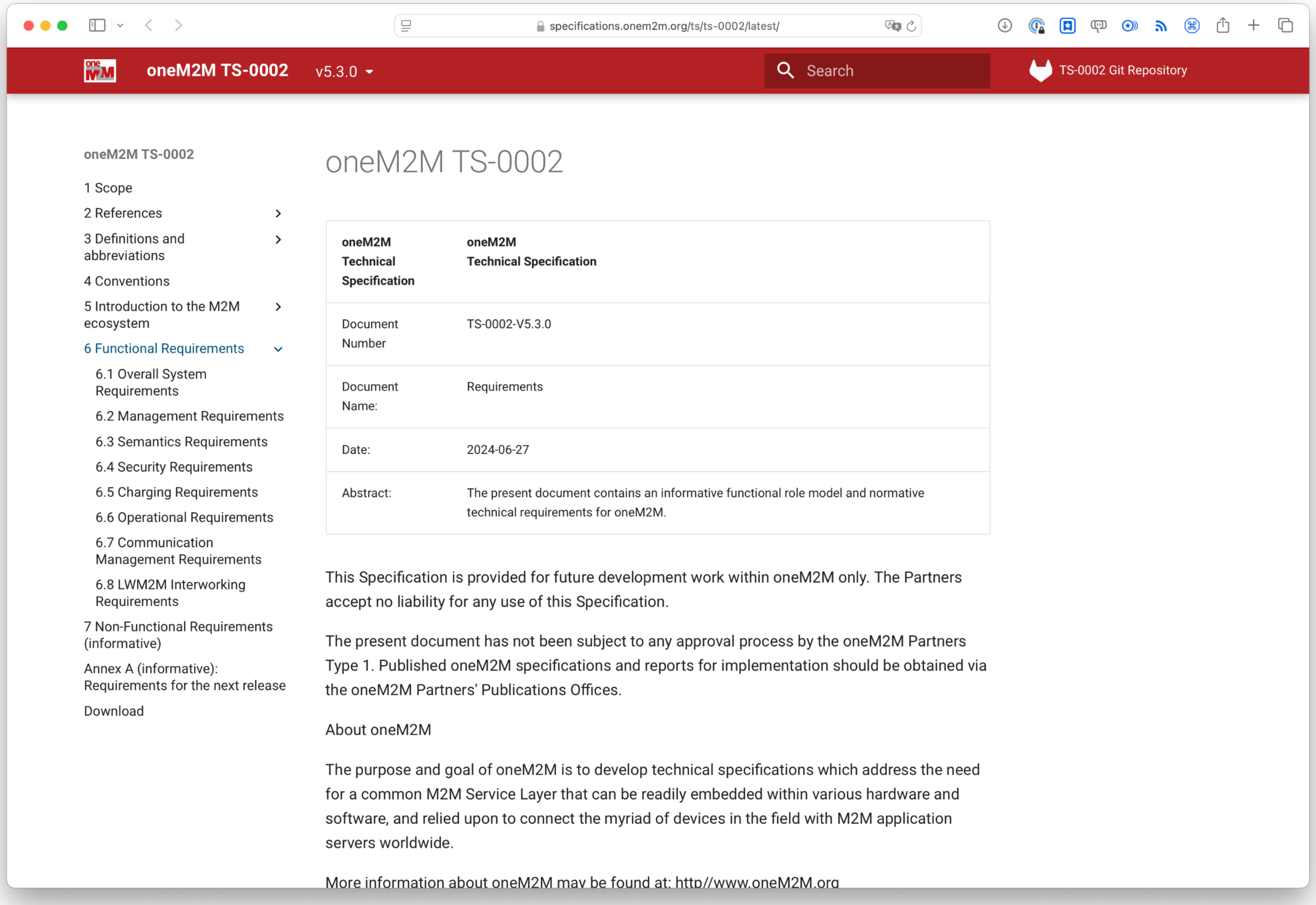
oneM2M follows a RESTful approach for its request and response procedures. This means that for every request that is sent from an *originator* to a *receiver*, the *receiver* must send a response back to the *originator* with the result of the request processing. The response is sent back to the *originator* using the same protocol that was used for the request.

Sending of requests and receiving of responses via the *Mca* or *Mcc* reference points is done by *protocol bindings* that implement the technical transport protocols between *originators* and *receivers*. This could be, for example, HTTP, CoAP, MQTT, or WebSockets.

The following figure shows the basic request and response procedures in oneM2M.

Table of contents

oneM2M Specifications: Ongoing Work



specifications.onem2m.org/ts/ts-0002/latest/

oneM2M TS-0002 v5.3.0 Search TS-0002 Git Repository

oneM2M TS-0002

oneM2M Technical Specification	oneM2M Technical Specification
Document Number	TS-0002-V5.3.0
Document Name:	Requirements
Date:	2024-06-27
Abstract:	The present document contains an informative functional role model and normative technical requirements for oneM2M.

This Specification is provided for future development work within oneM2M only. The Partners accept no liability for any use of this Specification.

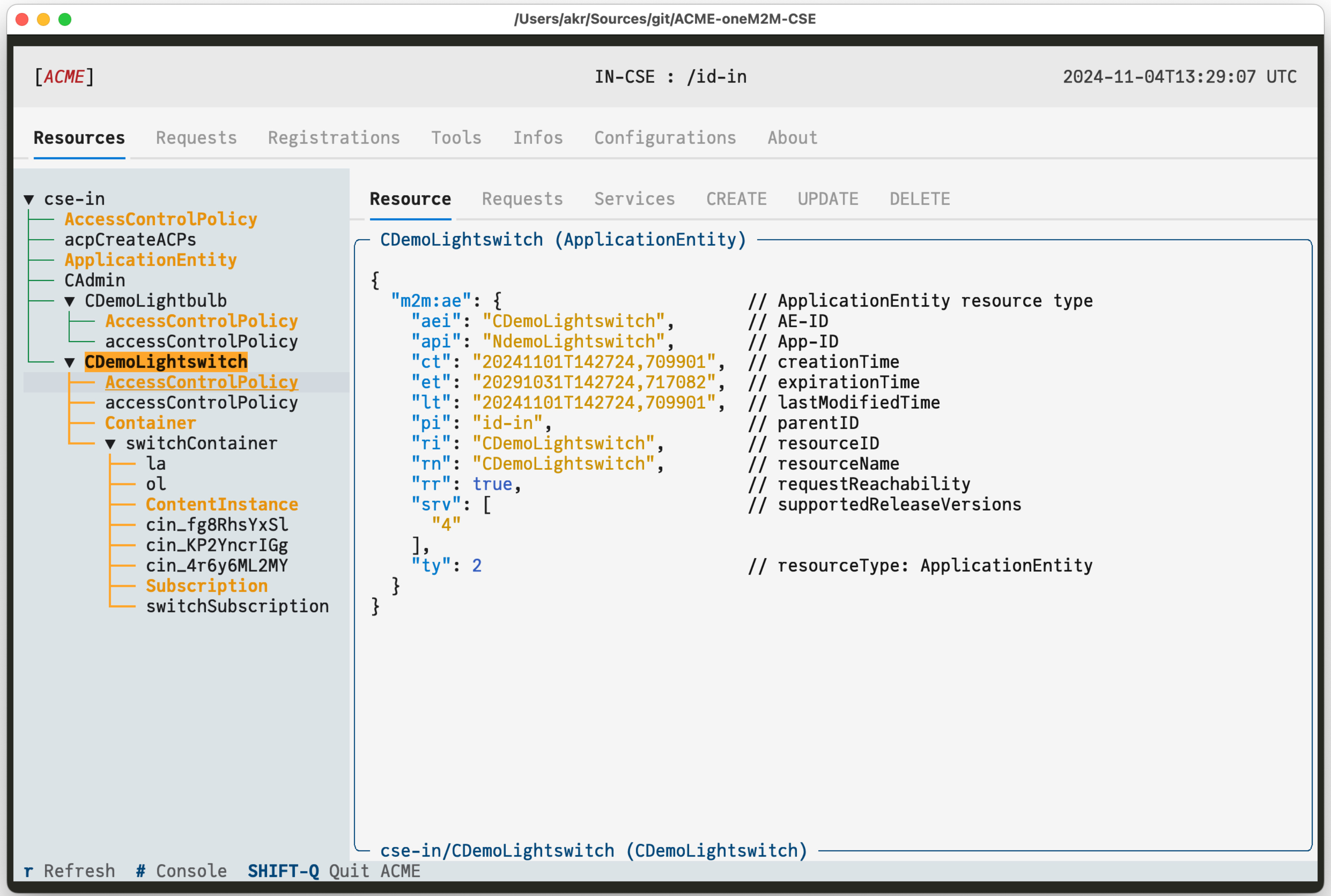
The present document has not been subject to any approval process by the oneM2M Partners Type 1. Published oneM2M specifications and reports for implementation should be obtained via the oneM2M Partners' Publications Offices.

About oneM2M

The purpose and goal of oneM2M is to develop technical specifications which address the need for a common M2M Service Layer that can be readily embedded within various hardware and software, and relied upon to connect the myriad of devices in the field with M2M application servers worldwide.

More information about oneM2M may be found at: http//www.oneM2M.org

ACME CSE: An OSS oneM2M Implementation



/Users/akr/Sources/git/ACME-oneM2M-CSE

```
[ACME]                         IN-CSE : /id-in                         2024-11-04T13:29:07 UTC

Resources   Requests   Registrations   Tools   Infos   Configurations   About

▼ cse-in                          Resource   Requests   Services   CREATE   UPDATE   DELETE
├── AccessControlPolicy           ┌ CDemoLightswitch (ApplicationEntity) ─────────────────────────
├── acpCreateACPs                 │
├── ApplicationEntity             │ {
├── CAdmin                        │   "m2m:ae": {                          // ApplicationEntity resource type
├── ▼ CDemoLightbulb              │     "aei": "CDemoLightswitch",         // AE-ID
│   ├── AccessControlPolicy       │     "api": "NdemoLightswitch",         // App-ID
│   └── accessControlPolicy       │     "ct": "20241101T142724,709901",    // creationTime
└── ▼ CDemoLightswitch            │     "et": "20291031T142724,717082",    // expirationTime
    ├── AccessControlPolicy       │     "lt": "20241101T142724,709901",    // lastModifiedTime
    ├── accessControlPolicy       │     "pi": "id-in",                     // parentID
    ├── Container                 │     "ri": "CDemoLightswitch",          // resourceID
    └── ▼ switchContainer         │     "rn": "CDemoLightswitch",          // resourceName
        ├── la                    │     "rr": true,                        // requestReachability
        ├── ol                    │     "srv": [                           // supportedReleaseVersions
        ├── ContentInstance       │       "4"
        ├── cin_fg8RhsYxSl        │     ],
        ├── cin_KP2YncrIGg        │     "ty": 2                            // resourceType: ApplicationEntity
        ├── cin_4r6y6ML2MY        │   }
        ├── Subscription          │ }
        └── switchSubscription    │
                                  └ cse-in/CDemoLightswitch (CDemoLightswitch) ─────────────────────
r Refresh  # Console  SHIFT-Q Quit ACME
```

It's a poor sort of memory that only works backwards

Experiences and Recommendations

Choose open and friendly licenses to allow for reuse, modification, and contributions

- ➡ For code
- ➡ For tutorials, extra documentation and presentations, and art-ware

Eat you own dog food

- ➡ Make use of (your own) Open Source tools in standards development

Plan for the future of your Open Source projects

- ➡ Plan for resources to maintain and further develop the projects
- ➡ Set up a “Software Development Group” to give guidance and long-term support to the projects and the community
- ➡ Support relevant third-party projects

Begin at the beginning, and go on till you come to the end: then stop

Thank You!