Introduction

Ike Alisson
LF Edge Akraino Documentation Sub-committee TSC Chair
September 15h, 2021
Table of Contents

1. LF Edge Akraino Overview
2. IoT Market Overview
3. LF Edge Akraino IoT Area
   3.1 Akraino IoT Blueprint (BP) Families
   3.2 OPC UA IEC 62 541 IoT Standard for M2M and M2E
   3.3 oneM2M IoT Service Layer (SL) Standard
1.1 LF Edge Akraino Project Overview
Number of publicly known "IoT Platforms" (2015-2019)

Source(s): IoT Analytics Research

40+ example providers

Copyright © 2019 by www.iot-analytics.com. All rights reserved.
Number of Identified IoT Platforms – By industry (Dec 2019)

Note: Percentages do not add up to 100% as most companies focus on several segments  
Source: IoT Analytics 2020 List of IoT Platform Companies, n=620
IoT Area

Blueprint Families

ELIOT: Edge Lightweight and IoT Blueprint Family
IloT at the Smart Device Edge (family)
oneM2M IoT Service Layer (SL) Platform
Project Cassini - IoT and Infrastructure Edge Blueprint Family

Security

PARSEc, the opensource CNCF project has been adopted for edge deployments as it offers a common API that abstracts secure roots of trust which are required to protect devices outside of the datacenter. This enables the cloud native principle of being able to freely move your applications from one platform to another while maintaining level of security that was not possible in the past.

Building on this abstraction, PARSEC can mediate access to hardware security primitives and create isolated key stores for a multi-tenancy environment.

Starting your project with the right platform for security will accelerate your deployments and scale.

Come read about Parsec at: https://parallaxsecond.github.io/parsec-book
And talk to the experts during our weekly community calls (see github).
Or join us on the CNCF slack channel: https://cloud-native.slack.com
1. 4 Akraino IoT Area

3. IoT Area
   - ELIoT: Edge Lightweight and IoT Blueprint Family
   - ILoT at the Smart Device Edge (family)

4. oneM2M IoT Service Layer (SL) Platform
   - 3GPP 5G HMTC (High Performance Machine Type Communication) SST 🇫🇷
   - oneM2M Cloud Vendor Independent & ETSI MEC support
   - oneM2M IoT SL and AI/ML use
   - oneM2M IoT SL and CIM NGSI-LD (Context Information Management Next Generation Service Interface for Legacy Devices)
   - oneM2M IoT SL Architecture
   - oneM2M IoT SL Common Service Functions (CSFs) (applied to all IoT Devices)
   - oneM2M IoT SL pre-integrated with 5G (3GPP) Specifications for IoT & Cloud
   - oneM2M IoT SL Release Roadmap
   - oneM2M Semantic enablement and ASD (Advanced Semantic Discovery)
   - oneM2M Test Suite Structure (TSS) and Test Purposes
   - oneM2M Use Cases (UCs) and SAREF (Smart Applications REference) Ontologies

5. OPC UA Standard IEC 62 541 for Open Platform Communication Unified

Dashboard / ... / IoT Area

oneM2M IoT Service Layer (SL) Platform
Skapad av ke Alsson, senast ändrad den 04.2021

oneM2M Overview

The oneM2M Global Organization creates Technical Specifications (TSs) to ensure that Machine-to-Machine (M2M) Communications can effectively operate on a Worldwide scale.

Seven (7) of the World’s leading Information and Communications Technology (ICT) Standards Development Organizations (SDOs) launched in July 2012 a new Global Organization to ensure the most efficient Deployment of Machine-to-Machine (M2M) Communications Systems.

The new organization, called oneM2M, develops specifications to ensure the Global Functionality of M2M—allowing a range of Industries to effectively take advantage of the benefits of this emerging Technology.

The seven (7) major ICT SDO founders of oneM2M are:

- The European Telecommunications Standards Institute (ETSI), Europe
- The Association of Radio Industries and Businesses (ARIB), Japan
- The Telecommunication Technology Committee (TTC), Japan
- The Alliance for Telecommunications Industry Solutions (ATIS), USA
- The Telecommunications Industry Association (TIA), USA
- The China Communications Standards Association (CCSA), China
- The Telecommunications Technology Association (TTA), Korea

The members of the organization are devoted to developing Technical Specifications and Reports to ensure M2M Devices can successfully communicate on a Global scale.

The oneM2M Standardization work is split in five (5) WG:
1.4 Akraino IoT Area

OPC UA Standard IEC 62 541 for Open Platform Communication Unified Architecture

OPC UA (Open Platform Communication Unified Architecture) provides the necessary infrastructure for interoperability across the Enterprise, from Machine-to-Machine (M2M), Machine-to-Enterprise (M2E) and everything in-between.

The OPC UA was initially released in 2005 - 2008 and has a very broad Market deployment footprint since then. OPC UA specifies a Platform independent Service-oriented Architecture, that integrates all the functionality of the individual OPC Classic Specifications into one (1) extensible Framework.


The current version of the OPC UA specification is on 1.04 (22 November 2017). The new version of OPC UA has added Publish/Subscribe in addition to the Client/Server communications infrastructure.

The OPC UA Information Model is a so-called Full Mesh Network based on nodes. The OPC UA Architecture supports two (2) Protocols. This is visible to Application programmers only via changes to the URL. The binary protocol is opc.tcp://Server and http://Server is for Web Service. Otherwise OPC UA works completely transparent to the API.

After the initial release in 1996, the OPC Foundation was created to maintain the Standard. As OPC has been adopted beyond the field of Process Control, the OPC Foundation changed the name to Open Platform Communications in 2011. The change in name reflects the Applications of OPC Technology for Applications in Building Automation, Discrete Manufacturing, Process Control and many others. OPC has also grown beyond its original OLE (Object Linking and Embedding) implementation to include other Data transportation Technologies including Microsoft’s .NET Framework, XML, and even the OPC Foundation’s binary-encoded TCP format.

The OPC UA Multi-Layered approach accomplishes the original design specification goals of:

- **functional equivalence**: all COM OPC Classic specifications are mapped to UA
- **Platform independence**: from an embedded micro-controller to cloud-based infrastructure
- **Secure**: encryption, authentication, and auditing
- **Extensible**: ability to add new features without affecting existing applications
- **Comprehensive information modeling**: for defining complex information

**Functional Equivalence**

Building on the success of OPC Classic, OPC UA was designed to enhance and surpass the capabilities of the OPC Classic specifications. OPC UA is functionally equivalent to OPC Classic, yet capable of much more:
For further information on the OPC UA, please see attached below the OPC UA Open IEC 62 541 (current) Documentation from Jan 2021.
1.4 Akraino IoT Area - 5 (OPC UA IEC 62 541)

2.1 Building the Library

2.1.1 Building with CMake on Ubuntu or Debian

```
sudo apt-get install git build-essential gcc pkg-config cmake python
# enable additional features
sudo apt-get install cmake-curses-gui # for the cmake graphical interface
sudo apt-get install lib ldap # for encryption support
sudo apt-get install check libkdeutil4-dev # for unit tests
sudo apt-get install python-sphinx graphviz # for documentation generation
sudo apt-get install python-sphinx-rtd-theme # documentation style
```

- cd OPEN5241
- mkdir build
- cd build
- cmake ..
- make

# select additional features
- cmake ..
- make

# build documentation
- make doc # html documentation
- make doc_pdf # pdf documentation (requires LATeX)

2.1.2 Building with CMake on Windows

Here we explain the build process for Visual Studio (2013 or newer). To build with MinGW, just replace the compiler selection in the call to CMake.

- **Download and install**
  - Python 2.7.x (Python 3.x works as well): https://python.org/downloads
  - CMake: http://www.cmake.org/cmake/resources/software.html

2.3 Building the Examples

```
Make sure that you can build the shared library as explained in the previous steps. Even easier way to build the examples is to install open5241 in your operating system (see building open5241).
```

Then the compiler should automatically find the includes and the shared library.

```
cp /path/to/examples/tutorial_server_firststeps.c . # copy the example server
gcc stdc++99 -o server tutorial_server_firststeps.c -lopen5241
```

2.4 Building for specific architectures

The open5241 library can be build for many operating systems and embedded systems. This document shows a small except of already tested architectures. Since the code is only using the C99 standard, there are many more supported architectures.

A full list of implemented architecture support can be found in the arch folder.

- **2.4.1 Windows, Linux, MacOS**

These architectures are supported by default and are automatically chosen by CMake.

- **2.4.2 freeRTOS + LwIP**

Ceilios to @ceballosfras

This documentation is based on the discussion of the PR: https://github.com/open5241/open5241/pull/2511. If you have any doubts, please first check the discussion there.

2.3 Prebuilt packages

2.3.1 Pack branches

Github allows you to download a specific branch as .zip package. Just using this .zip package for open5241 will likely fail:

- CMake uses git describe --tags to automatically detect the version string. The .zip package does not include any git information
- Specific options during the build stack require additional git submodules which are not included in the .zip

Therefore we provide prebuilding branches. They have the prefix pack and are automatically updated to match the referenced branch.

Here are some examples:

- pack/master.zip
- pack/1.0.zip

These pack branches have refined submodules and the version string is hardened. If you need to build from source but do not want to use git, use these specific pack versions.

2.3.2 Prebuilt binaries

You can always find prebuilt binaries for every release on our Github Release Page.

Nightly single the releases for Linux and Windows of the last 50 commits can be found here: https://open5241.org/releases

2.3.3 Debian

Debian packages can be found in our official PPA:

- Daily Builds based on master branch: https://launchpad.net/~open5241-teams/ppa/debian/daily
- Release Builds (starting with Version 0.4): https://launchpad.net/~open5241-teams/ppa/debian/ppa

Install them with:

```
sudo add-apt-repository ppa:open5241-teams/ppa
sudo apt-get update
sudo apt-get install libopen5241-1-dev
```

2.3.4 Arch

Arch packages are available in the AUR:

- Stable Builds: https://aur.archlinux.org/packages/open5241/
- Unstable Builds (current master): https://aur.archlinux.org/packages/open5241-gtk/

In order to add custom build options (Build Options), you can set the environment variable OPEN5241_CMAKE_FLAGS
oneM2M Overview

The oneM2M Global Organization creates Technical Specifications (TSS) to ensure that Machine-to-Machine (M2M) Communications can effectively operate on a Worldwide scale.

Seven (7) of the World’s leading Information and Communications Technology (ICT) Standards Development Organizations (SDOs) launched in July 2012 a new Global Organization to ensure the most efficient Deployment of Machine-to-Machine (M2M) Communications Systems.

The new organization, called oneM2M, develops specifications to ensure the Global Functionality of M2M—allowing a range of Industries to effectively take advantage of the benefits of this emerging Technology.

The seven (7) majors ICT SDO founders of oneM2M are:

- The European Telecommunications Standards Institute (ETSI), Europe
- The Association of Radio Industries and Businesses (ARIB), Japan
- The Telecommunication Technology Committee (TTC), Japan
- The Alliance for Telecommunications Industry Solutions (ATIS), USA
- The Telecommunications Industry Association (TIA), USA
- The China Communications Standards Association (CCSA), China
- The Telecommunications Technology Association (TTA), Korea

The members of the organization are devoted to developing Technical Specifications and Reports to ensure M2M Devices can successfully communicate on a Global scale.

The oneM2M Standardization work is split in five (5) WG:
1. 4 Akraino IoT Area

- ELIoT: Edge Lightweight and IoT Blueprint Fam
- IIoT at the Smart Device Edge (family)

**oneM2M IoT Service Layer (SL) Platform**

- 3GPP 5G HMTC (High Performance Machine)
- oneM2M Cloud Vendor Independent & ETSI
- oneM2M IoT SL and AI/ML use
- oneM2M IoT SL and CIM NGSI-LD (Context I)
- oneM2M IoT SL Architecture
- oneM2M IoT SL Common Service Functions
  - oneM2M IoT SL pre-integrated with 5G (3GF)
  - oneM2M IoT SL Release Roadmap
  - oneM2M Semantic enablement and ASD (Ac
  - oneM2M Test Suite Structure (TSS) and Test I
  - oneM2M Use Cases (UCs) and SAREF (Smart
  - OPC UA Standard IEC 62 541 for Open Platf
- Project Cassini - IoT and Infrastructure Edge Bl
- Tami COVID-19 Blueprint Family
- Automotive Area
- Blueprint Proposals
- Akraino Feature Projects (a.k.a Development Project)

---

**SAREF and its extensions**

- Semantic interoperability
- SAREF Core
- Entry
- Building
- Environment
- Smart Cities
- Marketing
- Smart Agriculture
- Automotive
- E Health
- Smart Water

03/2017   03/2018   03/2019   03/2020
Welcome and Enjoy Akraino IoT Area Regional Developer Meetup!