

OPC UA ANWENDUNGSENTWICKLUNG MIT OPEN62541

IOT Konferenz – 11/2019 – Stuttgart

AGENDA

- Über uns
- Was ist open62541?
- Ein Blick auf das Ökosystem
- Erstellung eines Servers – ein Tutorial
- Zusammenfassung



ÜBER UNS



Über basysKom

Software Engineering Services

HMI und Anwendungsentwicklung

Typischer Kunde: Maschinenbau, Messtechnik, Automatisierung, Konsumgüter (DACH)

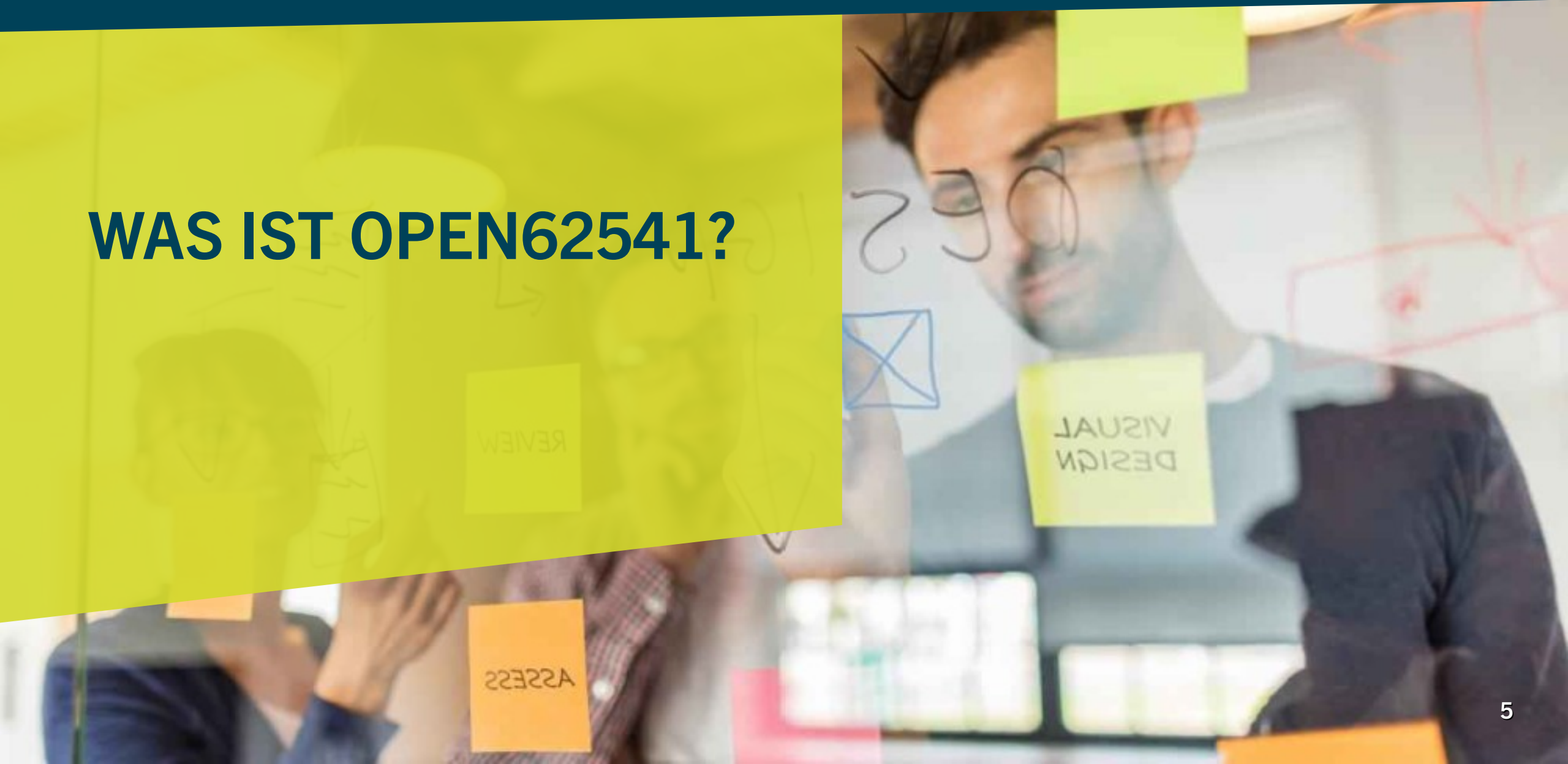
30 Mitarbeiter

Darmstadt und Nürnberg

Teil der Münchner UX-Gruppe

Open Source => open62541 & Qt OPC UA

WAS IST OPEN62541?



Was ist open62541?

Eine unabhängige Implementierung von IEC62541 / OPC UA

Was ist OPC UA?

- M2M-Protokoll/Framework (Entstanden im Kontext Industrie-Automatisierung/Prozesskontrolle)
- Nutzung in immer mehr Bereichen (Food/Beverage, Woodworking, ...)
- Offener Standard (entwickelt und gepflegt von der OPCFoundation)

Was ist open62541?

- Protokoll-Stack + SDK (Client & Server) + Tooling (u.a. für Code-Generierung)
- Umgesetzt in einem Subset C99/C++98
- Open Source

Der Fokus von open62541

Portabilität

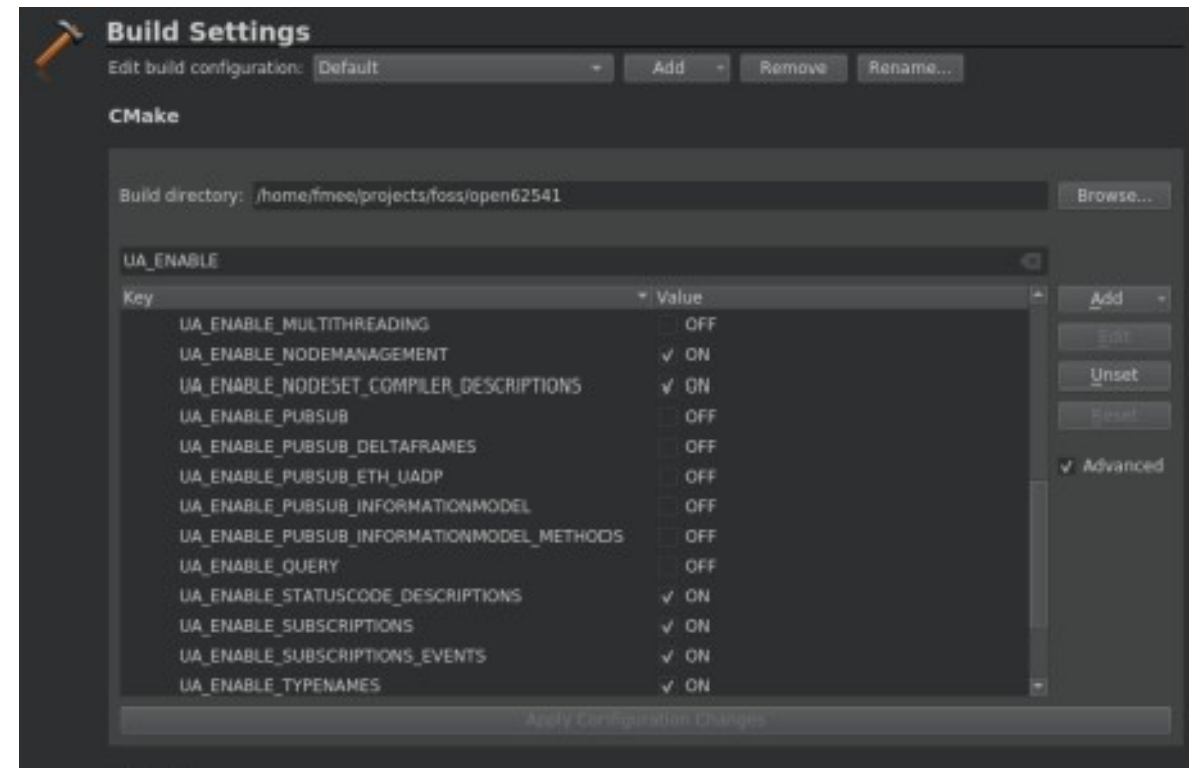
- Plattform-spezifische Funktionalität über „Plugins“
- Windows (Visual Studio, MinGW), Linux, Android, iOS, Microcontroller (FreeRTOS), ...

Schlank, konfigurierbar

Begrenzte Anzahl von Abhängigkeiten

Embedded Use-cases

- Implementiert das „Micro Embedded Device Server“ Profile (plus viele weitere Features)



Entwicklungsmodell

Entwicklung auf dem Master

Branches für Releases

- API Stabilität für die Releases
- Bugfixes

1.0 Release September 2019

Unterstützte Features 1.0 Release

Read/Write von Attributen

Monitoring von Wertänderungen (DataChange)

Monitoring von Events (Client- und Serverseite)

Methodenaufrufe

Browse

Auflösen von Browse-Pfaden

Anlegen und Löschen von Nodes und Referenzen

Sichere Verbindungen (Server- und Clientseite)

Publish & Subscribe (OPC UA Part 14)

Local Discovery

...

Unterstützte Features 1.0 Release

Liste ist eine Auswahl

- Details: <https://github.com/open62541/open62541/blob/master/FEATURES.md>
- Achtung: „Some Assembly Required“!

Zertifizierung

OPC UA Part 7 definiert Profile

- Beschreibung anwendungsrelevanter Subsets aus dem Gesamtstandard

Ein mit open62541 1.0 entwickelter Server (server_ctt) wurde 09/2019 zertifiziert

- „... supporting OPC UA client/server communication, subscriptions, method calls and security (encryption) with the security policies 'Basic128Rsa15', 'Basic256' and 'Basic256Sha256' and the facets 'method server' and 'node management'...“
- Siehe <https://open62541.org/certified-sdk.html>

Bedeutet nicht, dass open62541 „zertifiziert“ ist

Bedeutet, dass die Zertifizierung von eigenen Produkten machbar ist

ÖKOSYSTEM



Open Source

Actives Project

- Hosted @ github
- github.com/open62541/open62541/
- Erster Commit Ende 2013

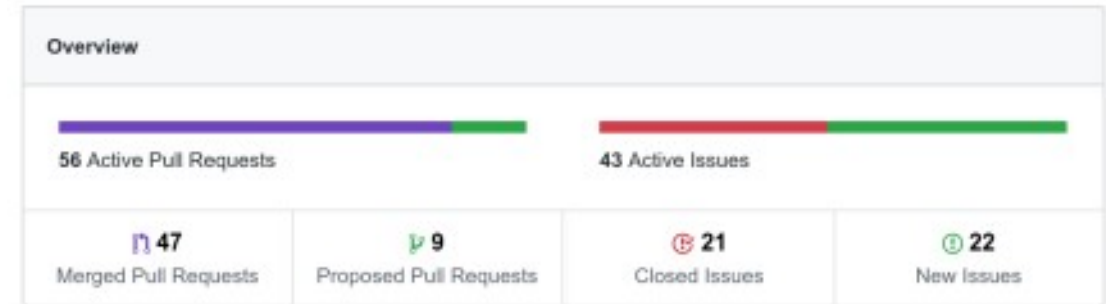
Lizenz: MPL2

- Unkompliziert in kommerziellen Projekten nutzbar

50k+ downloads of the 0.3 release

January 6, 2019 – February 6, 2019

Period: 1 month ▾



Excluding merges, **17 authors** have pushed **99 commits** to master and **113 commits** to all branches. On master, **146 files** have changed and there have been **9,343 additions** and **4,195 deletions**.



47 Pull requests merged by 14 people

- Merged #2362 Client history update 9 hours ago
- Merged #2401 history_plugin: Fix and test random index backend 9 hours ago
- Merged #2420 Fix include of client config 23 hours ago
- Merged #2422 Janitor stuff a day ago

Maintainer

Projektsteuerung

Das Gros der bisherigen Entwicklung



fortiss

Ökosystem

Contributing Companies

- Aktive Mitarbeit in den letzten zwei Jahren



Commercial Support

- Müssen im Projekt aktiv sein



ERSTELLUNG EINES SERVERS - EIN TUTORIAL



Eine erste Server-Anwendung

Aufgabe

- „Ein Server, der eine Variable enthält, die per (Remote) Methodcall inkrementiert werden kann“
- Unterstützt Browsing, Lesen/Schreiben von Werten
- Unterstützt DataChanges, Subscriptions

Nutzt Modellierung und Code-Generierung

Build-System

Nutzt selbst CMake

- Erstmal nur für das Bauen/Konfigurieren relevant
- Stellt z.B. für Code-Generierung CMake-Makros zur Verfügung

Nutzung als

- eine statische/dynamische Bibliothek
- „Amalgamation-Build“ (für erste Schritte/einfache Projekte OK)

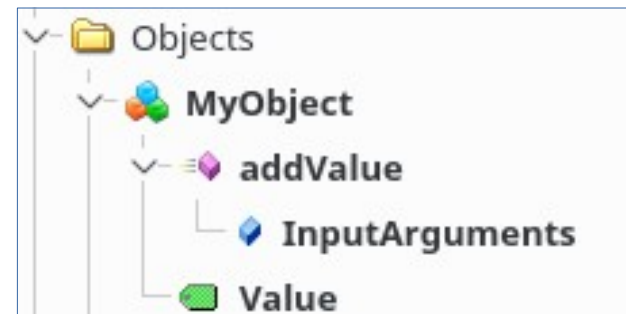
Für das Tutorial

- Kein Build-System + Amalgamation

Beispiel-Modell

Einfaches Modell

- Objekt (i=5002) mit zwei Komponenten
 - VariableNode (i=6002) mit Double-Wert
 - Methode (i=7001) addiert Parameter zum Double-Wert



z.B. mit dem UaModeler erzeugt

XML Export

- → example.xml

Code-Generierung

nodeset_compiler.py

- Teil der von open62541 zur Verfügung gestellten Werkzeuge

Commandline for our example

- `../open62541/tools/nodeset_compiler/nodeset_compiler.py --types-array=UA_TYPES --types-array=UA_TYPES --existing ../open62541/deps/ua-nodeset/Schema/Opc.Ua.NodeSet2.xml --xml example.xml namespace_example_generated`
- → `ua_namespace_example.c/.h`

Ein vollständiger Server

```
UA_StatusCode methodHandler(UA_Server *server, const UA_NodeId *sessionId, void *sessionHandle, const UA_NodeId *methodId,
                             void *methodContext, const UA_NodeId *objectId, void *objectContext, size_t inputSize,
                             const UA_Variant *input, size_t outputSize, UA_Variant *output)
{
    UA_Variant currentValue;
    UA_Server_readValue(server, variableId, &currentValue);
    double sum = *((double *)input[0].data) + *((double *)currentValue.data);
    UA_Variant result;
    UA_Variant_init(&result);
    UA_Variant_setScalar(&result, &sum, &UA_TYPES[UA_TYPES_DOUBLE]);
    return UA_Server_writeValue(server, variableId, result);
}

int main()
{
    UA_Server *server = UA_Server_new();
    UA_ServerConfig_setDefault(UA_Server_getConfig(server));

    namespace_example_generated(server);

    UA_UInt16 namespaceIndex = UA_Server_addNamespace(server, "http://yourorganisation.org/SlideExample/");
    variableId = UA_NODEID_NUMERIC(namespaceIndex, 6002);
    UA_Server_setMethodNode_callback(server, UA_NODEID_NUMERIC(namespaceIndex, 7001), methodHandler);

    bool running = true;
    return UA_Server_run(server, &running);
}
```

Compilation + Demo

Commandline

- `gcc -DUA_ENABLE_AMALGAMATION -I . -I ../open62541 demoserver.c ../open62541/open62541.c namespace_example_generated.c`

Basierend auf open62541: Qt OPC UA

**C++/Qt Modul mit dem Ziel einer einfachen OPC UA
Integration in Qt-HMIs**

- Open Source

API, kein eigener Stack

<https://doc-snapshots.qt.io/qtopcua/index.html>

<https://blog.basyskom.com/building-qt-opc-ua-with-open62541/>



ZUSAMMEN- FASSUNG



ZUSAMMENFASSUNG

open62541 ist

- Aktuell die mit Abstand aktivste Open-Source Community um einen C/C++ OPC UA Stack
- Eine gute Wahl für Embedded Usecases von OPC UA
- Für bestimmte Profile zertifizierbar
- Weiterhin stark in der Entwicklung

DANKE SCHÖN!

FRAGEN?

спасибо 谢谢
GRACIAS 谢谢
THANK YOU
ありがとうございました MERCI
DANKE धन्यवाद
شُكْرًا OBRIGADO