Whitepaper

Implementation: OPC Unified Architecture

Considerations regarding Software Development Kits

Feature rich cross platform communication in industry and beyond
→ selection criteria for software development libraries enabling OPC UA

written by:
Uwe Steinkrauss, ascolab GmbH
July 2016

Abstract

This whitepaper discusses decision criteria for choosing OPC UA development libraries, in particular the question if it's better to buy a ready-to-use OPC UA application or to implement it yourself. If devices or machines are to be equipped with OPC UA communication and brought to market as soon as possible, development and testing time are crucial. Commercial platform independent software development kits (SDKs) are examined as well as how they, together with the suitable tools, simplify the development process. In addition, migration strategies for existing implementation are shown, furthermore tools to compare and validate different products are presented.

COMMERCIAL VS OPEN SOURCE

Implementing OPC UA technology and integrating it into existing products poses a special challenge. There are three possible approaches to choose from, based on the particular requirements.

1) “Build”
OPC UA is an open standard and available free of charge for everybody. Using the OPC UA Specification or the respective IEC standard, the implementation can be completely realized by your own development department.

2) “Open Source”
The OPC Foundation provides their example implementations, which have been used to verify the Specification, as Open Source for members as well as non-members. The code is offered with a dual license on GitHub. Members of the OPC foundation receive a reciprocal community license, i.e. they don’t have to reveal their own code. Non-members, however, receive a GPLv2 license, forcing the user to publish their own source code. Beside the example code of the OPC Foundation, there are numerous other Open Source implementations by universities, institutes, and individuals, offered with different license models in a variety of programming languages.

3) “Buy”
This is the most popular option for commercial products. Software companies offer commercial libraries, so-called toolkits and software development kits (SDKs) under commercial license and support models. Typically, these libraries are regularly tested at interoperability workshops (IOP) and certificated for conformity (OPC UA Compliance). There are versions for a variety of programming languages and with different functionality (UA Profiles). Specialized SDK variants for smallest embedded systems are available, adapted for particular requirements, like limited memory and low CPU speed (miniature devices).

In the following chapters, we will look at the “buy” variant, in particular Unified Automation’s SDKs. Here, we will first have to estimate which SDK is the best for our use case. Beside pure functional criteria, architecture and extensibility for future features are crucial deciding factors.

LANGUAGE

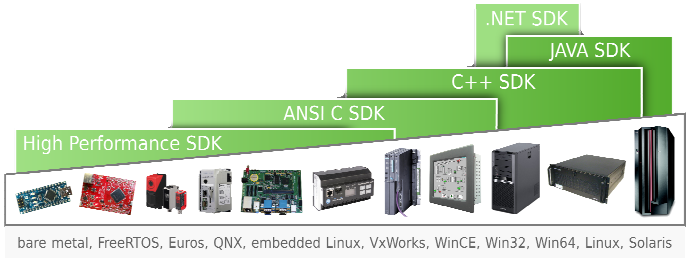
The first and probably the simplest decision to make is the programming language. Here, Unified Automation offers four SDKs in three languages: ANSI C, C++, and C# .NET.

1) High Performance ANSI C SDK
   for low performance embedded systems, e.g. I/O sensors, microcontrollers
2) ANSI C SDK
   for embedded systems with moderate performance, e.g. industrial controllers
3) C++ SDK
   for PC based systems and high performance controllers
4) C# .NET SDK
   for PC based systems running Windows

When integrating OPC UA functionality to an existing system, the already used programming language is the decisive criterion. The OPC UA library should always be used in the native language to avoid the need for copying data between components that have been developed in different programming languages.

When connecting the OPC UA server library to the data source that is to be represented, it is necessary to manage several interfaces. The so-called data provider contains the I/O interface for read and write access from and to the data source as well as the node interface for server address space administration and navigation of the client in the information model (browse).

Figure 1) Different device classes, typical operating systems and preferred languages

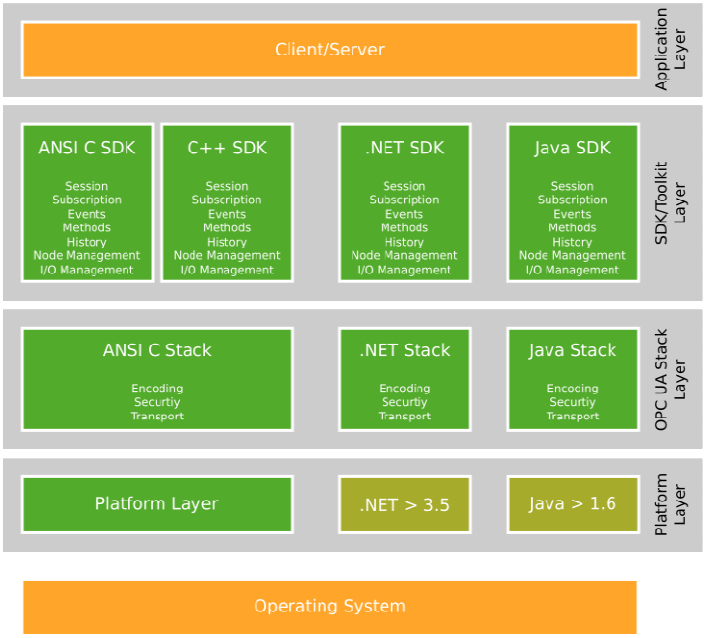


OPERATING SYSTEM

The second decision to make is choosing the target platform. Unified Automation's SDKs are, except of the .NET SDK, in principle platform independent: The code can be compiled for a variety of operating systems and CPU architectures. All platform dependent functions are abstracted using a platform layer. Only this platform layer is replaced or adapted for the target platform. The structure of all SDKs is generally similar and consists of the following components:

1) Platform layer
   contains operating system specific code
2) OPC UA Stack
   contains network implementation, encoding/decoding, and OPC UA message security
3) OPC UA SDK libraries
   provide OPC UA features like Session, Subscription, and node management

Figure 2) OPC UA SDK design principle consisting of UA Stack and SDK



OPC UA FUNCTIONALITY

The third decision is to match the required with the available OPC UA functionality. OPC UA Profiles offer the most precise description of functionalities. They combine so-called Conformance Units to

function blocks that are logically connected and consistent in themselves. In this context we can distinguish between complete Profiles and so-called Facets. Complete Profiles, like “Embedded OPC UA Server” define basic functionalities that an OPC UA Server has to offer. Facets define additional features on top, like Historical Data Access or Events.

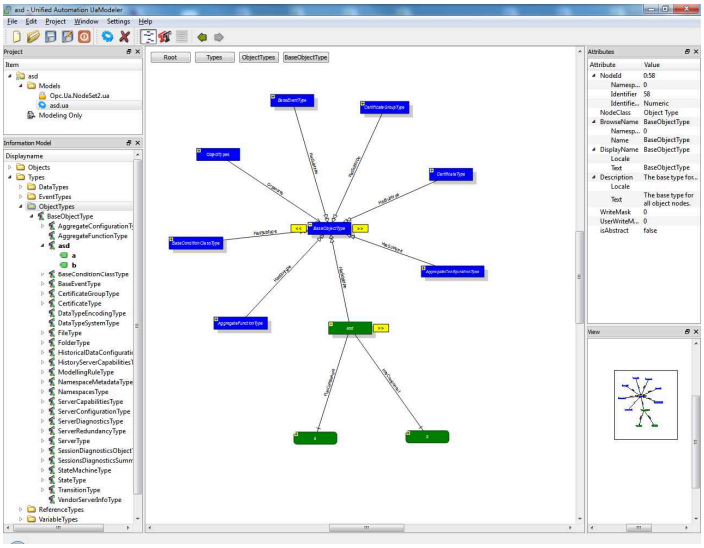
An OPC UA enabled product typically supports Profile and several Facets. It is also possible to provide additional OPC UA Features and thus Facets in further stages of development of a product. Especially if it is planned to implement a Companion Specification, i.e. a specification of another organization or industrial sector built on top of OPC UA, it is mandatory that the SDK provides the required Profiles and Facets. To realize the AutoID information model for RFID or barcode scanners for instance, in addition to the basic server profile OPC UA Methods, Events, and Structured DataTypes have to be supported. Thus, a pure “Embedded OPC UA Server Profile” with Data Access functionality is not sufficient to implement this device model.

USABILITY

A fourth decision criterion is usability and thus the time needed for implementation. Here, the design of the user interface is crucial. The developer or integrator wants an easy and consistent interface for being able to connect their application data quickly and error-free. Helper functions and encapsulation of recurring tasks are welcome as well.

Linking the data is typically not a big challenge. The toolkit interfaces of Unified Automation greatly simplify the access to data and usually broken down to a simple Read/Write. Building and implementing the address space can be much more complex, especially if a given information model has to be reproduced. For this reason, Unified Automation was the first vendor to offer its libraries with a modeling tool containing an integrated code generator. With this tool, it is possible to load standardized information models, create own information models, or extend standard models with product specific information. For this model, or several models built on top of each other, one can generate source code that can be directly used in the SDK. This massively speeds up implementation and considerably improves code quality. Moreover, first prototypes for validation can be created without much effort. Even the OPC Foundation working groups use UaModeler©, Unified Automation’s modeling tool, to quickly and effectively create and implement information models for Companion Specifications.

Figure 3) Screenshot UaModeler for generating information models and source code



INTERFACE DESIGN

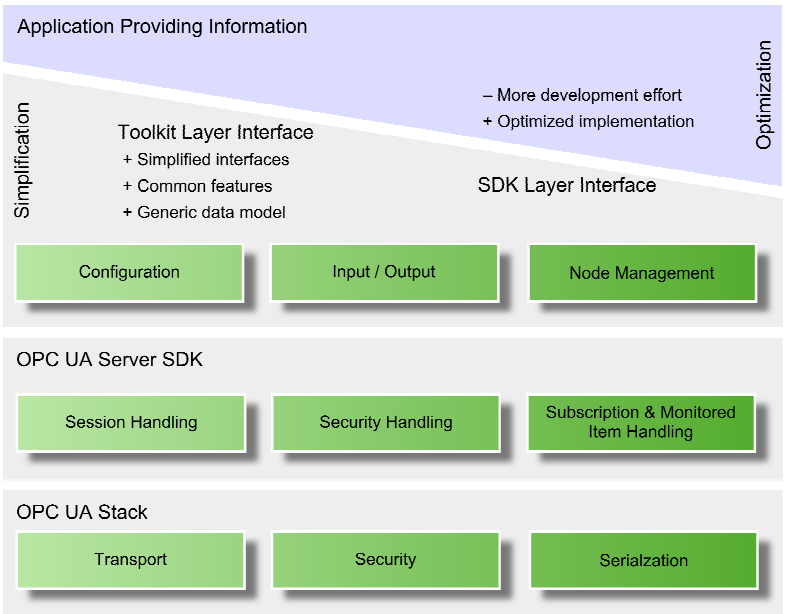
The fifth decision refers to the flexibility of the SDK. Besides a highly simplified interface for rapid development, a good SDK should also offer an “expert interface” with extended possibilities for application-specific optimization. Highly simplified interfaces often give little scope to perfectly master specific applications. If, for instance, an extremely large quantity of objects and variables are to be provided by an OPC UA server, optimized access can be realized using a deeper interface level in a customizable way.

Thus, Unified Automation's SDKs always offer two interface levels. The so-called toolkit layer allows to link OPC UA functionalities with highly simplified interfaces. This enables quick development of the desired OPC UA features for most application cases.

When required, parts of the data connection can be realized over the second layer, the SDK interface. Here, the developer has direct access to the functions that are used to provide application data to OPC UA clients. Using this interface requires expert knowledge of OPC UA, but offers maximum flexibility. The SDK allows to switch between these layers for every single functionality.

Even if this flexibility isn't used in a first implementation, it is nevertheless important and may become necessary in further development steps. Then single features can be further optimized without having to re-implement the complete OPC UA connectivity or even to replace the OPC UA library.

Figure 4) Flexibility of the SDK by providing several interface layers



EXTENSIONS

The sixth and last criterium for choosing the right SDK is the possibility to expand in respect of external libraries and features. Here, the additional benefit added with further libraries is in the focus. The final product can thus be extended with additional functionalities. With so-called add-ons, for instance redundancy between OPC UA servers may be realized. The add-on module synchronizes the necessary information between redundant servers. Further add-ons for connectivity to miscellaneous databases can extend an OPC UA server and transform it into a data logger. The nodes chosen for being archived can be provided again using the OPC UA functionality Historical Access

(HA). It is also possible to further extend security features of the server via add-ons. Typically, OPC UA servers file their certificates and trust lists in the file system of the device. The security of this file storage can be significantly improved when private keys are relocated to specialized hardware chips (secure elements). These specialized chips are able to compute cryptographic algorithms that require considerable processing power. Thus, the load of the device's main CPU is lowered and a stronger encryption is possible. In addition, a stronger entropy source could be connected to the device. Those chips are, wherever possible, integrated using standard interface like TPM (Trusted Platform Module). Unified Automation provides a partner program enabling other companies to offer such additional benefit. Wibu-Systems AG for instance offers connectivity of the OPC UA SDK to their CodeMeter technology, thus creating additional benefit for OPC UA devices. Further add-ons provide protocol extensions like OPC UA Pub/Sub with AMQP or MQTT for direct cloud integration on the one hand and the possibility for data transfer over TSN (Time Sensitive Network) on the other hand. Regardless of that, further helper modules (utilities) to persist information models or convert into efficient data formats are available.

PERFORMANCE AND RESOURCES

If the resources of the target system are heavily limited, it is necessary to use a specialized SDK. Although OPC UA is highly scalable by using profiles, reduced OPC UA functionality is often not desired, especially if nothing is left of the features that make up OPC UA, like security, monitoring or a detailed information model. To shift the field of application for OPC UA technology further to smallest devices and sensors/actuators, Unified Automation developed a specially optimized SDK. The High Performance SDK has been implemented based on a completely new architecture and consequently considering memory usage and performance. From the network layer up to the application layer, everything, including the functions of the UA Stack, has been optimized for the application in smallest devices. The limitations of implementations based on the OPC UA stack of the OPC Foundation have thus been outweighed. This SDK reduces the already low requirements of the ANSI C SDK by half, whereas the speed (data throughput) is doubled. Now, with this SDK, fully functional UA server on ARM M4 processors with less than 200kB RAM are possible. The new High Performance SDK is not only attractive for smallest devices. The low resource consumption enables new applications for high-performing devices. For example, several 1000 connections can be managed or huge address spaces with more than 10 million nodes can be configured, like it is required in large cloud or big data applications.

MIRGATION OF EXISTING EQUIPMENT

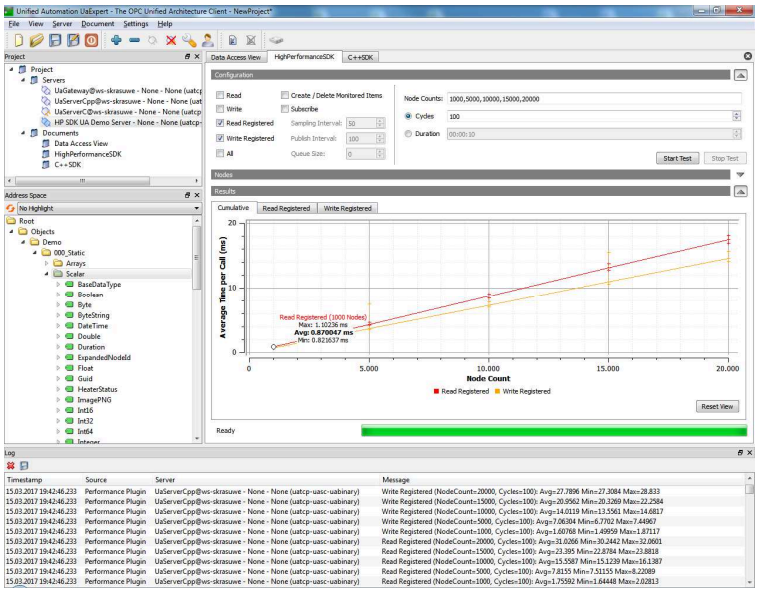
There are diverse possibilities to connect existing plants or products. Although the scope and possibilities of old “classic”, on Microsoft's COM technology based, OPC interfaces do reach nowhere near OPC UA, they are widely spread and frequently used. It is possible to connect those legacy systems via software gateways. The scope reaches from simple “transparent” gateways up to the enrichment with OPC UA specific information. Unified Automation's UaGateway (c) is such a transparent converter. The Windows software is installed directly on the PC where the “old” COM OPC Data Access server is running. The UaGateway opens a local connection to the “classic” OPC server and can now for its part provide OPC UA data over the network. In doing so, OPC UA's security features are made available for the network communication and transmitted data can be signed and encrypted. The often complicated DCOM configuration is no longer necessary, because the UaGateway only keeps a local connection to the COM DA server. Similarly, tunneling solutions over secure OPC UA connections can be established by utilizing tunneling solutions on both sides. Thus, existing plants can be migrated to OPC UA step by step.

TESTING AND PERFORMANCE

To compare and test the functionality of OPC UA servers, Unified Automation GmbH offers UaExpert, a free UA test client, that has evolved into the reference client for developers as well as commissioning engineers and end users. UaExpert is a Qt application developed using the company's own C++ SDK and available for Windows or Linux. During commissioning, the connection to the server is tested and all diagnostic values specified by the OPC Foundation, e.g. the number of currently active sessions or the total number of subscriptions created since server start-up, are shown in the Server Diagnostics View.

Especially when comparing different SDKs or server implementations, the performance measurement is very handy. UaExpert performs a measurement series of UA operations (eg. cyclic UA Read) and thereby variegates the number of nodes/variables used during the Read service call. The result is displayed in a graph, the average value as well as every single measurement. For the example shown below, 1000 data points (of type UInt32) of previously registered NodeIds have been read cyclically 100 times and averaged. The average round trip time of the High Performance SDK (demo server) is 0,87 milliseconds. The values of 20.000 nodes are delivered in an average time of 17ms.

Figure 5) Screenshot of a performance measurement on the High Performance server with 1000 to 20000 tags per operation using the UaExpert client



SUMMARY

Implementing professional, industrial grade software based on OPC UA technology, requires the usage of commercial SDK/Toolkits like those from Unified Automation, because such SDKs are rich in functionality, proved by thousands of implementations and the developer gets, besides professional support and documentation, additional help from sophisticated tooling. The speed and quality of the implementation is raised to a new level which can not be reached with other toolkits nor with the Open Source implementations as available today.