OPC UA vs. MTConnect – Sicherheit von Standards für Industrie 4.0

David Fuhr

HiSolutions AG
fuhr@hisolutions.com

Zusammenfassung

Die Kommunikationsstandards OPC UA und MTConnect kämpfen um die Vorherrschaft in der Automatisierung des *Internet of Things*. Während ersterer auf Webservices und eine verschachtelte Baumstruktur komplexer Sicherheitsstandards aufbaut, setzt letzterer auf simplere Webtechnologien und einige wenige Standardprotokolle wie HTTP und XML, ohne auf Sicherheitsanforderungen explizit einzugehen. Der Kampf SOAP gegen REST, den im Bereich der über den Hoheitsbereich einer Organisation hinaus verteilten Anwendungen die serviceorientierten Ansätze im Wesentlichen verloren haben, wird hier also noch einmal neu aufgeführt. Beide Ansätze haben aus Sicherheitssicht ihre Vorzüge, bedingen jedoch aufgrund der sehr unterschiedlichen implizierten Sicherheitsmodelle grundsätzlich andere Grundannahmen und eine höchst unterschiedliche sicherheitstechnische und prozessuale Einbettung durch Entwickler/Hersteller, Integratoren und Betreiber. In beiden Fällen verbleiben bei den genannten Gruppen eine hohe Verantwortung und ein hoher Bedarf an Security-Fachwissen.

1 Protokolle für die Automatisierung

Aktuell tobt ein Kampf in der Automatisierungstechnik. Nachdem sich seit den Achtzigerjahren die ursprüngliche Vielfalt proprietärer Kommunikationsprotokolle ohne jegliche Sicherheitsfunktionen vielerorts langsam aber stetig gelichtet hat in Richtung auf wenige, meist TCP(oder UDP)/IP-basierte Varianten wie Profinet, EtherNet/IP oder Modbus TCP, ist seit etwa zehn Jahren eine weitere Konzentrierung zu beobachten. Im Bereich der cyberphysischen Systeme und ihrer europäischen („Industrie 4.0“) bzw. angelsächsischen („*Industrial Internet*“) Geschmacksrichtungen schicken sich vor allem zwei Kandidaten an, um die Vorherrschaft im *Internet of Things* (IoT) außerhalb der Spielzeuge und Gadgets zu buhlen: OPC UA und MTConnect. Die zwei Bewerber kommen hierbei technisch wie kulturell und industriepolitisch aus sehr unterschiedlichen Traditionen und bringen daher gerade auch aus Sicherheitssicht höchst verschiedene Eigenschaften, Probleme und Grenzen mit. Der vorliegende Beitrag versucht nach einer kurzen Darstellung der zwei Standards, die relevanten Unterschiede bezüglich Gefährdungen, intrinsischen Maßnahmen und Restrisiken darzustellen und zu bewerten. Abschließend werden Folgerungen für die Absicherung von Industrie 4.0 vorgenommen und ein Ausblick auf zukünftige Entwicklungen gegeben.

2 OPC UA vs. MTConnect

Während OPC UA, welches stark von einer Reihe vor allem europäischer Hersteller verfochten wird, aus der Tradition der Webservices kommt und zunächst auf komplexe XML-Funktionen

P. Schartner · K. Lemke-Rust · M. Ullmann (Hrsg.) · D•A•CH Security 2015 · syssec (2015) 409-416.

gepaart mit mächtigen Sicherheitsstandards setzt, beruft sich das stärker im angelsächsischen Raum verwurzelte und kulturell in der hemdsärmeligeren Web-/Tech-Community verortete MTConnect auf klassische Webarchitekturen wie HTTP, XML-Dateien und REST. Zwar haben die beiden *Sponsoring Organizations* MTConnect Institute und OPC Foundation im Jahr 2013 eine Zusammenarbeit vereinbart, um ihren Nutzern in einem gewissen Rahmen Cross-Kompatibilität bieten zu können ([MTOF13]); trotzdem verfolgen die beiden Standardfamilien grundsätzlich unterschiedliche Architekturansätze nicht nur auf der Ebene der vorgeschlagenen Technik und Protokolle, sondern auch auf der Metaebene der Standardentwicklung an sich.

Im Folgenden werden daher die Grundzüge der beiden Familien zunächst getrennt dargestellt und danach aus Sicherheitssicht verglichen. Schließlich wird versucht, aus den Ergebnissen des Vergleichs wichtige Rückschlüsse auf sinnvolle Methoden einer soliden und sicheren Anwendung für Industrie 4.0 zu ziehen.

2.1 OPC UA

Das Kommunikationsprotokoll OPC UA ([OPCF12]) bietet mit seinem auf offenen Standards basierenden Sicherheitsmodell eine solide Basis für moderne Automatisierung mittels Webservices. Gleichzeitig bringt die technische Neuorientierung eine Reihe nichttrivialer Sicherheitsproblematiken mit sich, insbesondere durch von SOA geerbte Bedrohungen und die baumartige Abhängigkeit von Standards ([FUHR15]).

OPC UA – „Unified Architecture“, wobei OPC für „OLE for Process Control” steht und OLE wiederum ursprünglich für das Windows-spezifische “Object Linking and Embedding” – ist als Datenaustauschtechnik im Bereich industrieller Automatisierung stark im Kommen. Zwar wurden erste Teile der Spezifikation bereits seit 2003 entwickelt und 2006 veröffentlicht, jedoch hat sich ein über die Nische hinausgehender Markt von *industrial grade* OPC UA Clients und Servern erst in den letzten Jahren herausgebildet. Neuerdings ist gerade im Zuge von Industrie 4.0 und der sprunghaft gestiegenen Beschäftigung mit Cybersicherheit ein Boom zu verzeichnen. Anders als das thematisch verwandte Paradigma „SOA“ aus der Enterprisewelt scheint OPC UA also nicht den Weg in die Versenkung des Hype Cycles zu nehmen. Das hat gute Gründe: Neben einer besseren Skalierbarkeit und größeren Systemunabhängigkeit kann OPC UA mit offenen Standards und Schnittstellen sowie der Nutzung von über anderthalb Dekaden lang entwickelten Sicherheitstechnologien wie XML-Verschlüsselung und Security-Tokens punkten.

Die folgende Abbildung zeigt die Struktur des Multi-Part-Standards OPC UA in Version 1.02 sowie diejenigen Teildokumente, die für die Security Relevantes enthalten:

Dabei ist zu beachten, dass Part 12 *Discovery* in Version 1.02 des Standards noch nicht enthalten war. Hier liegt nun ein erster Release Candidate im Rahmen der Version 1.03 vor.

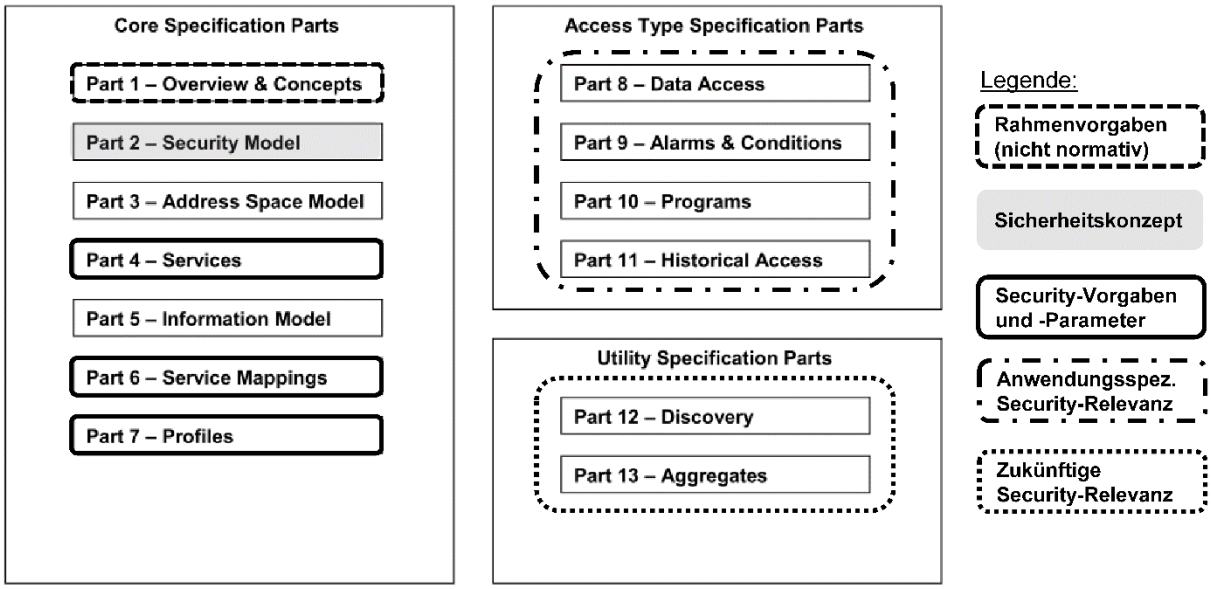

Abb. 1: Security in OPC UA

2.2 MTConnect

MTConnect ([MTCO14]), welches vom gleichnamigen Institut herausgegeben wird, geht einen anderen Weg. Zu einer ähnlichen Zeit entstanden wie OPC UA, werden zwar auch hier XML-Nachrichten verwendet, allerdings lediglich für die Formulierung der Antworten. Für Transport wie sonstige „Sicherheitsfunktionen“ (im engeren Sinn werden solche gar nicht benannt) kommen wesentlich simplere Standardprotokolle zum Einsatz. So besteht eine Anfrage lediglich aus einem HTTP GET-Request im REST-Stil – also mit „sprechender“ URL und zustandslosem Server –, wo OPC UA eine Vielzahl an Kombinationsmöglichkeiten aus Encoding (XML oder binär), Transport (SOAP auf HTTP/HTTPS oder ein spezielles „UA TCP“) und Verschlüsselung (WS-SecureConversation oder das handgemachte UA SecureConversation) bietet.

Ein typischer MTConnect-Request ist etwa ein HTTP GET auf

> http://10.0.1.23:3000/mill-1/sample?path=//Axes//DataItem[@type=”POSITION” and 928 @subType=”ACTUAL”]&from=50&count=100

(vgl. [MTCO14, Part 1]).

Zur Absicherung von MTConnect ist also „lediglich“ die Absicherung des Kommunikationskanals – etwa per HTTPS – möglich und ggf. notwendig, aber der Standard selbst legt hierauf noch nicht einmal einen besonderen Fokus.

Auf der anderen Seite ist laut Standardtext überhaupt nur lesender Zugriff vorgesehen, sodass bestimmte Angriffsarten grundsätzlich ins Leere laufen sollten – solange eine sichere Implementierung an dieser Stelle vorausgesetzt werden kann und die Hersteller nicht eigenständig zusätzliche, nicht spezifizierte Funktionen hinzufügen.

Da wie bereits beschrieben keinerlei Sicherheitsmerkmale explizit genannt werden, beschränkt sich die Sicherheitsrelevanz nichtfunktionaler Parameter im Wesentlichen auf wenige implizite Bemerkungen und mögliche Rückschlüsse, insbesondere in Teil 1 *Overview and Protocol*.

Die Struktur des Multi-Part-Standards stellt sich wie folgt dar (Version 1.3.0):

Tab. 1: Struktur MTConnect

Part	Titel
1	Overview and Protocol
2	Device Information Model
3	Streams, Events, Samples, and Condition
3.1	Interfaces
4	Assets
4.1	Cutting Tools

3 Risiken in den Standards

3.1 Sicherheitsmodell

In MTConnect ist Security im Sinn von Informationssicherheit kein Thema, weder ex- noch implizit. Die Dokumente enthalten keinerlei Überlegungen zu Sicherheitszielen, Gefährdungen oder Maßnahmen geschweige denn eine wie auch immer geartete Risikoanalyse. Dies war zwar bei technischen Standards bis vor einer Weile durchaus üblich, inzwischen hat sich aber die Abhandlung von *Security Considerations* zumindest in Form eines kurzen Kapitels als gute Praxis eingebürgert. MTConnect überlässt die ganze Last der Absicherung sowie der vorgelagerten Definition der Sicherheitsanforderungen den Anwendern. Leider wird dies nicht explizit herausgestellt, sodass unklar bleibt, in wessen Verantwortungsbereich diese Arbeit fällt: des Herstellers/Entwicklers, des Integrators oder aber des Betreibers.

OPC UA auf der anderen Seite stellt sich an dieser Stelle als vorbildlicher Standard dar: Nicht nur wird die Schaffung und Aufrechterhaltung der Informationssicherheit gleich zu Beginn im Standard als eines der zentralen Ziele definiert ([OPCF12, Part 1]), es wird sogar gleich ein ganzer Standardteil als Sicherheitskonzept geführt ([OPCF12, Part 2 *Security Model*]). Hier werden nicht nur mögliche Bedrohungen aufgelistet, sondern auch Sicherheitsziele wie Vertraulichkeit, Integrität, Verfügbarkeit und weitere für OPC UA definiert und in Folge beides – Sicherheitsziele wie Bedrohungen – mit den im Standard definierten Sicherheitsfunktionen (Maßnahmen) „abgestimmt“ (*Reconciliation*), ähnlich der Betrachtung von Funktionalität und Vertrauenswürdigkeit (Qualität) im Rahmen einer Evaluierung nach Common Criteria ([COMM12]) – nur dass in diesem Fall sämtliche Assets Dokumente sind.

3.2 Probleme in MTConnect

Da MTConnect selbst praktisch keine Sicherheitsanforderungen definiert und nur wenige Vorgaben macht, die eine direkte Auswirkung auf die Sicherheit haben, ist die grundsätzliche Schwäche leicht benannt: Das Sicherheitsmodell steht und fällt mit dem Grad, in dem Integritäts-, Vertraulichkeits- und Verfügbarkeitsbedarf des jeweiligen Anwendungsfalls durch zusätzliche Sicherheitsmaßnahmen auf Transportebene (HTTPS, VPN) oder sonst wo (Zertifikate, Redundanz etc.) gewährleistet werden können. Allein dies stellt aufgrund der in den letzten Jahren entdeckten Schwachstellen etwa in TLS-Algorithmen, -Protokollen und -Implementierungen ein nichttriviales Problem dar.

Zudem sind die Implementierungen der Parser und Interpreter daraufhin zu überprüfen, ob diese robust gegen typische Web- und XML-Angriffe wie etwa Path Traversal oder XML eXternal Entity-Attacken (XEE) sind.

Bei OPC UA stellt sich die Sachlage ungleich komplexer dar: Da in diese Standardfamilie im Gegensatz zu MTConnect gleich eine ganze Reihe an Sicherheitsanforderungen, Zusicherungen und Security-Technologien eingebaut wurden, liegt zum einen der Maßstab des Versprechens höher, an dem OPC UA gemessen werden muss – zumal hier auch Schreibzugriffe vorgesehen und üblich sind. Zum anderen bergen die vielen komplexen Substandards, auf welchen OPC UA aufbaut, neben Chancen für mehr Sicherheit jeweils auch Gefahren der Vererbung von Schwachstellen.

3.3 Probleme in OPC UA

Das Sicherheitsmodell von OPC UA, das in bestimmten Betriebsmodi entscheidend auf Webservice-Sicherheitsstandards wie WS-Security und WS-SecureConversation aufbaut, ist offener, stabiler und leichter verifizierbar als dasjenige des Vorgängers OPC, wenn man dort von einem solchen überhaupt sprechen konnte. Gleichzeitig bringt das Rebasing auf Webservices neben unbestrittenen Vorteilen auch eine Reihe unerwünschter und zum Teil nicht allen Akteuren bewusster Gefahren mit sich.

3.3.1 Komplexe Substandards

Webservice-Sicherheitsstandards, auf denen letztendlich auch die Sicherheit von OPC UA beruht, sind komplex in mehrfacher Hinsicht: Es gibt sehr viele davon, sie bauen in baumartiger Weise aufeinander auf, wobei die Versionierung eine weitere Dimension hineinbringt, und viele sind für sich genommen schon komplex zu erfüllen bzw. auch nur auf Korrektheit zu prüfen. Allein ein XML-Parser – der grundlegende Teil eines jeden SOAP-Prozessors bzw. Webservice-Clients oder Servers – ist alles andere als trivial sicher zu implementieren. Auf höheren Protokollebenen potenzieren sich diese Schwierigkeiten im wahrsten Sinn dieses mathematischen Fachausdrucks.

3.3.2 Baumartige Abhängigkeit

Diese baumartige indirekte Abhängigkeit der Sicherheit von Substandards von Substandards führt zu einer sehr schlechten Pflegbarkeit des Standards. Wenn eine Schwachstelle in einem Basisstandard entdeckt und behoben wurde, muss nach und nach die ganze Kette der Abhängigkeiten nach oben angepasst werden, bis zuletzt OPC UA selbst auch über mehrere Zwischenstufen auf eine sichere Version etwa von XML Encryption verweist. Dies kann sich über einen Zeitraum von mehreren Jahre ziehen, während dessen der Standard angreifbar bleibt.

3.3.3 Festlegungen zur Kryptographie

Im Gegensatz zu MTConnect, welches sich aus Fragen der Absicherung der Kommunikation weitgehend heraushält und diese höchstens indirekt beantwortet durch die strikte Verwendung von absoluten Standards, welche zwar nicht unbedingt trivial, aber immerhin mit Standardmethoden abzusichern sind, wagt sich OPC UA durchaus an die Festlegung diverser kryptographischer Parameter, ohne allerdings an dieser Stelle Vollständigkeit zu erreichen.

Die genauen Vorgaben sind nicht ganz einfach aus den gut 1000 Seiten der Standardfamilie herauszulesen, da sie sich über mehrere Dokumente und Fundstellen innerhalb dieser verteilen und sich erst nach einer genauen Nachverfolgung von Querverweisen zu einem Gesamtbild zusammensetzen lassen. Es darf davon ausgegangen werden, dass dies auch den Herstellern von standardkonformen Komponenten die sichere Produktentwicklung nicht gerade erleichtert.

Doch auch wenn man alle Informationen zur Kryptographie aus dem Standardköper herausdestilliert, ergibt sich ein unbefriedigendes Bild: Neben diversen weiteren Schwachpunkten und Auslassungen, welche immer die Gefahr einer unsicheren Interpretation lassen, setzt OPC UA an bestimmten Stellen RC4 als Verschlüsselungsalgorithmus ein. Dieses symmetrische Verfahren gilt jedoch schon seit mehreren Jahren als schwach und spätestens seit Anfang 2015 als gebrochen. Zwar benötigen die relevanten Angriffe immer noch einige Zusatzannahmen wie die, dass der Angreifer Code (z. B. JavaScript) im Kontext des Nutzers ausführen kann, jedoch sind solche Szenarien gerade aufgrund der extremen Vielfalt der Einsatzmöglichkeiten von OPC UA durchaus vorstellbar. Außerdem geht die Forschung an dieser Stelle weiter, und fast monatlich werden neue Schwächungen der einschlägigen veralteten Algorithmen und Protokolle bekannt.

Das Grundproblem an dieser Stelle ist jedoch nicht die Festlegung von OPC UA auf einen Algorithmus, der inzwischen als unsicher erkannt wurde. Dies kann und wird bei Standards, welche aufeinander verweisen, immer wieder passieren, und ein funktionierender Prozess der Risikobewertung und Standardfortschreibung sollte hier etabliert sein. Die Gefahr ist vielmehr, dass OPC UA tiefgreifende Dependencies verwendet, teilweise in diese hineinregiert und teilweise dies unterlässt ohne ein vollständiges Verständnis davon, was für die Absicherung der jeweils unteren Ebenen notwendig ist.

3.4 Security Level

In Version 1.02 des Standards sind noch sogenannte Security Levels enthalten, welche implizite Labels ohne eigenen Bedeutungsinhalt darstellen und dazu dienen sollen, bestimmte Profile – also garantierte Konfigurationen von (Sicherheits-)Parametern – miteinander vergleichbar zu machen. Zu den möglichen Risiken siehe [FUHR15]. Security Labels wurden bei der aktuell in der Kommentierung befindlichen Version 1.03 entfernt.

4 Ausblick

Beide Standardfamilien entwickeln sich selbstverständlich beständig weiter. Während von OPC UA relativ verlässlich etwa alle drei Jahre eine neue Version verabschiedet wird, erfolgt die Freigabe einer neuen MTConnect-Fassung etwa alle zwei Jahre.

MTConnect wurde zuletzt im Herbst 2014 upgedatet und wird in weiteren Versionen höchst wahrscheinlich neue Features, Funktionalitäten und Komponenten bringen, aber vermutlich auch mittelfristig keine explizite Beschäftigung mit dem Thema Security.

Anders OPC UA: Hier geht Feedback aus der zunehmend interessierten Security-Community in den Entwicklungsprozess mit ein und trägt bereits in der überfälligen Erneuerung 1.03 Früchte, welche zurzeit in der Kommentierung ist. So wurden mit der Ausmusterung sämtlicher WS-*-Standards große Mengen an Komplexität und Altlasten bereits über Bord geworfen. In Diskussion ist zudem die Migration weg von TLS 1.0 und sogar TLS 1.1 hin zu reinem TLS 1.2 – gelöst wäre damit auch das Problem mit RC4. Dies betrifft allerdings momentan nur eine kleine Nutzerbasis, da fast alle verfügbaren Produkte auf „natives“ UA TCP aufsetzen. Hierfür steht eine exakte Sicherheitsanalyse noch aus.

Möglicherweise in die nächste Version werden es noch Cipher-Suites mit Forward Secrecy (PFS) schaffen, welche bis jetzt gar nicht enthalten sind. Dies würde es ermöglichen, in An-

wendungsfällen mit hohem Vertraulichkeitsbedarf (z. B. in IoT-Anwendungen mit medizinischen Daten) die nachträgliche Entschlüsselung aufgezeichneter Kommunikation durch einen Angreifer zu erschweren.

5 Folgerungen

OPC UA und MTConnect gehen gegensätzliche Wege, um die Sicherheit der Kommunikation in der Automatisierungstechnik zu gewährleisten. Während OPC UA alle Varianten der Kommunikation offen lassen möchte und die Risiken durch eine Vielzahl von Substandards und Sicherheitstechniken einzudämmen versucht, beschneidet MTConnect von Anfang an die Möglichkeiten sehr stark (nur Lesezugriff, nur REST/HTTP), um dann die Absicherung der Kommunikation komplett dem Hersteller oder sogar Betreiber zu überlassen.

Dafür macht MTConnect so gut wie keine Sicherheitsvorgaben. Im Gegenteil werden durch Beispiele im Standard eher unsichere Varianten gefördert – z. B. die konsequente Vermeidung von HTTPS. OPC UA wiederum schießt teilweise über das Ziel hinaus und regiert – inkonsistent – an einigen Stellen in Substandards hinein, nicht immer mit sicheren Parametervorgaben.

Beide grundsätzlichen Herangehensweisen bieten Risiken und Chancen, sodass ein klarer Sieger nicht feststeht. Es ist jedoch leicht nachzuweisen, dass OPC UA als Standard in dieser Form sehr schwer unter Aufrechterhaltung aller Sicherheitsanforderungen zu pflegen ist und sich vermutlich als zu schwerfällig erweisen dürfte. MTConnect hingegen fehlen in der vorliegenden Form noch leichtgewichtige Module, die für eine sichere Verschlüsselung der Kommunikation herangezogen werden können, sodass die Sicherheit der Gesamtinstallation nicht allein von Wissen und Fähigkeiten des jeweiligen Betreibers abzuhängen braucht.

Die in der Einleitung angedeutete Kooperation der beiden Welten könnte daher theoretisch in Richtung einer sinnvollen Allianz weisen. Allerdings deuten sowohl die Tatsache, dass die MTConnect-OPC UA Companion Specification von 2013 ([MTOF13]) bis heute den Status „Release Candidate“ trägt als auch das Fehlen nennenswerter Mengen hybrider Produkte in der Praxis auf politische Absichtserklärungen, welchen andere Realitäten und wirtschaftliche Interessen gegenüberstehen.

Die Grundsatzentscheidung steht also noch aus, auf welcher Basis das *Internet of Everything* zukünftig laufen soll. Sicherheitstechnisch könnten und sollten beide Standards viel voneinander lernen. Entscheiden über die Verbreitung wird jedoch letztendlich der Markt, nicht zuletzt auch im Consumer-Bereich. Aufgabe der Anwender wie der Security-Community gleichermaßen bleibt es, sich an der Entwicklung zu beteiligen und Sicherheit auch und gerade in technischen Standards aktiv einzufordern.

Literatur

[COMM12] Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 4, 2012, http://www.commoncriteriaportal.org/cc/.

[FUHR15] D. Fuhr: OPC is Dead – Let's Hide it in the Cloud. Risiken der Serviceorientierten Automatisierung, in: BSI, IT-Sicherheitskongress 2015, Kongressband, Bonn 2015, Folien zum Beitrag: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Veranstaltungen/ITSiKongress/14ter/Vortraege-20-05-2015/David_Fuhr.pdf.

[MTCO14] MTConnect Institute: MTConnect Standard, Version 1.3.0, Parts 1-4, 2014, http://mtconnect.org/.

[MTOF13] MTConnect Institute und OPC Foundation: MTConnect-OPC UA Companion Specification, 2013, http://mtconnect.org/.

[OPCF12] OPC Foundation: OPC Unified Architecture Specification, Version 1.02, Parts 1-11 + 13, 2012, https://opcfoundation.org/developer-tools/specifications-unified-architecture.