Bei diesem Beitrag handelt es sich um einen wissenschaftlich begutachteten und freigegebenen Fachaufsatz („reviewed paper“).

doi.org/10.37544/1436–4980–2024–05–76

Analyse des OPC Foundation Whitepaper zur Nutzung in Modellierungstools und für Testfälle

OPC UA Best Practices in der Modellierung nutzen

T. Heinemann, T. König, A. Lechler, O. Riedel

ZUSAMMENFASSUNG Mit der Popularität von Open Platform Communications Unified Architecture (OPC UA) und Companion Specifications (CS) wird ein allgemeines Verständnis von OPC-UA-Modellen wichtiger. Dazu hat die OPC Foundation ein Best Practices Whitepaper verfasst, um die Modellierenden zu unterstützen. Dieser Beitrag beschreibt, welche Inhalte in verschiedener Weise zur Nutzung und Überprüfung von OPC-UA-Modellen und Implementierungen geeignet sind und inwiefern diese Nutzung automatisiert erfolgen kann.

STICHWÖRTER

Forschung, Normen/Richtlinien, Software

Using OPC UA Best Practices in Modelling

ABSTRACT With the rising popularity of Open Platform Communications Unified Architecture (OPC UA) and Companion Specifications (CS), a universal understanding of OPC UA models becomes increasingly important. Thus, the OPC Foundation published a Best Practices Whitepaper to support modellers. This paper describes which of its contents can be used to implement and validate OPC UA models and applications. It is also considered whether this usage can be automated by tools.

1 Einleitung

Open Platform Communications Unified Architecture (OPC UA) ist ein offener und herstellerunabhängiger Kommunikationsstandard, der sich im letzten Jahrzehnt zunehmend im Maschinen- und Anlagenbau etabliert hat [1]. OPC UA definiert zum einen die Kommunikationsmechanismen, die auf etablierten Protokollen des Internet Protocol aufbauen und diese in den oberen Schichten des ISO/OSI-Modells (Open Systems Interconnection model) erweitern. Zum anderen werden die Nachrichteninhalte über ein Informationsmodell spezifiziert [2].

Ein wesentliches Ziel der OPC Foundation und des VDMA ist es, die semantische Interoperabilität bei der Kommunikation zwischen und mit Maschinen zu stärken. Dies soll unter anderem durch die Standardisierung von Base- und Companion-Spezifikationen erreicht werden [3]. Darüber hinaus können hersteller- oder kundenspezifische Erweiterungen definiert werden.

OPC-UA-Modelle werden von verschiedenen Teilnehmern erstellt, erweitert und implementiert. Um Interoperabilität zu erreichen, müssen Missverständnisse zwischen diesen Teilnehmern ausgeräumt werden. Die OPC Foundation hat hierzu ein Whitepaper zu Best Practices in der Modellierung veröffentlicht [4]. Die Inhalte des Whitepapers direkt in Vorschläge zur Modellierung oder zur Erstellung von Modelltests zu integrieren, unterstützt die breite Nutzung dieser Best Practices. Letztlich wird so die Nutzung ähnlicher Modellierungsansätze durch verschiedene, untereinander anonyme Akteure in vergleichbaren Situationen vereinfacht.

2 OPC-UA-Modellierung, Spezifikation und Testfälle

Open Platforms Communications Unified Architecture (OPC UA) besteht seit 2008 als Standard [2]. Dieser Standard legt die technischen Inhalte für die Datenübertragung, wie Protokolle, Serialisierung und Sicherheitskonzepte fest. Um die Inhalte für die Übertragung abzubilden, bietet IEC 62541 Regeln für die Erstellung von Informationsmodellen. Der Standard umfasst eine Beschreibung der Services, welche es erlaubt, auf diese Modelle zuzugreifen und mit ihnen zu interagieren. Zudem beschreibt OPC UA die wesentlichen Modellelemente, auf die spezialisiertere Modelle aufbauen können [2].

Allgemeine Modelle wie etwa ein Modell für die generische Darstellung von kommunizierenden Geräten oder einzelnen Elementen für Automatisierungsgeräte werden ebenfalls von der OPC Foundation erstellt [5, 6]. Modelle, die branchenspezifische Informationen verallgemeinern, werden außerdem von einem Branchenverband oder einer bestehenden Nutzerorganisation erstellt und über die OPC Foundation veröffentlicht. Solche Modelle heißen Companion Specifications (CS) [7]. CS bestehen aus einer formalen Modellbeschreibung. Die in dieser Modellbeschreibung genutzten Modellelemente werden in OPC UA als Nodes bezeichnet. Das für die formalen Modellbeschreibungen definierte Austauschformat „NodeSetXML“ ist von der OPC Foundation vorgegeben [2]. Zusätzlich werden die Elemente in einem Dokument mit textuellen Beschreibungen erläutert. Auch Informationen zur Nutzung und zum Verhalten des Modells werden in diesem Textdokument transportiert. Um Implementierun-

https://doi.org/10.37544/1436-4980-2024-05-76 https://www.inlibra.com/de/page/agb - Open Access - CC BY

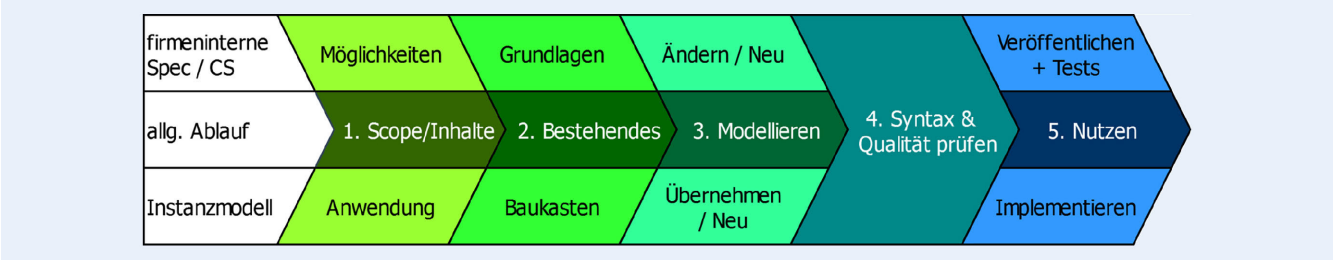

Bild 1. Ablauf der Erstellung von OPC-UA-Modellen für Spezifikationen (oben), allgemein (mittig) und Instanzmodelle (unten). *Grafik: Universität Stuttgart, ISW*

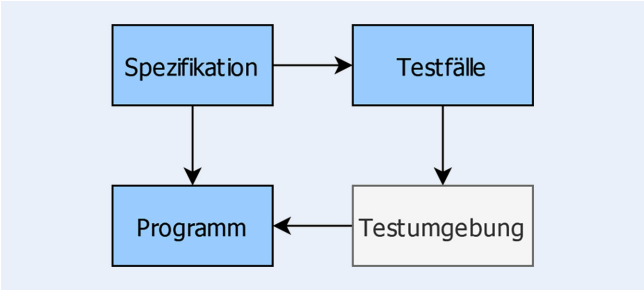

Bild 2. Zusammenhang zwischen Spezifikation, Testfällen und implementiertem OPC-UA-Programm. *Grafik: Universität Stuttgart, ISW*

gen einer CS auf Übereinstimmung mit diesen Dokumenten zu prüfen, werden Conformance-Testfälle erstellt [8].

Für Modelle mit geringerer Allgemeingültigkeit, zum Beispiel firmeninterne Vorgaben, ist eine Veröffentlichung über die OPC Foundation nicht notwendig. Solche Modelle können weiterhin über NodeSetXML ausgetauscht werden. Eine Beschreibung der Inhalte ist hier als Dokumentation sinnvoll, ebenso die Erstellung von Testfällen. Im Gegensatz zu veröffentlichten CS ist eine Einhaltung der Richtlinien der OPC Foundation in Bezug auf Beschreibungsdokument und Testfälle nicht notwendig.

Die spezifischste Form von Modellen sind diejenigen, die Instanzen beschreiben. Während CS und mögliche firmeninterne Vorgaben Typen von möglichen Geräten/Anlagen/Softwareanwendungen etc. beschreiben, legen Instanzmodelle die Daten fest, die von genau einem Gerät/einer Anlage/einer Softwareanwendung transportiert werden.

Die Standardisierung bedeutet also die Erstellung von OPC-UA-Informationsmodellen für die CS und zugehörigen Testfällen. Um in der Anwendung Flexibilität zu bieten, enthalten die standardisierten Informationsmodelle oft optionale Anteile. Deren Konkretisierung und gegebenenfalls eine Aggregation mehrerer Standard-Informationsmodelle erfolgt in einem produktspezifischen Informationsmodell, dem Instanzmodell.

In beiden Fällen erfolgt diese Modellerstellung nach demselben Muster (**Bild 1**), unterscheidet sich aber in Details.

Zunächst wird festgelegt, welche Inhalte im Modell abgebildet werden sollen. Dabei wird für die Erstellung von Spezifikationen betrachtet, welche Möglichkeiten die Spezifikation bieten soll. Für Instanzmodelle fokussiert sich der Inhalt auf die konkrete Anwendung. Der Blick auf eine Integration bestehender Modelle ist ebenfalls verschieden: Spezifikationen nutzen Grundlagen und erweitern diese. Instanzmodelle nutzen eine oder mehrere Spezifikationen vergleichbar mit einem Baukasten. Für neue Spezifikationsmodelle muss immer modelliert werden: Hier werden zum Teil die bestehenden Modelle passend erweitert und ansonsten Modellteile erstellt. Das ist auch für Instanzmodelle möglich: Hier können zudem die Inhalte von bestehenden Spezifikationen übernommen und konkretisiert werden. Für jedes Modell muss die Syntax korrekt eingehalten sein, und für die Verständlichkeit und Interoperabilität muss eine Mindestqualität erfüllt sein. Deren Prüfung unterscheidet sich für verschieden genutzte Modelle nicht wesentlich. Spezifikationen werden „genutzt“, indem sie für die geplante Zielgruppe zur Verfügung gestellt werden. Instanzmodelle werden genutzt, indem OPC-UA-Softwareanwendungen implementiert werden, welche dieses Modell abbilden. [9]

Die parallel zur Spezifikation erstellten Testfälle haben den Zweck sicherzustellen, dass ein nach Spezifikation erstelltes Programm dieser Spezifikation auch tatsächlich entspricht. **Bild 2** verdeutlicht diesen Zusammenhang: Programm und Testfälle werden basierend auf der Spezifikation erstellt. Die Testfälle liegen als Anweisungen oder als Programmcode für eine automatisierte Testumgebung vor. Durch Tester oder automatisiert mittels dieser Testumgebung werden die Testfälle ausgeführt, um das Programm zu testen. [8]

In erster Linie befassen sich typische Testfälle für CS mit einer fehlerfreien Wiedergabe der Modellinhalte, wie sie im NodeSetXML definiert sind [10]. Um sicherzustellen, dass auch das Modellverhalten korrekt abgebildet wird und geplante Anwendungen mit der Schnittstelle ermöglicht werden, lohnt sich die Erstellung weiterer Testfälle, welche verhaltensrelevante Aspekte berücksichtigen. Dabei werden alle Testfälle als Anweisungen mit erwartetem Ergebnis formuliert. Ihre Durchführung erfolgt je nach Möglichkeit automatisiert mit einem Testprogramm oder durch manuelle Tester, die Zustände auslösen, welche nicht per Software übernommen werden können [8]. In welchem Umfang Testfälle bereitgestellt werden, hängt von der Arbeitsgruppe ab, welche die CS erstellt.

Im Zuge der steigenden Anzahl an CS hat die interdisziplinäre Harmonization Working Group in der OPC Foundation ein Best Practices Whitepaper als Modellierungshilfe erstellt [11, 4]. Hier werden verschiedene Regeln vorgestellt, die in der Modellierung zu beachten sind. Auch werden Lösungsvorschläge für bestimmte Modellierungsprobleme gegeben. Ziel ist es, dass CS möglichst allgemein verständlich und untereinander ähnlich erstellt werden.

3 Nutzung von Best Practices in Modellierungstools vereinfachen

Im Erstellungsprozess für Informationsmodelle sind letztlich unterschiedliche Akteure involviert: die Ersteller der CS, die Ersteller firmeninterner Spezifikationen und die Ersteller der finalen Instanzmodelle. Mehrere Modelle können verknüpft und verbunden sein. Durch mehrere Teilnehmende werden verschiedene Sichtweisen und Stile eingebracht, dadurch unterscheiden sich die Spezifikationen in ihrer Form [7]. Einheitliche Grundprinzipien

https://doi.org/10.37544/1436-4980-2024-05-76 https://www.inlibra.com/de/page/agb - Open Access - (cc) BY

helfen, übergreifendes Verständnis zu fördern. Dazu wird im Forschungsprojekt „Clean OPC UA Information Modeling“ (CLOU) ein Hinweis auf übliche Best Practices bereits im Modellierungstool angestrebt [9].

Zur Erstellung der Testfälle dient in erster Linie das formale Modell als Basis, im Weiteren auch das in der Spezifikation beschriebene Modellverhalten. In einem zweiten Forschungsprojekt wird untersucht, wie die Testfallerstellung automatisiert unterstützt werden kann [12]. In diesem Beitrag wird zusätzlich zur Nutzung der Best Practices in einer Entwicklungsumgebung für OPC-UA-Informationsmodelle untersucht, ob bisher unberücksichtigte Testfälle aus Best Practices abgeleitet werden können.

4 Analyse und Vorbereitung der Tool-Regeln

Als Quelle für Best Practices in OPC UA wird das Best Practices Whitepaper der OPC Foundation genutzt [4]. Es wird sowohl für die Nutzung in einer Entwicklungsumgebung unter den Aspekten Linting und Metrik sowie bezüglich der Nutzung zur Erstellung von Testfällen aus Informationsmodellen analysiert. Dabei wird für die einzelnen Regeln und Hinweise neben der Relevanz für den Anwendungsfall auch betrachtet, ob die Anwendung automatisiert stattfinden kann.

Für die Modellerstellung wird zunächst geprüft, ob die einzelnen Regeln des Whitepaper für die Modellierung relevant sind und aus dem NodeSetXML ohne weitere Informationen geprüft werden können. Der Einbezug des Spezifikationsdokuments, das tiefer gehende Beschreibungen enthalten kann, wird im ersten Schritt nicht betrachtet. Ein weiterer zu beachtender Punkt ist die automatisierbare Prüfung und Ausgabe von Verbesserungshinweisen für das Linting. Für die Ableitung einer Modellbewertung wird der Einfluss beziehungsweise die Tragweite eines Regelverstoßes für die weitere Verwendung betrachtet.

Um die Testfallerstellung zu unterstützen, werden ebenfalls passende Regeln aus dem Whitepaper zusammengetragen. Es wird vermerkt, welche Inhalte genau zur Testfallerstellung aus dem XML benötigt werden. Auch wird vermerkt, welche Inhalte zusätzlich benötigt werden. Beispielhafte Testfallbeschreibungen werden skizziert. Mit diesen Vorlagen beziehungsweise Beispielen können Ersteller von CS analoge Testfälle für ihr Modell definieren.

5 Aus dem Whitepaper extrahierte Regeln

Für die Modellerstellung gibt das Kapitel „Naming Conventions“ eindeutige Regeln für die Benennung der Modellelemente vor. Diese Regeln umfassen einerseits Anforderungen, welche zwingend erfüllt sein müssen, wie die Eindeutigkeit der Nodes innerhalb eines Namespaces. Andererseits werden Regeln beschrieben, welche die Übersichtlichkeit des Modells verbessern und über alle Spezifikationen hinweg zu einem einheitlichen Stil führen, wie etwa die Konvention, die BrowseNames in „upper camel case“ zu definieren.

Des Weiteren werden Konzepte zur Modellierung im Best Practices Whitepaper behandelt. Zu deren Anwendung ist häufig Hintergrundwissen notwendig, um die Modellierungssituation korrekt zu identifizieren. Dabei können gewisse Eigenschaften als Indikator herangezogen werden, beispielsweise flache Listen mit einer Vielzahl an Variablen. In diesem Beispiel sind in vielen Fällen weitere Hierarchieebenen hilfreich, um die Übersichtlichkeit der Strukturen zu verbessern. Ein weiteres Beispiel sind Rekursionen innerhalb von Strukturen. An dieser Stelle muss bei der Implementierung des Modells sichergestellt werden, dass die Anwendungen zur Laufzeit nicht ausfallen.

Ein weiterer wichtiger Aspekt ist die Abwärtskompatibilität, welche in den meisten Fällen gewünscht ist, um einen Standard konsistent fortzuführen und bestehende Implementierungen weiterhin einsetzen zu können. Dafür ist es beispielsweise wichtig, dass neue Nodes als optional definiert werden und keine bestehenden Nodes entfernt werden. Des Weiteren können Enumerations und OptionSets nicht verändert werden. Andererseits kann ein Breaking Change auch bewusst in Kauf genommen werden, wenn Basisfunktionalitäten im Modell fehlen oder sich Fehler eingeschlichen haben. Ein weiterer Grund für einen Breaking Change kann die Steigerung der Interoperabilität sein, wenn sich bestimmte Teilmodelle in anderen Spezifikationen durchgesetzt haben und die bisherige Funktionalität angeglichen werden soll. In diesen Fällen empfiehlt das Whitepaper, die Typdefinitionen der betroffenen Nodes durch eine Versionsnummer zu ergänzen oder bei größeren Änderungen den Namespace zu ändern.

Für die Testfallerstellung lassen sich die passenden Inhalte des Best Practices Whitepaper im Wesentlichen zwei Gruppen zuordnen. In der ersten Gruppe finden sich allgemeine Modellregeln, wie die bereits angesprochenen Regeln für die Benennung von Modellelementen. All diese Inhalte können ohne weitere Informationen geprüft werden.

Die zweite Gruppe behandelt Strategien zum Umgang mit „ModellingRules“, einem Konzept in OPC-UA-Modellen. Die OPC-UA-Modellinhalte können optional oder verpflichtend (mandatory) definiert werden. Speziell in Fällen, in denen Inhalte als mandatory definiert werden, aber davon ausgegangen wird, dass diese Inhalte nicht jederzeit angegeben werden können, bietet das Best Practices Whitepaper Möglichkeiten zum Umgang mit dieser Situation. Beispiele sind die Nutzung von speziellen Werten oder die Bereitstellung als schreibbar für Clients. In diesen Fällen wird zur Testfallerstellung Information über das gewählte Verfahren benötigt, eine automatische Erstellung aus dem NodeSetXML ist also nicht möglich.

Darüber hinaus sind zwei einzelne Regeln des Whitepapers aufgefallen. Erstens, eine Prüfung der Source- und Target-Nodes von References, falls diese durch den ReferenceType eingeschränkt werden. Zweitens, Pattern für applikationsspezifische Stati von Methoden. Hier wird in der Methode ein numerischer Parameter definiert, dessen Wert eine in der Spezifikation beschriebene Bedeutung transportiert. In beiden Fällen werden Informationen benötigt, die dem NodeSetXML nicht entnommen werden können.

6 Nutzung der Best-Practices-Regeln

Beim Linting während der Modellerstellung können die Regeln der Naming Conventions als allgemeine Modellregeln automatisiert überprüft werden. Bei Regelverstößen kann die entsprechende Stelle markiert und ein Hinweis zur Lösung des Problems oder zur Verbesserung des Modells angezeigt werden. Ähnliches gilt für die Abwärtskompatibilität. In diesem Fall ist es für das unterstützende Tooling wichtig, die vorherige Spezifikation zur Verfügung zu haben. Dann kann während der Modellierung ein Hinweis eingeblendet werden, dass eine Änderung zu einem Breaking Change führt und welche Regel verletzt wurde.

https://doi.org/10.37544/1436-4980-2024-05-76 https://www.inlibra.com/de/page/agb - Open Access - (cc) BY

So kann während der Modellierung abgewogen werden, ob dieser Bruch in der Spezifikation so gewollt ist.

Bei Modellierungskonzepten ist es schwierig, den Modellierungsprozess über ein Linting zu unterstützen, da in den meisten Fällen Hintergrundwissen nötig ist. Allgemeine Auffälligkeiten können jedoch hervorgehoben und durch Korrekturmaßnahmen unterstützt werden. In diesem Fall liegt die Entscheidung über den sinnvollen Einsatz bei den Modellierenden. In **Bild 3** sind die bei der Analyse ermittelten Regelkategorien in den Zusammenhang mit den Modellkategorien gesetzt, um das Einflussverhalten zu beschreiben.

Für die Bestimmung der Modellqualität durch die Entwicklung einer Metrik liefern die Best Practices einige Ansätze. Für die meisten Hinweise kann jedoch nicht direkt eine Qualitätsbewertung abgeleitet werden. Grundsätzlich können auch hier die Naming Conventions sowie einige Modellkonzepte berücksichtigt werden. Insbesondere bei der Rückwärtskompatibilität können Abweichungen nicht allein auf Basis der NodeSetXML Dateien bewertet werden, da ein Breaking Change hier in vielen Fällen die Struktur verbessert, indem der Funktionsumfang erhöht wird oder Fehler aus der Vergangenheit korrigiert werden.

Für die Definition von Testfällen ist das Best Practices Whitepaper eine unterstützende Quelle. Primärquellen sind unter anderem die Modellinhalte und die Verhaltensbeschreibungen in der Spezifikation. Trotzdem bietet das Whitepaper eine relevante Sichtweise auf die Überprüfung von Implementierungen: Wenn Best Practices als allgemein bekannt angenommen und nicht dediziert in Spezifikationen erwähnt werden, können Missverständnisse in der Implementierung entstehen. Ebenso ist eine Abweichung von Best Practices eine Erwähnung in der Spezifikation wert – das kann beim Erstellen der Testfälle auffallen, wenn sich Vorschläge für Testfälle an Best Practices orientieren.

In Bezug auf die allgemeinen Modellregeln ist eine Prüfung immer möglich und vom tatsächlichen Modell unabhängig. Die Einhaltung von Namenskonventionen zu überprüfen benötigt vorab kein Modellwissen. In Bezug auf technische Regeln und übliche Einschränkungen sind solche Tests sinnvoll, um Interoperabilität sicherzustellen. Im zugehörigen Forschungsprojekt wird diese Kategorie von Testfällen nun mit berücksichtigt.

Der Umgang mit ModellingRules ist in der Anwendung von OPC UA entscheidend. Die allgemeine Annahme, dass als mandatory deklarierte Werte vorhanden sein müssen, wird erst durch die Information, welcher Wert oder welches Verhalten erwartet wird, semantisch wertvoll. Auch hier ist das Definieren von Testfällen sinnvoll: So kann sichergestellt werden, dass das Serververhalten der Spezifikation entspricht und dass benötigte Funktionalitäten (zum Beispiel Schreibzugang für Clients) in Implementierungen berücksichtigt sind. Für alle betrachteten Beispiele des Whitepapers werden zum NodeSetXML zusätzliche Informationen benötigt – allen voran die Information, dass eine besondere Nutzungsregel für eine bestimme Node existiert. Bei vorgegebenen Werten müssen exakt diese Werte bekannt sein.

Für Einschränkungen der Source- und Target-Nodes von References sind im NodeSetXML keine Informationen enthalten. Diese Informationen müssen also von den Erstellern der Spezifikation kommen und sollten auch im Spezifikationstext zu finden sein, um korrekt implementiert zu werden. Zur Erstellung von Testfällen ist initial die Kenntnis notwendig, ob eine solche Einschränkung existiert und wie diese Einschränkung gestaltet ist.

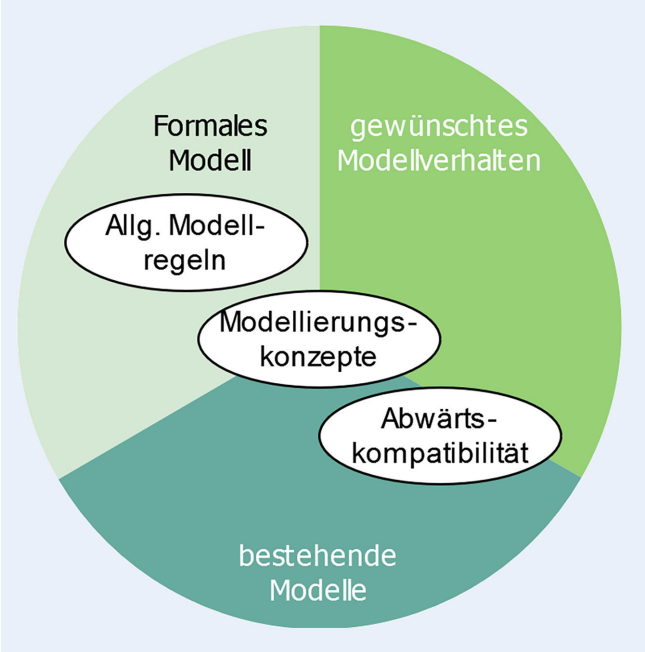

Bild 3. Im Linting genutzte Regeln im Zusammenhang mit Modellkategorien. *Grafik: Universität Stuttgart, ISW*

Ebenso verhält es sich für Stati von Methoden. Soll ein applikationsspezifischer Status zurückgegeben werden, so muss dessen Bedeutung erklärt sein. Im NodeSetXML ist ein Vorkommen solcher Stati schwer zu erkennen. Die Empfehlung ist, einen „Int32“ als Datentyp zu nutzen und den Wert als letztes Output-Argument der Methode zu definieren. Der umgekehrte Schluss kann allerdings nicht gezogen werden, ein Output-Argument des Datentyps Int32 kann auch andere Bedeutungen als einen Status haben. Auch hier ist erforderlich, dass die Ersteller der Spezifikation aktiv an der Testfallerstellung teilnehmen, sofern eine Überprüfung dieser Codes gewünscht ist.

Allgemein können die Gründe, Testfälle zu erstellen in drei Kategorien eingeteilt werden, wie in **Bild 4** dargestellt.

Zum einen, um das formal definierte Modell zu testen, wie es im NodeSetXML beziehungsweise auf dem OPC-UA-Server vorliegt. Hier müssen alle Definitionen exakt mit der Spezifikation übereinstimmen. Diese Testfälle sind oftmals automatisch aus dem Modell ableitbar. Die zweite Kategorie ist das Modellverhalten. Bestimmte Werte oder Wertänderungen, deren Grund im OPC-UA-Server oder außerhalb liegt, können sowohl vorgegeben und beschrieben als auch überprüft werden. Nötig ist dabei, dass die betreffende Situation im Testlabor nachgestellt werden kann. Die dritte Kategorie bezieht sich auf den Umfang des Modells: sollen bestimmte Funktionalitäten geboten werden, ist meist ein Mindestumfang des Modells erforderlich. Die vollständige Umsetzung dieses Umfangs kann durch Testfälle bestätigt werden.

Die aus dem Best-Practices-Dokument extrahierten möglichen Testfälle liegen fast alle im Bereich des Modellverhaltens: ein gesonderter Umfang der Nutzung von optional/mandatory geht über das formale Modell hinaus, wie auch die Einschränkung der Source- und Target-Nodes von References, die im NodeSetXML nicht enthalten ist. Alle speziell definierten Werte und Wertebereiche sind Teil des Modellverhaltens – so auch applikationsspezifische Stati von Methoden. Nur die allgemeinen Modellregeln fallen in den Bereich des formalen Modells. Die Unabhängigkeit

https://doi.org/10.37544/1436-4980-2024-05-76 https://www.inlibra.com/de/page/agb - Open Access - (cc) BY

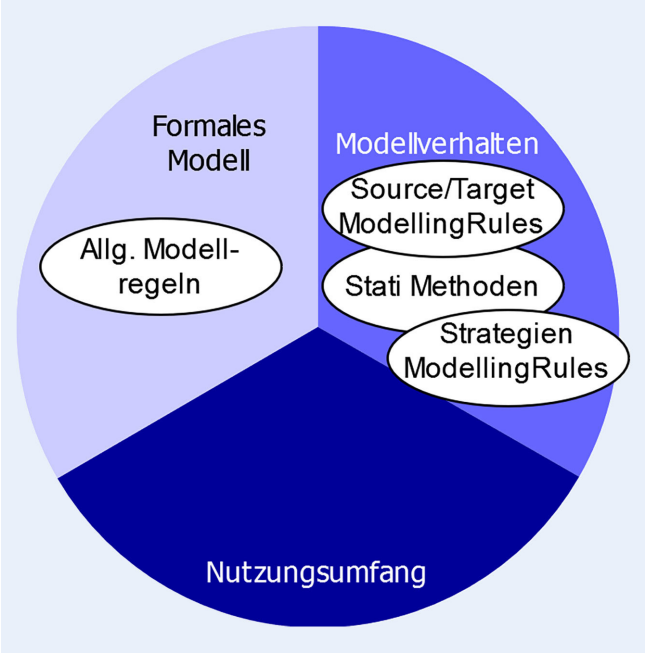

Bild 4. Zur Testfallgenerierung genutzte Regeln im Zusammenhang mit Testzwecken. *Grafik: Universität Stuttgart, ISW*

dieser Testfälle vom vorliegenden Modell spielt allerdings eine Sonderrolle in dieser Kategorie.

Testfälle für die allgemeinen Modellregeln können Tools wie das Compliance Test Tool (CTT) [8] in Zukunft bereichern und müssen dabei nicht je CS erstellt werden. Die Testfälle zu References, Stati von Methoden und dem Umgang mit ModellingRules müssen dagegen für jede CS erstellt werden. Dabei ist ein Vorteil, dass den Erstellern der Spezifikation vor Augen geführt wird, die notwendigen Zusammenhänge auch klar in der Spezifikation zu beschreiben. Ein weiterer Vorteil besteht vor allem in Testfällen für die ModellingRules. Hier werden wesentliche Modellinhalte abgeprüft, deren korrekte Umsetzung oft für eine Nutzung des Modells grundlegend ist. Es lohnt sich also, Testfallersteller darauf hinzuweisen, auf diese Kategorie von Testfällen besonders zu achten.

FÖRDERHINWEIS

Im Forschungsprojekt „Entwicklung eines Expertensystems für eine informationstechnische Automatisierung der Erstellung von OPC-UA-Testfällen und -skripten in Industrial-Internet-of-Things-Anwendungen“ wird die automatisierte Erstellung von Testfällen thematisiert. Das Projekt wird vom Bundesministerium für Wirtschaft und Klimaschutz (BMWK) aufgrund eines Beschlusses des Deutschen Bundestages gefördert.

Das Projekt „Clean OPC UA Information Modeling“ (23084 BG / 1) der Forschungsvereinigung VDW-Forschungsinstitut e.V. wird im Rahmen des Programms zur Förderung der Industriellen Gemeinschaftsforschung (IGF) vom Bundesministerium für Wirtschaft und Klimaschutz aufgrund eines Beschlusses des Deutschen Bundestages gefördert.

7 Zusammenfassung und Ausblick

In Summe liefert das Best Practices Whitepaper wichtige Hinweise, welche den gesamten Lebenszyklus einer CS begleiten. Die Analyse des Dokuments bezogen auf die Entwicklung weiterer Tools zur Unterstützung des Modellierungsprozesses hat relevante Regeln identifiziert, welche sich voraussichtlich für eine automatisierbare Überprüfung sowie Anzeige von Hinweisen in Form eines Linting eignen. Zur Erstellung einer Modellmetrik liefern dieselben Regeln Anhaltspunkte, um ein umfassenderes Bild der Modellqualität zu erzeugen. Von diesem Ausgangspunkt werden im Weiteren auch Modellmetriken aus anderen Domänen untersucht und deren Übertragbarkeit ermittelt.

Für die Testfallerstellung sind vor allem Anstöße für Testkonzepte zu ModellingRules aus dem Best Practices Whitepaper hervorzuheben. Die Intention der Spezifikation in diesem Bezug ist von großer Bedeutung für interoperable Softwarelösungen, daher ist der Nutzen dieser Testfälle voraussichtlich hoch. Testfälle für die allgemeinen Modellregeln liefern einen schnellen und einfachen Weg zur Vermeidung von Formfehlern. Um eine konsistente Menge an Testfällen für eine CS zu erstellen, werden im Verlauf des zugehörigen Projekts Anhaltspunkte für Testfälle aus allen drei hier vorgestellten Kategorien zusammengestellt.

Literatur

[1] VDMA e. V. (Hrsg.): Studie zur Interoperabilität im Maschinen- und Anlagenbau – Die Weltsprache der Produktion als Grundlage für Industrie 4.0. Stand: 2021. Internet: www.vdma.org/documents/34570/4802302/Studie+OPC+UA.pdf/16364989-bed9–705b-0e22–08daac09c173?t=1618219349899. Zugriff am 26.03.2024

[2] International Electrotechnical Commission.: IEC TR 62541–1:2020. OPC Unified Architecture – Part 1: Overview and Concepts. Deutsche Fassung, Ausgabe November 2020

[3] Drath, R. et al.: Diskussionspapier – Interoperabilität mit der Verwaltungsschale, OPC UA und AutomationML – Zielbild und Handlungsempfehlungen für industrielle Interoperabilität. Stand: 11.04.2023. Internet: opcfoundation.org/wp-content/uploads/2023/04/Diskussionspapier-Zielbild-und-Handlungsempfehlungen-fur-industrielle-Interoperabilitat-5.3-protected.pdf. Zugriff am 26.03.2024

[4] OPC Foundation.: OPC 11030 UA Modelling Best Practices. Draft 1.02.00 [noch nicht veröffentlicht]

[5] OPC Foundation.: OPC 10000–100: Devices. Released 1.04. Stand: 03.11.2022. Internet: reference.opcfoundation.org/DI/v104/docs/. Zugriff am 02.04.2024

[6] OPC Foundation.: OPC 10000–200: Industrial Automation – Basics. Released 1.01.2. Stand: 17.06.2022. Internet: opcfoundation.org/developer-tools/documents/view/199. Zugriff am 02.04.2024

[7] Heinemann, T.; Friedl, S.; Lechler, A.: OPC-UA-Domänenmodelle heute und morgen. wt Werkstattstechnik online 112 (2022) 5, S. 320–324

[8] OPC Foundation.: OPC 21010–1 OPC TestLab – Part 1: Concepts. Stand 16.02.2023. Internet: opcfoundation.org/developer-tools/documents/view/325. Zugriff am 02.04.2024

[9] Keilig, C.: Clean OPC UA Information Modeling – CLOU. Fraunhofer-Institut für Werkzeugmaschinen und Umformtechnik IWU. Internet: www.iwu.fraunhofer.de/de/projekte/clean-opc-ua-information-modeling-clou.html. Zugriff am 26.03.2024

[10] Heinemann, T. et al.: Automating Test Cases for OPC UA Information Models. 27th International Conference on Production Research Cluj-Napoca, Romania, 23–28 July 2023. Lecture Notes in Production Engineering. Cham: Springer International Publishing, 2023, pp. 101-112

[11] OPC Foundation.:OPC-F Working Groups. Internet: opcfoundation.org/opcf-wg/. Zugriff am 26.03.2024

[12] Universität Stuttgart, ISW.: OPC UA Testfall Generator. Internet: www.isw.uni-stuttgart.de/forschung/kommunikation/OPC-UA-Testfall-Generator/. Zugriff am 26.03.2024

https://doi.org/10.37544/1436-4980-2024-05-76 https://www.inlibra.com/de/page/agb - Open Access - (cc) BY

Tonja Heinemann, M.Sc.
tonja.heinemann@isw.uni-stuttgart.de
Tel. +49 711 / 685-84626
Foto: ISW

Timo König, M.Sc.

Dr.-Ing. Armin Lechler

Prof. Dr.-Ing. Oliver Riedel

Universität Stuttgart
Institut für Steuerungstechnik der Werkzeugmaschinen und Fertigungseinrichtungen (ISW)
Seidenstr. 36, 70174 Stuttgart
www.isw.uni-stuttgart.de

LIZENZ

Dieser Fachaufsatz steht unter der Lizenz Creative Commons Namensnennung 4.0 International (CC BY 4.0)

https://doi.org/10.37544/1436-4980-2024-05-76 https://www.inlibra.com/de/page/agb - Open Access -