Absicherung vernetzter IoT-Funktionen mit selbstlernenden Modellen

Gereon Weiß, Christian Drabek
{gereon.weiss|christian.drabek}@esk.fraunhofer.de
Fraunhofer ESK, Hansastr. 32, 80686 München

Zusammenfassung. Mit dem Internet-of-Things (IoT) wird die Verknüpfung diverser Systeme von eingebetteten Steuerungen über Cloud-Services und diverse Frameworks sowie Plattformen möglich. Für den Nachrichtenaustausch der beteiligten Komponenten existiert bereits heute eine Vielzahl von grundlegenden Kommunikationsprotokollen. Die darauf aufbauenden Anwendungsprotokolle sind jedoch in der Regel spezifisch für einzelne Applikationen definiert und umgesetzt. Um auch das korrekte Funktionieren einer vernetzten Anwendung sicherzustellen, muss deren Interaktion mit anderen Komponenten abgesichert und verifiziert werden. Damit dies effizient möglich ist, sind neue Verfahren zur automatisierten Absicherung notwendig, so dass die korrekte Interaktion solcher verteilter Anwendungen sichergestellt werden kann. Hierfür wird ein modellgetriebenes Verfahren vorgestellt, welches es erlaubt Fehler automatisiert anhand des Kommunikationsverhaltens festzustellen. Um den Aufwand hierfür gering zu halten und auch neue, zunächst unbekannte Anwendungen absichern zu können, wird ein selbstlernendes Verfahren eingesetzt. Das kann teilautomatisiert Modelle aus Anwendungsprotokollen erzeugen, die dann wiederum überprüft und weiterverwendet werden können. So können diese Modelle auch wieder zur automatisierten Absicherung der Anwendungen genutzt werden. Das Verfahren wurde in verschiedenen Projekten und Anwendungsszenarios, wie am Beispiel einer Modell-Produktionsanlage, bereits erfolgreich erprobt.

1 Vernetzte IoT-Funktionen

Neue Anwendungen im sogenannten *Internet-of-Things (IoT)* bestehen aus einer Vielzahl verknüpfter Funktionen, die wiederum verteilt ausgeführt werden. Das kann von Apps, über Cloud-Dienste bis hin zu eingebetteten Systemen reichen. Vor allem zuletzt genannte benötigen jedoch eine hohe Verlässlichkeit, insbesondere wenn sie auch sicherheitsrelevante Aufgaben übernehmen. Aus diesem Grund muss auch die gesamte Anwendung entsprechend ausgelegt und abgesichert werden. Beispiele hierfür sind untereinander und

Die in dieser Arbeit gezeigten Ergebnisse wurden in Teilen mit Förderung durch das Bundesministerium für Wirtschaft und Energie (BMWi) und das Bayerische Staatsministerium für Wirtschaft und Medien, Energie und Technologie erzielt.

mit der Infrastruktur vernetzte Fahrzeuge (vgl. **Abbildung 1**) oder Industriesteuerungen (vgl. **Abbildung 3**).

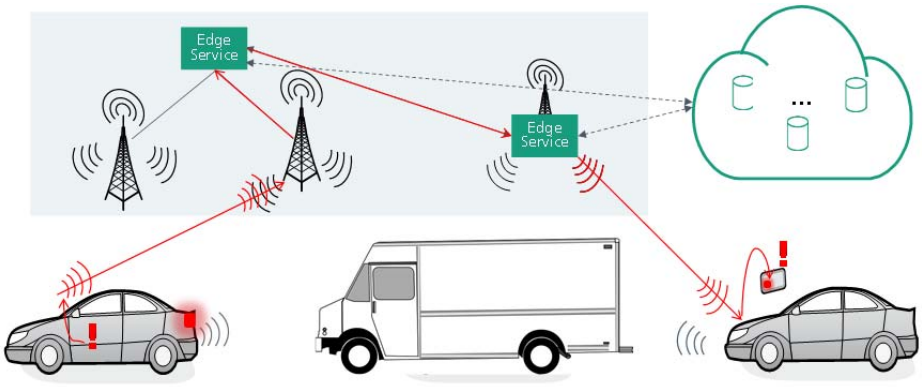

Abbildung 1 Beispiel einer vernetzten Fahrfunktion mit einem Car2X Gefahrenwarner [1]

Vor allem aus Sicht eingebetteter Systeme verändert sich in diesen Szenarien die Komplexität der Kommunikation von einfachen Broadcast Informationen hin zu interaktiven Nachrichten (s. **Abbildung 2**). Dadurch werden auch die Schnittstellen der Software-Komponenten komplexer [2].

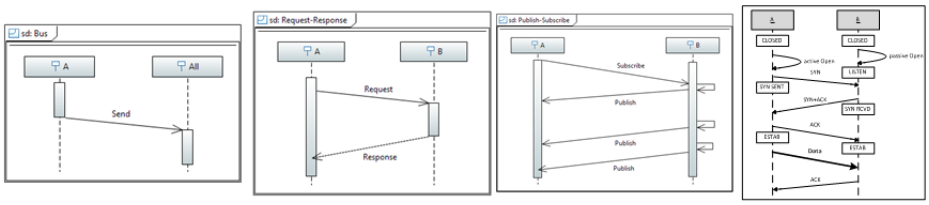

Abbildung 2 Gegenüberstellung der Kommunikationskomplexität von BUS-basierter, Request-Response, Publish-Subscribe und TCP/IP Kommunikation

Wenn nun die im IoT involvierten Systeme flexibler und adaptiver werden, muss die *Interoperabilität* der verschiedenen Systeme sichergestellt werden. So verändert sich insbesondere, dass Systeme nicht mehr klar abgetrennt und abgeschlossen von einer Umgebung geplant und verwendet werden, sondern mit unbekannten Systemen interagieren. Vernetzte mobile Produktionsanlagen sind nur ein Beispiel, an dem klar wird, wie sich hier grundlegende Systemannahmen verändern, sobald mehr Flexibilität notwendig wird. Um jedoch den hohen Anforderungen an Qualität und Sicherheit sicherheitsrelevanter Systeme gerecht zu werden, müssen solche vernetzte Funktionen abgesichert werden.

2 Absicherung von Anwendungsprotokollen

Da vernetzte Anwendungen durch den Verbund verteilter Funktionen entstehen, muss auch gesondert diese Interaktion abgesichert werden. Für grundlegende und standardisierte Kommunikationsprotokolle kommen hier häufig *Conformance Tests* zum Einsatz. Diese überprüfen mit speziellen Tests, ob Standards in den einzelnen Systemen entsprechend korrekt umgesetzt wurden. Die Idee dahinter ist, dass wenn diese Kommunikations-Stacks ausreichend konform implementiert sind, sie mit anderen standard-konformen Systemen funktionieren. Darauf aufbauende Anwendungen, wie Server-basierte Steuerung von Produktionsanlagen, sind jedoch häufig sehr individuell, so dass keine allgemeinen Standards oder Testsuites dafür existieren. Einzelne *Unit Tests* der Anwendungen können jedoch häufig nur das statische Verhalten absichern. Jedoch ist hier natürlich besonders die Absicherung der verschiedenen Funktionen in Interaktion wichtig. Dies betrifft insbesondere das *dynamische Verhalten der Schnittstellen*. Das heißt beispielsweise gegenüber den statischen Schnittstellen-Anteilen, wie Funktionsname und Parameter, in welcher Reihenfolge Funktionen mit anderen kommunizieren. Um eine solche Interoperabilität sicherzustellen, sind modellgetriebene Ansätze hilfreich.
Sogenannte *Referenzmodelle* können das gültige Kommunikationsverhalten von Software-Komponenten beschreiben. Sie stellen somit eine Art *semiformale Spezifikation* dar. Ein Abgleich von beobachtetem System-Verhalten mit den Modellen erlaubt es, Abweichungen durch sogenanntes *Monitoring* zu erfassen. Dabei kann das System passiv beobachtet werden, ohne dass spezielle Tests entwickelt werden müssen. Spezielle *Synchronisationsmechanismen* [3] erlauben es, solche Referenzmodelle auch trotz auftretender Fehler weiterhin parallel zu realen Systemen auszuführen.
Allerdings müssen solche Modelle aktuell manuell entworfen und überprüft werden. Das stellt einen initialen Aufwand dar, der jedoch weiter reduziert werden kann. So können verschiedene *Lernverfahren* (u.a. Conceptual / Hierarchical Clustering, Decision Mining oder Maximal Pattern Mining) genutzt werden, um aus bereits existierenden Systemen Referenzmodelle teilautomatisiert abzuleiten. Dazu wird das reale Verhalten anhand der mitgeschnittenen Kommunikation als Zustandsautomaten gelernt. Dieses Vorgehen bietet den Vorteil, dass die Automaten angepasst werden können. Bei Nutzung Neuronaler Netze beispielsweise, die von außen eine Black-Box darstellen, wäre das so ohne weiteres nicht möglich ist. Somit können die gelernten Modelle manuell überprüft und angepasst werden, um ggf. nicht gewollte Abweichungen oder nicht erfasstes Verhalten zu ergänzen. Daher können solche Modelle sowohl von fertigen Implementierungen als auch von Prototypen als Ausgangsbasis gelernt werden. Werden letztere verwendet, so können die Referenzmodelle entsprechend aktualisiert und erweitert werden. In jedem Fall

kann so der Aufwand für die initiale Erstellung eines Modells stark reduziert werden.

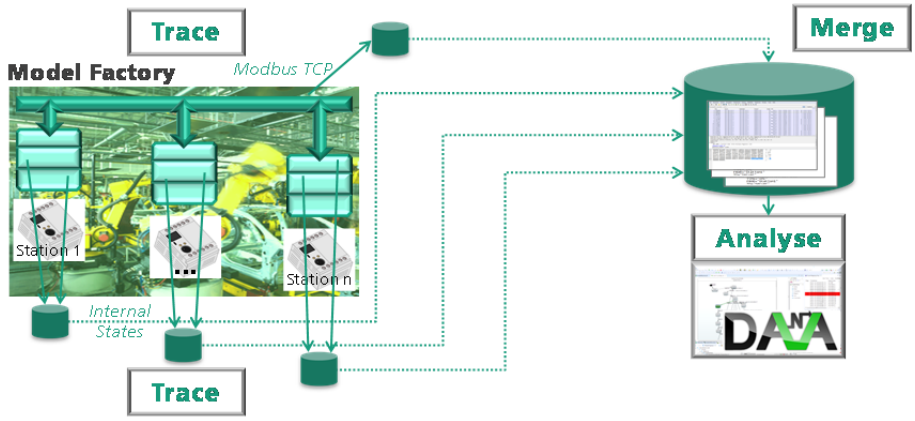

Abbildung 3 Beispiel einer Modell-Produktionsanlage mit verschiedenen Steuerungen, internen Zuständen und über ModbusTCP kommunizierter Daten, die durch Referenzmodelle analysiert werden.

Dieses Verfahren wurde bereits erfolgreich in einem Eclipse-basierten Werkzeug *DANA* [4] implementiert und anhand eines Anwendungsbeispiels erprobt. Am Beispiel einer Modell-Produktionsanlage (vgl. **Abbildung 3**) konnte das Verhalten der Steuerungen gelernt werden, um ein erstes Referenzmodell der Anlage zu erhalten (s. **Abbildung 4**).

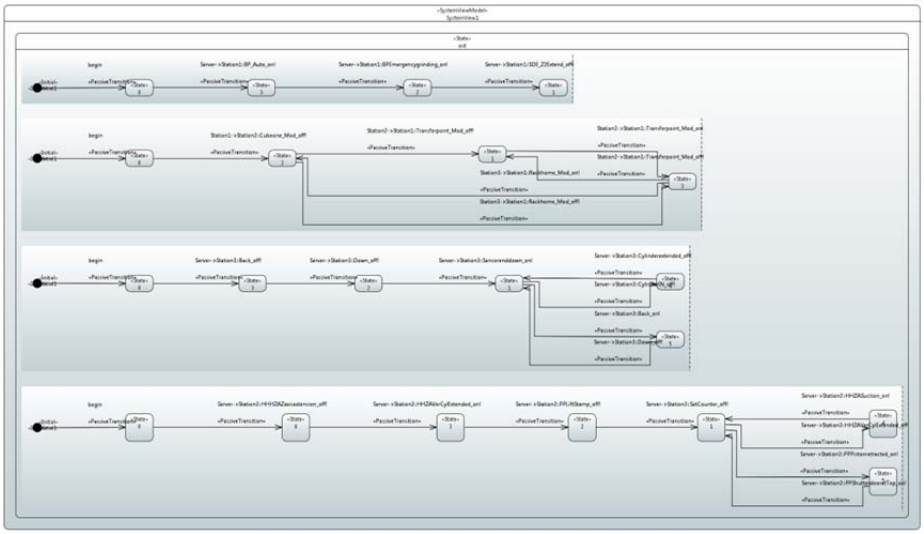

Abbildung 4 Beispiel eines gelernten Referenzmodells der Modell-Produktionsanlage

Das vorgestellte Verfahren ist jedoch nicht auf einzelne Anwendungen beschränkt, sondern kann auf verschiedene vernetzte IoT-Funktionen und Kommunikationsprotokolle angepasst werden.

3 Zusammenfassung & Ausblick

Mit der Etablierung des Internet-of-Things wird eine Vielzahl von Systemen miteinander vernetzt. Damit diese vernetzten Systeme korrekt miteinander interagieren, ist eine funktionale Absicherung der Funktionen unerlässlich. Da die Interoperabilität der Funktionen besonders wichtig ist, müssen deren Interaktionen abgesichert werden. Hierzu können Referenzmodelle genutzt werden, welche das korrekte, gewünschte Sollverhalten beschreiben. Verhält sich das System anders als modelliert, so können Abweichungen im Vergleich direkt erkannt und behoben werden. Um das schnelle Erstellen von Referenzmodellen zu unterstützen, können Lernverfahren eingesetzt werden, die Modelle teilautomatisiert anhand des Verhaltens vorhandener Systeme generieren. Auf deren Basis können dann einfacher komplexe Referenzmodelle zur Absicherung erzeugt werden.

Literaturverzeichnis

[1] G. Weiß and J. Jiru, "Software implementieren und absichern - Mit Modellierung zum schnelleren Prototyping," *Embedded Design*, vol. 3, no. 8, pp. 42-44, 2017.

[2] G. Weiß and C. Drabek, "Absicherung von komplexen Software-Komponenten vernetzter Fahrzeuge," *Automotive Software Kongress*, 2016.

[3] C. Drabek, G. Weiß, and B. Bauer, "Method for automatic resumption of runtime verification monitors," *International Conference on Advances and Trends in Software Engineering (SOFTENG)*, 2017.

[4] DANA - Description and Analysis of Networked Applications (letzter Zugriff: 26.09.2017). [Online]. http://s.fhg.de/DANA