www.itk-engineering.de

itk ENGINEERING

SERVICE OVERVIEW

CHARGE ENABLER:
OSS für die Absicherung ISO 15118 konformen Ladens

Der Charge Enabler als Open-Source-Tool liefert eine neue ISO 15118-konforme Basis, marktspezifische Anforderungen zur reibungslosen Kommunikation zwischen Ladesäule und Fahrzeug zu testen.

Charge Enabler

SW-Test Laboratory

- Open Source
- Highly Modular
- No Hardware needed

Motivation

Mit steigendem Nachhaltigkeitsbewusstsein und dem Ausbau erneuerbarer Energien steigt auch das Interesse an der Elektromobilität. Jedoch bringt die steigende Zahl an Elektrofahrzeugen im europäischen Markt auch eine erhöhte Nachfrage an geeigneter Ladeinfrastruktur mit sich. Noch immer ist diese unzureichend ausgebaut und die damit verbundenen potenziellen Herausforderungen der Routenplanung ein Grund auf Elektrofahrzeuge zu verzichten. Diesen Markt haben zahlreiche Unternehmen neu für sich entdeckt und wollen sich positionieren. Allerdings nehmen Neuentwicklungen mehr Zeit in Anspruch als eingeplant, weil Fehler in der Softwareentwicklung erst spät auffallen oder bis zur Freigabe häufig unentdeckt bleiben und erst im Betrieb, in Form von Ladeabbrüchen, erkannt werden. Auch erfahrene Unternehmen in der Elektromobilität, sowohl OEMs im Automotive-Umfeld als auch Hersteller von Ladeinfrastrukturen, stehen immer wieder vor der Herausforderung, bestehende Ladeabbrüche frühzeitig mit geeignetem Testsetup zu erkennen, zu minimieren und letztendlich ihre Entwicklungen ganzheitlich zu optimieren.

Neben einem potenziellen Imageverlust, bedeutet dies für die Entwicklung auch erheblichen Mehraufwand und -kosten. Zum einen gilt es die Ursache zu finden und gegebenenfalls softwareseitig zu lösen, zum anderen muss die Software im Betrieb angepasst werden.

Um diesen Herausforderungen Rechnung zu tragen, wurde der Charge Enabler entwickelt. Er bildet ein Open Source Testingframework für ISO 15118 normkonforme Ladevorgänge in frühen Entwicklungsphasen für die softwareseitige Absicherung anhand von Konformitätstests, um Softwarefehler frühzeitig zu entdecken, die Entwicklung zu optimieren und somit Ladeabbrüche im Betrieb zu verringern. Hierdurch kann die Entwicklung beschleunigt, spätes Auffinden von Fehlern minimiert und somit Entwicklungs- und Wartungskosten verringert werden.

Elektrisches Laden

In Europa gibt es verschiedene Möglichkeiten der elektrischen Energieversorgung von Elektrofahrzeugen. Hierbei wird zwischen verschiedenen Ladearten unterschieden: vom konduktiven Laden mit Wechselstrom (AC-Laden), über das Laden mit Gleichstrom (DC), bis hin zum induktiven Laden. Je nach Ladeart befinden sich das eigentliche Ladegerät und die Ladeleitung in dem Fahrzeug selbst, oder in einer Ladestation mit unterschiedlichen Kommunikationsschnittstellen.

Darüber hinaus werden die Ladevorgänge durch ihre Ladeleistung unterschieden. Bis zu 22 kW spricht man hier von „Normalladen“, zwischen 22 kW und 50kW von „Schnellladen“ und oberhalb von 50 kW von „Hochleistungsladen“. AC-Laden ist im Bereich des Normal- und Schnellladens vorgesehen, während DC-Laden auch im Bereich des Hochleistungsladens, gleichwertig zu High-Power-Charging (HPC), möglich ist.

Aus diesen verschiedenen Ladearten und -vorgängen ergeben sich für das kabelgebundene Laden die folgenden Ladebetriebsarten [DIN EN IEC 61851-1]:

Ladebetriebsart 1 / Mode 1 beschreibt das Laden mit Wechselstrom an einer Schutzkontaktsteckdose oder einer Industriesteckdose mit einer beziehungsweise drei Phasen. Diese Ladebetriebsart kommt ohne eine definierte Kommunikationsschnittstelle zwischen Fahrzeug und Ladeinfrastruktur aus.

Ladebetriebsart 2 / Mode 2 beschreibt ebenfalls das Laden mit Wechselstrom. Der Unterschied zu Mode 1 besteht jedoch in einer im Fahrzeug integrierten Ladeleitung, welche Steuer- und Schutzeinrichtung beinhaltet. Über ein Pilotsignal wird der Informationsaustausch und die Überwachung angestoßen.

Ladebetriebsart 3 / Mode 3 beschreibt eine dritte Variante des Ladens mit Wechselstrom. Auch hier werden ein beziehungsweise drei Phasen genutzt, allerdings ist eine fest installierte Ladestation nötig, an welcher die Ladeleitung angeschlossen ist. Sicherheitsfunktionalitäten mit Fehlerstrom-Schutzeinrichtung sind in der Gesamtinstallation integriert. Die gesamte Kommunikation verläuft über die Ladeleitung.

Ladebetriebsart 4 / Mode 4 wiederum beschreibt das Laden mit Gleichstrom. Dieser Ladevorgang benötigt fest installierte Ladestationen, die über eine Ladeleitung und dem eigentlichen Ladegerät verfügen. Wie auch in Mode 3 verläuft die gesamte Kommunikation über die Ladeleitung.

Bei den Modes 2, 3 und 4 liegt die sogenannte „Low Level" Kommunikation nach IEC 61851-1 zugrunde, über welche grundlegende Informationen ausgetauscht werden. Eine zusätzliche Kommunikation gemäß der Norm ISO 15118 ist bei der Ladebetriebsart 3 optional möglich.

In der Entwicklung muss diese Vielzahl an Ladebetriebsarten und den damit verbundenen Variationen beachtet werden, um Fehlfunktionen im Betrieb zu vermeiden. Nicht zuletzt, da es neben möglichen Unzufriedenheiten der Nutzer, auch um Schadensminimierung geht. In den meisten Fällen sind die im Betrieb auftretenden, vom Kunden erlebbaren Ladeabbrüche auf Kommunikationsprobleme zwischen Fahrzeug und Ladeinfrastruktur zurückzuführen, welche insbesondere auf potenzielle Software-Schwächen und unzureichende Implementierung des Kommunikationsstandards hindeuten. Für die Entwicklung gilt es also, die Herausforderung der normkonformen Kommunikation zu meistern. In Europa wird hierfür in der Regel die ISO 15118 herangezogen und gilt als Standard.

ISO 15118 als Rahmen

Der Kommunikationsstandard „ISO 15118 - Road Vehicles - Vehicle to Grid Communication Interface" bildet die notwendige Kommunikations-, Identifikations-, Authentifizierungs- und Autorisierungsprozesse zwischen E-Fahrzeug und Ladeinfrastruktur ab. Die bidirektionale Kommunikation nach diesem internationalen Standard erlaubt den Austausch von Daten zwischen Elektrofahrzeug und Ladesäule, wie beispielsweise Ladestand der Batterie (State of Charge, SoC), Energiebedarf und Ladeleistung und sorgt für effektive Ladevorgänge sowie die Entwicklung komfortabler Abrechnungssysteme. Sie garantiert somit eine zukunftssichere und einfache Ladeinfrastruktur. Diese offene und standardisierte Schnittstelle soll sich stetig weiterentwickeln und mit zunehmender Ausbreitung der Elektromobilität im Allgemeinen weiter ausgebaut werden. Um die Kommunikation während eines Ladevorgangs auf ihre Normkonformität zu prüfen, bieten sich Konformitätstests an, welche zumindest anteilig in der ISO 15118 bereitgestellt werden.

Konformitätstests

Wie zuvor bereits erwähnt, ist die hohe Anzahl an Ladeabbrüchen hauptsächlich auf Softwarefehler in der Kommunikation zurückzuführen. Aus diesem Grund bietet es sich an, bereits in frühen Entwicklungsphasen mit der Absicherung und Qualitätssicherung zu beginnen. Insbesondere Konformitätstests sollten hier im Fokus stehen.

Der Begriff Konformität im Bereich der Softwareabsicherung beschreibt: „Die Fähigkeit eines Softwareprodukts, anwendungsspezifische Normen oder Vereinbarungen oder gesetzliche Bestimmungen und ähnliche Vorschriften zu erfüllen [ISO 9126]." mit Blick auf die folgenden Qualitätsmerkmale: Funktionalität, Zuverlässigkeit, Benutzbarkeit, Effizienz, Änderbarkeit bzw. Übertragbarkeit. Kurz: die Prüfung auf Normverträglichkeit. Hierfür werden in der Regel funktionale Blackboxtestfälle verwendet, welche die Kommunikation gegen das zugrundeliegende Protokoll verifizieren. Diese Testfälle sind aus der Spezifikation abgeleitet und ignorieren die Implementierung und innere Struktur des zu testenden Systems („Black-Box").

Um Interoperabilität zwischen dem Fahrzeug (EVCC - Electric Vehicle Communication Controller) und der Ladeinfrastruktur (SECC - Supply Equipment Communication Controller) sicherzustellen, sollte diese ladestandardspezifische Verifikation durchgeführt werden. Die in der ISO 15118-4 enthaltenen standardisierte Konformitätstests sichern die Übereinstimmung von Netzwerk- und Anwendungsprotokollen gemäß ISO 15118-2 ab. Diese Testfallimplementierungen liegen in einer domänenspezifischen, formalen Programmiersprache zum Testen kommunikationsbasierter Systeme – namens TTCN-3 „Testing and Test Control Notation"– vor und können beispielsweise durch den kostenfreien Eclipse Titan Compiler interpretiert werden. Dieser bietet eine integrierte Entwicklungs- und Ausführungsumgebung für die modulare und logische Testfallbeschreibung.

Der Charge Enabler bietet die Möglichkeit der Stimulation der zu testenden Komponente und führt die standardisierten Konformitätstests mit anschließender Bewertung aus.

Charge Enabler: Komplexität beherrschbar machen

Der Charge Enabler hat sich daher als Ziel gesetzt, das frühe Auffinden von Softwarefehlern in der Kommunikation zu ermöglichen und somit zur Qualitätssicherung beizutragen, um Kosten einzusparen. Auf dem Markt existieren bereits einige Testframeworks zur Absicherung von Ladevorgängen in der Elektromobilität. Aber zum einen sind diese sehr kostenintensiv und zum anderen ist die Zielhardware erforderlich. Der Charge Enabler bietet ein modulares Open Source Testframework, welches bereits in frühen Entwicklungsstadien Anwendung findet und benötigt erstmal keine Hardware. Dadurch kann die Konformität softwareseitig und unabhängig von möglichen Lieferengpässen der Hardware sichergestellt werden.

Die kundenspezifische SECC- oder EVCC-Software kann mit wenigen Adaptionen eingebunden und auf Konformität geprüft werden.

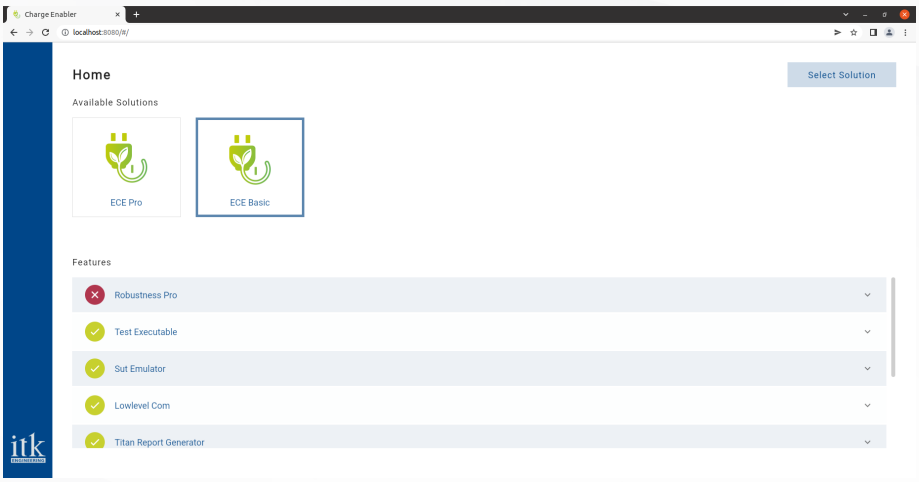

Abbildung 1 Home Scrreen

Verschiedene Testsuiten aus der ISO 15118 stehen hierfür zur Verfügung. Die Testergebnisse geben Aufschluss über nicht normkonforme Implementierung des Ladestandards. Durch das reine Softwaretesten wird die Fehlersuche erleichtert, da Wechselwirkungen mit der Hardware ausgeschlossen sind. Über eine benutzerfreundliche Web-Oberfläche hat der Nutzer die Möglichkeit, die Bedienung des Charge Enablers vorzunehmen, Testsuiten für Konformitätstests anwendungsfallbasiert zu parametrisieren und bei Bedarf auch für spätere Ausführungen zu speichern.

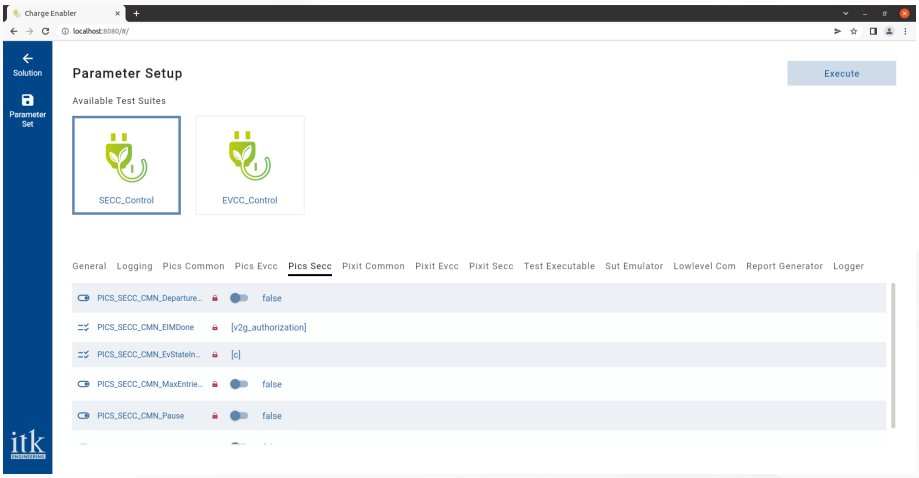

Abbildung 2 Parameter Setup Screen

Basierend auf den eingestellten Parametern werden die Testfälle vollautomatisiert ausgeführt. Während der Ausführung werden die Status der Testfälle aktualisiert. Der Nutzer hat dabei immer die Möglichkeit, die Testausführung zu pausieren, fortzusetzen oder abzubrechen.

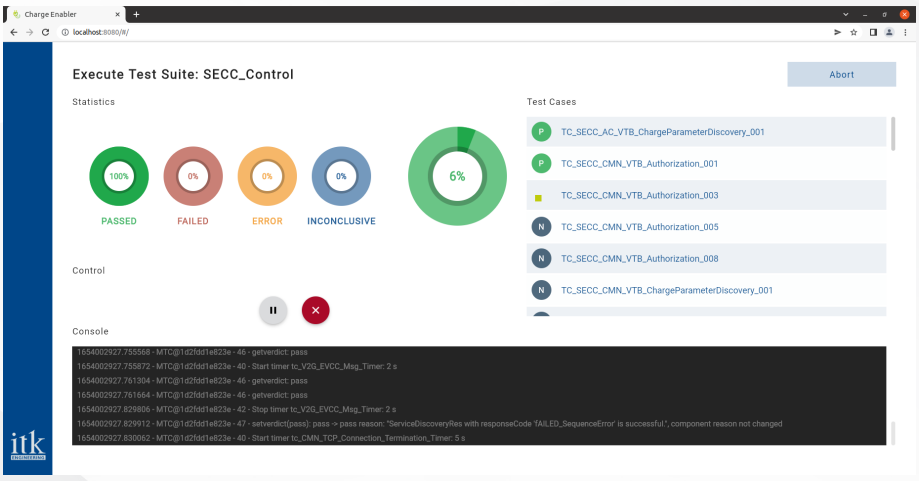

Abbildung 3 Test Execution Screen

Ist die Testsuite beendet, werden die Testergebnisse des Testlaufes dargestellt. Hierbei wird das Endergebnis visualisiert und die prozentualen Anteile, gegliedert in „PASSED“, „FAILED“, „ERROR“ und „INCONCLUSIVE“, angezeigt.

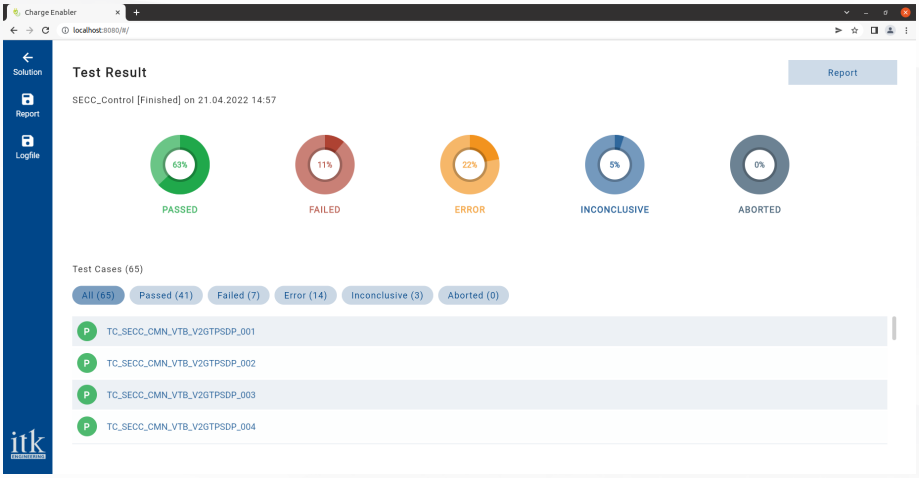

Abbildung 4 Test Result Screen

Die Testläufe werden abgespeichert und können im Nachgang mit den dazugehörigen Parametern erneut ausgeführt werden.

Abbildung 5 Test Run Screen

Durch den modularen Aufbau des Charge Enablers und die darin verwendeten Technologien ist es verhältnismäßig einfach möglich, die bereits bestehenden Funktionalitäten um verschiedene Features, wie zum Beispiel kundenspezifische Screens oder Auswertungen, zu erweitern.

Die Architektur

Wie zuvor beschrieben, besticht der Charge Enabler durch seinen modularen Aufbau und wenigen Abhängigkeiten innerhalb der Softwarearchitektur. Die grauen Anteile kennzeichnen verwendete Open Source Software (OSS) innerhalb des Charge Enablers, die weißen Anteile kennzeichnen die von ITK entwickelte OSS.

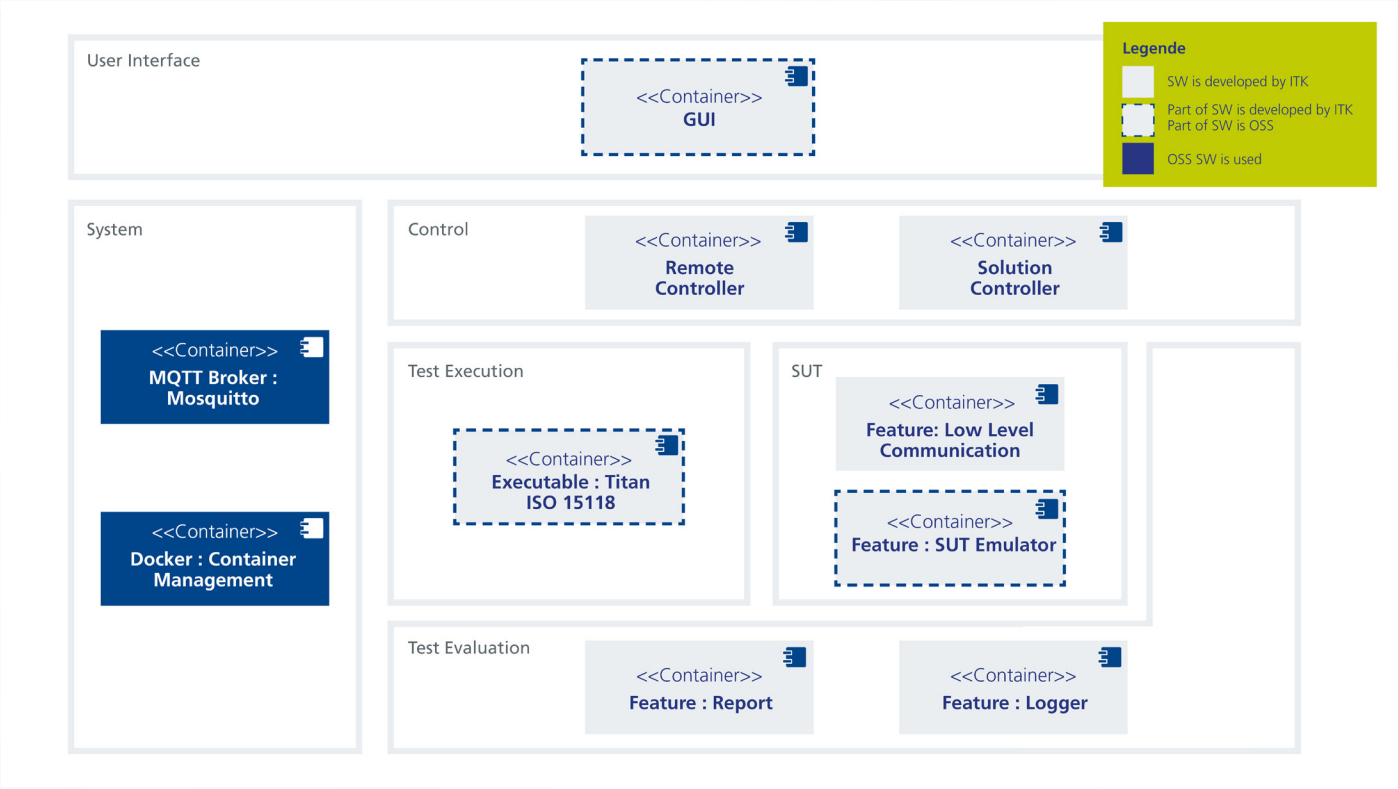

Abbildung 6 Software Layer

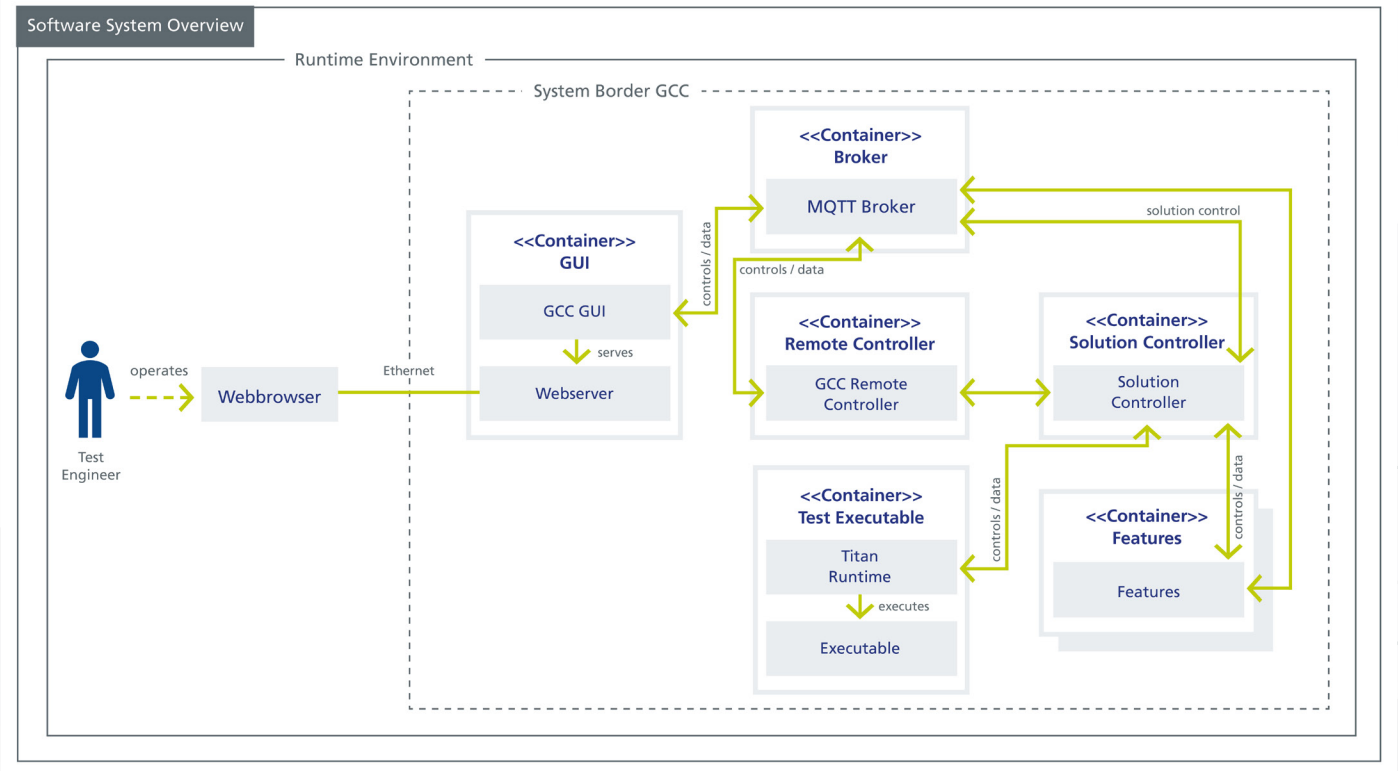

Abbildung 7 Container Abhängigkeiten

Die Funktionalitäten der Softwarekomponenten sind weitestgehend inhaltlich getrennt:

- GUI: Die GUI (als Web-Applikation) liefert dem Nutzer die Möglichkeit der Vorbereitung, des Aufsetzens und der Ausführung der Konformitätstests
- Remote Controller: Der Remote Controller ist die wichtigste Komponente des Softwaresystems und steuert alle anderen Komponenten. Er verbindet die Benutzeroberfläche mit der eigentlichen Softwarelösung und liefert alle notwendigen Informationen.
- Solution Controller: Der Solution Controller kontrolliert alle Softwarekomponenten, die für die eigentliche Softwarelösung genutzt werden und ist insbesondre für das Pre- und Postprocessing dieser zuständig.
- Test Executable: Das Test Executable enthält alle notwendigen Funktionalitäten für die Testausführung und steuert diese.
- Featues:
  - Low Level: Die Low Level Kommunikation stellt den IEC 61851-1 Simulator zur Verfügung. Dabei handelt es sich um die Virtualisierung des Kabels und dessen Eigenschaften. Der Charge Enabler stellt eine API zur Verfügung, so dass das System Under Test dieses „Kabel" ansprechen kann.
  - SUT Emulator: Der SUT Emulator beinhaltet das „System Under Test", emuliert durch das Rise-V2G, einer zertifizierten Softwarekomponente, welche konform zum Kommunikationsprotokoll aus der ISO 15118 ist (sowohl für SECC, als auch für EVCC)
  - Report Generator: Der Report Generator stellt Informationen der Testausführung im JUnit xml-Format zusammen. Der Report beinhaltet Informationen wie die Liste der ausgeführten und abgebrochenen Testfälle, das Testergebnis der einzelnen Testfälle, die Dauer der Ausführung und potenzielle aufgetretene Fehler in der Testfallausführung.
  - Logger: Der Logger stellt Informationen der einzelnen Softwarekomponenten während der Testausführung der eigentlichen Softwarelösung zusammen und legt diese als separate Datei zusammen mit den Testergebnissen ab.
- MQTT Blocker: Der MQTT-Broker ist die Basis der Kommunikation über MQTT.

Die einzelnen Softwarekomponenten sind in Docker Containern organisiert und kommunizieren über MQTT. Diese Technologien und deren Besonderheiten werden im Weiteren genauer beschrieben.

Docker Container

Docker Container eignen sich in erster Linie für die Isolierung von Softwareanwendungen. Ein Docker- Image bildet die Basis für die Container in Form eines Festplattenabbildes. Der eigentliche Code und alle Abhängigkeiten werden hier verpackt und ermöglichen eine schnelle und zuverlässige Übertragung auf verschiedene Umgebungen. Somit ist die containerisierte Software unabhängig von der genutzten Infrastruktur und kann unter beliebigen Betriebssystemen wie Linux oder Windows genutzt werden. Für die Verwendung der Docker Container im Charge Enabler Kontext ist die Plattformunabhängigkeit nicht direkt gegeben. Docker unterstützt unter Windows aktuell das Internet Protocol Version 6 IPv6 noch nicht, welche nach Vorgaben der ISO 15118 für die direkte Kommunikation zwischen dem SUT und dem Test Executable verwendet wird. Diese Unabhängigkeit kann jedoch über den Umweg einer Virtuellen Linux Maschine unter Windows hergestellt werden.

Der Charge Enabler weist die folgenden Abhängigkeiten der Container auf:

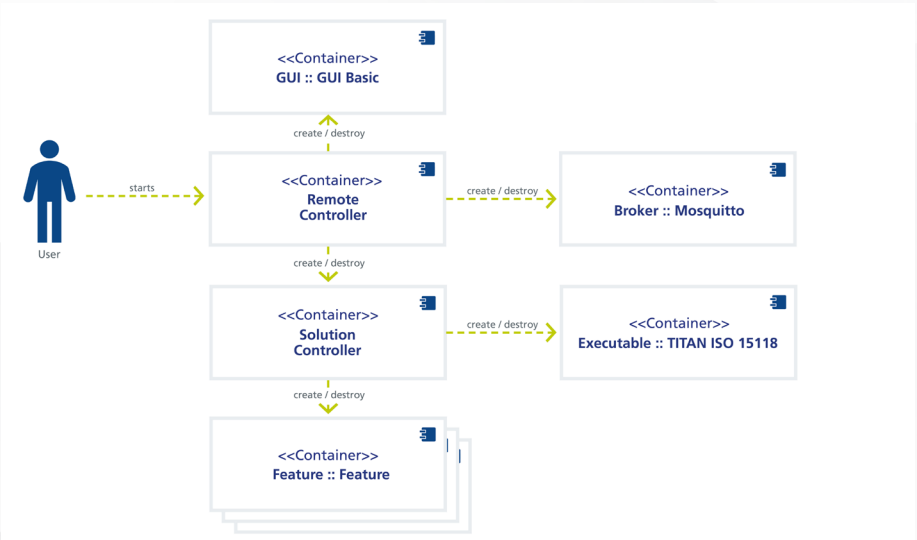

Abbildung 8 Container Abhängigkeiten

Auffällig ist, dass der Remote Controller die einzige Komponente ist, welche eine Interaktion zwischen dem Nutzer und der eigentlichen Software steuert. Außerdem wird deutlich, dass der Solution Controller wiederum die Steuerung der eigentlichen Softwarelösung übernimmt. Durch diese unabhängige Architektur werden Wartungsaufwände geringgehalten.

MQTT

Das Message Queuing Telemetry Transport (MQTT) ist ein offenes Netzwerk für Machine-to-Machine-Kommunikation (M2M), basierend auf einem sehr simplen Client-Server-Protokoll. MQTT ist besonders geeignet für Systeme mit geringer Bandbreite oder unzuverlässigen Netzwerken, da das Protokoll die Ressourcenauslastung verringert und eine stabile Nachrichtenübertragung sicherstellt. Dies sichert die Versorgung im Allgemeinen und ermöglicht die stabile Nachrichtenübertragung zwischen Geräten. Der Server, auch MQTT-Broker genannt, spielt hierbei eine Schlüsselrolle. Die gesamte Kommunikation aller Clients (Kommunikationspartner) verläuft über diesen Broker. Die Clients können im Vorfeld sogenannte Topics abonnieren und der Broker wird die veröffentlichten Nachrichten nur an die tatsächlichen Abonnenten senden.

Die Kommunikation des Charge Enablers über MQTT entspricht der folgenden Struktur:

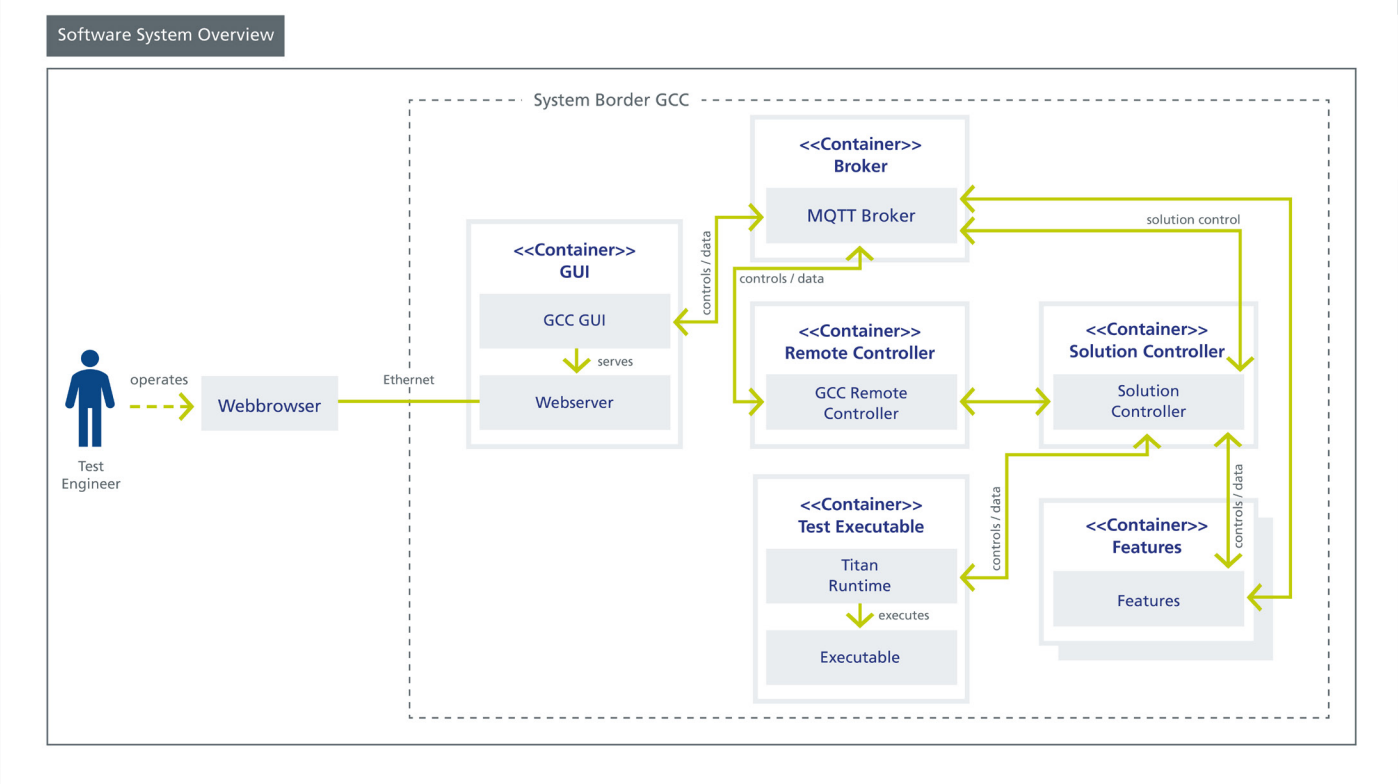

Abbildung 9 MQTT Kommunikation

MQTT gilt nicht als besonders sicher, da die Publisher und Subscriber in der Regel unverschlüsselt und ohne Authentifizierung kommunizieren. So ist es potenziellen Angreifern möglich, nicht nur mitzulesen, sondern auch Sequenzen zu manipulieren. Mit der richtigen Konfiguration des Brokers unter Verwendung von Authentifizierungen via Benutzername und Passwort und der Verwendung einer TLS Verschlüsselung, kann dieses Sicherheitsproblem umgangen werden.

Im Charge Enabler wird der MQTT Datenstrom jedoch nicht verschlüsselt, da die Software auf einem lokalen Rechner, im VPN Netz oder hinter einem Proxy mit Verschlüsselung betrieben wird. Somit ist eine zusätzliche Absicherung der MQTT Kommunikation nicht nötig.

Rise-V2G

Das Rise-V2G ist Bestandteil des System under Test (SUT). Der SUT Emulator kann sowohl das Fahrzeug, als auch die Ladeinfrastruktur sein. Rise-V2G steht hierbei für "Reference Implementation Supporting the Evolution of the Vehicle-2-Grid Communication Interface ISO 15118". Es steht als ISO 15118 normkonforme Open Source Softwarekomponente zur Verfügung. Hierbei handelt es sich nicht um eine ITK-Eigenentwicklung: https://github.com/SwitchEV/RISE-V2G

Zusammenfassung

Durch den Einsatz dieser innovativen Technologien und deren Unabhängigkeiten gelingt es dem Charge Enabler, die Komplexität in der Ladekonformität beherrschbar zu machen und ermöglicht dem Nutzer eine simple Bedienung der softwareseitigen Verifikation auf Normkonformität. Auch neue kundenspezifische Anforderungen an neue Feature sind aufgrund der eingesetzten Technologien verlässlich umsetzbar.

Ausblick

Der Charge Enabler bietet als Open Source Testingframework eine Möglichkeit des frühen und kostengünstigen softwareseitigen Testens auf Normkonformität. Aufgrund seines modularen Aufbaus und der simplen Kommunikationsstruktur, können annährend beliebige Erweiterungen an die Charge Enabler Basissoftware angebunden werden.

Geplante Erweiterungen sind beispielsweise das Pro-Reporting-Feature für detaillierte Testergebnisse, eine Erweiterung der Testsuites um kundenspezifische Testfälle oder Robustheitstestfälle und auch die Verwendung weiterer Ladeprotokolle, wie beispielsweise CHAdeMO und OCPP.

Aber auch der spätere Einsatz spezifischer Kundenhardware ist durch die Architektur mit wenigen Modifikationen umsetzbar, sodass das Tool auch nach dem Softwaretesten im HiL (Hardware-in-the-Loop) Bereich weiterverwendet werden kann. So können identische Testfälle in verschiedenen Testumgebungen reproduzierbar ausgeführt und analysiert werden. Wechselwirkungen zwischen Software und Hardware können hierdurch mit minimiertem Aufwand analysiert und verstanden werden.

Mit der internen Weiterentwicklung der kostenlosen Basisversion des Charge Enablers werden sukzessive neue kostenpflichtige Pro Features entwickelt. Diese Feature können je nach Anwendungsfall beliebig in die Charge Enabler Basis integriert werden. Somit entsteht ein auf die Bedürfnisse des Kunden abgestimmtes Testsystem. Da alle Feature unter der Anforderung der Rückwärtskompatibilität entwickelt werden, stellt auch ein späteres Nachrüsten und Aufstocken des Basistools keine größere Herausforderung dar.

Kontaktieren Sie uns gerne! Jederzeit stehen wir bei Fragen beratend zur Seite und unterstützen bei einer maßgeschneiderten Lösung für Ihre komplexen Probleme! Zusätzlich bieten wir Ihnen auf Anfrage individuelle Workshops und passgenaue Schulungen an.

https://www.itk-engineering.de/stories/charge-enabler/