# Microservices im Kontext Industrie 4.0

Industrie 4.0 ist eines der aktuellen Topthemen bei vielen industriellen Unternehmen. Einerseits bietet die damit verbundene Digitalisierung und Vernetzung von Prozessen viele Möglichkeiten der Effizienzsteigerung und neue Geschäftsmodelle, aber auch neue Herausforderungen, insbesondere beim Schutz der wertvollen Unternehmensdaten. Viele befürchten durch die zunehmende Auslagerung der Datenverarbeitung in eine Cloud einen Kontrollverlust über die eigenen Daten. In einem mit der Firma LCA Automation durchgeführten Projekt haben wir eine Webapplikation zur Zustandsüberwachung von Produktionsanlagen entwickelt unter dem Aspekt, dass die Datenhoheit gewährleistet werden kann. Dabei greifen wir auf eine Software-Architektur basierend auf Microservices zurück, welche einen standortunabhängigen Datenzugriff ohne Auslagerung kritischer Daten ermöglicht.

Matthias Krebs, Mark Zeman | matthias.krebs@fhnw.ch

Hauptziel von Industrie 4.0 ist es, die industrielle Produktion mit moderner Informations- und Kommunikationstechnik zu verbinden. Durch eine verstärkte Vernetzung und einem erweiterten Einbezug im ICT soll eine verbesserte Kommunikation und Kooperation zwischen Menschen, Anlagen und Produkten und damit einhergehend eine erhöhte Flexibilität und Automatisierung in der industriellen Produktion ermöglicht werden.

Im hier vorgestellten Projekt[1] liegt der Fokus auf Anlagen und Produktionssystemen. In Zusammenarbeit mit dem Institut für Automation (IA) der FHNW und der Firma LCA Automation [LCAA] haben wir eine Software entwickelt, welche industrielle Anlagen dahingehend überwacht, dass diese immer auf der maximalen Produktivitätsstufe betrieben werden können. Dabei spielt die Analyse-Software CM+ des IA eine zentrale Rolle, welche anhand der Anlagedaten (z.B. Stromverbrauch, Betriebstemperatur, Durchsatz) den Zustand bestimmt und bei Über- oder Unterschreiten gewisser Schwellwerte Warnungen und Empfehlungen zur Problemvermeidung generiert.

Solche Empfehlungen und Warnungen müssen für alle Personen einsehbar sein, welche damit arbeiten sollen. In einem Unternehmen mit verschiedenen Anlage- und Überwachungsstandorten bietet sich eine Webapplikation für diesen Zweck ideal an. In Abbildung 1 ist die von uns entwickelte Webapplikation abgebildet, welche die Daten von CM+ grafisch darstellt. Diese Applikation ermöglicht es, dass die Daten nicht nur direkt auf der Anlage selbst ausgelesen werden können, sondern auch via Internet an anderen Standorten der Firma oder bei einer Präsentation bei einem Kunden abrufbar sind. Damit dienen diese Anlageinformationen nicht nur der Anlagesteuerung, sondern können auch für Marketingzwecke eingesetzt werden.

Durch Monitoring von Anlagen können auch Daten entstehen, welche eine Firma als vertraulich einstuft. Daher muss der Datenhoheit und dem Zugriffsschutz in einem verteilten System gebührend Beachtung geschenkt werden.

**Datenhoheit**

Viele Lösungsvorschläge und bestehende Software-Pakete für Industrie 4.0 setzen auf Cloud-Dienste und vertrauen darauf, dass die Daten beim Dienstleister sicher sind. Für den Fall, dass diese Sicherheit nicht ausreichend ist, oder wenn die Daten nicht zu einem Dienstleister im Ausland gelangen dürfen, sollte es daher in einer gut gebauten Software möglich sein, seine Daten selber zu verwalten und bei sich zu behalten.

Aus diesem Grund haben wir in unserem Projekt eine Architektur gewählt, welche eine Unabhängigkeit von Cloud-Dienstanbietern ermöglicht. Dazu haben wir die Funktionalität auf mehrere Microservices verteilt, welche frei konfigurierbar sind [MicS]. Diese Microservices können selber gehostet oder z.B. von LCA bereitgestellt werden. Damit kann sichergestellt werden, dass die erhobenen Daten innerhalb der Firma bleiben.

Gleichzeitig ist die Architektur aber auch so flexibel, dass es möglich ist, den Betrieb von Services auszulagern, zum Beispiel an den Hersteller der Anlagen. Damit kann der eigene Aufwand minimiert und doch die Vorteile der Software genutzt werden.

Diese Fähigkeit der Software, mehrere Standorte separat zu verwalten, ermöglicht einem Dienstleister mehrere Mandanten mit einer einzigen Instanz der Software zu verwalten. Ebenso kann eine Firma damit den potentiell weit verteilten Standorten ermöglichen, sich selbst zu überwachen und im Hauptsitz der Firma alle Standorte gleichzeitig zu überwachen.

1 KTI-Nr. 15958.2 PFES-ES: Maximierung der Verfügbarkeit und Produktivität von Montagelinien mit zustandsbasierter Wartung, intelligentem Sensornetzwerk und mobiler Visualisierung

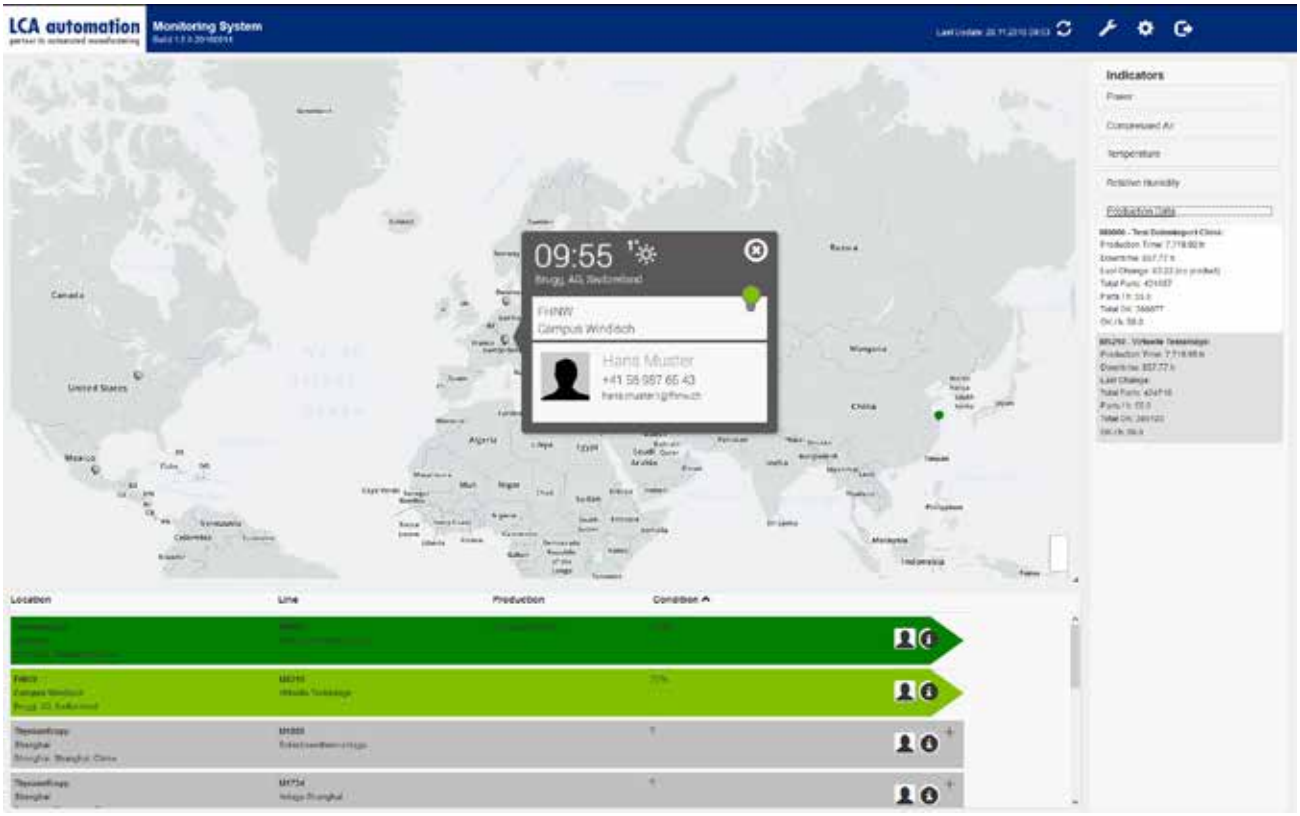


Abbildung 1: Web-App zur Überwachung des Zustands von Produktionsanlagen

**Flexible Systemarchitektur mit Microservices**

Im Gegensatz zu einer monolithischen Applikation bilden Microservices ein verteiltes System aus mehreren Services, wobei jeder individuelle Microservice eine spezifische Aufgabe übernimmt. Der Datenaustausch erfolgt über genau spezifizierte Netzwerkschnittstellen, damit die einzelnen Services möglichst gut austauschbar sind. Aufgrund der verteilten Architektur eignen sich Microservices besonders gut für Anwendungsfälle, bei denen Applikationen standortübergreifend betrieben werden.

Im Falle des Anlagen-Monitorings besteht die Systemarchitektur aus mehreren Arten von Microservices, welchen allesamt eine gemeinsame Java-Codebasis zugrunde liegt. Die Verwendung des *Spring Frameworks* [SprF] ermöglicht eine einfache Implementierung von *RESTful Webservices* [RFWS] und erlaubt mit Hilfe von Dependency Injection eine flexible Konfiguration einer Applikation. Die Erweiterung *Spring Boot* [SprB] ist bei der Entwicklung von Microservices besonders nützlich, weil sich damit ein Java-Webservice dank integriertem Applikationsserver als eigenständige ausführbare Applikation betreiben lässt.

Wir unterscheiden zwischen drei Typen von Microservices:

- *Site Data Service:* Dieser Service sammelt die Messdaten von verschiedenen Produktionsanlagen an einem oder mehreren Standorten, speichert sie in einer Datenbank und stellt die Daten wiederum als Webservice bereit. Je nach Konfiguration erfasst ein Site Data Service beispielsweise die Daten einer bestimmten Firma, oder auch eines bestimmten Firmenstandorts.
- *HQ Data Service:* Als übergeordneter Service dient ein HQ Data Service der Datenauswertung sowie der Überwachung mehrerer Site Data Services. Ein HQ kann dabei beispielsweise als Firmenzentrale (Headquarter) interpretiert werden. Mit diesem Service können standortübergreifende Daten in aggregierter Form über eine einheitliche Schnittstelle ausgegeben werden. Der HQ Data Service speichert dabei selbst keine Daten, so dass die Datenhoheit der einzelnen Standorte gewährleistet bleibt. Zudem ist auf einem HQ Data Service nur lesender Zugriff möglich.
- *Data Agent:* Da ein direkter Datenabruf von Messdaten aus CM+ aus Gründen der Netzwerksicherheit nicht erwünscht ist, wird ein Data Agent Service als Vermittler eingesetzt. Dieser wird im selben lokalen Netzwerk wie CM+ betrieben und ruft in konfigurierbaren Intervallen die aktuellsten Messdaten ab. Anschliessend werden die Messdaten an einen Site Data Service übertragen.

Abbildung 2 zeigt die Systemarchitektur anhand eines Setup mit zwei separaten Standorten

A und B, auf die über ein HQ zugegriffen werden kann. Die Data Services können standortunabhängig, also entweder on-site oder bei einem Cloud-Dienstleister betrieben werden. Lediglich die Datenerfassung und -analyse (CM+) sowie der Data Agent müssen am Standort der Produktionsanlage betrieben werden. Damit die erfassten Daten ebenfalls standortunabhängig sind, greift jeder Microservice auf eine eigene Datenbank zu.

Die Web-Apps sind unabhängige Applikationen, welche komplett im Web-Browser laufen und via REST API auf einen Site- oder HQ Data Service zugreifen. Sie können deshalb ebenfalls standortunabhängig betrieben werden und sind bei Site Data Services sogar optional, sofern ein HQ existiert.

Der Datenaustausch zwischen Clients (Web-App, Data Agent, HQ Service Backend) und Data Services, also die Übertragung und Abruf von Anlageninformationen und Messdaten, erfolgt immer via REST API. Der Austausch administrativer Nachrichten zwischen Site- und HQ Data Service erfolgt über einen separaten Kanal. Hierzu werden Message-Queues mit RabbitMQ [RbMQ] verwendet.

Die Implementierung eines Data Service ist in mehreren Schichten aufgebaut, wie in Abbildung 3 am Beispiel der Anlageninformationen („Lines“) gezeigt wird. Die oberste Schicht ist die Controller-Schicht, welche den RESTful Webservice bereitstellt. Sie dient ausschliesslich dem Datenaustausch und der Umwandlung der übertragenen JSON-Daten. Die eigentliche Geschäftslogik wird durch ein Service Interface (Bsp.: *LineService*) abgebildet, für das zwei Implementierungen existieren. Die Standard-Implementierung für einen Site

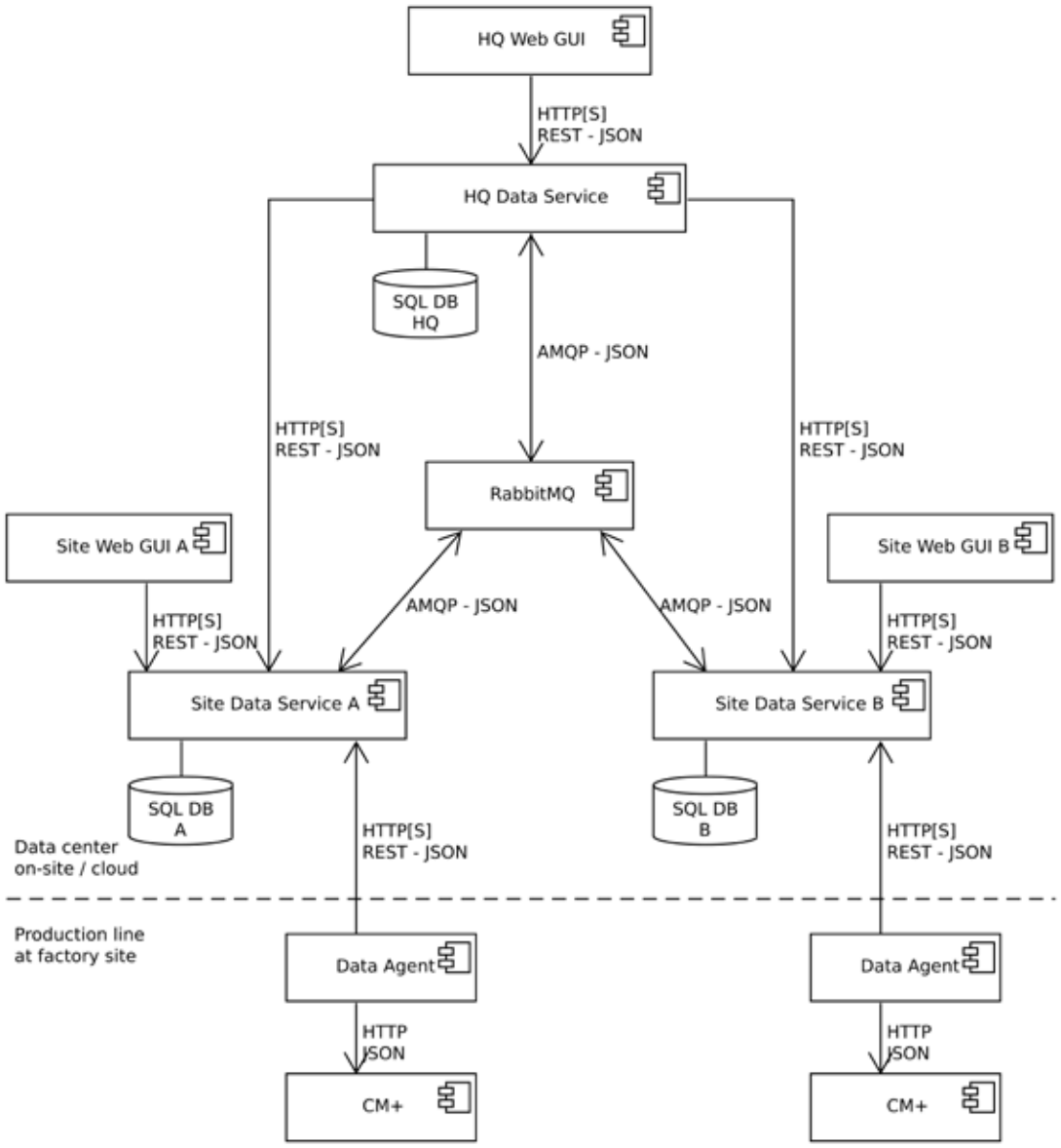


Abbildung 2: Systemarchitektur mit zwei Standorten und Headquarter HQ

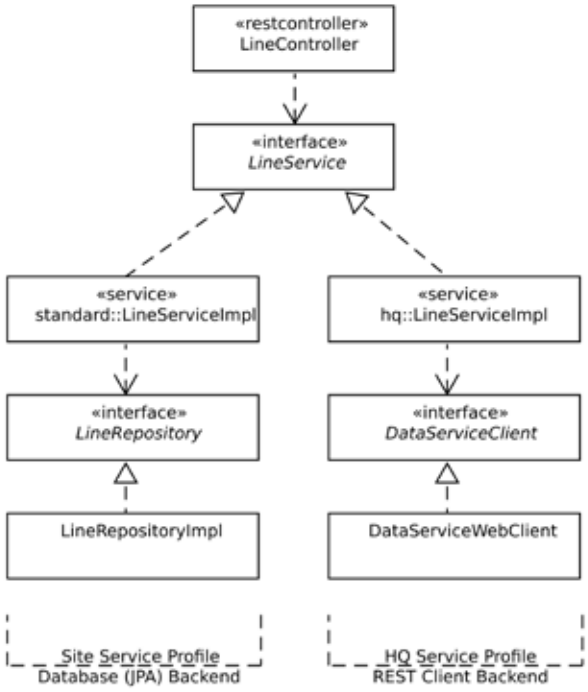


Abbildung 3: Service zum Abruf von Anlageninformation mit zwei Implementierungen

Data Service enthält Methoden zur Datenverarbeitung und bindet via JPA eine SQL-Datenbank an, um die verarbeiteten Daten zu speichern. Die Implementierung für einen HQ Data Service speichert keine Daten, sondern enthält als Backend einen Webservice Client, welcher Daten von einem entfernten Site Data Service abfragt. Welche Implementierung verwendet werden soll, wird durch Angabe des korrekten Spring-Profils in der Applikationskonfiguration festgelegt. Wird das Profil „hq" angegeben, wird die HQ-Implementierung verwendet, ansonsten die Standard-Implementierung.

**Einheitliche Schnittstelle**

Die Übertragung von Messdaten erfolgt via *RESTful Webservices* [RFWS]. Jeder Microservice implementiert also ein REST API als Client, Server oder im Falle des HQ Data Service auch beides. Als Datenübertragungsformat wird JSON verwendet.

Weil der Site und der HQ Data Service die gleiche Codebasis besitzen und sich nur durch ihre Backend-Implementierung unterscheiden, welche via Spring-Profil konfiguriert werden kann, ist das REST API für beide Services quasi identisch. Dies vereinfacht die Entwicklung der Web-App, welche zur Betrachtung der Monitoring-Daten verwendet wird. Es spielt für die Web-App keine Rolle, ob sie auf einen Site- oder einen HQ Data Service zugreift, deshalb muss sie nur einmal implementiert werden.

In einem einfachen Setup mit nur einem Standort kann ganz auf einen HQ Data Service verzichtet werden, indem die Web-App direkt auf den Site Data Service zugreift.

**Sichere Datenübertragung**

Mit dem standortübergreifenden Datenaustausch und der weltweit möglichen Datenabfrage via Web-App bekommt die Sicherheit der Datenübertragung eine entscheidende Bedeutung. Es kann davon ausgegangen werden, dass die Kommunikation im Wesentlichen über öffentliche Internetverbindungen abgewickelt wird. Deshalb müssen unautorisierte Zugriffe auf die Daten verhindert werden und die Kommunikation muss verschlüsselt erfolgen.

Beim öffentlichen, respektive standortübergreifenden, Zugriff auf Site oder HQ Data Services wird konsequent HTTPS eingesetzt. Auf diese Weise wird verhindert, dass Daten unterwegs mitgeschnitten werden können. Jeder Client, sei es ein User der Web-App oder ein Data Agent, muss sich zudem authentifizieren, um auf einen der Webservices zugreifen zu können.

In Bezug auf die Netzwerksicherheit hat die gewählte Systemarchitektur den Vorteil, dass kein direkter Zugriff auf betriebskritische Komponenten der Produktionsanlagen benötigt wird. CM+, welches die Monitoring-Daten über einen eigenen Webservice bereitstellt, ist nur im lokalen Netzwerk erreichbar. Die Übertragung von Daten zum Site Data Service wird ausschliesslich vom Data Agent angestossen, welcher auch die Verbindung zum Server initiiert. Ähnliches gilt auch für den administrativen Nachrichtenaustausch. Die Data Services greifen auf einen zentralen RabbitMQ Broker zu, wobei die Verbindung vom Data Service aufgebaut wird. Deshalb muss einzig der RabbitMQ Broker direkt durch die Services erreichbar sein. Auch bei RabbitMQ müssen sich angeschlossene Services authentifizieren und die Kommunikation kann verschlüsselt mittels Transport Layer Security (TLS) erfolgen.

**Zentrales Management**

In einem verteilten System von Microservices müssen die einzelnen Microservices einander bekanntgemacht werden, damit sie kommunizieren zu können. Es wird also eine Registrierung benötigt. Durch bestmögliche Automatisierung des Registrierungsprozesses kann der administrative Aufwand zum Nachführen der Konfiguration reduziert werden.

In unserem Fall geschieht das mit Hilfe der RabbitMQ Message-Queues. Einem Site Data Service muss lediglich die Queue des zugehörigen HQ Data Service bekanntgemacht werden, damit er sich nach dem Start selbst beim HQ Data Service registrieren kann. Auf diese Weise können einem HQ zusätzliche Standorte hinzugefügt werden,

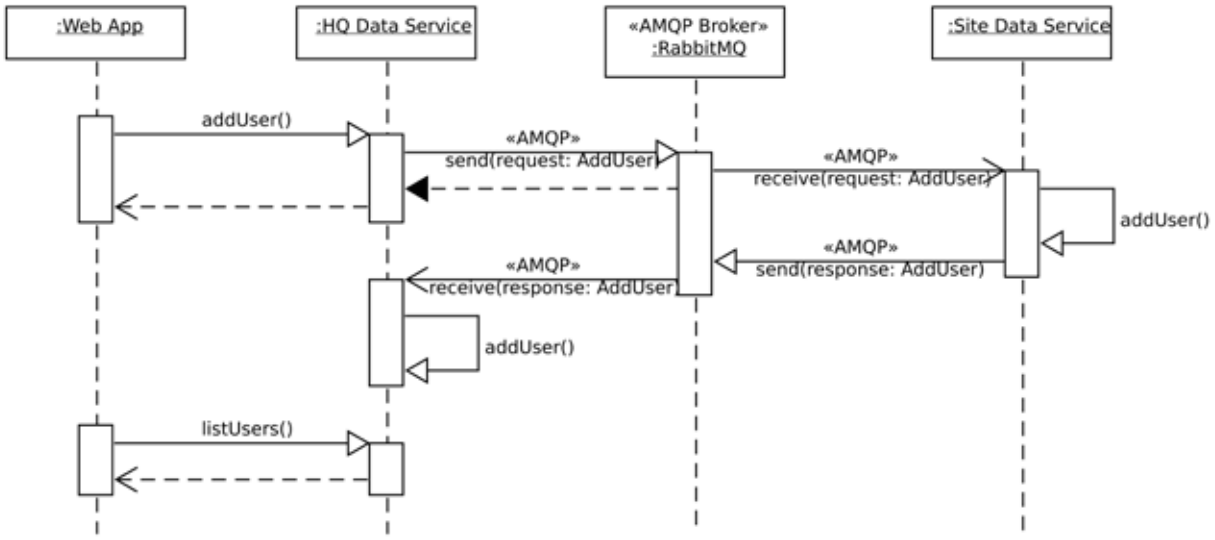


Abbildung 4: Service zum Abruf von Anlageninformation mit zwei Implementierungen

ohne dass die Konfiguration des HQ Data Service von Hand angepasst werden muss. Eine zweite Funktion der Message-Queues ist die Überwachung der Verfügbarkeit der angeschlossenen Standorte. In regelmässigen Abständen prüft der HQ Data Service mit einer „Ping"-Meldung, ob die Standorte noch aktiv sind. Bleibt eine Antwort aus, kann ein Standort als inaktiv markiert werden.

Auch die Verwaltung der Benutzerkonten für den Web-Zugriff kann zentral erfolgen. Hierfür bietet die Web-App einen Administrationsbereich zur Benutzerverwaltung an. Wird ein HQ Data Service administriert, können auf diesem auch Benutzerkonten angeschlossener Site Data Services erstellt und bearbeitet werden. Änderungen werden, wie in Abbildung 4 gezeigt, via RabbitMQ-Message-Queue an den entsprechenden Service propagiert und im HQ erst nach erfolgreicher Verbreitung gespeichert.

Trotz zentralem Management arbeitet ein Site Data Service autonom, denn die für die Benutzer-Authentifizierung nötigen Informationen werden auch lokal gespeichert. Sollte also z.B. der HQ Data Service ausfallen, können am Standort weiterhin Daten von den Produktionsanlagen erfasst werden.

**Fazit**

Die hier gezeigte Systemarchitektur mit Microservices ermöglicht einen standortübergreifenden und bei Bedarf weltweiten Zugriff auf Daten, welche gerade im Kontext der Industrie 4.0 zentrale Aspekte sind. Gleichzeitig wird aber auch dem Bedürfnis nach Datensicherheit Rechnung getragen, da die Betreiber der von LCA hergestellten Produktionsanlagen die Hoheit über ihre betriebsrelevanten Daten nicht durch eine Auslagerung des Betriebs der Dienste abgeben müssen. Sowohl der Anlagenbauer LCA als auch die Anlagenbetreiber können letztendlich von dieser Lösung profitieren, da sie für den Anlagenbauer eine kostengünstigere Wartung und für den Betreiber eine Reduktion der Stillstandzeiten und damit unmittelbare finanzielle Anreize ermöglicht.

**Referenzen**


[LCAA] LCA Automation: https://www.lca.ch/

[MicS] Definition Microservices: http://www.martinfowler.com/articles/microservices.html

[SprF] Spring Framework: https://spring.io/

[SprB] Spring Boot: https://projects.spring.io/spring-boot/

[RFWS] Definition RESTful Web Services: https://de.wikipedia.org/wiki/Representational_State_Transfer

[RbMQ] RabbitMQ: https://www.rabbitmq.com/