# Qualitätsverbesserungen durch einen Plattform-Mock am Beispiel EnStadt:Pfaff

Frank Elberzhager, Phil Stüpfert
Fraunhofer IESE, Fraunhofer-Platz 1, 67663 Kaiserslautern
{frank.elberzhager, phil.stuepfert}@iese.fraunhofer.de

**Zusammenfassung**:
Im Rahmen des Leuchtturmprojekts EnStadt:Pfaff, welches zum Ziel hat, ein smartes Stadtquartier zu entwickeln, in dem die Klimaneutralität adressiert wird, sollen digitale Technologien und Lösungen dabei unterstützen. Der Kern ist eine Quartiersplattform mit darauf laufenden Diensten, welche Bürgern, Unternehmen oder auch die öffentliche Hand unterstützen. Im Rahmen des Forschungsprojekts entstehen viele Ideen für solche digitalen Dienste – ein schnelles Ausprobieren ist aber oftmals schwierig. Deshalb haben wir einen Plattform-Mock als Zwischenlösung entwickelt, der das vorrangige Ziel verfolgt, frühe Versionen solcher Dienste (MVPs) erproben zu können. Rückmeldungen von frühen Nutzern gehen dann in die Weiterentwicklung des MVPs hin zum produktiven Dienst ein, in dem dann auch bisher unberücksichtigte Qualitäten zu adressieren sind. Der Plattform-Mock selber soll dazu unterschiedliche Assistenzsysteme bereitstellen, welche dem Entwickler zusätzliche Rückmeldungen geben können, beispielsweise zur Codequalität, zur UI oder auch weiteren Qualitäten.

**Schlüsselworte:** Qualitätssicherung, Nutzerfeedback, Plattform, Stadtquartier

## 1. Einleitung und Motivation

Die Software Qualitätssicherung ist ein wichtiger Bestandteil der Softwareentwicklung. Aus diesem Grund gibt es bereits sehr lange eine Vielzahl an standardisierten Test- und Reviewverfahren, um ein hohes Maß an Softwarequalität sicherzustellen. Bei der Entwicklung von Diensten für stark vernetzte Ökosysteme kann die Qualitätssicherung eine große Herausforderung darstellen, da ganz neue Anforderungen wie eine erhöhte Vernetzung, verschiedenste Stakeholder oder eine schwierigere Integration eine Rolle spielen. Damit Qualitätsprobleme früh auffallen, stellen wir einen Plattform-Mock vor, welcher dabei unterstützen kann, frühe Versionen von vernetzten Diensten für ein Software-Ökosystem schneller testen zu können und somit frühzeitig Probleme zu erkennen.

## 2. Hintergrund und Herausforderung

Im Zuge des Forschungsprojekts EnStadt:Pfaff [1] entwickeln wir eine Quartiersplattform, welche dabei helfen soll ein klimaneutrales Stadtquartier zu schaffen. Dabei ist diese Plattform Teil eines Softwareökosystems, in dem auch verschiedene Diensten laufen, z.B. als Apps auf den Smartphones der Bewohner. Diese Dienste sind so über die Plattform miteinander vernetzt, dass sie durch gemeinsame Kommunikation einen Mehrwert für den Nutzenden generieren können.

Möchte man neue Konzeptideen in diesem Ökosystem testen, gibt es jedoch meist nur die Möglichkeit, dies in der Produktivumgebung (oder eben einer aufwendig erstellten Testumgebung) zu tun. Das Andocken dieser Prototypen an die Plattform bedarf eines hohen Implementierungsaufwands. Dieser erhöhte Aufwand stellte für meist eine große Hürde dar, sodass manche Ideen erst gar nicht ausprobiert und weiterverfolgt werden konnten.

Um diese Hürde abzuschwächen, entwickelten wir den Plattform Mock. Diese stark vereinfachte Plattform dient uns als eine Art „Spielwiese“, welche jedoch weiterhin die Grundfunktionalität der Vernetzung bieten. Dabei verzichtet sie jedoch auf Eigenschaften wie z.B. Datenschutzmaßnamen oder Sicherheitsmechanismen. Dadurch entsteht kein unnötiger Mehraufwand bei der Implementierung eines Prototypens, und es können einfach neue Ideen getestet werden.

## 3. Der Plattform Mock für ein schnelleres Ausprobieren

Der Plattform Mock besteht im Wesentlichen aus zwei Teilen. Im Kern befindet sich ein MQTT-Broker, welcher als asynchrone Kommunikationszentrale dient. Über diese können alle Dienste (auch Prototypen) Event-Nachrichten veröffentlichen. Da das Kommunikationsmuster des MQTT-Protokolls dem Publish-Subscribe Pattern folgt, können Nachrichten nicht an spezifische Empfänger adressiert werden, sondern lediglich auf bestimmten Topics veröffentlich werden. Durch aktives Abonnieren eines Topics durch einen Dienst, werden diese Nachrichten dann an diesen Dienst weitergeleitet.

Um dem Sender und Empfänger die Kommunikation zu vereinfachen, haben wir sogenannte „Shared-Topics“

eingeführt. Diese sind spezielle Topics, welche genau vorgeben, wie und in welcher Form eine Information auf diesen Topics veröffentlicht werden muss. Ein Beispiel ist das „Smart-Home-Topic“, über welches Raumtemperaturen oder Stromverbräuche veröffentlicht werden können. Diese Regelung macht es gerade für Empfänger einfach, da diese immer wissen, wie sie die empfangenen Daten interpretieren müssen. Darüber hinaus ermöglicht dieses Konzept auch eine verbesserte Erweiterbarkeit der Dienste, da Empfänger auch mit Nachrichten von z.B. neuen Prototypen umgehen können.

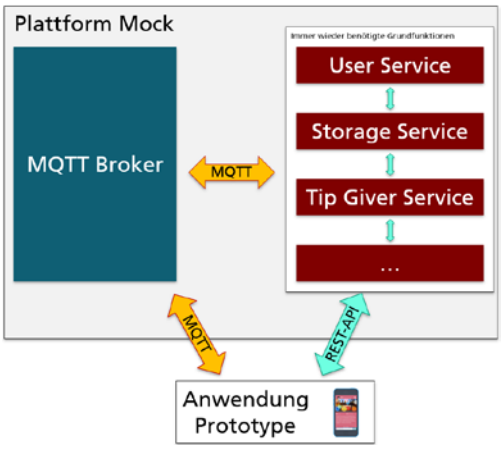


**Abbildung 1: Konzept des Plattform Mocks**

Zweiter Bestandteil des Plattform Mocks ist eine Sammlung von verschiedenen Diensten, welche bei der Prototypentwicklung immer wieder benötigte Grundfunktionalitäten zur Verfügung stellt. So können z.B. mit dem User-Service plattformweite Benutzer anlegen und auf dem Storage Service zentral Daten abgelegt werden. Durch diese Bereitstellung kann die Prototypentwicklung nochmals beschleunigt werden. Da die Verwendung der Dienste eine synchrone Kommunikation voraussetzt, wird diese durch eine Rest-API realisiert. Dennoch können diese Dienste Event-Nachrichten über den MQTT-Broker veröffentlichen.

## 4. Frühe Rückmeldung

Der Plattform Mock unterstützt eine schnellere und effizientere Entwicklung von frühen, aber lauffähigen Prototypen (MVP) im Plattformkontext. Dadurch können Entwickler sehr schnell Feedback über viele technische Eigenschaften des neuen Dienstes erproben. Dazu zählen Erfahrungen über die Integration in das Ökosystem, Funktionalitäten und Qualitätsverbesserungsmöglichkeiten, noch bevor sie mit der eigentlichen Implementierung der finalen Anwendung beginnen. Darüber hinaus, eröffnet sich die Möglichkeit der frühen Rückmeldung über die Idee durch Testnutzer, sprich: Diese können explizit eine Meinung äußern, ob der Dienst sinnvoll ist, und was für Features noch gewünscht werden.

Weiterhin können aber ebenfalls konkrete Testmethoden eingesetzt werden, von explorativen Tests des Dienstes auf der Plattform über strukturierte Integrations- und funktionale Tests. Vorstellbar ist auch eine Vielzahl von automatisierten Prüfungen, welche die Plattform anbietet. Das können Lasttests sein, oder auch Bots, die den Dienst absichtlich stören und prüfen, wie robust dieser ausgelegt ist.

Alle diese Informationen können dann an die Entwickler des neuen Dienstes zurückgespielt werden und dabei helfen, den Dienst zu verbessern. Der große Vorteil des Plattform-Mocks dabei ist, dass eben der Entwickler nicht selber eine entsprechende Umgebung aufsetzen oder „mocken“ muss, sondern dies vom Plattform-Mock bereitgestellt wird.

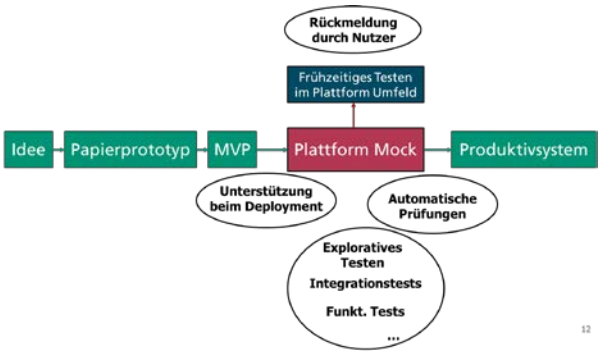


**Abbildung 2: Einsatz des Plattform-Mocks in der Qualitätssicherung**

## 5. Fazit und Ausblick

Ökosysteme und vernetzte Dienste spielen eine zunehmend wichtige Rolle, was auch die Qualitätssicherung vor neue Herausforderungen stellt. Um neue Dienste in einem Softwareökosystem schnell erproben zu können, haben wir einen Plattform-Mock erstellt, der Entwickler auf unterschiedliche Weise unterstützen kann. Im Rahmen eines Forschungsprojekts wurde dieser bereits erfolgreich eingesetzt, z.B. bei Hackthons, und wird kontinuierlich weiterentwickelt.

**Danksagung**

Dieser Beitrag wurde im Kontext des Projekts EnStadt:Pfaff (Förderkennzeichen: 03SBE112D und 03SBE112G) erstellt, gefördert vom Bundesministerium für Wirtschaft und Energie (BMWi) sowie vom Bundesministerium für Bildung und Forschung (BMBF).

**Referenzen**

[1] https://pfaff-reallabor.de/