Innovation Hub 13
fast track to transfer

Testbed-Serverarchitektur des Innovation Hub 13

SERVERKONFIGURATION IM TESTBED

Das Testbed-System besteht im Wesentlichen aus zwei Servern, dem **Entwicklungssystem und dem Produktivsystem**. Die Serverkonfiguration fällt minimal aus und wird automatisert konstant gehalten. Probleme durch Unterschiede zwischen Entwicklungs-, Produktiv- und lokaler Testumgebung werden dadurch größtenteils vermieden.

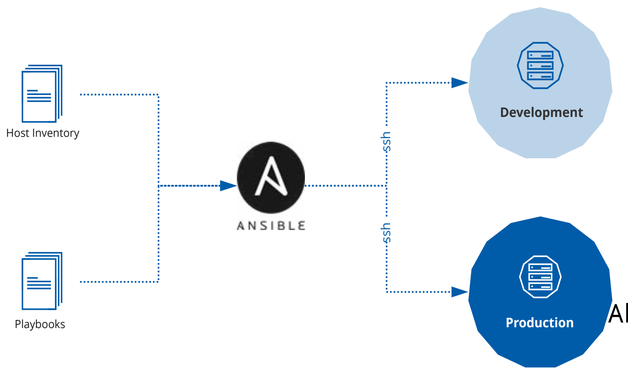

Abb. 1

Zur automatisierten Übertragung etwaiger Änderungen auf beide Systeme und zu Dokumentationszwecken werden **Ansible Playbooks** (Abb. 1) genutzt. Somit wird sichergestellt, dass beide Server die exakt gleichen Einstellungen haben. Diese Einstellungen setzen sich aus grundlegenden Sicherheitsfunktionen zusammen, sowie der für den Betrieb benötigten Software.

Grundsätzliche Sicherheitskonfiguration
- Nutzerzugang über SSH nur mit „Public Key Authenticaton“ möglich
- Firewallkonfiguration
- automatisierte Updates

Applikationsspezifische Konfiguration
- Installation von Docker inkl. docker-compose
- Hochschulspezifische Konfiguration des Systems und von Docker
- Konfiguration der Log-Dateien
- Monitoring-Schnittstelle

MICROSERVICES-ARCHITEKTUR

Im Hintergrund der Testbed-Plattformentwicklung ist die Infrastruktur sämtlicher Software-Produkte modernisiert und als „Microservices-Architektur“ (Abb. 2) angelegt. Dadurch lassen sich einzelne digitale Produkte einfacher verwalten, dokumentieren und in unterschiedliche Bereiche integrieren.

Diese einheitliche, übersichtliche und moderne Art der Infrastruktur bildet u.a. auch die Grundlage für die Nutzung der Projektergebnisse über den Projektzeitraum hinaus. Die Verwaltung ist ohne tiefgehende inhaltliche Kenntnis der Produkte möglich.

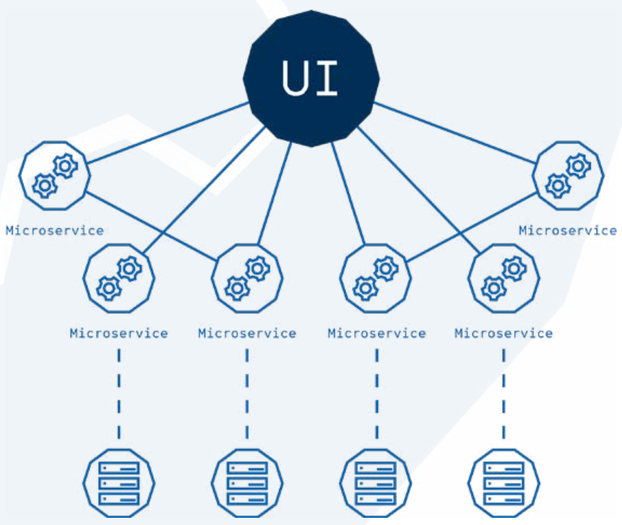

Abb. 2

DevOps-STRATEGIE

Der dauerhafte Betrieb und die kontinuierliche Weiterentwicklung einer Vielzahl von zum Teil äußerst komplexen Systemen in der Testbed-Umgebung bringt verschiedene Herausforderungen mit sich. Um diesen entgegenzukommen, sind unterschiedliche Methoden/Systeme aus dem DevOps-Bereich adaptiert.
DevOps **minimiert Grenzen zwischen Entwicklern (Development) und Betreibern (Operations)**. Methoden aus dem Entwicklungsbereich werden genutzt, um den Betrieb und die Installation der Applikation zu vereinfachen (bspw. automatisiertes, kontinuierliches Testen). Probleme der „Ops“ können durch Anpassungen der Entwickler minimiert bzw. komplett vermieden werden.

DOCKER

Ein zentraler Bestandteil der DevOps-Strategie und auch der Testbed-Architektur ist Docker – eine offene Plattform für die Entwicklung, Auslieferung und den Betrieb von Applikationen. Die wichtigste Komponente von Docker ist der Container, eine leichtgewichtige isolierte Laufzeitumgebung für die **Bereitstellung von Applikationen**. Durch die Nutzung von Containern ist es möglich, eine Vielzahl von Applikationen auf einem Host laufen zu lassen (Abb. 3), ohne dass sich die verschiedenen Laufzeitumgebungen behindern. Container können sehr einfach weitergegeben werden und problemlos auf jedem System eingesetzt werden.

Durch Kombination von **Dockerfiles und docker-compose.yml**-Dateien können selbst komplexe Applikationen problemlos in verschiedenen Kontexten genutzt werden (Hochschulbibliotheken, Showroom, Testbed).

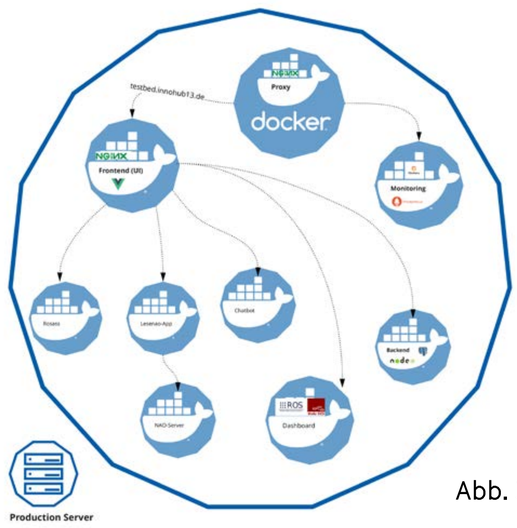

Abb. 3

GIT

Die Kommandozeilensoftware „git“ bildet den eigentlichen Kern des Versionierungstools, mit dem sich Änderungen an Dateien aufzeichnen, darstellen und wiederherstellen lassen. Für alle regelmäßig weiterentwickelte Projekte sind **CI/CD Pipelines** erstellt, um die sonst manuell durchzuführenden Schritte des Testens und Deployments automatisiert im Hintergrund erledigen. Durch den Einsatz von git-Automatisierungsmechanismen reduzieren sich viele Arbeitsschritte. Integrierte Applikationen müssen kaum noch manuell gesteuert werden.

MONITORING

Die Überwachung von Servern, Applikationen und Prozessen ist essentiell für jede Art von Test- oder Produktivsystem. Für **frühzeitiges Erkennen und Lokalisieren von Fehlern** wird im Testbed ein Monitoring-System auf Basis von Grafana und Prometheus genutzt. **Prometheus** ist ein „Service Monitoring“-System, welches Daten aus unterschiedlichen Quellen verarbeitet und in einer Zeitreihendatenbank abspeichert. **Grafana**, eine plattformübergreifende Open-Source-Anwendung zur grafischen Darstellung von Daten, wird dann genutzt, um diese Daten für Menschen anschaulich zu visualisieren und bei Überschreitung von frei definierbaren Grenzwerten zu benachrichtigen.

EINE GEMEINSAME INITIATIVE VON
Innovative Hochschule
Bundesministerium für Bildung und Forschung
Gemeinsame Wissenschaftskonferenz GWK

Der „Innovation Hub 13 – fast track to transfer“ der Technischen Hochschule Wildau und der Brandenburgischen Technischen Universität Cottbus-Senftenberg gehört zu den 29 ausgewählten Gewinnern der Bund-Länder-Förderinitiative „Innovative Hochschule”, ausgestattet mit Mitteln des Bundesministeriums für Bildung und Forschung BMBF und des Landes Brandenburg.

Homepage: https://innohub13.de

Innovation Hub13:
Prof. Janett Mohnke
Fabian Firlay
Henning Wiechers

Tel. +49 (0) 3375 508 435
fabian.firlay@th-wildau.de