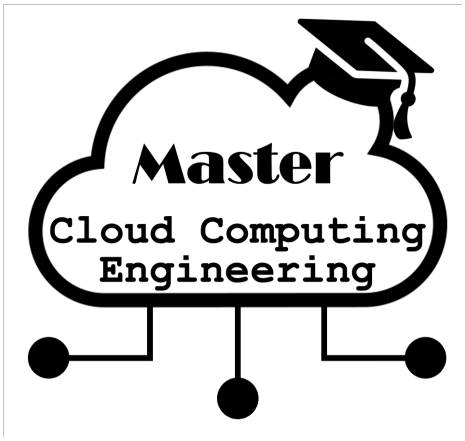

Aufgabenstellung 1 Taucher

Harald Beier* Susanne Peer† Patrick Prugger‡
Philipp Palatin§

23. April 2025

*2410781028@hochschule-burgenland.at
†2410781002@hochschule-burgenland.at
‡2410781029@hochschule-burgenland.at
§2310781027@hochschule-burgenland.at

Inhaltsverzeichnis

1 Aufgabenstellung 1 Taucher

1.1 Teilbereich Build and Code

1.1.1 Vorgehensmodelle

Fragestellung: Welche bereits vorgestellten bzw. zusätzlich recherchierte Vorgehensmodelle ermöglichen ein schnelles Iterieren und somit die Möglichkeit Feedback zeitnah durch die jeweiligen Stakeholder einzubringen? Zusätzlich soll auch darauf eingegangen werden, weshalb die gewählten Vorgehensmodelle dies im Vergleich zu anderen Modellen ermöglichen.

Scrum

Extreme Programming

Test-Driven Development

Kanban

1.1.2 Extreme Programming

Fragestellung: Beschreiben Sie mindestens 3 Praktiken aus Extreme Programming im Detail und den Einfluss die diese Praktik auf die Softwareentwicklung und dem Ergebnis haben.

Pair Programming Beim Pair Programming arbeiten zwei Entwickler gemeinsam an einem Computer. Eine Person (der "Driver") schreibt den Code, während die andere Person (der "Navigator") den Code überprüft, Probleme identifiziert und strategische Entscheidungen trifft. Die Rollen werden regelmäßig gewechselt.

Einfluss auf die Softwareentwicklung:

- **Verbesserte Codequalität:** Durch kontinuierliches Review werden Fehler früher erkannt und behoben.
- **Wissenstransfer:** Entwickler lernen voneinander, was zu einer breiteren Verteilung von Wissen im Team führt.
- **Bessere Lösungsansätze:** Durch die Kombination verschiedener Perspektiven entstehen oft kreativere und effizientere Lösungen.
- **Reduzierte technische Schulden:** Die kontinuierliche Überprüfung verhindert Abkürzungen und schlechte Praktiken.

Continuous Integration (CI) Bei der Continuous Integration werden Codeänderungen mehrmals täglich in ein gemeinsames Repository integriert. Nach jeder Integration werden automatisierte Tests durchgeführt, um sicherzustellen, dass die Änderungen keine Fehler verursachen.

Einfluss auf die Softwareentwicklung:

- **Frühe Fehlererkennung:** Probleme werden unmittelbar nach ihrer Entstehung identifiziert, was die Behebungskosten drastisch reduziert.
- **Reduzierte Integrationszeit:** Durch häufige kleine Integrationen werden große, problematische Merges vermieden.
- **Höhere Softwarestabilität:** Die Software bleibt kontinuierlich in einem funktionsfähigen Zustand.
- **Schnelleres Feedback:** Entwickler erhalten unmittelbare Rückmeldung zu ihren Änderungen.

Test-Driven Development (TDD) Bei TDD werden Tests geschrieben, bevor der eigentliche Code implementiert wird. Der Entwicklungsprozess folgt einem "Red-Green-RefactorZyklus: Zuerst wird ein fehlschlagender Test geschrieben (Red), dann wird gerade genug Code implementiert, um den Test zu bestehen (Green), und schließlich wird der Code verbessert, ohne seine Funktionalität zu ändern (Refactor).

Einfluss auf die Softwareentwicklung:

- **Klarere Anforderungen:** Das Schreiben von Tests zwingt Entwickler, die Anforderungen genau zu verstehen.
- **Besseres Design:** TDD fördert modularen, entkoppelten Code, der leichter zu testen ist.
- **Umfassende Testabdeckung:** Jede Funktionalität wird durch Tests abgedeckt, was zu robusterer Software führt.
- **Dokumentation durch Tests:** Tests dienen als lebende Dokumentation, die zeigt, wie der Code verwendet werden soll.
- **Sicherheit bei Refactoring:** Entwickler können Code mit Vertrauen umgestalten, da Tests Regressionen aufdecken.

1.1.3 Zusammenfassung von Artikel Spotify Scaling

Aufgabenstellung:Lesen Sie den Artikel https://blog.crisp.se/wp-content/uploads/2012/11/SpotifyScaling.pdf. Diese Informationen fassen Sie bitte in 1 bis maximal 2 Seiten zusammen

1.1.4 Git Features

Fragestellung: Beschreiben Sie Git im Detail sowie mindestens 2 Features im Detail.

Was ist Git? Git ist ein verteiltes Versionskontrollsystem, das 2005 von Linus Torvalds entwickelt wurde. Es ermöglicht die Verwaltung von Änderungen an Dateien und Projekten, wobei jeder Entwickler eine vollständige Kopie des Projekts und seiner Historie auf seinem lokalen System hat. Git ist besonders für seine Geschwindigkeit, Datenintegrität und Unterstützung für nicht-lineare, verteilte Workflows bekannt [GitHub, 2024].

Feature 1: Branching und Merging Git bietet ein leistungsfähiges Branching-System, das es ermöglicht, parallele Entwicklungslinien zu erstellen und zu verwalten. Branches sind leichtgewichtige Zeiger auf einen bestimmten Commit. Entwickler können schnell neue Branches erstellen, zwischen ihnen wechseln und sie zusammenführen. Dies ermöglicht:

- Isolierte Entwicklung neuer Features
- Experimentieren ohne Risiko für den Hauptcode
- Parallele Arbeit an verschiedenen Aspekten des Projekts
- Einfaches Zusammenführen von Änderungen durch Merging

Feature 2: Distributed Version Control Git ist ein vollständig verteiltes System, was bedeutet, dass jeder Entwickler eine vollständige Kopie des Repositories besitzt. Dies bietet mehrere Vorteile:

- Offline-Arbeit ist möglich
- Schnelle Operationen durch lokale Ausführung
- Redundanz und Backup durch multiple Kopien
- Flexible Workflow-Möglichkeiten
- Keine zentrale Schwachstelle

1.1.5 Qualitätssteigernde Maßnahmen

Fragestellung: Welche qualitätssteigenden Maßnahmen kennen Sie? Beschreiben Sie 2 beliebige im Detail.

1.2 Teilbereich DevOps

1.2.1 Aufgabenstellung

Das Unternehmen ABC Ad Tech stellt eine SaaS-Lösung für Kunden bereit mit denen ihre Werbekampagnen verwaltet werden können. Aktuell gibt es ein Entwicklungsteam namens Äd-Dev"mit 5 Personen die gemeinsam die Lösung entwickeln. Der Source Code ist hierzu in git abgelegt. Der Output des Build-Prozesses (=Artefakt) wird auf dem Firmen-PC von einem speziellen Mitarbeiter erstellt und dann händisch auf eine Netzwerkdateifreigabe kopiert. Für die

Kunden wird die Lösung aktuell durch das Team namens Äd-Ops"betrieben. Die beiden Teams arbeiten unabhängig voneinander. Das Entwicklungsteam Äd-Dev"liefert alle 3 Monate eine neue "Produktivversionünd alle zwei Woche eine neue "Testversion". Beide werden von Team Äd-Ops"bei Verfügbarkeit eingespielt, d.h. es wird das Artefakt von der Netzwerkdateifreigabe heruntergkopiert und dann "händischëingespielt. Die "Testversion"kommt immer auf eine Staging-Umgebung, welche vom Team Äd-QS"getestet wird und anschließend wird das Feedback mittels einer Besprechung an die Entwicklung rückgemeldet. Auch die "Produktivversion"wird zuerst auf der Staging-Umgebung getestet und sobald die Freigabe seitens Äd-QS"gegeben wird erfolgt die Einspielung auf das Produktivsystem durch das Team Äd-Ops". Bei Problemen und sonstigen Auffälligkeiten wird vom Team Äd-Opsäus Kontakt mit dem Entwicklungsteam Äd-Deväufgenommen.

1.2.2 Mögliche Probleme in der aktuellen Organisation und Arbeitsaufteilung

Fragestellung: Beschreibe kurz mögliche Probleme in der aktuellen Organisation und Arbeitsaufteilung

1.2.3 Notwendige Schritte um in dieser Organisation DevOps einzuführen

Aufgabenstellung: Beschreibe die notwendigen Schritte um in dieser Organisation DevOps (bis einschließlich Continuous Delivery) einzuführen.

1.2.4 Reihenfolge der Schritte

Fragestellung: Welche Schritte sind in welcher Reihenfolge notwendig?

1.2.5 Benötigte Tools

Fragestellung: Welche Tools werden hierzu benötigt?

1.2.6 Wesentliche Stakeholder und Argumente

Fragestellung: Beachte das solche Änderung Schritt für Schritt eingeführt werden müssen damit diese erfolgreich sein können. Ebenso müssen die wesentlichen Stakeholder überzeugt werden. Identifiziere die wesentlichen Stakeholder und liefere ihnen Argumente, die sie davon überzeugen das die Einführung von DevOps Praktiken vorteilhaft ist.

Literaturverzeichnis

[GitHub, 2024] GitHub (2024). About git.