Technische Hochschule
Brandenburg
University of
Applied Sciences
Fachbereich
Wirtschaft

Versionsverwaltung im Softwareengineering

Projektbetreuer*innen:
Michael Höding, Ralf Teusner

1 Das Projektstudium in der WI, dem tollsten Jahrgang aller Zeiten

Das Projektstudium Wirtschaftsinformatik wird im ersten Studienjahr angeboten. Es dient sowohl der Vermittlung fachlicher als auch überfachlicher Kompetenzen. Zum anderen werden durch Gruppenarbeit Fähigkeiten wie Kooperation und Kommunikation, Zeitmanagement und Projektorganisation entwickelt.

2 Was sind Branching-Strategien?

Branching-Strategien definieren strukturierte Vorgehensweisen zur Erstellung, Pflege und Zusammenführung paralleler Entwicklungszweige in Versionsverwaltungssystemen. Sie steuern, wie Teams neue Features entwickeln, Fehler beheben oder Releases vorbereiten, ohne Code-Stabilität zu gefährden. Sauber definierte Branch-Modelle reduzieren Konflikte, beschleunigen Reviews und erhöhen Nachvollziehbarkeit. Typische Ansätze sind Feature Branching, GitFlow, Trunk-Based Development und Release-Branches.

3 Trunk-Based Development

Trunk-Based Development ist ein agiler Ansatz, bei dem alle Entwickler kontinuierlich in einen gemeinsamen Hauptzweig („Trunk“) integrieren. Änderungen werden in kleinen, häufigen Commits vorgenommen, wodurch Merge-Konflikte reduziert und die Software stets in einem lauffähigen Zustand gehalten wird. Feature-Flags ermöglichen das schrittweise Aktivieren neuer Funktionen. Dieser Ansatz fördert schnelle Feedbackzyklen, höhere Codequalität und eine stabilere Release-Pipeline.

4 Feature Branching

Feature Branching ist eine gängige Strategie in der Versionskontrolle (z.B. Git), bei der Entwickler für jede neue Funktion, Fehlerbehebung oder jedes Experiment einen separaten Branch von der Hauptentwicklungslinie erstellen. Dadurch wird die Isolation des Codes gewährleistet. Die Arbeit kann unabhängig und ohne Risiko, die stabile Hauptversion zu beeinträchtigen, durchgeführt werden. Erst nach Fertigstellung und erfolgreicher Überprüfung wird der Feature Branch wieder in den Main Branch zusammengeführt.

5 Forking Workflow

Der Forking-Workflow unterscheidet sich fundamental von anderen Arbeitsweisen, da er keine Schreibrechte für alle Entwickler auf einem zentralen Repository voraussetzt. Stattdessen erstellt jeder Beteiligte eine serverseitige Kopie (Fork) des Projekts. Dieses Modell hat sich als De-facto-Standard in der Open-Source-Entwicklung etabliert, da es eine sichere Zusammenarbeit ohne zentrale Verwaltung von Zugriffsrechten ermöglicht. Dabei fungieren eine oder mehrere vertrauenswürdige Personen (Maintainer) als „Gatekeeper“. Nur sie besitzen die Berechtigung, Änderungen in das offizielle Hauptrepository zu integrieren, wodurch eine klare Hierarchie und Qualitätssicherung gewährleistet wird.

6 Wie beeinflussen unterschiedliche Branching-Strategien die Softwarequalität, insbesondere im Hinblick auf Fehlerhäufigkeit, Code-Stabilität und Merge-Konflikte?

	Trunk-Based Development	**Feature Branching**	**Forking Workflow**
Fehlerhäufigkeit	- Kurze Branch-Lebensdauer minimiert Divergenzen - Feature Flags isolieren unfertige Features -Hohe Integrationsfrequenz deckt Fehler früh auf -Hohe Testautomatisierung nötig	-Separate Feature-Branches schützen Hauptbranch -Risiko steigt bei langen Branches -CI reduziert frühe Fehler -Fehlerrisiko insgesamt mittel	-Isolierte Forks verhindern frühe Fehler im Hauptbranch -Pull Requests & Reviews senken Fehlerquote -CI erkennt Fehler vor Merge - Systemische Fehler können später auftreten
Code-Stabilität	-Trunk ist permanent produktionsbereit - Kurze Feedbackzyklen, Tests & Peer-Review sichern Stabilität - Feature Flags isolieren riskante Entwicklungen	-Hauptbranch stabil bei diszipliniertem Merge -Stabilisierung nicht garantiert -Kontrollierte Integration möglich	-Hauptbranch stabil durch Maintainer & Reviews - Forks trennen experimentelle von stabiler Entwicklung - Robuste Integration durch Guidelines & Tests
Merge-Konflikte	- Selten, da Branches kurzlebig -Konflikte inkrementell & schnell lösbar -Minimierter Merge-Umfang reduziert	-Häufig bei langen Feature-Branches -Konflikte gebündelt, nicht iterativ -Branch-Owner verantwortlich	-Höhere Konfliktwahrscheinlichkeit bei längerer Fork-Arbeit - Divergenzen → komplexe PR-Konflikte - Konflikte im Fork gelöst, Hauptbranch bleibt stabil

Dieses Poster ist im Rahmen der Lehrveranstaltung *Projektstudium - Wissenschaftliches Arbeiten* im Sommersemester 2022 entstanden.

Ihr Kontakt: Prof. Dr. Michael Höding
TH Brandenburg, Postfach 2132
D-14737 Brandenburg, Germany
E-Mail: hoeding@th-brandenburg.de

.der zweite Punkt