Verbesserung der MBSE-Ausbildung durch Versionskontrolle und automatisiertes Feedback

Levente Bajczi[1] , Dániel Szekeres[1] , Daniel Siegl[2] , Vince Molnár [1]

[1] Technische und Wirtschaftswissenschaftliche Universität Budapest
Abteilung für Mess- und Informationssysteme

[2] LieberLieber Software GmbH

Track (*Wissenschaftlicher Beitrag*): *Innovationen im Systems Engineering*

Schlüsselwörter: *MBSE, SysML, Ausbildung, Versionskontrolle, Modellzusammenführung*

Zusammenfassung: In diesem Beitrag wird ein innovativer Ansatz zur Durchführung einer MBSE(Model-Based Systems Engineering) -Lehrveranstaltung (Model-Based Systems Engineering) vorgestellt, an dem jährlich über 80 Teilnehmer:innen teilnehmen. Der Kurs ist um gemeinsame Gruppenaufgaben herum strukturiert, bei denen die Student:innen Enterprise Architect verwenden, um komplexe SystemsEngineering-Aufgaben über sechs Abgaben zu erledigen. In diesem Jahr haben wir mehrere technologische Neuerungen eingeführt, um die Lernerfahrung zu verbessern, darunter die Verwendung von LemonTree, SmartGit und GitHub. Die Student:innen arbeiteten an gemeinsam genutzten Repositories in GitHub, erhielten kontinuierliches Feedback durch automatisierte Prüfungen mit LemonTree.Automation und dokumentierten ihren Fortschritt mit vorgerenderten, sich ständig aktualisierenden Diagrammen. Darüber hinaus verwalteten sie 2-Wege- und 3-Wege-Merges direkt in SmartGit, wobei Merge-Probleme, Aktualisierungen und Modellstatistiken für jeden Work-in-Progress-Beitrag sofort verfügbar waren. Der Prozess der Korrektur und des manuellen Feedbacks wurde dank der zugänglichen Changelogs und Renderings in GitHub rationalisiert. Ein Feedbackformular am Ende des Kurses ergab eine hohe Zufriedenheit der Studenten.

Danksagung: Wir danken der LieberLieber Software GmbH, Sparx Systems Pty Ltd. und der syntevo GmbH für die Bereitstellung von Ausbildungslizenzen für unseren Kurs.

1 Einleitung

Das modellbasierte Systems Engineering (MBSE) hat sich zu einem Eckpfeiler des modernen Engineerings entwickelt. Angesichts der zunehmenden Komplexität des Systemdesigns bietet MBSE eine strukturierte Methodik, die die Klarheit, Nachvollziehbarkeit und Kommunikation zwischen den Stakeholdern verbessert. Mit der Verlagerung der Industrie hin zu immer komplexeren und vernetzten Systemen steigt die Nachfrage nach Ingenieuren, die MBSE-Methoden und -Werkzeuge beherrschen [1]. Dies erfordert ein Ausbildungsparadigma, das den Studierenden sowohl theoretisches Wissen als auch praktische Erfahrung im Umgang mit modernsten Werkzeugen vermittelt.

In der heutigen Industrielandschaft ist die Integration von automatisierten Überprüfungen und Versionskontrolle unerlässlich geworden [3,4]. Automatisierte Überprüfungen dienen als kritischer erster Schritt, um Fehler und Inkonsistenzen in einem frühen Stadium des Entwicklungsprozesses aufzuspüren. Dadurch wird nicht nur die Qualität der zu entwickelnden Systeme verbessert, sondern auch der Zeit- und Arbeitsaufwand für manuelle Überprüfungen erheblich reduziert. Ebenso sind Versionskontrollsysteme wie Git von entscheidender Bedeutung für die Nachverfolgung von Änderungen, die Erleichterung der Zusammenarbeit und die Aufzeichnung der Modellentwicklung. Indem wir sicherstellen, dass die Studierenden mit diesen Werkzeugen vertraut sind, bereiten wir sie auf den Eintritt in die Arbeitswelt mit den Fähigkeiten vor, die zur Rationalisierung und Optimierung der technischen Arbeitsabläufe erforderlich sind.

Unsere MBSE-Lehrveranstaltung unterrichtet jährlich über 80 Studenten:innen in Systems Engineering mit praktischen Beispielen in SysML [2]. In diesem Studienjahr haben wir den praktischen Teil des Kurses durch den Einsatz von Tools wie LemonTree, SmartGit und GitHub erneuert, um die Lücke zwischen akademischem Lernen und den Anforderungen der Industrie [7] zu schließen, den Studierenden bei der Erstellung besserer Modelle zu helfen und die Arbeitsbelastung der Reviewer zu verringern. Durch die Integration dieser Werkzeuge in den Lehrplan bieten wir den Studierenden praktische Erfahrungen in einer kollaborativen und automatisierten Umgebung, die die gängige Praxis widerspiegelt, denen sie in ihrer beruflichen Laufbahn begegnen werden. Diese Bemühungen stehen im Gegensatz zu früheren Berichten über die MBSE-Ausbildung, wie z. B. [1,5,6], und stellen eine Zusammenarbeit zwischen Hochschulen und Tool-Anbietern [7] dar, die auf die Bedürfnisse von Industriepartnern zurückgeht. In diesem Beitrag werden die Struktur, die Umsetzung und die Ergebnisse dieses innovativen Ansatzes beschrieben und seine Effektivität bei der Verbesserung der Lernerfahrung demonstriert.

1.1 Details zur Toolchain

Enterprise Architect: Enterprise Architect ist das zentrale Tool in unserem MBSE-Kurs, es unterstützt verschiedene Modellierungssprachen wie SysML und hilft den Studierenden, Systemanforderungen, Architektur und Verhalten effektiv zu erfassen.

GitHub: GitHub ermöglicht die Zusammenarbeit und Versionskontrolle in unserem Kurs. Studierende nutzen gemeinsame Repositories, um Projekte zu verwalten und Änderungen zu verfolgen, was das Verständnis für Versionskontrolle und Projektmanagement fördert.

GitHub Actions: GitHub Actions automatisiert Workflows und bietet kontinuierliches Feedback. Automatisierte Prüfungen bei Pull Requests ermöglichen eine kontinuierliche Integration. LemonTree.Automation wird zur Validierung von Modelländerungen eingesetzt.

SmartGit: SmartGit erleichtert die Versionskontrolle und die Lösung von Konflikten durch 2-Wege- und 3-Wege-Merges. Studierende lernen so wichtige Fähigkeiten zur Konfliktlösung und Versionskontrolle.

LemonTree: LemonTree verbessert unseren Kurs erheblich, indem es grafische Diff/Merge-Funktionen sowie automatische Überprüfungen und kontinuierliche Rückmeldungen über die Fortschritte der Student:innen bietet. LemonTree analysiert Modelländerungen und stellt die Konsistenz und die Einhaltung vordefinierter Standards sicher, so dass die Studenten einen unmittelbaren Einblick in ihre Arbeit erhalten. LemonTree.Automation unterstützt die Dokumentation durch Diagramm-Generierung und ist in den GitHub-Workflow integriert, um den Feedback-Prozess zu optimieren.

1.2 Details zum Kurs

Von den Studierenden erwartete Vorkenntnisse: Von den Studierenden, die sich für den MBSE-Kurs anmelden, wird erwartet, dass sie über ein grundlegendes Verständnis der Prinzipien der modellbasierten Entwicklung (wie UML im Kontext der Softwareentwicklung) und über grundlegende Kenntnisse im Umgang mit Modellierungswerkzeugen verfügen.
Kursbeschreibung Der Kurs deckt ein umfassendes Spektrum an Themen ab, die für MBSE wesentlich sind. Während der theoretische Teil des Kurses versucht, so allgemein wie möglich zu bleiben, behandelt der praktische Teil (die 6-stufige Hausaufgabe) die folgenden Schlüsselthemen speziell in SysML:

1. Anforderungsanalyse
2. Strukturelle Modellierung
3. Fehlertoleranz
4. Modellierung von Verhalten
5. Modellierung der Architektur
6. Verifizierung & Validierung

Bis auf den Schritt der Fehlertoleranz sollten alle diese Themen in Enterprise Architect unter Verwendung der in diesem Beitrag vorgestellten Infrastruktur durchgeführt werden.

Erwartete Lernergebnisse Es wird erwartet, dass die Studierenden nach Abschluss des Kurses die folgenden Ergebnisse erreicht haben:

- Beherrschung von MBSE-Tools: Nachweis der Kompetenz im Umgang mit Enterprise Architect, LemonTree, SmartGit und GitHub für kollaborative modellbasierte Systementwicklungsaufgaben.
- Verbesserte Fähigkeiten zur Zusammenarbeit: Effektive Arbeit in Teams, Verwaltung gemeinsamer Repositories und Lösung von Konflikten zur Wahrung der Modellintegrität.
- Automatisierte Überprüfungsprozesse: Schnittstelle zu automatisierten Prüfungen und kontinuierlichen Feedback-Mechanismen zur Verbesserung der Qualität und Konsistenz der Modelle.
- Umfassende Dokumentation: Sie erstellen und pflegen eine genaue, aktuelle Dokumentation, die die laufenden Modelländerungen und -aktualisierungen widerspiegelt.
- Industrietauglichkeit: Sie sammeln praktische Erfahrungen mit industriellen Werkzeugen und Prozessen und bereiten sich so auf eine nahtlose Integration in professionelle technische Umgebungen vor.

2 Der Arbeitsablauf für die Studierenden

In unserem MBSE-Lehrveranstaltung ist der Arbeitsablauf der Studierenden so gestaltet, dass er die state-of-the-art Prozesse aus der Industrie nachahmt, indem Automatisierung zur Rationalisierung von Aufgaben und Einreichungen eingesetzt wird. Die Übersicht über den Prozess ist in Abbildung 1 zu finden.

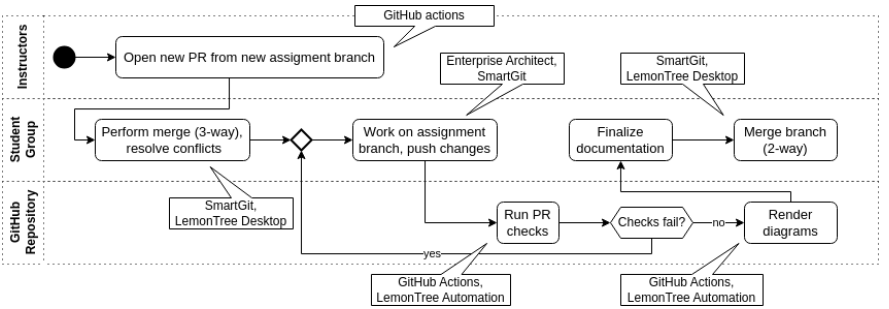

Abbildung 1. Überblick über den Arbeitsablauf der Studierenden

2.1 Automatische Zuweisung mittels Pull Requests

Der Arbeitsablauf beginnt mit der automatischen Zuweisung von Pull Requests (PRs) auf GitHub für jede Gruppenaufgabe. Zu Beginn eines jeden Aufgabenzyklus wird durch einen automatisierten Prozess eine neue Verzweigung im gemeinsamen Repository der Gruppe erstellt. In diesem Zweig entwickeln die Studierenden ihre Modelle und erledigen ihre Aufgaben. Die Automatisierung generiert auch einen ersten PRs, der einen strukturierten Rahmen für die Verfolgung von Änderungen schafft und die kontinuierliche Integration erleichtert.

Diese Automatisierung gewährleistet Konsistenz und bietet einen klaren Ausgangspunkt für jede Aufgabe. So können sich die Studierenden auf ihre technischen Aufgaben konzentrieren, ohne sich um die Feinheiten der Einrichtung der Versionskontrollumgebung kümmern zu müssen. Außerdem wird der Prozess standardisiert, was es den Lehrkräften erleichtert, die Fortschritte aller Gruppen zu verwalten und zu überwachen.

2.2 Abgabe mittels Pull Requests

Die Studierenden müssen zunächst mit einem 3-fache Merge ihrer früheren Änderungen zusammenführen. Sobald die Studierenden ihre Aufgabe abgeschlossen haben, wird der PR in den Hauptzweig ihres Repositories eingefügt. Diese Schritte werden durch SmartGit und LemonTree erleichtert, mit denen die Studierende etwaige modellbasierte Merge-Konflikte behandeln können. Vor dem Zusammenführen durchläuft der PR eine Reihe automatisierter Prüfungen über GitHub Actions und LemonTree.Automation, um die Qualität, Konsistenz und Übereinstimmung des Modells mit den Anforderungen zu überprüfen.

Die Ergebnisse dieser Prüfungen werden als Feedback im PR bereitgestellt, so dass die Studierenden vor der endgültigen Zusammenführung notwendige Anpassungen vornehmen können. Sobald die automatisierten Prüfungen und alle Konflikte gelöst sind, können die Studierenden den PR zusammenführen und den Einreichungsprozess abschließen. Dieser Arbeitsablauf stellt sicher, dass die Modelle gründlich geprüft werden, und gibt den Studierenden ein unmittelbares Feedback zu ihrer Arbeit, wodurch sie lernen und sich kontinuierlich verbessern können, durch die Verwendung von "required"-Prüfungen für die PRs konnten wir sicherstellen, dass die Studenten:innen nur Modelle zusammenführen, die sie für ihre Aufgaben einreichen können.

2.3. Dokumentation in Markdown mit dynamisch aktualisierten Diagramm-Renderings

In unserem Kurs spielt die Dokumentation eine entscheidende Rolle bei der Erfassung der Anforderungen, Entwurtsentscheidungen und der Entwicklung der von den Studenten erstellten Modelle. Um diesen Prozess zu rationalisieren, verwenden wir Markdown und dynamisch aktualisierte Diagramm-Renderings für die Erstellung der Dokumentation. Dies bietet mehrere Vorteile in Bezug auf Zugänglichkeit, Lesbarkeit und Echtzeit-Updates.

Die Markdown-Dateien werden im selben GitHub-Repository wie die Modelle gespeichert, um sicherzustellen, dass die Dokumentation und die Modelle zusammenbleiben und versionskontrolliert sind.

Die Integration von dynamisch aktualisierten Diagramm-Renderings wird durch automatisierte Workflows mit GitHub Actions und LemonTree.Automation erleichtert. Immer wenn Studierende Änderungen an ihrem Modell vornehmen und diese an das Repository übertragen, werden die Diagramme automatisch aktualisiert und gerendert. Diese aktualisierten Diagramme werden dann direkt in die Markdown-Dateien eingebettet und bieten eine aktuelle visuelle Darstellung der Modelle. Dieser dynamische Dokumentationsprozess stellt sicher, dass die Diagramme immer den neuesten Stand der Modelle widerspiegeln, wodurch manuelle Aktualisierungen überflüssig werden und das Risiko von Inkonsistenzen zwischen der Dokumentation und den tatsächlichen Modellen verringert wird. Durch den Einsatz von Markdown mit dynamisch aktualisierten Diagrammen lernen die Studierenden, während des gesamten Projektlebenszyklus eine genaue und aktuelle Dokumentation zu führen. Dies spiegelt die Industriestandards wieder, bei denen die Dokumentation ständig aktualisiert werden muss, um den aktuellen Stand des Systemdesigns wiederzuspiegeln. Außerdem werden die Lernergebnisse verbessert, indem die Bedeutung einer klaren und umfassenden Dokumentation hervorgehoben wird.

3 Umsetzung

In diesem Abschnitt wird die Verwendung der zuvor vorgestellten Werkzeuge zur Erreichung unserer Ziele beschrieben, einschließlich der Verwaltung von Kursmaterialien und studentischen Beiträgen, der Automatisierung von Feedback und der kontinuierlichen Integration von Dokumentation.

3.1 Kursverwaltung über GitHub

Wir haben uns für GitHub als primäre Plattform für die Verwaltung der studentischen Beiträge entschieden, da es über robuste Versionskontrollfunktionen verfügt und unter Softwareentwicklern weithin bekannt ist. Wir haben uns aus verschiedenen Gründen gegen die Verwendung von GitHub Classroom entschieden, vor allem wegen der Notwendigkeit, die Historie des Template-Repositorys in den Repositories der Studierenden aufrechtzuerhalten und um PRs zu erleichtern.

Template-Repository und Forking Anstatt GitHub Classroom zu verwenden, das isolierte Repositories für jeden Studierenden erstellt, ohne die Historie des Template-Repositories beizubehalten, haben wir ein zentrales Template-Repository erstellt und Forks für Studentengruppen angelegt. Auf diese Weise blieb die gesamte Commit-Historie der Vorlage in den Repositories der Studierenden erhalten, so dass wir PRs effektiv nutzen konnten. Dies war entscheidend für unseren Arbeitsablauf, der darauf angewiesen war, dass die Studenten:innen ihre Arbeit über bereits geöffnete PRs einreichten.

Repository-Struktur und Zugriffskontrolle Jede Studentengruppe erhielt einen privaten Branch des zentralen Template-Repositorys. Diese Struktur ermöglichte Folgendes:

- Unabhängige Arbeitsbereiche: Jede Gruppe hatte ihr eigenes privates Repository, um sicherzustellen, dass ihre Arbeit von anderen isoliert blieb.
- Zentrales Feedback: Durch die Verwendung des PR-Mechanismus konnten wir Feedback und Diskussionen zu bestimmten Beiträgen direkt auf GitHub zentralisieren und so eine klare und organisierte Aufzeichnung der Interaktionen beibehalten.
- Zugriffskontrolle über die integrierten Team- und Berechtigungseinstellungen von GitHub: Dadurch wurde sichergestellt, dass nur autorisiertes Personal (Dozent:innen und Lehrassistent:innen) den notwendigen Zugang zu den Repositories der Studierenden für die Überprüfung und das Feedback hatte.

3.2 Das Tutorial

Um die Studierenden auf die Arbeit mit Enterprise Architect, SmartGit und LemonTree vorzubereiten, haben wir den Kurs mit einem optionalen Tutorial begonnen. Dieses Tutorial sollte sie durch die Installation, Konfiguration und grundlegende Nutzung dieser Tools führen.

Grundlegende Aufgaben und Einarbeitung in den Workflow Das Tutorial ging nach einer detaillierten Installationsanweisung dann zu praktischen Aufgaben über, um die Studenten:innen mit den wesentlichen Funktionen der Werkzeuge vertraut zu machen. Die Studierende öffneten ein Beispielmodell in Enterprise Architect und nahmen eine einfache Änderung vor, indem sie eine Klasse umbenannten. Anschließend übergaben sie diese Änderungen an einen neuen Zweig in SmartGit und nutzten LemonTree, um die Änderungen vor dem Übergeben zu visualisieren und zu überprüfen. Sie wurden angewiesen, ihre Änderungen auf GitHub zu veröffentlichen und einen PR von ihrem Branch zum Main-

Branch zu öffnen. Im Tutorial wurde erklärt, wie man die durch den PR ausgelösten automatischen Prüfungen überwacht und die Ergebnisse interpretiert.

Um den Merge durchzuführen, wiederholten die Studierenden die Änderungsaufgabe auf einem anderen Branch, was zu Konflikten führte. Das Tutorial bot eine Schritt-für-Schritt-Anleitung zur Verwendung von SmartGit und LemonTree, um diese Konflikte zu lösen. LemonTrees Visualisierung von 3-Way-Merge wurde als Schlüsselwerkzeug für das Verständnis und die Lösung von Konflikten zwischen Branches hervorgehoben. Anschließend wurde den Teilnehmer:innen gezeigt, wie sie die zusammengeführten Änderungen übertragen und auf GitHub veröffentlichen können. Sie wurden angewiesen, den PR zusammenzuführen, um den Zyklus von der Änderung bis zur Integration zu vollenden.

3.3 Automatisierte Prüfungen für WiP-Einreichungen

Um die Lernerfahrung durch kontinuierliches Feedback zu verbessern, haben wir mit Hilfe von GitHub Actions eine Reihe von automatisierten Prüfungen für die von den Studierenden eingereichten Work-in-Progress (WiP) integriert. Diese Prüfungen wurden entwickelt, um die Qualität, Konsistenz und Korrektheit der von den Studierenden eingereichten Modelle sicherzustellen.

Zu den automatisierten Prüfungen gehörte eine "Diff-Prüfung", die Änderungen zwischen dem Basis- und dem Main-Branch eines PRs verglich, so dass Studierende und Lehrende die Änderungen Visualisiert bekamen und überprüfen konnten. Außerdem wurde eine "Modellprüfung" implementiert, um die Integrität und Struktur der eingereichten Modelle zu validieren und sicherzustellen, dass sie einige erforderliche Standards einhalten. Darüber hinaus wurde eine "Konsistenzprüfung" durchgeführt, um etwaige Inkonsistenzen innerhalb der Modelle zu erkennen und zu melden und so die Qualität der Arbeit weiter zu sichern.

Diese automatisierten Prozesse lieferten zeitnahes Feedback direkt in der GitHub-Pull-Anfrage, so dass die Studenten:innen Probleme umgehend angehen konnten und der gesamte Arbeitsablauf verbessert wurde.

4 Bewertung

Die Effektivität der Infrastruktur wurde qualitativ durch das Feedback der Prüfer:innen zur Infrastruktur und zur Qualität der eingereichten Arbeiten sowie durch das Feedback der Studenten zur Gesamterfahrung bewertet.

4.1 Rückmeldung der Lehrenden

Was die Infrastruktur betrifft, so konnten die eingereichten Arbeiten dank der automatischen Renderings in der Dokumentation viel schneller bewertet werden. Es gab Fälle, in denen es notwendig war, das Modell selbst zu öffnen, weil ein Rendering-Problem auftrat, aber es war immer noch besser, als sich immer auf die manuellen Exporte der

Studenten:innen zu verlassen oder das Modell zu öffnen. Durch die automatischen Konsistenzprüfungen wurden Fehler in den Einsendungen herausgefiltert, die wir zuvor einzeln prüfen mussten, so dass sich die Prüfer auf die Designentscheidungen und die semantische Korrektheit konzentrieren konnten.

Die modellbasierte Versionskontrolle und die Option des 3-Wege-Merge ermöglichten es den Studentengruppen, den Gruppenmitgliedern individuelle Aufgaben zuzuweisen und die Hausaufgaben viel einfacher parallel zu bearbeiten. Durch das bessere Zeitmanagement und die frühzeitige Rückmeldung durch die automatischen Kontrollen war die Qualität der eingereichten Arbeiten höher als zuvor.

4.2 Rückmeldungen der Studierenden

Am Ende der Lehrveranstaltung wurden die Studierenden gebeten, ihre Erfahrungen mit der neuen Infrastruktur zu bewerten (siehe Abbildung 2). Die Rückmeldungen zeigten ein relativ hohes Maß an Zufriedenheit, wobei mehr als 70 % der Antworten neutral oder positiv waren. Dies kann als ein sehr gutes Ergebnis angesehen werden, da die Studierenden zuvor noch keine Erfahrungen mit einer ähnlichen Technologie gemacht hatten. Daher ist die Bewertung dieser Frage nicht als Vergleich mit anderen Ansätzen zu verstehen, sondern eher als allgemeiner Eindruck von Komfort und Benutzerfreundlichkeit, wobei eine neutrale Antwort darauf hinweist, dass es keine ernsthaften Probleme oder Komplikationen gab.

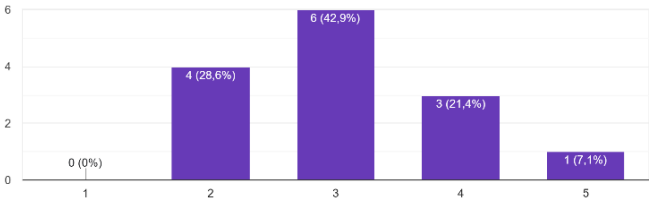

Abbildung 2. Verteilung der Antworten auf einer 5-stufigen Likert-Skala zu "Wie zufrieden waren Sie mit der gesamten Infrastruktur (Enterprise Architect, LemonTree, Github)?" (1 - "Ich möchte es nie wieder verwenden"; 5 - "Ich denke, es war ausgezeichnet")

Informelle Rückmeldungen zeigten, dass die Studierenden LemonTree als eigenständiges Werkzeug schätzten, sich aber mit Konflikten bei der Zusammenführung im Allgemeinen schwertaten. Basierend auf dem Feedback aus den Vorjahren, wo die typische Lösung für Merge-Konflikte darin bestand, die konfliktären Versionen zu verwerfen, kann dies als große Verbesserung angesehen werden. Ein Student wies darauf hin, dass das Zusammenführen von Modellen im Kursmaterial hervorgehoben werden sollte und das optionale Tutorial obligatorisch sein sollte.

Insgesamt konnte die neue Infrastruktur sowohl die Lernerfahrung als auch die Qualität der eingereichten Arbeiten verbessern, was sich in der Abschlussquote (79 %) und dem positiven Feedback widerspiegelt.

5 Zusammenfassung und zukünftige Arbeit

Die Integration von GitHub, LemonTree, SmartGit und GitHub Actions in unseren Kurs "Model-Based Systems Engineering" hat die Effizienz und Qualität der studentischen Erfahrung deutlich verbessert. Durch die Automatisierung des Einreichungs- und Feedbackprozesses waren wir in der Lage, kontinuierliche, umsetzbare Erkenntnisse zu liefern, die das Verständnis und das Engagement der Studierenden verbesserten. Die neue Infrastruktur ermöglichte eine reibungslosere Zusammenarbeit, eine effektive Konfliktlösung und zeitnahes Feedback, was zu höheren Abschlussquoten und einer verbesserten Qualität der eingereichten Arbeiten führte. Die Evaluierung der neuen Infrastruktur hat gezeigt, dass sie erfolgreich war, da eine beträchtliche Anzahl von Gruppen die Aufgaben mit einem zufriedenstellenden oder außergewöhnlichen Ergebnis abgeschlossen hat. Die Messung der Zufriedenheit der Studierenden ergab überwiegend neutrale und positive Antworten, was auf eine reibungslose Erfahrung für die Studierenden hinweist.

In Zukunft wollen wir unsere Infrastruktur durch die Integration eines Muster-basierten Modellvalidators weiter ausbauen, um nicht nur syntaktische, sondern auch semantische Anforderungen an das Systemdesign automatisch überprüfen zu können. Durch die kontinuierliche Innovation und Integration neuer Technologien in unseren Kurs sind wir bestrebt, eine hochmoderne Ausbildung zu bieten, die unsere Studierenden auf die komplexen Herausforderungen der realen Welt des Engineerings vorbereitet.

Literaturverzeichnis

[1] A. Khandoker, et al. "An Interdisciplinary Course on Model-Based Systems Engineering," 2023 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), Västerås, Sweden, 2023, pp. 102-109, doi: 10.1109/MODELS-C59198.2023.00033.

[2] OMG Systems Modeling Language Spezifikation. Verfügbar unter: https://www.omg.org/spec/SysML/1.6/About-SysML

[3] D. Siegl, "Bridging the Gap between OpenMBEE and Git," 2021 ACM/IEEE International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), Fukuoka, Japan, 2021, pp. 465-466, doi: 10.1109/MODELS-C53483.2021.00072

[4] D. Lehner, et al. "Git-based Model Management for Quality Monitoring of Systems Engineering Models". INCOSE International Symposium. Vol. 32. No. 1. 2022.

[5] S. Ugurlu, S. Bougain, C. Nigischer und D. Gerhard, "Application of model based systems-engineering in Austrian vocational schools", Proc. of the 18th International Conference on Engineering and Product Design Education (E&PDE16), pp. 089-094, 2016.

[6] A. Butting, S. Konar, B. Rumpe und A. Wortmann, "Teaching model-based systems engineering for industry 4.0: student challenges and expectations", Proc. of the 21st ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings (MODELS), S. 74-81, 2018.

[7] A. Khandoker, S. Sint, G. Gessl, K. Zeman, F. Jungreitmayr, H. Wahl, et al., "Towards a logical framework for ideal MBSE tool selection based on discipline specific requirements", Journal of Systems and Software, vol. 189, pp. 111306, 2022.