Wie lehrt man Requirements Engineering? Ein Erfahrungsbericht

Andrea Herrmann
Freiberufliche Trainerin für Software Engineering, D-70372 Stuttgart, herrmann@herrmann-ehrlich.de

Abstract

Bei diesem Artikel handelt es sich um einen Erfahrungsbericht darüber, wie man Requirements Engineering (RE) lehren kann, was gut ankommt und was eher nicht.

Motivation und Kontext

Es gibt Lehrkonzepte, die zwar in didaktischen Lehrbüchern empfohlen werden, aber nicht unbedingt gut funktionieren. In der relevanten Fachliteratur fehlen mir auch Erfahrungsberichte darüber, was gar nicht oder weniger gut klappt. Eines der besten Bücher, die ich zur Didaktik gelesen habe, ist das von Brauer [1], das jedoch themenunabhängig gehalten ist.

Ich lehre RE nun schon seit zehn Jahren. Zum Thema „Lehre für RE“ habe ich schon einige Erfahrungsberichte zu einzelnen Kursen publiziert ([2] bis [6]) und auch einen Workshop organisiert [7], aber das ist mein erster zusammenfassender Artikel zu diesem Thema.

Der Kontext meiner Kurse ist ein wechselnder: Ich gebe Vorlesungen, offene Seminare für Praktiker und Inhouse-Schulungen in Firmen, IREB-Prüfungsvorbereitungskurse (www.ireb.org) und praxis-orientierte Seminare, die für den Alltag fit machen.

Unterschiedliche Vorerfahrungen der Teilnehmer

Eine besondere Herausforderung ist es, einen Kurs anzubieten, der sowohl für Anfänger geeignet ist als auch erfahrenen Praktikern noch neue Erkenntnisse beschert. Das gilt auch für Vorlesungen. Viele Informatik-Studierende sind z.B. erfahrene Fachinformatiker, die nun noch einen Hochschulabschluss dazu erwerben.

Schlüsselelemente für einen solchen Kurs sind:

- Theorieteile knapp und präzise halten. Auch erfahrene Praktiker nehmen es einem nicht übel, die nötigen Fachbegriffe kurz zusammenzufassen und alle Teilnehmer auf denselben Stand zu bringen.
- Übungen im RE können so gestaltet werden, dass sie sowohl den Anfängern als auch den Fortgeschrittenen zugute kommen. So können erfahrene Teilnehmer komplexere Beispiele bearbeiten oder in Rollenspielen eine andere Rolle übernehmen, z.B. die des Kunden.

Phasen eines RE-Kurses

Die Phasen eines RE-Kurses sind:

- Motivation
- Begriffsdefinitionen
- Theorie
- Übung
- Zusammenfassung

Theorie und Übung wiederholen sich dabei iterativ in jeder Kurseinheit. Kurse bestehen letztlich aus modularen Lerneinheiten, schon allein aus dem praktischen Grund, dass man irgendwann Pausen einschieben muss und auch Kurse immer wieder neu erstellt, kürzt oder erweitert. Für die Teilnehmenden ist eine klare Kursstruktur wichtig zur Orientierung und es ist auch gut, wenn man die Agenda entweder an die Wand klebt oder als Handout verteilt.

Erfahrungen

Zu jeder der Kursphasen fasse ich hier meine wichtigsten Erfahrungen zusammen:

Motivation: RE muss motiviert werden, gerade für Teilnehmer, die sehr codeorientiert sind oder den Kurs nicht freiwillig besuchen. Dabei muss klar werden: RE ist wichtig, RE ist schwierig und RE ist erlernbar. Der übliche RE-Kurs startet mit den üblichen Statistiken über die hohe Anzahl misslungener IT-Projekte und weitere Studien, die aufzeigen, dass viele Fehler bereits in den Anforderungen steckten. So frappierend diese Zahlen sind, berühren sie die Teilnehmer aber nicht unbedingt emotional. Als einzige Motivation reichen sie nicht aus. Zusätzlich verwende ich zur Motivation am liebsten ein aktuelles Thema. Der Abgas-Skandal hat sich neulich sehr bewährt. In diesem Fall stellt sich die Frage, wie die Anforderungen definiert waren. War Stories aus meiner eigenen Erfahrung bringe ich in dieser Phase lieber nicht, weil ein misslungenes eigenes Projekt nicht automatisch ein gutes Licht auf die Kompetenz des Trainers wirft. Die War Stories kommen aber zu späteren Zeitpunkten gut an.

Es ist hier nicht nur wichtig zu erklären, warum ich mich für RE begeistere, sondern vor allem auch die Motivation der Teilnehmenden abzufragen. So kann man frühzeitig das Missverständnis aufräumen, wenn ein/e Teilnehmer/in glaubt, RE nicht zu benötigen.

Begriffsdefinitionen: Spaß machen Begriffs-definitionen selten. Ich verteile sie darum in homöopathischen Dosen über die ganze Schulung. Begriffe werden erst eingeführt, wenn sie gebraucht werden. Gleich zu Beginn eine Stunde lang ein Glossar vorzulesen wäre das Schlimmste, was man machen kann. Der erste Eindruck zählt. Am besten kommt es hier schon zu einer Diskussion, damit klar wird: Dies

ist ein Mitmachkurs! So kann man z.B. Definitionen gemeinsam erarbeiten.

Theorie: Am Anfang einer Lerneinheit sind die Teilnehmer von der Pause frisch erholt, neugierig und aufnahmefähig. Darum kann man hier Theorie vortragen, v.a. Methoden und Prinzipien vorstellen. Von theorielosen didaktischen Konzepten wie Problem-Orientiertem Lernen oder reinen Fallstudien-Kursen bin ich inzwischen völlig abgekommen: Die Teilnehmenden erwarten, dass der Trainer die Hälfte der Zeit redet, damit sie sich zwischen den Übungen entspannen können. Außerdem brauchen sie eine ordentliche Vorbereitung und Einweisung für die Übungen.

Wichtig ist auch eine lernzielorientierte Auswahl der Kursinhalte. Ein RE-Kurs von weniger als 100 Stunden Umfang muss ohnehin unvollständig sein. Ein fundamentaler Unterschied besteht zwischen einer praxisorientierten Schulung und der Vorbereitung auf eine Zertifizierungsprüfung. Die Zertifizierungsprüfung gibt umfangreiche Theorie-Inhalte verbindlich vor, die man am besten auch mehrmals wiederholt, und die Übungen dürfen und sollen einfach sein, um ganz konzentriert ein bestimmtes Wissen zu üben. Die praxisorientierte Schulung widmet ihre Zeit eher realistisch komplexen Fallstudien, anhand derer praktische Fragen diskutiert werden können. Die Theorie kann sich auf das Notwendige beschränken.

Einzelübungen: Bei den Einzelübungen geht es darum, Methoden in aller Ruhe selbst zu erfahren oder Wissen zu wiederholen. Z.B. UML lernt man nicht durch Lesen, sondern dadurch, dass man selbst modelliert. Bei den Einzelübungen kann man den erfahrenen Teilnehmern schwierigere Aufgaben geben als den Anfängern, und die Anfänger intensiver betreuen. Während die Anfänger ein vorbereitetes einfaches Beispiel bearbeiten, für das sie später eine Musterlösung erhalten, wenden die erfahrenen Teilnehmer die Methode auf ein Beispiel aus ihrer Berufspraxis an und nehmen so für den Alltag sogar ein handfestes Ergebnis mit nach Hause.

Gruppenübungen: Da RE viel Kommunikation beinhaltet, kann man nur wenige RE-Tätigkeiten sinnvoll einzeln üben. Gerade bei heterogenen Kursgruppen kommen Gruppenübungen besser an als Einzelübungen. Bei Programmierkursen hat eine Untersuchung gezeigt, dass der Lerneffekt am besten ist, wenn die Gruppenmitglieder ungefähr gleiches Wissensniveau haben [8]. Im RE gilt das aber nicht, wenn z.B. im Rollenspiel Interviews geführt werden, Planning Poker ausprobiert oder gemeinsam ein Prototyp erstellt wird. Hier übernehmen zwar die erfahrenen Teilnehmer ganz schnell Stift oder Maus, aber da es im RE für kaum eine Übung genau eine optimale Lösung gibt und außer der Notation / Methode auch der Inhalt diskutiert werden muss, entwickelt sich auch bei heterogenen Kleingruppen schnell eine lebhafte, lehrreiche Diskussion, bei der alle etwas lernen. Die Aufgabenstellung darf aber nicht zu offen sein, weil sonst v.a. darüber diskutiert wird, wie die Aufgabe zu bearbeiten ist.

Zusammenfassung: Am Ende der Kurseinheit wird das Gelernte zusammengefasst, zur Wiederholung und um zu klären, ob noch Fragen offen sind. Die Zusammenfassung des Gelernten kann man auch den Teilnehmern überlassen. Am Kursende fragt man die Kursteilnehmer nicht nur: „Hat´s Spaß gemacht und habt ihr etwas gelernt?“, sondern es ist auch gut, wenn diese sich konkret überlegen, was von dem Gelernten sie wie in der Praxis umsetzen möchten. Mir gefällt die Empfehlung von Gris [9], nach einem Kurs noch ein Coaching anzuschließen, um die Umsetzung des Gelernten in der Praxis sicherzustellen, aber das habe ich noch nicht praktisch ausprobiert. Außerdem dient das Feedback der Teilnehmer der kontinuierlichen Verbesserung des Kurses.

Referenzen

[1] Markus Brauer: An der Hochschule lehren - Praktische Ratschläge, Tricks und Lehrmethoden, Springer, 2014

[2] A. Herrmann: Lernen durch Feedback aus Inspektionen. Treffen der Fachgruppe Requirements Engineering der GI, 2013, Ilmenau

[3] A. Herrmann: Was bedeuten die Hofstede-Dimensionen für Training und Lehre des Software Engineering? Metrikon, 7. Dezember 2014, Stuttgart

[4] A. Herrmann, A. Hoffmann, R. Weißbach: Research on Intercultural Teaching for RE – Proposal for a Multi Case Study. Workshop on Requirements Engineering Education and Training REET at RE Conference, 25 August 2014

[5] A. Herrmann, A. Hoffmann, D. Landes R. Weißbach: Experience-Oriented Approaches for Teaching and Training Requirements Engineering: An Experience Report. Proceedings of the 20th International Working Conference, REFSQ 2014, Essen, Germany, April 7-10 2014, Camille Salinesi, Inge van de Weerd (eds), Springer, Lecture Notes of Computer Science LNCS 8396, S. 254-267

[6] M. Jastram, A. Herrmann: Eclipse for Teaching Systems Engineering. EclipseCon, 2014, https://www.eclipsecon.org/europe2014/session/eclipse-teaching-systems-engineering-35-minute-standard-talk

[7] A. Herrmann, A. Hoffmann, D. Landes, R. Weißbach: LehRE: Lehre für Requirements Engineering am 25.02.2014 in Kiel auf der Software Engineering Konferenz, http://www.haw-hamburg.de/index.php?id=31332, http://ceur-ws.org/Vol-1129/

[8] N. Katira, L. Williams, J. Osborne: Towards increasing the compatibility of student pair programmers. Proceedings of the 27th International Conference on Software Engineering, ICSE 2005, S. 625-626

[9] Richard Gris: Die Weiterbildungslüge - Warum Seminar und Trainings Kapital vernichten und Karrieren knicken. Campus Verlag, 2008