Andreas Zeller · Marcus Deininger (Hrsg.) (2007): Software Engineering im Unterricht der Hochschulen
SEUH 10, Stuttgart, dpunkt.verlag, Heidelberg

Positive Effekte von Szenarien und Features in einem Softwarepraktikum[1]

Kim Lauenroth[1] · *Ernst Sikora*[1, 2] · *Klaus Pohl*[1, 2]

1. Software Systems Engineering
   Institute for Computer Science and Business Information Systems (ICB)
   University of Duisburg-Essen, 45117 Essen
   {kim.lauenroth|ernst.sikora|klaus.pohl}@sse.uni-due.de
2. Lero – The Irish Software Engineering Research Centre
   University of Limerick, Limerick, Ireland
   {ernst.sikora|pohl}@lero.ie

Zusammenfassung

In der Bachelor-Veranstaltung Softwareentwicklung und Programmierung (SEP) führen die Teilnehmer die wesentlichen Aktivitäten der Softwareentwicklung vom Requirements Engineering über die Implementierung bis zum Softwaretesten durch.

In diesem Beitrag beschreiben wir die von uns beobachteten Probleme bei der Durchführung der Softwareentwicklungsaktivitäten durch die Studierenden und präsentieren einen szenario- und featurebasierten Ansatz zur Verminderung dieser Probleme. Wir erläutern die positiven Effekte von Szenarien und Features auf das SEP und belegen die Akzeptanz von Szenarien und Features anhand einer Umfrage unter den Teilnehmern.

1 Einleitung

Die Veranstaltung Softwareentwicklung und Programmierung (SEP) ist ein Bestandteil der praktischen Ausbildung in den Studiengängen Angewandte Informatik – Systems Engineering, Wirtschaftsinformatik, Mathematical Engineering und Lehramt Informatik an der Universität Duisburg-Essen. Das SEP hat pro

1 Das Verfassen dieser Arbeit wurde teilweise gefördert durch SFI Grant No. 03/CE2/I303_1.

Semester ca. 150 bis 250 Teilnehmer. Das SEP verfolgt drei wesentliche Zielsetzungen:

- Die Vertiefung von Programmierkenntnissen
- Die Entwicklung von Soft Skills (z.B. teamorientierte Arbeitsweise oder die angemessene Präsentation von Arbeitsergebnissen)
- Die Vermittlung eines methodischen Vorgehens zur Entwicklung von Software

Über vergleichbare Zielsetzungen in einem Softwarepraktikum wurde bereits berichtet, vgl. z.B. [Göh 05]. Schwerpunkt dieses Beitrags ist die Umsetzung des dritten Ziels, d.h. die Vermittlung methodischer Vorgehensweisen zur Softwareentwicklung.

In der industriellen Praxis hat die Bedeutung methodischer Vorgehensweisen zur Entwicklung von Software in den letzten Jahren erheblich zugenommen. Themen wie Requirements Engineering, Softwareentwurf und Softwaretesten spielen in IT-Unternehmen eine zentrale Rolle. Empirische Untersuchungen schreiben dabei insbesondere dem Requirements Engineering eine Schlüsselrolle für den Erfolg oder Misserfolg von Softwareprojekten zu (vgl. z.B. [Ha 02; Stan 04]).

Im SEP werden alle wesentlichen Aktivitäten der Softwareentwicklung durchgeführt, um den Studierenden die Zusammenhänge zwischen den einzelnen Aktivitäten anhand eines konkreten Beispiels zu verdeutlichen und ihnen somit wichtige Grundlagen der Softwareentwicklung anwendungsorientiert zu vermitteln. Um dies zu erreichen werden beispielsweise die Anforderungen an das zu entwickelnde System den Teilnehmern nicht im Detail vorgegeben, sondern müssen von den Teilnehmern (unter Berücksichtigung der gegebenen Aufgabenstellung) entwickelt werden.

Im Folgenden beschreiben wir den Aufbau des SEPs und legen die von uns beobachteten Probleme bei der Durchführung von Requirements-Engineering-, Entwurfs- und Testaktivitäten in früheren SEP-Veranstaltungen dar. In Abschnitt 2 erläutern wir die Umstellung des SEPs auf ein szenario- und featurebasiertes Vorgehen sowie die dadurch erzielten positiven Effekte. In Abschnitt 3 präsentieren wir die Ergebnisse einer Umfrage unter den Teilnehmern zu der Verwendung von Szenarien und Features im Requirements Engineering.

1.1 Aufbau und Durchführung des SEPs

Die Veranstaltung Softwareentwicklung und Programmierung baut auf einer Vorlesung »Programmierung« mit einer begleitenden Übung auf, in der u.a. die Programmiersprache Java vermittelt wird. Die Veranstaltung besteht aus den Teilen Einstiegsaufgabe (4 Wochen) und Hauptaufgabe (9 Wochen). Die Veranstaltung ist mit 3 ECTS-CP bewertet.

Die Einstiegsaufgabe beinhaltet Programmierübungen sowie eine darauf aufbauende Testaufgabe, mit der die Programmierkenntnisse der Teilnehmer über-

Andreas Zeller · Marcus Deininger (Hrsg.) (2007): Software Engineering im Unterricht der Hochschulen
SEUH 10, Stuttgart, dpunkt.verlag, Heidelberg

prüft werden. Nur die Teilnehmer, die die Testaufgabe bestehen (ggf. im Wiederholungsversuch), werden zur Teilnahme an der Hauptaufgabe zugelassen. Eine typische Testaufgabe beinhaltet die Ergänzung eines vorgegebenen, den Teilnehmern bekannten Programms um eine zusätzliche Funktionalität.

In der Hauptaufgabe führen Teams von jeweils 6 bis 10 Studierenden, begleitet durch die SEP-Betreuer, einen vorgegebenen Softwareprozess (im Folgenden als SEP-Prozess bezeichnet) durch. Der SEP-Prozess orientiert sich an bekannten Vorgehensmodellen, wie z.B. dem Rational Unified Process [Kru 03] oder dem V-Modell XT [Rau 07]. Da das SEP von Studierenden im 2. Semester belegt wird, können bei den Teilnehmern keine fundierten Kenntnisse von Softwareprozessmodellen, Modellierungstechniken oder CASE-Tools vorausgesetzt werden. Somit kommen für das SEP nur Techniken und Werkzeuge in Frage, die die Teilnehmer nach einer geringen Einarbeitungszeit anwenden können. Um den SEP-Prozess zu etablieren, finden wöchentliche Gruppensitzungen statt. Der Betreuer stellt in einer Gruppensitzung das Ziel für die nächste Woche vor (z.B. Durchführung des Requirements Engineering) und erläutert die zur Zielerreichung benötigten Techniken anhand eines durchgängigen Beispiels. Die Teilnehmer sind dafür verantwortlich, Teilgruppen zu bilden, die die jeweilige Aufgabenstellung bearbeiten, die Arbeitsergebnisse in eine vorgegebene Dokumentationsstruktur integrieren und diese dem Betreuer zur Verfügung stellen. Zur Vorbereitung der folgenden Gruppensitzung unterzieht der Betreuer die Ergebnisse einem Review. Die Teilnehmer werden in der folgenden Gruppensitzung über die identifizierten Fehler und die Verbesserungsvorschläge des Betreuers in Kenntnis gesetzt und müssen die notwendigen Korrekturen so durchführen, dass die Konsistenz von Anforderungen, Entwurf etc. gewahrt wird.

Zum Ende der Hauptaufgabe findet eine Abnahme statt, bei der jeweils ein Projektteam den SEP-Betreuern und den anderen Projektteams die eigenen Arbeitsergebnisse vorstellt. Eine solche Abschlusspräsentation besteht aus einem Folienvortrag, in dem die zentralen Entwicklungsartefakte kurz vorgestellt werden, sowie einer Programmdemonstration. Alle SEP-Teilnehmer erhalten die Gelegenheit, die erstellte Software auszuprobieren. In die Entscheidung über die Projektabnahme durch die Betreuer fließen die Begutachtung des lauffähigen Systems, der Projektdokumentation sowie der Abschlusspräsentation durch die SEP-Betreuer ein.

Tabelle 1 enthält einen Überblick über einige bisher bearbeitete Aufgabenstellungen in der Hauptaufgabe. Die Komplexität der Aufgabenstellungen wurde so gewählt, dass die Lösung der Aufgabe für die Teams eine Herausforderung darstellte, die Teams jedoch nicht überforderte (vgl. [Göh 05]). Eine angemessene Komplexität der Aufgaben ist insbesondere vonnöten, um ein methodisches Vorgehen zu motivieren.

Bezeichnung	Kurzbeschreibung
WikiSEPedia	Einfaches Wikisystem
BSepCW	Dokumenten- und Versionsmanagementsystem
BoulderSEP	Geschicklichkeitsspiel in Boulder-Dash-Manier
SEPTakis	Action-Spiel (Horizontalscroller) in Katakis-Manier

Tab. 1 *In der SEP-Hauptaufgabe bearbeitete Aufgabenstellungen*

1.2 Beobachtete Probleme mit Entwicklungsaktivitäten

In früheren SEP-Veranstaltungen wurden verschiedene Probleme bei der Durchführung von Requirements-Engineering-, Entwurfs- und Testaktivitäten beobachtet. Ein signifikanter Teil der Probleme wurde dabei durch Mängel in Requirements-Engineering-Aktivitäten verursacht. Diese Probleme stellen wir im Folgenden vor.

Problem 1: Mangelnde Detaillierung von Anforderungen

Eine häufig anzutreffende Fehlannahme der Teilnehmer war, dass die Aufgabenstellung bereits die Anforderungen an das zu entwickelnde System in einem hinreichenden Detailgrad enthalten würde. Die Teilnehmer setzten sich folglich nur unzureichend mit den Anforderungen an das System auseinander. Daten, Funktionen und Verhalten des geplanten Systems wurden somit zu Beginn des Projekts nur unzureichend erarbeitet und dokumentiert. Die Zeit, die für die Entwicklung von Anforderungen zur Verfügung stand, wurde nicht effektiv genutzt.

Problem 2: Vermischung von Anforderungen und Entwurf

In anderen Fällen vermischten die Teilnehmer Anforderungen und Entwurf. Dies führte u.a. dazu, dass die Teilnehmer Entwurfsdetails festlegten, ohne dass sie sich eingehend mit den Anforderungen an das System befasst hatten. Zeit, die für die Entwicklung von Anforderungen zur Verfügung stand, wurde von diesen Teilnehmern genutzt, um sich mit Entwurfsdetails zu befassen. In mehreren Fällen mussten die entwickelten Entwürfe jedoch wieder verworfen werden, nachdem die Anforderungen geklärt worden waren.

Problem 3: Entwurf und Implementierung mit unklaren Anforderungen

Die Teilnehmer gingen mit unklaren Vorstellungen darüber, was das geplante System genau können sollte, in die Entwurfs- und die Implementierungsphase ein. Folglich hatten die einzelnen Teilnehmer entweder sehr vage oder untereinander widersprüchliche Vorstellungen von dem zu entwickelnden System. Der einzelne Teilnehmer wusste somit nicht, was er entwerfen oder implementieren sollte. Die

Planung und arbeitsteilige Durchführung der Entwicklungsaktivitäten wurde daher erschwert. Anforderungen und Entwurf wurden letztlich nicht in der Gruppe abgestimmt, sondern von einem Kern von Projektmitgliedern festgelegt, die das System implementierten. Die weiteren Teammitglieder konnten keinen Einfluss auf die Anforderungen und den Entwurf nehmen, sondern diese nur noch nachträglich in die Anforderungsdokumentation übernehmen. Die Entwicklung des Systems war für die Gruppe insgesamt nicht mehr transparent und daher auch nicht kontrollierbar.

Problem 4: Fehlende Basis für Testfallerstellung

Für die Erstellung von Testfällen stand keine geeignete Ausgangsbasis, d.h. keine Testreferenzen z.B. in Form von Anforderungen, zur Verfügung. Ein systematisches Vorgehen zur Testfallerstellung, Testdurchführung und Testprotokollierung konnte den Teilnehmern folglich nicht vermittelt werden. Den Teilnehmern war es somit z.B. nicht möglich, (detaillierte) Testfälle zu erstellen, bevor die Implementierung beendet war. In der Zeit, die eigentlich für die Testdurchführung zur Verfügung stand, mussten daher auch alle Testfälle entwickelt werden.

2 Szenarien und Features im SEP

Zur Abmilderung der von uns beobachteten Probleme wurde das SEP im Wintersemester 2005/2006 auf ein auf Szenarien und Features basierendes Vorgehen umgestellt. Durch die Verwendung von Szenarien und Features sollte im SEP eine verbesserte Unterstützung der Requirements-Engineering-, Entwurfs- und Testaktivitäten erreicht und nahtlose Übergänge zwischen den einzelnen Aktivitäten ermöglicht werden.

Szenarien dokumentieren Interaktionsfolgen, die zu einem Mehrwert für den Systemnutzer (oder einen anderen Stakeholder) führen. Das Entwickeln von Szenarien unterstützt ein Projektteam bspw. darin, über zusammenhängende Abläufe nachzudenken und diese im Detail zu dokumentieren. Zahlreiche Autoren berichten von positiven Erfahrungen mit Szenarien (vgl. z.B. [Hau 98; Car 00]). Details zur Verwendung von Szenarien im Requirements Engineering sind u.a. in [Poh 07] zu finden. Features sind nach Kang für den Endbenutzer sichtbare Eigenschaften eines Systems (vgl. z.B. [Kan 90]). Features sind bspw. in Form von Produktmerkmalen in zahlreichen Produktbeschreibungen enthalten (z.B. von Mobiltelefonen oder von Fotokameras).

Über die Anwendung von Szenarien und Features in einem Softwarepraktikum liegen unseres Wissens keine Berichte vor. Daher beschreiben wir im Folgenden, wie Szenarien und Features im SEP eingesetzt werden. Die Verwendung von Szenarien und Features wird dabei anhand der von einem SEP-Team erstellten Entwicklungsartefakte zu der Aufgabenstellung »SEPTakis« veranschaulicht.

Andreas Zeller · Marcus Deininger (Hrsg.) (2007): Software Engineering im Unterricht der Hochschulen
SEUH 10, Stuttgart, dpunkt.verlag, Heidelberg

2.1 Requirements Engineering

Abbildung 1 gibt einen Überblick über das Requirements Engineering mit den Eingaben und Ausgaben im SEP-Prozess. Den Teilnehmern wird eine (bewusst vage gehaltene) Beschreibung des zu entwickelnden Systems (Aufgabenstellung) zur Verfügung gestellt. Ausgehend von dieser Beschreibung definieren die Teilnehmer die Anforderungen an das zu entwickelnde System in Form von Features, (Interaktions-)Szenarien und einem Datenmodell (siehe Abb. 1).

Einige Features können direkt aus der Aufgabenstellung abgeleitet werden. Die Teilnehmer müssen die aus der Aufgabenstellung übernommenen Features jedoch detaillieren sowie die Verwendung der Features durch den Systemnutzer mittels Szenarien konkretisieren. Die Formulierung von Szenarien verbessert zum einen das Verständnis der Teilnehmer für die vorhandenen Features und stimuliert zum anderen das Formulieren neuer Features. Somit decken die Teilnehmer im Verlauf der Szenarioerstellung weitere Features auf bzw. detaillieren die zuvor identifizierten Features. Für diese Features müssen wiederum neue Szenarien formuliert werden. Auf diese Weise werden Features und Szenarien für das geplante System iterativ weiterentwickelt. Die iterative Entwicklung von Szenarien und Features trägt u.a. zur Verminderung von Problem 1 (»Mangelnde Detaillierung von Anforderungen«; siehe Abschnitt 1.2) bei. Problem 2 (»Vermischung von Anforderungen und Entwurf«) konnte aufgrund der Ausrichtung von Features und Interaktionsszenarien auf für den Endbenutzer sichtbare Systemeigenschaften ebenfalls vermindert werden.

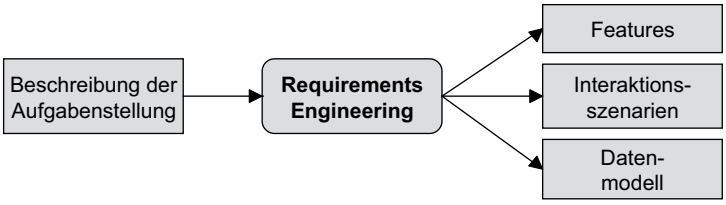

Abb. 1 *Requirements-Engineering-Aktivität mit Ein- und Ausgaben im SEP-Prozess*

Beispiel: Featuremodell

Basierend auf der Aufgabenstellung haben die Teilnehmer unter anderem die in Abbildung 2 dargestellten Features identifiziert. Die Features wurden von den Studierenden in einem Feature-Baum dokumentiert, der eine hierarchische Strukturierung von Features ermöglicht. Das Spiel SEPTakis als Wurzel-Feature wird unterteilt in die groben Features Grafik, Funktion, Steuerung und Sound. Diese Features werden wiederum weiterverfeinert, wie in Abbildung 2 dargestellt. Zusätzlich wird jedes Feature textuell erläutert, um Missverständnisse über die Bedeutung der einzelnen Features zu vermeiden.

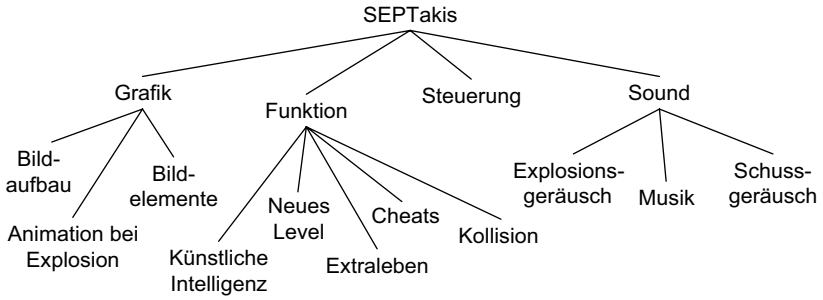

Abb. 2 *Auszug aus einem Feature-Baum für SEPTakis*

Beispielszenario

Szenarien werden im Requirements Engineering von den Studierenden in textueller Form dokumentiert. Um nachvollziehen zu können, ob jedes Feature mindestens einem Szenario zugeordnet ist, werden die Teilnehmer angehalten, die Features zu nummerieren und für jeden Szenarioschritt zu dokumentieren, welche Features in diesem Schritt zur Anwendung kommen. Abbildung 3 zeigt ein Beispielszenario zum Spiel SEPTakis.

Szenario:	Features:
1. Der Spieler steuert das Raumschiff gegen einen großen Asteroiden,die Kollision (2.6.1) zerstört das Raumschiff.	2.6.1: Kollision mit Objekten
2. Die Explosion des Raumschiffes wird durch ein Explosionsgeräusch (4.1) und Explosions-Animation (1.2) dargestellt.	4.1: Explosionsgeräusche 1.2: Animation bei Explosionen
3. Der Spieler verliert ein Leben (2.3).	2.3: Raumschiffzerstörung
4. Das Level beginnt am Levelanfang (2.2).	2.2: Levelneustart

Abb. 3 *Beispielszenario mit zugeordneten Features*

2.2 Entwurf

Der Entwurf ist im SEP in den Grobentwurf und den Feinentwurf untergliedert (siehe Abb. 4). Beim Grobentwurf sollen die Teilnehmer die Systemarchitektur skizzieren, d.h. das System in grobe Systemkomponenten strukturieren. Beim Feinentwurf erstellen die Teilnehmer für jede identifizierte Komponente im Grobentwurf ein (Entwurfs-)Klassendiagramm. Architekturszenarien unterstützen die Teilnehmer darin, Komponenten, Klassen und Methoden zu identifizieren. Die Teilnehmer erstellen für jedes Interaktionsszenario ein Architekturszenario, das die zur Ausführung des Interaktionsszenarios erforderlichen Interaktionen zwischen Systemkomponenten beschreibt. Die Komponenten und deren Interaktionen müssen wiederum durch Klassen und Methodenaufrufe realisiert werden.

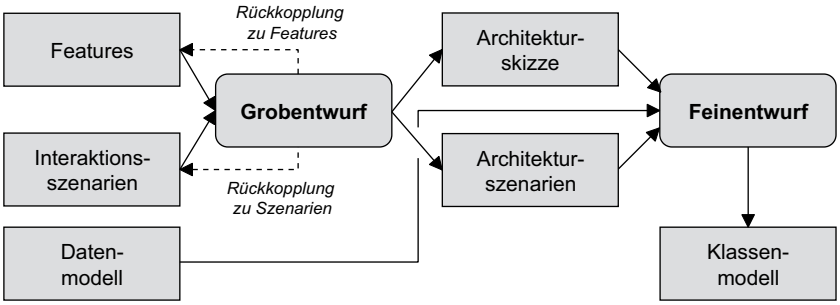

Abb. 4 *Entwurfsaktivitäten im SEP-Prozess*

Architekturszenarien bilden eine Brücke zwischen Anforderungen und Entwurf, d.h., sie erleichtern es den Teilnehmern, Beziehungen zwischen Anforderungen und Entwurf herzustellen. Bei der Entwicklung von Architekturszenarien reflektieren die Teilnehmer die zuvor formulierten Szenarien und Features. Häufig stellen die Teilnehmer fest, dass bestimmte Szenarien oder Features in der formulierten Weise nicht umzusetzen sind, oder dass Szenarien oder Features fehlen. Die zusätzlichen Erkenntnisse führen zu Anpassungen der bereits definierten Szenarien und Features. Diese stehen wiederum für die weiteren Entwurfsaktivitäten zur Verfügung. Durch die sukzessive Verfeinerung und Verbesserung der Anforderungen wird Problem 3 vermindert (»Entwurf und Implementierung mit unklaren Anforderungen«).

Beispiel: Komponenten von SEPTakis

Die Architektur des Systems wird im SEP mittels eines UML-Komponentendiagramms dokumentiert (siehe [Rum 05]). Abbildung 5 zeigt exemplarisch ein Komponentendiagramm für SEPTakis.

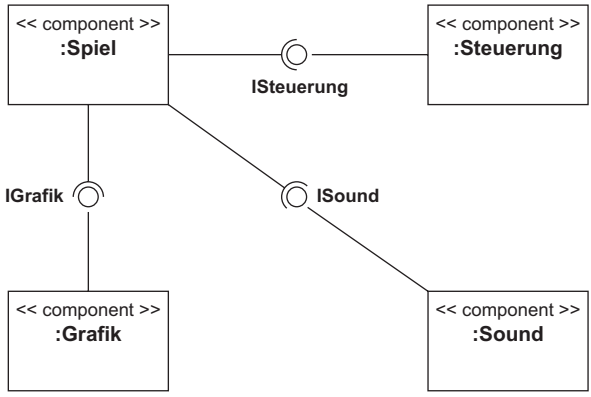

Abb. 5 *SEPTakis-Architektur als UML-Komponentendiagramm*

Beispiel: Architekturszenario

Architekturszenarien werden im SEP mittels UML-Sequenzdiagrammen (siehe [Rum 05]) dokumentiert. In Abbildung 6 wird ein Architekturszenario dargestellt, das das Szenario aus Abbildung 3 detailliert. Die Akteure des Architekturszenarios sind die in Abbildung 5 dargestellten Komponenten.

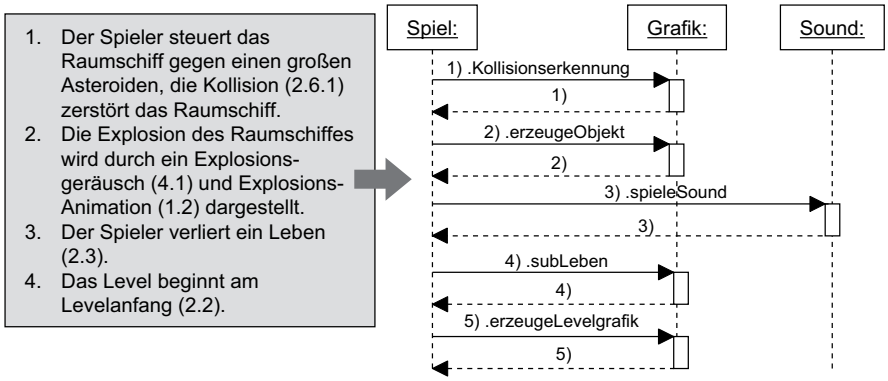

Abb. 6 *Architekturszenario von SEPTakis (rechts)*

2.3 Implementierung

Bei der Implementierung erstellen die Teilnehmer basierend auf den zuvor entwickelten Artefakten das System. Szenarien und Features unterstützen die Teilnehmer bei der Koordination der Implementierungsaktivitäten. Zum Beispiel können die Teilnehmer anhand der bereits durchführbaren Szenarien oder der realisierten Features den erreichten Projektfortschritt messen.

2.4 Testen

Bei der Entwicklung von Testfällen dienen Interaktionsszenarien als Grundlage für die Formulierung von Systemtestfällen und Architekturszenarien als Basis für die Entwicklung von Integrationstestfällen (siehe Abb. 7).

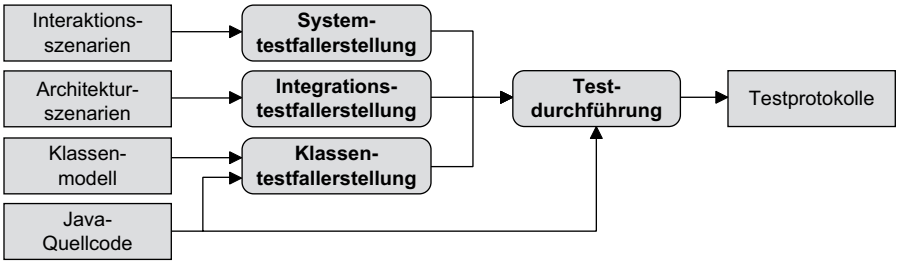

Abb. 7 *Testaktivitäten im SEP-Prozess*

Den Teilnehmern wird vermittelt, dass Testfälle bereits unmittelbar im Zusammenhang mit der Erstellung der jeweiligen Anforderungs- und Entwurfsartefakte entwickelt werden können. Durch das beschriebene Vorgehen wird Problem 4 (»Fehlende Basis für Testfallerstellung«) vermindert. Testdurchführung und Protokollierung erfolgen möglichst zeitnah zur Fertigstellung der für den Test benötigten Systemteile.

Beispiel: Systemtestfall

Von den SEP-Teilnehmern wird lediglich gefordert, exemplarisch jeweils zwei Modul-, zwei Integrations- und zwei Systemtestfälle zu erstellen. Darüber hinaus ist es den Teilnehmern selbst überlassen, wie intensiv sie das implementierte System testen. Abbildung 8 zeigt auszugsweise einen von SEP-Teilnehmern erstellten Systemtestfall.

Testziel:	Test des Szenarios »Raumschiff trifft auf Jäger«
Testumgebung:	Windows XP, JRE 1.5, JCreatorLE, SEPTakis
Vorbedingungen:	SEPTakis, kompiliert, noch nicht gestartet
Nachbedingungen:	Spiel läuft
Testfallszenario	
Tester	**System**
Der Tester startet das Spiel, indem er auf das Desktop-Icon klickt.	
	Das System stellt das Spielfenster mit allen Elementen dar, und das Spiel beginnt.
Der Tester prüft, ob • die Objekte (Raumschiff, Jäger, Asteroiden) im Spielfeld vorhanden sind, • die Lebenspunkte angegeben sind, • der Punktezähler vorhanden ist.	
Der Tester bewegt das Raumschiff über die Pfeiltasten.	
	Das System prüft die Eingaben auf Gültigkeit und führt diese aus. Das System bewegt das Raumschiff entsprechend der Eingabe.
Dem Raumschiff kommt ein flugbahnkreuzender Jäger entgegen. Der Tester versucht, dem Jäger auszuweichen, indem er die Pfeiltasten betätigt.	
	Das System bewegt das Raumschiff entsprechend der Eingabe.
Erfolgskriterien • Das System stellt das Spielfenster mit allen Elementen korrekt dar, und das Spiel beginnt. • Das System erkennt die Kollision zwischen Raumschiff und Objekten (Jäger, Asteroiden). • Das System erkennt die Benutzereingabe (Pfeiltasten für Richtungen und Leertaste für Schuss) und führt diese aus. • Das System zählt die erzielten Punkte im Punktezähler. • Das System zeigt die aktuellen Lebenspunkte an. • Das System reagiert nicht auf falsche Tastatureingaben. • Das System erkennt die Kollision mit dem Spielfeldrand und verhindert ein Herausfliegen.	

Abb. 8 *Exemplarischer Systemtestfall von SEPTakis*

3 Evaluierung

Die positiven Effekte von Szenarien und Features wurden an den Arbeitsergebnissen der Studierenden deutlich. Beispielsweise waren in den von Studierenden erstellten Features und Szenarien deutlich weniger Entwurfsinformationen enthalten als in den natürlichsprachlichen Anforderungen, die in früheren Semestern erstellt wurden. Die Entwicklung von Testfällen (insbesondere für den System- und Integrationstest) fiel den Studierenden deutlich leichter, da die erstellten Interaktions- und Architekturszenarien als Basis verwendet werden konnten. Für die Beurteilung des mit der Verwendung von Szenarien und Features erreichten Erfolgs war für uns die Meinung der SEP-Teilnehmer von Interesse. Maßgeblich waren für uns die folgenden Fragestellungen:

1. Inwieweit regten Szenarien und Features inhaltliche Diskussionen über Anforderungen an?
2. Inwieweit haben Wechselwirkungen zwischen Szenarien und Features zu einem verbesserten, inhaltlichen Verständnis des Systems beigetragen?

Um diese beiden Fragestellungen zu beantworten wurden vier Items formuliert und in einen Meinungstest zur Evaluierung der Veranstaltung integriert:

1. Die Szenarien des geplanten Spiels waren eine Grundlage für viele Diskussionen.
2. Die Features des geplanten Spiels waren eine Grundlage für viele Diskussionen.
3. Durch die Formulierung von Szenarien zu den einzelnen Features habe ich die Features besser verstanden.
4. Bei der Formulierung von Szenarien habe ich Ideen für weitere Features bekommen.

Die Items wurden von insgesamt 94 Teilnehmern auf folgender Skala bewertet: sehr zutreffend, zutreffend, mittel, unzutreffend, sehr unzutreffend. Abbildung 9 stellt die Resultate des Meinungstests zu den vier vorgestellten Items als Säulendiagramm dar.

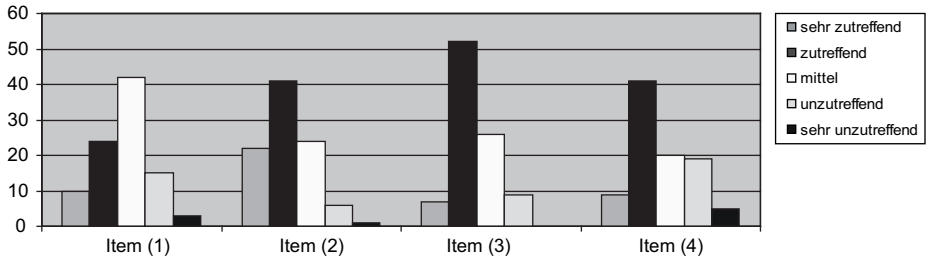

Abb. 9 *Antworthäufigkeiten zu den vier Items*

Im Folgenden interpretieren wir die Resultate der einzelnen Items in Hinblick auf unsere Ausgangsfragestellungen. Wir werten eine nicht negative Bewertung eines Teilnehmers, d.h. die Kategorien »mittel«, »zutreffend« und »sehr zutreffend«, als Zustimmung zu dem Item:

1. Zu diesem Item äußerten 76 Teilnehmer (81%) ihre Zustimmung. Wir schließen daraus, dass die Gruppen durch das Formulieren von Szenarien zu inhaltlichen Diskussionen angeregt wurden.
2. Zu diesem Item äußerten 87 Teilnehmer (93%) ihre Zustimmung. Wir schließen daraus, dass Features ein zentraler Aspekt von inhaltlichen Diskussionen innerhalb der Gruppen waren.
3. Zu diesem Item äußerten 70 Teilnehmer (74%) ihre Zustimmung. Wir schließen daraus, dass die Teilnehmer das Konkretisieren von Features durch Szenarien als positiv empfunden und im Projekt genutzt haben.
4. Zu diesem Item äußerten 85 Teilnehmer (90%) ihre Zustimmung. Wir schließen daraus, dass die Teilnehmer Szenarien als sehr hilfreich empfanden, um Features zu identifizieren.

Insgesamt lässt die Auswertung des Meinungstests auf einen guten bis sehr guten Erfolg der Verwendung von Szenarien und Features im SEP schließen. Szenarien und insbesondere Features regten einen signifikanten Anteil der Teilnehmer zu inhaltlichen Diskussionen an. Zudem erwiesen sich Szenarien als sehr hilfreich, um neue Features zu identifizieren sowie das Verständnis über bekannte Features zu vertiefen.

4 Fazit und Ausblick

In diesem Beitrag wurde die Verwendung von Szenarien und Features in einem Softwarepraktikum dargestellt. Die Arbeitsergebnisse der Teilnehmer zeigen, dass Szenarien und Features zu einer Verbesserung gegenüber früheren Veranstaltungen beigetragen haben. Die positive Bewertung von Szenarien und Features im Meinungstest ist ein Indiz dafür, dass die Teilnehmer die positiven Effekte von Szenarien und Features selbst erfahren haben. Diese Ergebnisse motivieren uns, weitere Evaluierungen zur Untermauerung unserer bisherigen Erkenntnisse durchzuführen.

5 Danksagung

Wir möchten an dieser Stelle einen besonderen Dank an unsere studentischen Hilfskräfte André Heuer und Sebastian Jackels für ihre wertvollen Beiträge zur Gestaltung des SEPs sowie für ihren unermüdlichen Einsatz bei der Betreuung von SEP-Gruppen aussprechen.

Literatur

[Car 00] J. M Carrol (Hrsg.): Making Use: Scenario-Based Design of Human Computer Interactions. MIT Press, Cambridge, 2000.

[Göh 05] P. Göhner, F. Bitsch, H. Mubarak: Softwaretechnik live – im Praktikum zur Projekterfahrung. In: SEUH '05, dpunkt.verlag, Heidelberg, 2005, S. 41-55.

[Ha 02] T. Hall, S. Beecham, A. Rainer: Requirements Problems in Twelve Companies – An Empirical Analysis. In: Proceedings of the 6th International Conference on Empirical Assessment and Evaluation in Software Engineering (EASE 2002), Keele University, 2002.

[Hau 98] P. Haumer, K. Pohl, K. Weidenhaupt: Requirements Elicitation and Validation with Real World Scenes. IEEE Transactions on Software Engineering, Vol. 24, Nr. 12, 1998, S. 1036-1054.

[Kan 90] K. Kang, S. Cohen, J. Hess, W. Nowak, S. Peterson, Feature-Oriented Domain Analysis (FODA) Feasibility Study. Technical Report CMU/SEI-90-TR-021, Software Engineering Institute, Carnegie-Mellon University, Pittsburg, 1990.

[Kru 03] P. Kruchten: The Rational Unified Process – An Introduction. Addison-Wesley, Reading, 2003.

[Poh 07] K. Pohl: Requirements Engineering – Grundlagen, Prinzipien und Techniken. dpunkt.verlag, Heidelberg, 2007.

[Rau 07] A. Rausch, M. Broy: Das V-Modell XT – Grundlagen, Erfahrungen, Werkzeuge. dpunkt.verlag, Heidelberg, erscheint 2007.

[Rum 05] J. Rumbaugh, I. Jacobson, G. Booch: The Unified Modeling Language Reference Manual. 2. Aufl., Addison-Wesley, Boston, 2005.

[Stan 04] The Standish Group: Chaos Demographics, 2004.