Proc. of Deutscher Software-Produktlinien Workshop (DSPL-1), Kaiserslautern, November 2000

Definition von Testfällen für kundenspezifische produktfamilienbasierte Anwendungen

Klaus Pohl, Mark Strembeck
Universität GH Essen
Software Systems Engineering
Altendorfer Str. 97-101
45117 Essen
{pohl | strembeck}@informatik.uni-essen.de

Zusammenfassung

Der Fokus einer aktuellen Kooperation des Unternehmens SHS mit der Software Systems Engineering Forschungsgruppe liegt auf der Reduktion des Testaufwands für kundenspezifische Anwendungen, die aus den generischen Artefakten einer Produktfamilie abgeleitet werden. Hierdurch wurde die Erweiterung des V-Modells durch eine szenariobasierte Vorgehensweise initiiert. Diese Vorgehensweise ermöglicht die explizite Berücksichtigung der Definition und Verfeinerung von Tests in jeder Entwicklungsphase. Durch die parallele Aufzeichnung von entsprechenden Traceability-Informationen kann zudem die Abschätzung des Änderungsaufwands sowie die Propagierung von Änderungen in das Testmodell unterstützt werden.

1. Einleitung und Motivation

Bevor mit der Argumentation begonnen wird, soll an dieser Stelle noch eine Definition für die im folgenden verwendeten Begriffe *Software-Produkt* und *Software-System* gegeben werden:

- Ein *Software-Produkt* wird hierbei als eine Konfiguration von Komponenten bzw. Architekturartefakten betrachtet.
- Ein *Software-System* stellt wiederum eine Konfiguration mehrerer Software-Produkte dar.

Zusätzlich muss darauf hingewiesen werden, das jedes der einzelnen Software-Produkte, die zu einem Gesamtsystem integriert werden i.d.R. eine Instanz einer eigenen Produktfamilie ist und prinzipiell auch als "Standalone-Produkt" vertrieben werden kann. Im Fall von SHS werden beispielsweise (u.a.) ein "Radiologie Informationssystem", ein sog. "Healthcare Information System" sowie ein sog. "Picture Archiving and Communication System" zu einem Gesamtsystem integriert. Jedes dieser Produkte kann jedoch auch separat vertrieben werden und mit entsprechenden Systemen anderer Hersteller integriert werden. Ein Software-System im obigen Sinne stellt demnach eine Familie von Produkt-Familien (bzw. von deren Instanzen) dar.

1.1. Problemstellung

In einer aktuellen Kooperation des Unternehmens Siemens Health Services (SHS) mit der Software Systems Engineering Forschungsgruppe der Universität Essen geht es darum, szenariobasierte Testfallableitung im Rahmen der Produktfamilienentwicklung zu untersuchen.

Im Rahmen der angesprochenen Kooperation wurde festgestellt, dass die ohnehin bereits hohen Anforderungen an die Testbarkeit der entwickelten Software sowie an die Möglichkeiten zur Testfallableitung (vgl. z.B. [Harr00], [Hetz88], [Myer79] und [Whit00]) im Produktfamilienkontext nochmals zunehmen. Dies folgt insbesondere aus dem Umstand, dass sowohl die generischen Teile einer Produktfamilie als auch jede der abgeleiteten kundenspezifischen Applikationen umfassend getestet werden müssen. Mit anderen Worten: eine Produktfamilie (bzw. die domänenspezifischen Artefakte) ist für sich bereits ein sehr komplexes System, welches darüberhinaus verwendet wird, um eine prinzipiell beliebige Anzahl kundenspezifischer Applikationen abzuleiten. Für jede abgeleitete Anwendung müssen daher sowohl die generischen als auch die kundenspezifischen Anforderungen erfüllt und dementsprechend auch getestet werden. Das Testen der essentiellen Kundenanforderungen auf korrekte Erfüllung durch die resultierende Software und die Entdeckung evtl. vorhandener Lücken ist kritisch für den Erfolg jedes (Sofware-)Produkts und Systems.

Die Tatsache, daß sowohl die generischen Teile der Produktfamilie als auch die abgeleiteten spezifischen Applikationen im Laufe ihres Lebens Änderungen unterworfen sind, führt zu einer weiteren Erhöhung der Komplexität. Entsprechend muss die Möglichkeit bestehen im Falle der Änderung eines Artefakts schnell die zugehörigen Testfälle zu identifizieren und evtl. an die Änderung fälle zu identifizieren und evtl. an die Änderung anzupassen. Hierdurch kann die Entwicklungszeit von Software-Produkten verkürzt werden.

Ebenso wie in der traditionellen Software-Entwicklung verursacht das Testen der Software in der produktfamilienbasierten Entwicklung einen Großteil der gesamten Software-Entwicklungskosten (vgl. [Harr00]). Dies folgt aus der Tatsache, dass der Entwicklungsaufwand bei der Ableitung von Applikationen aus einer Produktfamilie durch einen hohen Grad an Wiederverwendung reduziert werden kann, während die aufwendigen Integrations- und Systemtests jedoch für jede abgeleitete Applikation erneut durchgeführt werden müssen. Da es nicht möglich ist komplexe Software vollständig zu testen und weil darüberhinaus bereits die geringfügige Änderung einer Anforderung oder das Hinzufügen einer neuen kundenspezifischen Anforderung einen großen Testaufwand verursacht, ist es erforderlich aus allen möglichen Tests - nach Signifikanz - die wichtigsten auszuwählen (vgl. z.B. [Hetz88], [Harr00], [Myer79] und [Whit00]). In anderen Worten: es wird eine Möglichkeit zur sinnvollen Reduktion der durchzuführenden Testfälle sowie zur schnellen Anpassung der Testfälle an evtl. Änderungen benötigt.

Aufgrund der oben erwähnten, erhöhten Test-Komplexität im Rahmen der produktfamilienbasierten Entwicklung hätte ein entsprechendes Vorgehen das Potential, die positiven Effekte der Produktfamilien-Entwicklung, wie z.B. erhöhte Qualität der resultierenden Produkte oder verkürzte "Time-to-Market", weiter zu erhöhen.

1.2. Lösungsansatz

Szenarien im allgemeinen und Use Cases im besonderen sind ein bewährtes Mittel, um (funktionale) Kundenanforderungen zu identifizieren und zu modellieren (vgl. z.B. [Jaco92], [JaBC98] und [WPJH98]). Des weiteren ist ein Use-Case-Modell ein hervorragender Ausgangspunkt für die Auswahl der Funktionalität, die eine spezifische Applikation, welche von einer Produktfamilie abgeleitet wird, zur Verfügung stellen soll. Daher ist es sehr wahrscheinlich, dass (funktionale) Änderungen innerhalb einer Produktfamilie bzw. einer kundenspezifischen Anwendung zuerst auf der Use-Case-Ebene identifiziert werden. Aus diesen Gründen wurden Use-Cases von uns als Ausgangspunkt gewählt, um eine an Kundenanforderungen orientierte Reduktion der Testfälle zu ermöglichen.

SHS befindet sich zur Zeit in einer Übergangsphase von der traditionellen hin zu einer produktfamilienbasierten Software-Entwicklung. Der von SHS verwendete Software-Entwicklungsprozess ist an das V-Modell [VMod97] angelehnt. Entsprechend besteht die Aufgabe darin eine geeignete Anpassung des V-Modells vorzunehmen, um die szenariobasierte Testfallableitung im Rahmen einer produktfamilienbasierten Entwicklung zu ermöglichen.

Die Methodenzuordnung für das V-Modell [VMMZ97] verweist bereits auf Methoden für die Ableitung und die Durchführung von Tests. Im Verlauf des Projektes mit SHS wurde aber festgestellt, dass die im V-Modell vorgesehenen Methoden bei weitem nicht ausreichend sind, um die erhöhten Anforderungen im Kontext der produktfamilienbasierten Software-Entwicklung zu erfüllen. Insbesondere existiert keine durchgängige Methodik die eine sinnvolle Reduktion des Testaufwandes und die effiziente Anpassung der Tests bei Änderungen ermöglicht.

Eine an der Universität Essen derzeit in Entwicklung befindliche Erweiterung des V-Modells hat zum Ziel während jeder Entwicklungsphase sog. Testszenarien abzuleiten, die ihrerseits zur Definition von Systemtests und Integrationstests herangezogen werden können. Das besondere Augenmerk liegt auf den frühen Phasen der Software-Entwicklung, wie der Anforderungsdefinition und dem Architekturentwurf.

2. Ein Vorgehen zur szenariobasierten Definition von Testfällen

Im folgenden geben wir einen Überblick über die Erweiterung des V-Modells zur Ermöglichung einer kontinuierlichen, szenariobasierten Definition von Testfällen.

2.1. Das V-Modell

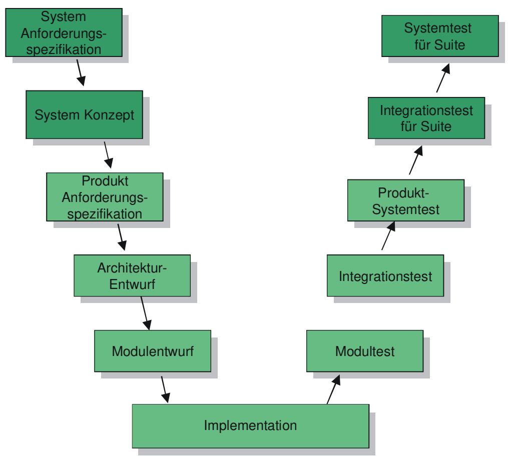

Abbildung 1: Das V-Modell

Das V-Modell (vgl. Abbildung 1) ist ein bekannter Standard für die Entwicklung von IT-Systemen und seit Juni 1996 in Deutschland vorgeschrieben für die Erstellung von IT-Systemen des Bundes (vgl. [VMod97]). Das V-Modell umfasst ein Prozessmodell, eine Methodenzuordnung sowie funktionale Werkzeuganforderungen. Es legt fest, welche Schritte im Einzelnen durchzuführen sind, welche Methoden innerhalb der verschiedenen Entwicklungsaktivitäten zur Anwendung gelangen und welche funktionalen Merkmale die verwendeten Werkzeuge besitzen müssen. Ein besonderes Charakteristikum des V-Modells ist die Möglichkeit zur Anpassung an verschiedenste Projekte und Organisationen ("Tailoring").

Bei SHS werden Software-Produkte und Software-Systeme, wie in Abschnitt 1 definiert, unterschieden. Nachfolgend soll nun jeweils kurz auf diejenigen Phasen eingegangen werden, die sich besonders auf die Erstellung von Software-Systemen beziehen (vgl. Abbildung 1):

- Die Phase *System-Anforderungsspezifikation* dient der Beschreibung der Anforderungen an ein aus mehreren Software-Produkten zusammengesetztes Software-System.
- Die Phase *Systemtest für Suite* korrespondiert mit der Phase System-Anforderungsspezifikation und ist für die Durchführung der Testszenarien vorgesehen, die das Gesamtsystem ("die Suite") auf seine korrekte Funktion überprüfen.
- In der Phase *System-Konzept* werden die Beziehungen zwischen den verschiedenen Software-Produkten beschrieben (z.B. Interaktionen und die physikalische Verteilung). In anderen Worten: es wird eine abstrakte System-Architektur beschrieben, in der die einzelnen Software-Produkte als Architekturbausteine fungieren.
- Die Phase *Integrationstest für Suite* korrespondiert mit der Phase System-Konzept und beinhaltet die Durchführung der Testszenarien, die die Integration von zwei oder mehreren Software-Produkten adressieren.

2.2. Szenariobasierte Ableitung von Tests

Mit Hilfe des 1,5 V-Modells (vgl. Abbildung 2) können in jeder Entwicklungsphase Testszenarien abgeleitet werden, die in der jeweils korrespondierenden Testphase ausgeführt werden müssen.

Die Bezeichnung "1,5 V-Modell" rührt von den zusätzlichen, in Abbildung 2 erkennbaren Aktivitäten her, die parallel zu den Entwurfs- und Erstellungsphasen verlaufen und speziell auf die kontinuierliche Ableitung von Testszenarien und die Aufzeichnung von sog. Traceability Informationen (vgl. z.B. [GoFi94]), zur effektiven Unterstützung von Change Management Prozessen, ausgerichtet sind. Im Rahmen der produktfamilienbasierten Entwicklung liegt hierbei ein besonderer Schwerpunkt auf der Erstellung generischer Test- und Traceability-Artefakte, die bei der Erstellung jeder Produktfamilien-Instanz wiederverwendet werden können.

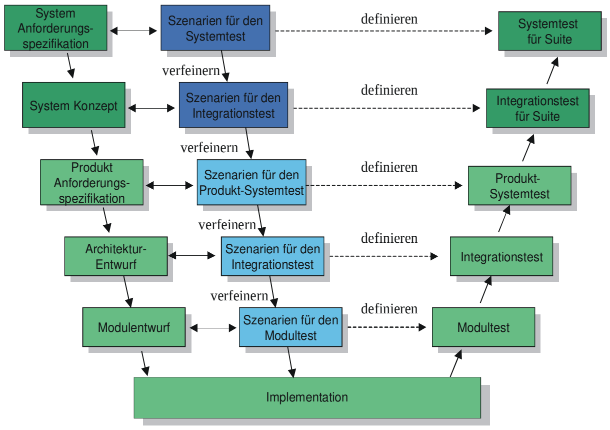

Abbildung 2: Das 1,5 V-Modell

Wie bereits kurz erwähnt zielt das in Abbildung 2 dargestellte 1,5 V-Modell u.a. darauf ab in jeder Phase der Software-Entwicklung Testszenarien abzuleiten, die in der korrespondierenden Testphase ausgeführt werden. Auf diese Weise kann sichergestellt werden, das die entsprechenden Tests in zeitlicher Nähe zu der Spezifikation entworfen werden, deren Erfüllung sie im Anschluss überprüfen sollen. Hierdurch kann die Gefahr verringert werden, dass wichtige Testfälle evtl. vergessen oder nur unzureichend spezifiziert werden. Weiterhin wird durch dieses Vorgehen die kontinuierliche Aufzeichnung der Beziehungen (Links) zwischen Entwurfsartefakten und den zugehörigen Testfällen gewährleistet. Diese Beziehungen ermöglichen im Fall einer Änderung die unmittelbare Bestimmung der evtl. betroffenen Testszenarien und bilden somit die Grundlage für ein effizientes Change Management mit Bezug auf Testfälle.

Dieses Vorgehen bietet somit auch eine direkte Unterstützung der sog. "Feedback-Schleife" zwischen Tests und Änderungen, wobei davon ausgegangen wird das Tests zu neuen Änderungswünschen und durchgeführte Änderungen zu einer erneuten Testinstanziierung führen.

Zwischen den Entwurfsartefakten wie Anforderungen oder Architekturkomponenten und den entsprechenden Testszenarien besteht eine n:m Beziehung. Das bedeutet, dass die korrekte Erfüllung einer Anforderung durch ein oder mehrere Testszenarien überprüft wird, während das gleiche Szenario u.U. eingesetzt werden kann, um die richtige Umsetzung mehrerer Anforderungen bzw. von Teilen verschiedener Anforderungen nachzuweisen. Die in Abbildung 2 sichtbaren (vertikalen) Pfeile zwischen den verschiedenen Ebenen der Testszenarien deuten entsprechend an, dass Szenarien einer höheren Ebene durch weitere Testszenarien auf einer tieferen Ebene verfeinert werden können. Eine Verfeinerung bedeutet immer eine Detaillierung der zugehörigen Testszenarien.

Entsprechend existieren Teile eines Testszenarios, die komplett auf einer Testebene ausgeführt werden, während andere Teile evtl. über mehrere Ebenen verteilt getestet werden müssen. Mit anderen Worten: Ein Szenario auf der Ebene des Produkt-Systemtests wird mit hoher Wahrscheinlichkeit durch ein oder mehrere Test-Szenarien auf der Ebene des Integrationstests verfeinert. Durch diese Verfeinerungsbeziehungen wird ein "Leveling" der Testszenarien erreicht. Dies erleichtert wiederum die Abschätzung des Testaufwandes sowie die Propagierung von Änderungen in das Testmodell.

Des weiteren besteht die einfache Möglichkeit z.B. nur die Testfälle auf der Ebene des Produkt-Systemtests auszuführen ("Test in die Breite") oder einen bestimmten Produkt-Systemtest auszuwählen und inklusive aller assoziierten Detailszenarien zu testen ("Test in die Tiefe").

2.3. Fokus der Kooperation

Fokus der Kooperation mit SHS ist die Ableitung von Testfällen in den frühen Produkt-Entwicklungsphasen, d.h. während der "Produkt-Anforderungsspezifikation" bzw. des "Use-Case-Entwurfs" und des "Architekturenwurfs". Hierzu werden zunächst die in Form von Szenarien modellierten Anforderungen analysiert, um anschließend geeignete Testfälle daraus ableiten zu können. Durch die fortlaufende Verfeinerung der Szenarien soll erreicht werden, dass die korrekte Erfüllung der Kundenanforderungen durch das resultierende Software-Produkt effizient getestet werden kann.

Da nicht nur die Anpassung eines bereits vorhandenen Artefakts sondern z.B. auch das Hinzufügen eines komplett neuen Artefakts als Änderung betrachtet wird (vgl. z.B. [Wein97]), kann die Ableitung einer speziellen Anwendung aus den generischen Artefakten einer Software-Produktfamilie ebenfalls als Änderungsprozess betrachtet werden. Aus diesem Grund muss zudem berücksichtigt werden wie sich Änderungen auf Use-Case- und Architekturebene auf die zugehörigen Testfälle auswirken. Um diese Auswirkungen korrekt abschätzen und in die betroffenen Artefakte integrieren zu können, ist es besonders wichtig, dass geeignete sog. Tracebility-Links zur Verfügung stehen. Um diese Links sowie die damit zusammenhängenden Daten in ähnlichen Änderungsfällen wiederverwenden zu können, müssen entsprechende Link-Typen identifiziert und geeignet beschrieben werden. Die durch diese Links definierten Beziehungen sowie die zugehörigen Attributwerte werden dauerhaft gespeichert, um Nachvollziehbarkeit zu gewährleisten und somit das Change-Management zu unterstützen. Nachfolgend werden einige einfache Typen von Beziehungen beispielhaft in kurzer Form charakterisiert:

- *erfüllt* zwischen Teilarchitektur und Anforderung
- *realisiert* zwischen Funktions-Spezifikation und Anforderung
- *abgedeckt* zwischen Test und Anforderung
- *verfeinert* zwischen Test und Test

Um die Aufzeichnung sinnvoller Trace-Informationen zu ermöglichen, werden geeignete Strukturen benötigt zwischen denen Beziehungen (Links) hergestellt werden können. Auf diese Weise wird die schnelle Identifikation der evtl. von einer Änderung betroffenen Artefakte (bzw. der betroffenen Strukturen innerhalb dieser Artefakte) ermöglicht. Die Änderung einer Vorbedingung in der Beschreibung eines bestimmten Use-Case könnte z.B. Auswirkungen auf die Schnittstellen der zugehörigen Architekturkomponenten haben. Ein entsprechender Link zwischen Vorbedingungen und Schnittstellen ermöglicht somit die schnelle Propagierung einer solchen Änderung und erleichtert die Abschätzung des zu erwartenden Änderungsaufwandes. Links zu den entsprechenden Testszenarien gewährleisten zudem die umgehende Abschätzung des durch eine bestimmte Änderung verursachten Testaufwandes und ermöglichen das gezielte Testen der geänderten Strukturen. Außerdem wird so die Propagierung von Änderungen in die Testszenarien ermöglicht, was wiederum zu einer Beschleunigung der evtl. nötigen Anpassung von Testszenarien an geänderte Entwurfsartefakte führt. Für die Identifikation geeigneter Link-Typen wurde das nachfolgend verkürzt dargestellte Vorgehen gewählt:

- Definition jeweils eines Metamodells für die Beschreibung von Use-Cases, Architekturartefakten und Testszenarien.
- Identifikation der Beziehungen zwischen den verschiedenen Bestandteilen zwischen Use-Case- und Architekturmetamodell auf der einen und dem Test-Metamodell auf der anderen Seite.

Auf diese Weise konnten Link-Typen zur Beschreibung von Beziehungen zwischen funktionalen Anforderungen (in Form von Use-Cases) und Testszenarien sowie Architekturartefakten und Testszenarien abgeleitet werden.

2.4. Vorteile

Ein szenariobasiertes Vorgehen für die Ableitung von Testfällen weißt folgende Vorteile auf:

- Sinnvolle Reduktion der Testfälle durch Orientierung an Kundenanforderungen.
- Explizite Berücksichtigung von Tests in jeder Entwicklungsphase.
- Fortlaufende Verfeinerung der Testszenarien und somit verbesserte Möglichkeit zur Erreichung einer guten Testabdeckung (in Bezug auf Kundenanforderungen).
- Unterstützung von Change Management Aktivitäten für Testfälle durch die parallel aufgezeichneten Traceability-Informationen.

3. Zusammenfassung und Ausblick

Im Rahmen der Kooperation unserer Forschungsgruppe mit dem Unternehmen SHS wurde die Entwicklung des 1,5 V-Modells als Erweiterung des V-Modell Standards [VMod97] initiiert. Ziel der Kooperation ist insbesondere die sinnvolle Reduktion des Testaufwands. Dies gilt sowohl für die Durchführung als auch für die Anpassung von Tests an Änderungen.

Der von uns gewählte Use-Case- bzw. szenariobasierte Ansatz ermöglicht die explizite Orientierung an Kundenwünschen und führt zu einer kundenspezifischen Selektion von Testfällen. Der besondere Fokus des 1,5 V-Modells liegt auf der expliziten und kontinuierlichen Berücksichtigung von Tests in jeder Phase der Entwicklung sowie der Aufzeichnung von entsprechenden Traceability-Informationen. Im Kontext der Produktfamilien-Entwicklung soll hierbei insbesondere auf die Erstellung generischer Traceability- und Test-Artefakte geachtet werden, die bei der Ableitung einer neuen Instanz der Produktfamilie wiederverwendet werden können. Der Ansatz zur Adaption des V-Modells wurde gewählt, da sich SHS aktuell in einer Übergangsphase von der traditionellen zur produktfamilienbasierten Software-Entwicklung befindet und das vielfach bewährte V-Modell auch weiterhin verwenden möchte.

Das initiale 1,5 V-Modell befindet sich zur Zeit in der Erprobung und wird im Detail kontinuierlich angepasst bzw. fortentwickelt. Die bisherigen Erfahrungen zeigen, dass das 1,5 V-Modell ein sinnvolles Vorgehen zur Reduktion des Testaufwandes im Rahmen der produktfamilienbasierten Software-Entwicklung liefert.

Zukünftige Arbeiten werden sich vor allem auf die weitere Betonung der Wiederverwendbarkeitsaspekte der verschiedenen Traceability- und Test-Artefakte sowie auf die Entwicklung einer szenariobasierten Methodik zur zielgerichteten Anpassung und Durchführung von Tests bei der Entwicklung von produktfamilienbasierten, kundenspezifischen Anwendungen konzentrieren.

4. Danksagung

Wir möchten uns für die intensive und kooperative Zusammenarbeit mit SHS und insbesondere bei Norbert Herzog, Josef Weingärtner und Christian Zapf bedanken. Diese Arbeit wurde im Rahmen des ESAPS Projekts teilweise durch das BMBF finanziert (Eureka ∑! 2023 Programme, ITEA project 99005).

5. Literatur

[GoFi94] Gotel, O.C.Z.; Finkelstein, A.C.W.: “An Analysis of the Requirements Traceability Problem”, In: Proc. of the 1st Int. Conference on Requirements Engineering, IEEE CS Press, 1994

[Harr00] Harrold, M.J.: "Testing: A Roadmap", In: A. Finkelstein (editor), The Future of Software Engineering, ACM Press, 2000

[Hetz88] Hetzel, B.: "The complete Guide to Software Testing - Second Edition", John Wiley & Sons, 1988

[JaBC98] Jarke, M.; Bui, X.T.; Carroll, J.M.: “Scenario Management: An Interdisciplinary Approach“, In: Requirements Engineering Journal, Springer-Verlag, 1998

[Jaco92] Jacobson, I.: “Object-Oriented Software Engineering : A Use Case Driven Approach”, Addison-Wesley, 1992

[Myer79] Myers, G.J.: "The Art of Software Testing", John Wiley & Sons, 1979

[VMod97] Das V-Modell - Entwicklungsstandard für IT-Systeme des Bundes, 1997, http://www.v-modell.iabg.de/, Zugriff Sept. 2000

[VMMZ97] V-Modell - Methodenzuordnung, allgemeiner Umdruck Nr. 251, 1997, http://www.v-modell.iabg.de/vm97.htm

[Wein97] Weinberg, G.M.: " Quality Software Management Volume 4: Anticipating Change", Dorset House Publishing, 1997

[Whit00] Whittaker, J.A.: "What is Software Testing ? And Why It Is So Hard ?", IEEE Software January/February 2000

[WPJH98] Weidenhaupt, K; Pohl, K.; Jarke, M.; Haumer, P.: “Scenario Usage in System Development: A Report on Current Practice“, In: IEEE Software, March 1998