Sascha Lity

(lity@ips.cs.tu-bs.de)
ist Doktorand an der Technischen Universität Braunschweig. Er erhielt 2011 seinen Master of Science und arbeitet seit 2012 im Institut für Programmierung und Reaktive Systeme als wissenschaftlicher Mitarbeiter. Zu seinen Forschungsinteressen gehört das modellbasierte Testen variantenreicher und evolvierender Softwaresysteme. Herr Lity wird im Rahmen des DFG-Schwerpunktprogramms SPP 1593: Design For Future – Managed Software Evolution gefördert.

Remo Lachmann

(r.lachmann@tu-bs.de)
ist Doktorand an der TU Braunschweig, wo er 2012 seinen Master of Science erhielt. Er arbeitet seit November 2012 als wissenschaftlicher Mitarbeiter am Institut für Softwaretechnik und Fahrzeuginformatik unter Leitung von Prof. Dr. Ina Schaefer. Zu seinen Forschungsinteressen gehört das Testen variantenreicher Systeme sowie das Testen von Fahrerassistenzsystemen.

Malte Lochau

(Malte.Lochau@es.tu-darmstadt.de)
ist seit Oktober 2012 Postdoc am Fachbereich Echtzeitsysteme von Prof. Andy Schürr an der TU Darmstadt. In der Forschungsgruppe zu Themen aus dem Bereich Softwareproduktlinien arbeitet er unter anderem im DFG-Projekt IMoTEP und dem SFB 1053 MAKI.

Michael Dukaczewski

(m.dukaczewski@tu-bs.de)
forscht seit 2009 an der TU Braunschweig im Bereich modellgetriebener Softwareentwicklung von mobilen und Webapplikationen. Seit 2012 ist er am Institut für Softwaretechnik und Fahrzeuginformatik tätig.

Ina Schaefer

(i.schaefer@tu-bs.de)
leitet das Institut für Softwaretechnik und Fahrzeuginformatik an der TU Braunschweig. Nach ihrer Promotion an der TU Kaiserslautern war sie Postdoc an der Chalmers University of Technology in Göteborg, Schweden. Zu ihren Forschungsinteressen gehören Verifikations- und Testmethoden für variantenreiche und evolvierende Softwaresysteme.

Delta-orientiertes Testen von variantenreichen Systemen

Testen gehört zu den essenziellen Qualitätssicherungsverfahren für moderne Softwaresysteme. Variantenreiche Softwaresysteme wie Softwareproduktlinien bringen Herausforderungen für die Qualitätssicherung durch ihren hohen Anteil an Variabilität und die dadurch steigende Komplexität mit sich. Sie erfordern Testmethoden, die ein verlässliches und zugleich effizientes Testen ermöglichen. Delta-orientiertes Testen betrachtet Unterschiede zwischen Produktvarianten innerhalb einer Produktlinie und zielt auf die systematische Wiederverwendung von Testfällen ab. In diesem Artikel wird die Methode des delta-orientierten modellbasierten Testens auf Komponenten- und Integrationstestebene sowie ein delta-orientiertes anforderungsbasiertes Systemtestverfahren vorgestellt.

Herausforderungen beim Testen variantenreicher Systeme

Moderne Softwaresysteme werden heutzutage oftmals in Form von *Softwareproduktlinien (SPLs)* entwickelt (vgl. [Poh05]). Der Einsatz von SPLs wird dabei maßgeblich durch den stetig steigenden Bedarf an kundenspezifisch konfigurierbaren Produkten sowie deren Herstellung (engl. *Mass Customization*) motiviert (vgl. [Poh05]). In der Automobilindustrie wird zum Beispiel auf Basis von Produktkonfiguratoren einem Kunden ermöglicht, ein Fahrzeug den eigenen Wünschen entsprechend anzupassen. Die Konfiguratoren bieten dabei die Möglichkeit, optionale Features flexibel an- und abzuwählen. Ein Beispiel für eine SPL aus dem Automobilbereich ist die Fallstudie *Body Comfort System (BCS)* (vgl. [Lit12]), in der ein Komfortsystem für Fahrzeuge als SPL definiert ist. Darin sind verschiedene optionale Funktionen enthalten, die durch einen Kunden ausgewählt werden können, z. B. eine Zentralverriegelung, ein Alarmsystem oder ein Funkschlüssel.

Diese modularisierte Art, ein Produkt zu gestalten, führt je nach Anzahl der angebotenen Features zu einer exponentiell steigenden Anzahl von möglichen Produktvarianten. Die Anzahl der potenziellen Produkte übersteigt dabei die Menge der möglichen Kunden meist um ein Vielfaches. In der BCS-Fallstudie lassen sich beispielsweise aus ca. 20 wählbaren Features über 11.000 verschiedene Produktvarianten erzeugen. Es lässt sich somit während der Entwicklung der Produktlinie nicht vorhersehen, welche Produktkonfigurationen tatsächlich produziert werden müssen. Da ausführliches Testen vor allem im Bereich sicherheitskritischer Software essenziell ist, müssten alle Produktvarianten vorab getestet werden. Auf Grund der sehr hohen Anzahl von Produkten ist das vollständige Testen jeder einzelnen Produktvariante jedoch wirtschaftlich nicht realisierbar. Daher werden Methoden benötigt, mit denen der Testaufwand für variantenreiche Systeme reduziert und gleichzeitig entsprechenden Normen und Vorgaben gerecht werden kann.

Für die Entwicklung von SPLs wurde die delta-orientierte Softwareentwicklung als sprachenunabhängiger und modularer Ansatz vorgestellt (vgl. [Sch10]). Der delta-orientierte Entwicklungsansatz sieht dabei vor, Produktvarianten durch deren Unterschiede zu einem Kernsystem in Form von Deltas zu definieren. Deltas beschreiben die Transformation des Kernsystems durch das Hinzufügen, Entfernen und Modifizieren von Elementen. Je nach Domäne können die erlaubten Operationen und somit deren Bedeutung variieren. Im Bereich von textuellen Anforderungen fügen Deltas beispielsweise Anforderungen für verschiedene Produkte hinzu oder entfernen diese. Auf Softwarearchitekturebene werden Deltas für Softwarekomponenten definiert. Betrachtet man Deltas auf Statecharts, ist es möglich, Zustände hinzuzufügen oder zu entfernen.

Dieser Variabilitätsmodellierungsansatz kann für ein delta-orientiertes Testverfahren aufgegriffen werden, um eine Reduktion des Testaufwands beim Testen von SPLs durch Deltas auf Testartefakten zu erreichen.

Delta-orientierte Testmethodik

Die grundlegende Idee des delta-orientierten Testverfahrens ist die Wiederverwendung von Testfällen und die systematische Testfallauswahl für eine Menge von Produktvarianten. Da SPLs auf Gemeinsamkeiten zwischen Produktvarianten aufbauen, ist eine Testfallwiederverwendung zwischen Varianten immer dann möglich, wenn die durch diese Testfälle geprüfte Funktionalität eine Gemeinsamkeit dieser Varianten darstellt. Dies führt zu einer Ersparnis bei der Testfallerstellung und -ausführung, was eine Reduktion des Testaufwands mit sich bringt.

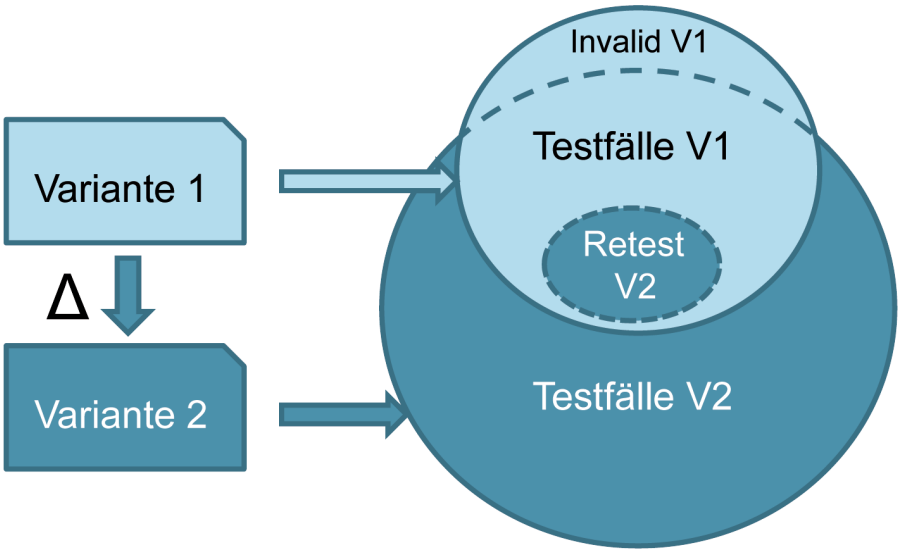

Abb. 1: Delta-orientierte Kategorisierung von Testfällen analog zu Regressionstestverfahren.

Da die Anzahl aller möglichen Produktvarianten zu hoch ist, um jede einzeln zu testen, beginnt das delta-orientierte Testverfahren mit einer Auswahl von zu testenden Produktvarianten. Bereits hier kann der Testaufwand reduziert werden, indem man die Auswahl zu testender Produkte möglichst gering hält. In diesem Kontext wurden kombinatorische Auswahlverfahren in der Literatur vorgestellt (vgl. [Per12]). Diese verwenden einen *pairwise feature coverage*-Ansatz, der die paarweisen Interaktionen von Features abdeckt, indem die Menge der zu testenden Produkte anhand der auftretenden Kombinationsmöglichkeiten aller Paare von Featureparametern festgelegt wird. Anschließend wird in der delta-orientierten Testmethodik das Kernprodukt als erstes zu testendes Produkt gewählt. Dieses Kernprodukt wurde bereits bei der Entwicklung der SPL beliebig gewählt, wobei bestimmte Faktoren die Wahl des Kerns positiv beeinflussen können. Beispielsweise kann es aus wirtschaftlicher Sicht von Vorteil sein, die größte oder kleinste Produktvariante zu wählen oder diejenige Produktvariante als Kern zu definieren, die als erstes ausgeliefert werden soll. Auf Basis des Kernprodukts wurden ebenfalls in der Entwicklungsphase Deltas definiert, um die Generierung möglicher Varianten zu ermöglichen. Diese Menge an Deltas wird in der Testmethodik verwendet, um explizit Gemeinsamkeiten und Unterschiede zwischen Varianten zu beschreiben. Nachdem das Kernprodukt getestet wurde, werden die restlichen Produkte inkrementell getestet, wobei ein hoher Grad an Wiederverwendung von Testfällen angestrebt wird.

Das delta-orientierte Testverfahren ermöglicht es, gezielt wiederverwendbare Testfälle zu identifizieren, indem die Testfälle für die unterschiedlichen Produktvarianten nach ihrer Anwendbarkeit kategorisiert werden. Die Kategorisierung ist angelehnt an Regressionstestansätze und ist in Abbildung 1 schematisch dargestellt.

Die Abbildung zeigt die Testfallmenge für zwei Produktvarianten, wobei Variante 2 durch die Anwendung eines Deltas aus der ersten Produktvariante, z. B. dem Kern, hervorgeht. Zwischen den beiden Testfallmengen gibt es eine Überschneidung. Diese Testfälle werden als *wiederverwendbar* bezeichnet. Einige Testfälle aus Variante 1 sind nicht erneut für Variante 2 zu verwenden, da ihre Voraussetzungen nicht mehr erfüllt sind, d. h. dass z. B. das zu testende Verhalten nicht mehr in

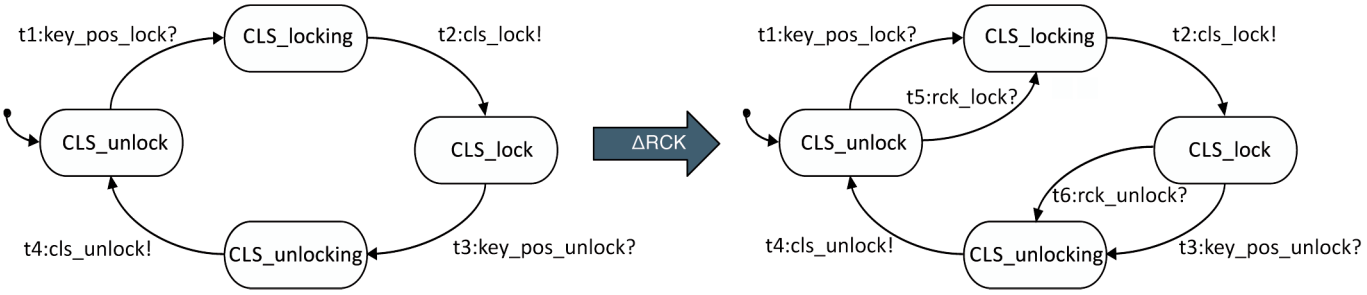

Abb. 2: Delta-orientierter Komponententest auf Basis von variablen Statechart-Testmodellen.

Variante 2 vorhanden ist. Die Testfälle sind daher ungültig (*Invalid*) für Variante 2. Die Menge der *neuen* Testfälle enthält jene Testfälle, die für Variante 2 gültig sind und zuvor nicht existierten. Innerhalb der Menge wiederverwendbarer Testfälle befindet sich die Teilmenge der erneut zu testenden Testfälle (*Retest*). Angelehnt an das Regressionstesten soll mit dieser Testfallmenge sichergestellt werden, dass bereits getestetes Verhalten aufgrund von Modifikationen nicht ungewünscht verändert wurde. Eine Herausforderung besteht dabei in der Auswahl eines geeigneten Retest-Kriteriums. Wir beschreiben einige Kriterien exemplarisch in den kommenden Abschnitten. Insgesamt müssen für jede Produktvariante die Menge der neuen und die erneut durchzuführenden Testfälle ausgeführt werden.

Das bisher abstrakt beschriebene delta-orientierte Testverfahren wurde für Komponenten-, Integrations- und Systemtests adaptiert. Je nach Testphase ergeben sich demnach unterschiedliche Ausprägungen des Testansatzes. Die Beispiele wurden der BCS-Fallstudie entnommen (vgl. [Lit12]).

Delta-orientiertes Komponententestverfahren

Das delta-orientierte Komponententestverfahren (vgl. [Loc12]) kombiniert das modellbasierte Testen mit dem Regressionstesten und verwendet Statecharts als Testmodelle. Statecharts beschreiben das interne Verhalten der zu testenden Komponenten. Auf Basis der Testmodelle können Testfälle für die zu testende Komponente anhand gewisser Abdeckungskriterien automatisch erstellt werden. Ein vereinfachtes Beispiel für ein Statechart aus der BCS-Fallstudie für die Komponente Zentralverriegelung (*Central Locking System (CLS)*) ist in Abbildung 2 dargestellt.

Das Statechart besteht aus vier Zuständen, die durch vier Transitionen miteinander verbunden sind. Transitionen können durch empfangene Signale ausgelöst werden. Dies wird in der Abbildung durch ein „?“ dargestellt. Sie können ebenfalls Aktionen auslösen, die durch ein „!“gekennzeichnet sind. Im Beispiel aus Abbildung 2 löst die Zentralverriegelung beim Abschließen des Fahrzeugs (*t1:key_pos_lock?*) die Aktivierung des CLS aus (*t2:cls_lock!*). Analog ist die Deaktivierung durch das Aufschließen des Fahrzeugs beschrieben.

In der Fallstudie kann die Zentralverriegelung um das Feature Funkschlüssel (*Remote Control Key (RCK)*) erweitert werden, damit die Aktivierung/Deaktivierung ebenfalls per Funk gesteuert werden kann. Die entsprechende Funktionalität wird durch ein Delta innerhalb der CLS-Komponente hinzufügt. Das Resultat ist in Abbildung 2 auf der rechten Seite dargestellt. Das Delta fügt in diesem Fall die zwei Transitionen *t5* und *t6* hinzu.

Durch Deltas auf Statecharts können für jede Produktvariante die Unterschiede im Verhalten zwischen den Varianten beschrieben werden. Dafür ist ein geeignetes Kernmodell zu wählen. Anschließend werden die Produktvarianten durch die Anwendung von Deltas erzeugt.

Für den Test müssen dann zunächst Testfälle aus dem Kernmodell, beispielsweise in Form von Transitionspfaden, generiert werden. Hierbei gibt ein Abdeckungskriterium wie Zustands- oder Transitionsabdeckung vor, wie viele Testfälle automatisch generiert werden sollen. Um Testfälle für weitere zu testende Varianten

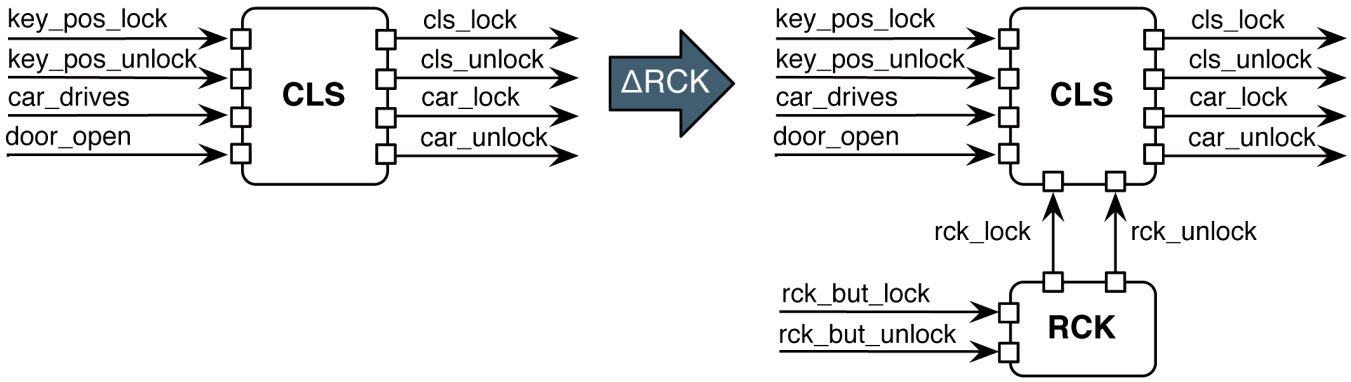

Abb. 3: Delta-orienterter Integrationstest auf Basis von variablen Architekturmodellen.

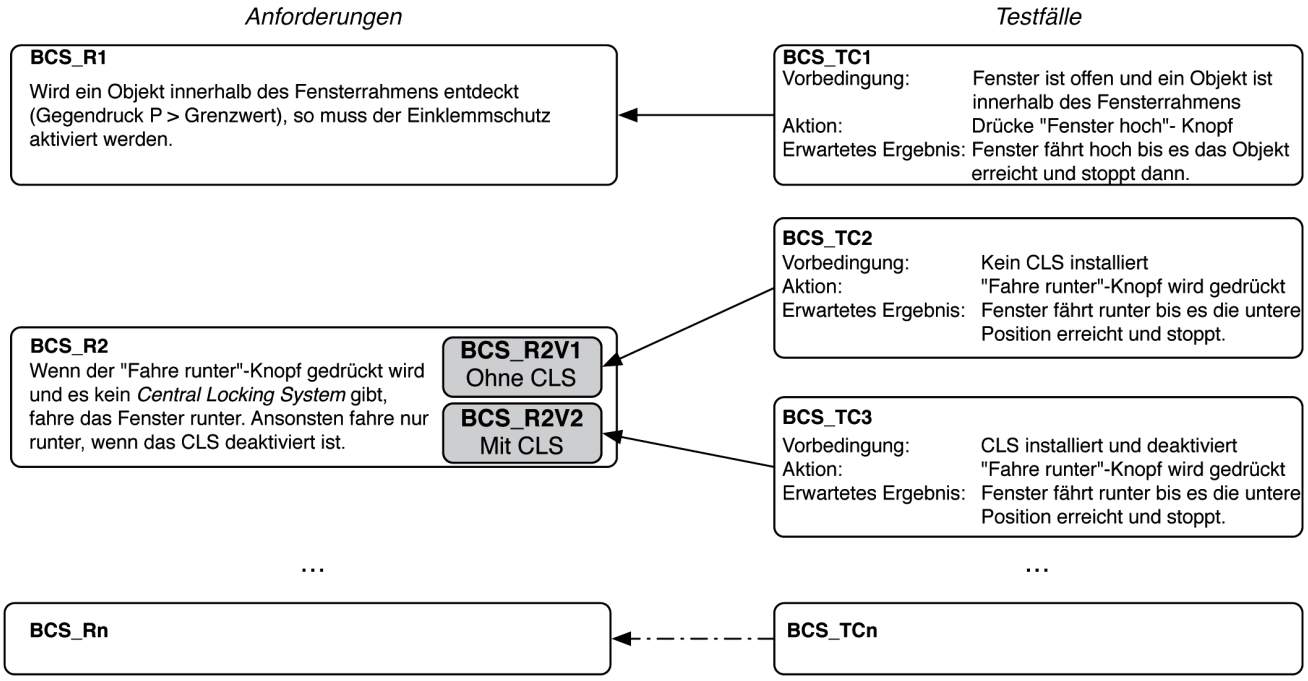

Abb. 4: Delta-orientierter Systemtest auf Basis von variablen Anforderungsbeschreibungen.

wiederzuverwenden, müssen die zum Testfall gehörenden Elemente (Zustände/Transitionen) in der betrachteten Variante weiterhin vorhanden sein. Anschließend muss ermittelt werden, welche Teilmenge der wiederverwendbaren Testfälle erneut ausgeführt werden muss. Hier können beispielsweise Statecharts-Slicing-Methoden verwendet werden, um betroffene Testfälle zu ermitteln (vgl. [Kam12]). Zusätzlich müssen für die Variante neue Testfälle erstellt und ausgeführt werden, wenn neue Zustände oder Transitionen hinzugefügt wurden.

Delta-orientiertes Integrationstestverfahren

Integrationstestverfahren testen die Interaktion zwischen Komponenten. Es wird davon ausgegangen, dass die einzelnen Komponenten bereits in Komponententests getestet wurden. Das hier vorgestellte Integrationstestverfahren basiert auf Softwarearchitekturbeschreibungen (vgl. [Lit12]). Architekturmodelle bestehen aus *Komponenten* und *Konnektoren*. Komponenten stellen Softwareteile dar, die eine geschlossene Einheit bilden und nach außen über *Ports* mit anderen Komponenten kommunizieren. Ports senden oder empfangen Signale, die von Konnektoren übertragen werden. Abbildung 3 stellt auf der linken Seite als einen Ausschnitt aus einer Softwarearchitektur die CLS-Komponente vor, welche die Funktionalität des Features *Zentralverriegelung* beinhaltet.

Der delta-orientierte Integrationstestansatz hat zum Ziel, möglichst wenig Interaktion zwischen Komponenten mehrfach ohne Änderungen zu testen. Zu diesem Zweck werden Deltas auf Ebene der Softwarearchitekturen definiert. Deltas erlauben es, Komponenten oder Konnektoren hinzuzufügen oder zu entfernen. Ein Beispiel für ein angewendetes Delta ist in Abbildung 3 schematisch dargestellt. Darin wird das Delta *Delta RCK* auf den Architekturausschnitt links angewendet. Das Delta fügt zunächst eine Komponente RCK zur Architektur hinzu. Im Anschluss definiert es noch zwei Änderungen, um jeweils zwei Eingangs-und Ausgangskonnektoren für RCK hinzufügen.

Durch die Anpassung der Architekturmodelle mittels Deltas kann eine Reduktion des Testaufwands durch die Wiederverwendung von Teilarchitekturen und damit zusammenhängenden Testfällen erzielt werden. Um dies zu erreichen, werden zunächst Testfälle für das Kernprodukt definiert. Die Testfallerstellung erfolgt für alle Komponenten und Konnektoren einer Architekturvariante. Die Interaktion dieser Elemente wird durch Szenarien in Sequenzdiagrammen modelliert. Ein Sequenzdiagramm beschreibt Interaktionsszenarien zwischen Komponenten.

Nachdem die Testfälle für das Kernprodukt definiert wurden, werden inkrementell die Testfälle der anderen Produktvarianten erstellt und kategorisiert. Die Kategorisierung erfolgt ähnlich zu dem für Statecharts vorgestellten Verfahren. Ein Testfall bzw. Sequenzdiagramm kann immer dann wiederverwendet werden, wenn die von ihm getesteten Komponenten und Konnektoren in dem zu testenden Produkt vorhanden sind. Andernfalls ist der Testfall für die untersuchte Produktvariante ungültig. Bei wiederverwendbaren Testfällen gilt es zu entscheiden, ob ein Testfall erneut ausgeführt werden muss. Beispielsweise kann geprüft werden, ob sich die Schnittstellen der zu testenden Komponenten im Vergleich zu vorherigen Produkten geändert haben. In diesem Fall wäre der Testfall erneut auszuführen, da eine Änderung an den Komponenten zu unerwünschtem Verhalten führen kann. Zusätzlich werden neue Testfälle für die in der Architekturvariante neu hinzugekommenen Komponenten und Konnektoren erstellt und ausgeführt.

Delta-orientiertes Systemtestverfahren

Die vorgestellten Komponenten- und Integrationstestverfahren basieren auf Verhaltens- bzw. Strukturmodellen. Derartige Spezifikationsmodelle liegen in der Praxis

jedoch oft nicht vor. Stattdessen werden Spezifikationen meist in natürlicher Sprache erfasst. Für diesen Bereich wurde daher ein anforderungsbasiertes delta-orientiertes Systemtestverfahren entwickelt. Das Verfahren baut auf Deltas auf, welche nach textuellen Anforderungen definiert werden (vgl. [Duk13]).

Anforderungen sind gewöhnlich mit einer Menge von Testfällen verknüpft, was die Traceability zwischen Anforderungen und ihren zugehörigen Testfällen ermöglicht. Wurde eine Anforderung beispielsweise verändert, können die betroffenen Testfälle ermittelt werden. Traceability wird in der Praxis häufig durch Werkzeuge wie IBM Rational DOORS unterstützt. Klassisches Anforderungsmanagement erfasst sämtliche Anforderungen für alle Produktvarianten in einem Dokument und kombiniert diese mit den entsprechenden Testfällen. Das delta-orientierte Testverfahren unterscheidet Anforderungen für die verschiedenen Produktvarianten. Da Anforderungen für viele Produktlinien bereits bestehen und für neue Generationen von Produkten adaptiert werden, existiert bereits eine Menge von Anforderungen und Testfällen.

Beim delta-orientierten Systemtestverfahren werden zunächst die Anforderungen des Kernprodukts extrahiert. Dies ist beispielsweise möglich, indem man Anforderungen zu Features zuordnet. Anforderungen, welche für mehrere Features definiert sind, müssen entsprechend der Features aufgeteilt und den jeweiligen Features explizit zugeordnet werden. Ein Beispiel hierfür ist in Abbildung 4 zu sehen. Dort wird die Anforderung *BCS_R2* in die beiden Anforderungen *BCS_R2_V1* und *BCS_R2_V2* unterteilt. Sie unterscheiden sich darin, ob das CLS-Feature in der untersuchten Produktvariante vorhanden ist oder nicht.

Alle weiteren Anforderungen werden ebenfalls den für sie gültigen Produktvarianten zugeordnet. Für sie werden Deltas definiert, welche Anforderungen hinzufügen oder entfernen. Das Modifizieren von Anforderungen ist durch das Entfernen und anschließende Hinzufügen der modifizierten Anforderung möglich.

Anschließend werden die einzelnen Produktvarianten getestet. Dabei geht man iterativ von einer Produktvariante zur nächsten und berechnet die Menge der Anfoderungen für die nächste Produktvariante anhand der Deltas. Die Testfälle, die mit den Anforderungen der aktuellen Produktvariante verknüpft sind, werden für die untersuchte Variante in die vier vorgestellten Testfallkategorien *Neu*, *Invalid*, *Wiederverwendbar* und *Retest* eingeteilt.

Wiederverwendbare Testfälle erfordern das Vorhandensein der entsprechenden Anforderungen in der zu testenden Produktvariante. Sind diese nicht vorhanden, so ist ein Testfall ungültig. Testfälle, die für die aktuelle Produktvariante erstmalig erstellt wurden, werden in die Kategorie der neuen Testfälle aufgenommen und ausgeführt. Für die Bestimmung der erneut zu testenden Testfälle aus der Menge der wiederverwendbaren Testfälle können verschiedene Kriterien verwendet werden. Denkbar ist z. B. die zufällige Auswahl, die Betrachtung von Metadaten wie der Kritikalität einer Anforderung oder der Testhistorie (vgl. [Duk13]).

Zusammenfassung und Fazit

In diesem Artikel wurden delta-orientierte Komponenten-, Integrations- und Systemtestverfahren für variantenreiche Softwaresysteme vorgestellt. Mit Hilfe der BCS-Fallstudie konnte belegt werden, dass bei den Testfällen für Komponententests insgesamt nur 16 % aller Testfälle generiert wurden, die für jedes Produkt jeweils einzeln erstellt worden wären. Auch für die Integrationstests konnte ermittelt werden, dass die Menge der erneut zu erstellenden Testfälle pro Produktvariante deutlich unter der Anzahl der komplett zu erstellenden Testfälle lag. Auf Systemtestebene mussten nur ca. 50 % aller wiederverwendbaren Testfälle erneut ausgeführt werden (vgl. [Duk13]).

Das delta-orientierte Testverfahren konnte für verschiedene Testphasen des V-Modells erfolgreich adaptiert werden, sodass eine Reduktion des Testaufwandes in allen betrachteten Testphasen möglich war. Im Vergleich zu klassischen Testmethoden für einzelne Softwaresysteme konnte in vorherigen Arbeiten gezeigt werden, dass das hier vorgestellte Verfahren in Bezug auf die Erstellung und Ausführung von Testfällen effizienter ist, um variantenreiche Systeme zu testen. ■

Referenzen

[Duk13] M. Dukaczewski, I. Schaefer, R. Lachmann, M. Lochau, Requirements-based Delta-oriented SPL Testing, PLEASE, 2013.

[Kam12] J. Kamischke, M. Lochau, H. Baller, Conditioned model slicing of feature-annotated state machines, FOSD'12, 2012.

[Lit12] S. Lity, R. Lachmann, M. Lochau, I. Schaefer, Delta-oriented Software Product Line Test Models - The Body Comfort System Case Study, Technischer Bericht, TU Braunschweig, 2012.

[Loc12] M. Lochau, I. Schaefer, J. Kamischke, S. Lity, Incremental Model-based Testing of Delta-oriented Software Product Lines, TAP, 2012.

[Per12] G. Perrouin, S. Oster, S. Sen, J. Klein, B. Baudry, Y. Le Traon, Pairwise testing for software product lines: comparison of two approaches, Software Quality Journal 20(3-4): 605-643 (2012).

[Poh05] K. Pohl, G. Böckle, F. van der Linden, Software Product Line Engineering: Foundations, Principles, and Techniques, Springer, Heidelberg, 2005.

[Sch10] I. Schaefer, L. Bettini, V. Bono, F. Damiani, N Tanzarella, Delta-Oriented Programming of Software Product Lines, SPLC 2010.