Aufrechterhaltung von Traceability Links während evolutionärer Softwareentwicklung

Patrick Mäder, Matthias Riebisch und Ilka Philippow

Technische Universität Ilmenau, Fachgebiet Softwaresysteme/Prozessinformatik

{patrick.maeder|matthias.riebisch|ilka.philippow}@tu-ilmenau.de

1 Einleitung

Softwaresysteme spielen in vielen Lebensbereichen eine immer wichtigere Rolle und werden immer komplexer. Sie unterliegen häufigen Änderungen, die notwendig sind, um weiteren Einsatz zu ermöglichen. Solche Änderungen erfordern hohen Aufwand und sind außerdem mit hohen Risiken verbunden, z.B. bezüglich Korrektheit, Termin- und Budgeteinhaltung. Häufige Änderungen führen zu Strukturverlust (englisch *Architectural Decay*), der wiederum nachfolgende Änderungen erschwert. Zur Vermeidung dieser Konsequenzen muss die evolutionäre Weiterentwicklung von Systemen durch softwaretechnische Konzepte, Methoden und Werkzeuge unterstützt werden.

Traceability stellt ein Konzept dar, das evolutionäre Softwareentwicklung fördert. Traceability Links verbinden Artefakte der Anforderungsanalyse, des Entwurfs und der Implementierung und tragen so zu besserer Verständlichkeit bei. Änderungen werden vereinfacht, indem beispielsweise Aufwandsschätzung, Verfolgung, Vollständigkeitsanalyse und Prüfung unterstützt werden. Die Nutzung von Traceability Links bietet selbst dann Vorteile, wenn dafür zusätzlicher Aufwand notwendig ist.

Um Traceability Links effektiv nutzen zu können, müssen sie gültig, das heißt korrekt und vollständig sein. Aufgrund der hohen Anzahl erfordert die Erstellung, Konsistenzsicherung und Prüfung der Links sehr hohen Aufwand.

Die hier vorgestellten Arbeiten zielen auf eine deutliche Reduzierung dieses Aufwands ab, um einen Nutzen in der praktischen Anwendung zu erreichen. Aufgrund der nur teilweise formal definierten Semantik von verbreiteten Spezifikations- und Modellierungssprachen ist eine Automatisierung der Erstellung, Nachführung und Konsistenzsicherung von Traceability Links derzeit nicht zu erwarten. Deshalb wird eine weitgehende Unterstützung der Entwickler angestrebt, um neben einer Minimierung des Aufwandes eine Beherrschung der Komplexität realer Modelle und ihrer Traceability Links zu erreichen.

Die Speicherung von Traceability Links ist in vielen verbreiteten Repositories bereits möglich. Die Aufgabe der Erstellung von solchen Links ist zwar prinzipiell gelöst, aber nicht automatisierbar und nur unzureichend durch Tools unterstützt.

Ziel dieser Arbeit ist es, eine Methodik für die Aufrechterhaltung und Prüfung von Traceability Links zu entwickeln. Dabei erfolgt die Erstellung, Aufrechterhaltung und die Ablage von Traceability Links im gleichen Repository wie die durch sie verbundenen Artefakte. Wird ein Artefakt geändert, so sind die betroffenen Traceability Links entsprechend zu aktualisieren. Durch Bezugnahme auf Elementaroperationen soll die vorgestellte Methodik in alle Software-Entwicklungsmethoden integrierbar sein.

2 Aktualisierung von Traceablity Links

Traceability Links sollen während der Erstellung von Modellen oder während weiterer Entwicklungsaktivitäten erstellt oder angepasst werden. Dazu ist eine Einbettung in Entwicklungsmethoden nötig. Diese Einbettung wird dadurch erreicht, dass zu den Aktivitäten einer Entwicklungsmethode die jeweils durchzuführenden Änderungsoperationen zur Nachführung der Traceability Links hinterlegt sind.

Jede Software-Entwicklungsmethodik, z.B. Objektorientierter Entwurf, Refactoring, Entity Relationship-Modellierung oder SADT beschreibt Aktivitäten, wie z.B. das Aufteilen oder Verallgemeinern von Artefakten. Die Autoren gehen von der Annahme aus, dass eine geringe Anzahl von Elementaroperationen existiert, in welche sich alle Aktivitäten der bekannten Entwicklungsmethoden zerlegen lassen. Baldwins Modular Operators [1] und Potts und Takahashis Types of Changes [4] bieten solche Ansätze zur Zurückführung von Entwicklungsaktivitäten auf elementare, atomare Schritte. Als Elementaroperationen identifizieren die Autoren dabei das Aufteilen, Ersetzen, Hinzufügen, Entfernen, Invertieren und Portieren.

Zu jeder identifizierten Elementaroperation wird gegenwärtig eine entsprechende Änderungsoperation zur Aktualisierung der beteiligten Traceability Links definiert. Dabei sind die Links so zu aktualisieren, dass ihre Konsistenz und Korrektheit gewahrt bleibt. Abbildung 1 stellt diese Vorgehensweise dar. Mit der Entwicklung einer grundlegenden Serie von Nachführungsoperationen für Traceability Links zu den identifizierten Elementaroperationen von Änderungen an Artefakten lässt sich diese Methodik leicht auf alle Entwicklungsmethoden erweitern.

Die vorgestellte Vorgehensweise wird gegenwärtig für Methoden des objektorientierten Entwurfs, der Architekturentwicklung sowie des Refactoring verfolgt. Dabei werden die Aktivitäten der Methode in Elementaroperationen zerlegt. Gleichzeitig werden zu den bisher identifizierten

Elementaroperationen die entsprechenden Tätigkeiten zur Aktualisierung der Traceability Links definiert.

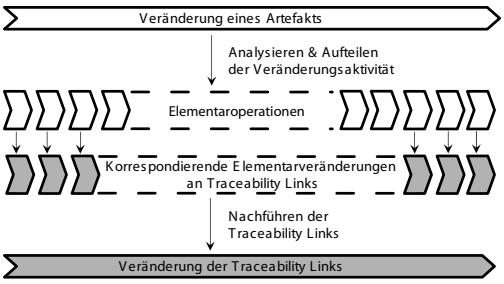

Abbildung 1: Aktualisierung von Traceability Links anhand von Elementaroperationen

Event-getriggerte Aktualisierung Cleland-Huang et al. beschreiben in [2] ein ereignisgestütztes Traceability System (EBT) zur Wartung von Verknüpfungen zwischen Anforderungen, das gut zur Steuerung der oben genannten Aktualisierungen geeignet ist.

Dabei wird von der Grundidee ausgegangen, dass sich die Evolution einer Anforderung durch eine Reihe von Change Events (elementare Veränderungsereignisse) abbilden lässt. Beim Eintreten eines solchen Veränderungsereignisses wird automatisch eine Nachricht mit Informationen über die Art der Veränderung erzeugt. Dazu gehören unter anderem der Typ der Veränderung, Verweise auf die betroffene Anforderung und auf abhängige Objekte sowie Stakeholder.

Die mit Anforderungen und weiteren Artefakten verbundenen Traceability Links werden unter Nutzung des Publish-Subscribe-Paradigmas verwaltet. Die Bedeutung eines Links wird durch eine LinkStrength definiert. Basierend auf dieser Bedeutung erfolgt eine Priorisierung der erzeugten Nachrichten. Abhängig von der Art der Veränderung und der betroffenen Objekte enthält jede Nachricht einen Tätigkeitsvorschlag zur Nachführung des Links.

Das vorgestellte Verfahren wurde ursprünglich für die Traceability von Anforderungen entwickelt. Es wird hier für alle Arten von Softwareartefakten angewendet. Der prioritätsbasierte Mechanismus zur Linkbewertung unterstützt den Umgang mit unsicheren und unvollständigen Informationen in Modellen. Er erlaubt eine Beeinflussung des Aufwands von Änderungen bei jedem Ereignis.

Beispiel zur Illustration Im Folgenden wird das bisher allgemein vorgestellte Vorgehen kurz anhand der Refactoring-Aktivität *Eliminate Duplication by Composition* nach [3] vorgestellt (vgl. Abbildung 2). Es wird von zwei Komponenten mit teilweise identischem Quellcode ausgegangen, der zu entfernen ist. Die hier verwendete Refactoring-Aktivität bewirkt das Auslagern von dupliziertem Code in eine separate Komponente. Es wird weiterhin angenommen, dass die erste der beiden ursprünglichen Komponenten Bezüge zu zwölf Anforderungen aufweist, die durch Traceability Links dargestellt werden.

Die genannte Refactoring-Aktivität besteht aus den Elementaroperationen Aufteilen, Portieren und Entfernen. Für jede der Elementaroperationen ist bekannt, wie Traceability Links potentiell beeinflusst werden. Dadurch können neun Traceability Links ermittelt werden, die genauer überprüft werden müssen. Durch Überprüfung der Linktypen und durch Regeln zu Begriffsbeziehungen lassen sich letztendlich sechs der Traceability Links identifizieren, welche direkt von der Änderung des Auslagerns des duplizierten Codes betroffen sind und ggf. aktualisiert werden müssen.

Dem Entwickler wird im Anschluss an die Refactoring-Aktivität eine Liste mit diesen sechs Links angeboten und jeweils die Möglichkeit zur Verlinkung mit der neuen, der ursprünglichen oder beiden Komponenten gegeben. Mit dieser Eingrenzung der Auswahl wird der Aufwand für die Anpassung und Aktualisierung gegenüber einer Auswahl aus der Gesamtmenge enorm verringert.

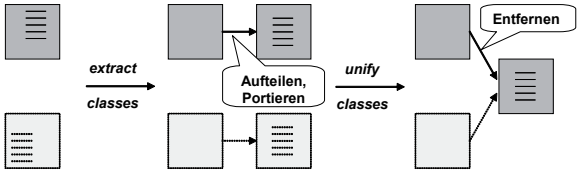

Abbildung 2: Elementaroperationen am Beispiel

3 Ausblick

Das vorgestellte Verfahren liefert einen Ansatz zur effizienten Nachführung von Traceability Links nach Änderungen an den beteiligten Artefakten.

Für den Einsatz im täglichen Entwicklungsprozess ist eine Werkzeugunterstützung der vorgestellten Methodik nötig. Die Aktualisierung von Links kann jedoch nicht automatisch erfolgen, es werden stattdessen interaktiv die in Frage kommenden Alternativen zur Korrektur des entsprechenden Links angeboten. Damit ergibt sich ein enormer Gewinn an Quantität und Qualität im Umgang mit Traceability Links.

Durch den vorgestellten Elementaroperationenansatz wird die Einbindung in übliche Entwicklungsmethoden und -werkzeuge möglich. Diese Werkzeuge und die darunter liegenden Repositories unterstützen häufig bereits das Erstellen und Ablegen von Traceability Links.

Danksagung Diese Arbeit wird teilweise durch die Deutsche Forschungsgemeinschaft DFG unter Projektnummer Ph49/7-1 gefördert.

Literatur

[1] C. Y. Baldwin and K. B. Clark. *Design Rules: Volume 1. The Power of Modularity*. The MIT Press, Cambridge, Massachusetts, 2000.

[2] J. Cleland-Huang, C. K. Chang, and M. J. Christensen. Event-based traceability for managing evolutionary change. *IEEE Trans. Software Eng*, 29(9):796–810, 2003.

[3] S. Gorts and P. T'Seyen. Refactoring thumbnails. `http://www.refactoring.be/thumbnails.html`, 2006.

[4] C. Potts and K. Takahashi. An active hypertext model for system requirements. In J. C. Wileden, editor, *Proceedings of the 7th International Workshop on Software Specification and Design*, pages 62–68, Redondo Beach, CA, Dec. 1993. IEEE Computer Society Press.