

# Keine Verifikation ohne Navigation

## Anforderungsmanagement zur Kontrolle und Realisierung iterativer Projekte

Eine wesentliche Voraussetzung für erfolgreiche Verifikation ist die genaue Kenntnis der Eigenschaften des zu entwickelnden Systems, die in jeder Phase des Prozesses nachgewiesen werden müssen. Die Anforderungen, die das Zielsystem in seinen funktionalen und nicht-funktionalen Eigenschaften beschreiben, sind in der Regel dokumentiert und können als Referenz für die Verifikation herangezogen werden. In heutigen Entwicklungsprojekten können sich die Anforderungen jedoch auch im Verlaufe des Projektes aufgrund aktueller neuer Marktgegebenheiten oder technischer Erfordernisse ändern. Daraus können sich erhebliche Probleme für die Verifikation ergeben, die den Erfolg des ganzen Projektes gefährden. RENATE STÜCKA



Dipl.-Informatikerin RENATE STÜCKA,  
Director of Marketing für den  
Bereich Technical Markets bei der  
Telelogic Deutschland GmbH

**KONTAKT**  
T +49/521/14503-0  
renate.stuecka@telelogic.de

**O**bwohl die meisten Werkzeuge für Anforderungsmanagement in der Lage sind, nicht nur die aktuelle Version sondern auch die Historie einer Anforderung zu speichern, ist die Darstellung der Beziehungen untereinander in der Regel "zeitlos" – Die "Traceability" Informationen, welche die Verknüpfungen von Anforderungen untereinander, mit Entwicklungsartefakten und mit Meilensteinen dokumentieren, repräsentieren in der Regel nur den gegenwärtigen Stand. Es gibt kein Gedächtnis, das Aufschluss über das Beziehungsgeflecht in früheren Versionen gibt.

Anforderungsmanagement muss immer zwei Dimensionen betrachten: die vertikale Dimension – Sammlung und Analyse der Anforderungen – und die Lebenszyklus-Dimension – die Verknüpfung und Verfolgung der Anforderungen über den gesamten Verlauf des Projektes. Dagegen haben die meisten anderen Disziplinen in der Entwicklung entweder nur eine vertikale (über eine oder zwei Phasen) oder nur die Lebenszyklus-Dimension. Letztere stellt die prozessbegleitende Infrastruktur da, die alle Phasen des Prozesses miteinander verbindet.

### Anforderungsmanagement – das Prinzip

Anforderungsmanagement wird heute bereits in vielen Entwicklungsbereichen eingesetzt und

ist ausführlich dokumentiert. Es umfasst nicht nur die Sammlung und Analyse von Anforderungen sondern auch deren Verfolgung und die Dokumentation darüber, wie die Anforderungen miteinander verknüpft sind und welche Abhängigkeiten zwischen ihnen bestehen (Traceability).

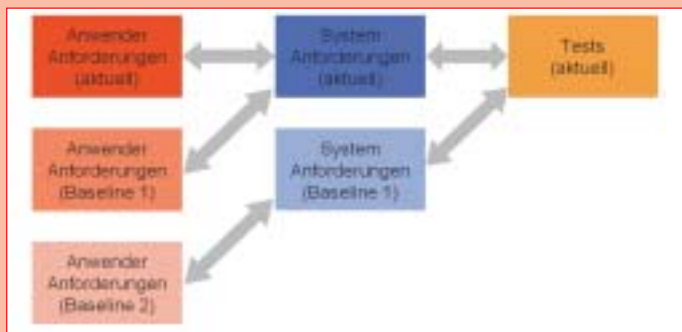
Die Notwendigkeit der Traceability wird noch deutlicher, wenn man sich vor Augen führt, dass Anforderungsmanagement alle Arten von Anforderungen berücksichtigt. Dazu zählen Anwender-, System- und Leistungsanforderungen funktionaler und nicht-funktionaler Art wie auch alle möglichen anderen Kriterien wie z.B. Entwicklungsstandards, Testpläne, Systemarchitekturen und mehr. Alle diese Anforderungen müssen im Verlaufe des Projektfortschritts verifiziert werden. Traceability hilft, die immer wiederkehrenden Fragen zu beantworten:

- ▶ Warum muss diese Funktion implementiert werden?
- ▶ Wie ist diese Anforderung realisiert worden?
- ▶ Wenn sich diese Anforderung ändert, welche Kosten wird die Änderung des entstehenden Systems verursachen? (Impact-Analyse)
- ▶ Ist diese Anforderung vollständig realisiert? (Compliance Matrix)
- ▶ Ist alles, was implementiert wurde, auch durch entsprechende Anforderungen gerechtfertigt? (Traceability Matrix) ▶





Der Idealfall eines Entwicklungs-Projektes



Gängige Praxis in großen Projekten: Verschiedene Versionen werden parallel bearbeitet

### Das Phänomen der iterativen Entwicklung

Im Idealfall werden die Anwenderanforderungen einmal aufgeschrieben und sind bereits beim ersten Mal vollständig und korrekt. Die Systemanforderungen werden abgeleitet und mit den Anwenderanforderungen verknüpft, die sie erfüllen, und sind wiederum sofort vollständig und korrekt. Die Testfälle werden aus den Systemanforderungen hergeleitet, mit diesen verknüpft und sind ebenfalls vollständig und korrekt. Wenn das Projekt abgeschlossen ist, wird das Ergebnis geliefert und ein neues Projekt beginnt. Die Architektur der Informationen ist dabei sehr einfach.

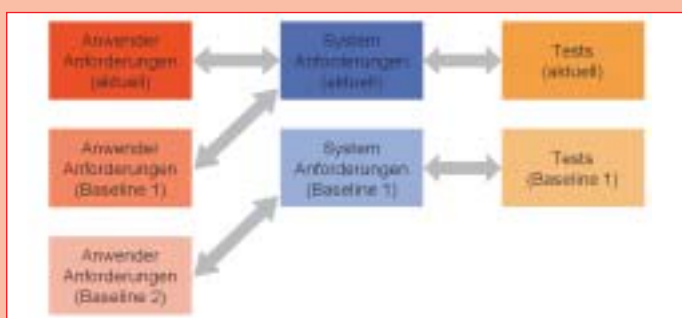
Der Nachteil dieses Modells ist, dass es in der Realität nicht zutrifft: Anforderungen werden selten bereits beim ersten Mal so vollständig erfasst, dass sie eine komplette Spezifikation ergeben. Hinzu kommt, dass sie von Menschen aufgeschrieben werden, denen Fehler unterlaufen können.

Mit zunehmender Größe der Projekte ergibt sich fast zwangsläufig, dass Anforderungen sich ändern können, oder neue hinzukommen. Will man eine schleichende Veränderung der Zielvorgabe vermeiden und das Projekt erfolgreich zu Ende führen, muss man berücksichtigen, dass in heutigen Projekten in der Regel iterativ entwickelt wird. Bei der iterativen Entwicklung wird eine Menge von Anforderungen für die erste Version definiert. Die Anforderungen werden

jedoch kontinuierlich weiter ergänzt und können für Folgeversionen berücksichtigt werden. Zudem können bei diesem Verfahren die Anforderungen für die erste Version eingefroren und in Form einer so genannten Baseline gespeichert werden. Daraus werden die Systemanforderungen für die erste Version hergeleitet, während die Sammlung von Anwenderwünschen für Folgeversionen fortgeführt wird.

Gängige Praxis in großen Projekten ist, dass die Arbeit an mehreren Versionen überlappt, so dass verschiedene Baselines gleichzeitig existieren können, auf denen parallel weitergearbeitet wird. Beispielsweise werden die Testfälle gleichzeitig für zwei verschiedene Spezifikationen entwickelt – die eingefrorene Baseline und die nächste Version. Dabei ist das Risiko, den "richtigen" Funktionsumfang aus den Augen zu verlieren natürlich besonders groß.

Für die meisten Anforderungsmanagement-Werkzeuge ist Traceability zweidimensional: sie verbindet zwei Anforderungsobjekte miteinander, kennt jedoch keine Zeitdimension. Diese wird entweder dem Konfigurationsmanagement-Werkzeug überlassen, oder schlimmstenfalls gar nicht berücksichtigt. Verknüpfungen können lediglich zwischen aktuellen Versionen der Spezifikationen angelegt werden, ein Bezug zu einem Objekt in einer bereits eingefrorenen Baseline kann nicht hergestellt werden. Folgt man jetzt der Verknüpfung in beide Richtungen, gelangt man immer zur jeweils aktuellen Version des Objektes.



Baseline 1 für die Testfälle ist angelegt

Ein Beispiel: Es soll ein Personal Computer entwickelt werden. In der Baseline 1 der Anwenderanforderungen wird festgelegt, dass der Computer ein Laptop mit präzise spezifizierten Abmessungen sein soll. In der Baseline 2 werden diese Abmessungen nun verringert, um einen Palmtop zu entwickeln. Für die Traceability zwischen Anwenderanforderungen und Systemanforderungen heißt das, es wird eine Verknüpfung zur letzten Version der Anforderung angelegt. Damit verändert sich die Zielvorgabe im laufenden Projekt, und es könnte bereits in der ersten Version ein Palmtop entwickelt werden.

### Traceability mit Gedächtnis – Intelligente Traceability.

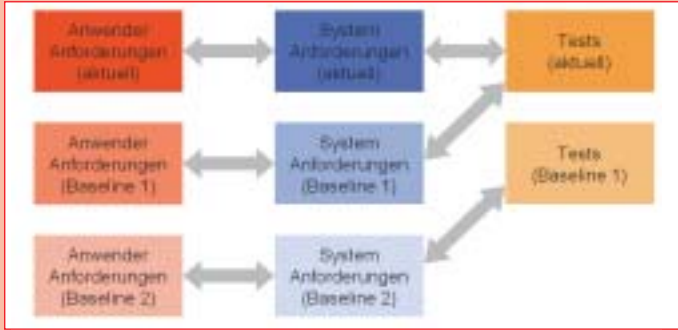
Uneingeschränkte Traceability in einer iterativen Entwicklungsumgebung erfordert die Möglichkeit, Verknüpfungen zu erlauben sowohl mit aktuellen Versionen der Objekte als auch mit solchen, die sich in einer eingefrorenen Baseline befinden. Darüber hinaus muss man zwischen den Objekten entlang der Verknüpfungen beliebig navigieren können. Intelligente Traceability bedeutet:

- ▶ Verknüpfungen anlegen zu können sowohl in aktuellen als auch mit und innerhalb von Baselines.
- ▶ Navigation entlang der Verknüpfungen, unabhängig davon, ob sich das Objekt am anderen Ende in der aktuellen Version oder in einer Baseline befindet.

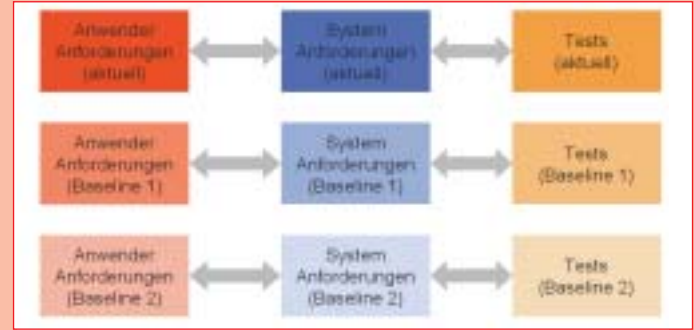
Betrachten wir das Beispiel von oben. Nehmen wir an, dass das Testteam in zwei Arbeitsgruppen aufgeteilt ist. Gruppe 1 entwickelt die Tests für Baseline 1 und verknüpft diese mit den zu testenden Objekten der Systemanforderungen, die zu Baseline 1 gehören. Gruppe 2 macht dasselbe für die aktuelle Version der Systemanforderungen. Mit intelligenter Traceability wird die Baseline 1 für die Tests gespeichert. Die Verknüpfungen zur Baseline 1 der Systemanforderungen gehen nun in die Baseline 1 der Tests, nicht mehr in die aktuelle Version der Tests.

Das Team, das die aktuellen Systemanforderungen bearbeitet, ist nun ebenfalls in zwei Gruppen aufgeteilt: eines für die Bearbeitung der aktuellen Anwenderanforderungen und eines für die Bearbeitung der Anwenderanforderungen in der Baseline 2. Wenn nun die Baseline 2 der Systemanforderungen eingefroren wird, entsteht folgendes Bild: Die Verknüpfungen zwischen den Systemanforderungen und den Anwenderanforderungen sind nun genau so verschoben, wie zuvor die Verknüpfungen mit den Tests. Die Verknüpfungen von den aktuellen Testfällen zu den Systemanforderungen gehen jetzt entweder in die Baseline 2 oder in die aktuelle Version. Wenn nun auch noch die ▶

Anzeige



Baseline 2 der Systemanforderungen wird eingefroren



Die Verknüpfungen zwischen den verschiedenen Versionen sind klar gegliedert

Baseline 2 der Tests erzeugt wird, ergibt sich ein klares Bild des Projektes.

### Baselines und Meilensteine

Eine Baseline allein kann einen Projektmeilenstein noch nicht hinreichend beschreiben. Ohne die Definition von Meilensteinen ist eine systematische Projektbegleitende Verifikation nicht möglich, und die intelligente Traceability wäre nicht vollständig.

Eine Baseline beschreibt einen Meilenstein in der Bearbeitung und Analyse von Anforderungsobjekten. In der Regel ist dies jedoch nur eine Teilmenge dessen, was man in einem Projektmeilenstein festlegen möchte. Typischerweise gehören zu einem Projektmeilenstein in iterativen Entwicklungsumgebungen mehrere Baselines (z.B. Anwenderwünsche und ihre Umsetzung in Systemanforderungen), sowie die Verknüpfungen zwischen den Objekten in diesen Baselines. Zudem geht die Bearbeitung der Anforderungen unterschiedlich schnell, und man möchte unter Umständen sogar Baselines setzen, ohne dass ein Meilenstein bereits komplett erfüllt wäre. Dies erfordert die Definition und Speicherung eines so genannten 'Baseline-Sets'. In einem Baseline-Set werden Mengen von Anforderungsobjekten zusammengefasst und mit ihren Verknüpfungen gespeichert. Damit wird dieses Konzept auch für die iterativen Entwicklungsprojekte in der täglichen Praxis einsetzbar.

Nun stellt sich die Frage, warum derselbe Nutzen nicht dadurch erreicht werden kann, dass man die Anforderungsobjekte in ein Kon-

figurationsmanagement-Werkzeug einbettet. Bei diesem Ansatz fallen jedoch die Verknüpfungen völlig heraus, die elementar sind, um in den heutigen iterativen Entwicklungsprojekten eine wirksame Kontrolle über die entstehenden und möglicherweise überlappenden Versionen zu gewährleisten. Diese Verknüpfungen, die Abhängigkeiten und der Prozess des Anforderungsmanagement sind dem Konfigurationsmanagement völlig unbekannt.

Intelligente Traceability kann nur innerhalb eines Werkzeuges bereitgestellt werden, das diesen Prozess versteht und unterstützt – in der Regel das Anforderungsmanagement-Werkzeug. Bei intelligenter Traceability geht es nicht um Versionsmanagement. Es geht auch nicht darum, die Historie von Änderungen an Anforderungsobjekten zu speichern oder den Prozess, der für die Änderung einer Anforderung erforderlich ist.

Es geht um die Definition von Meilensteinen und wie das Erreichen von Meilensteinen dokumentiert und systematisch verifiziert werden kann. Es geht darum, Verknüpfungen zu bestimmten Versionen von Anforderungsobjekten zu ermöglichen, und es geht darum, wie eine bestimmte Auswahl von Anforderungen inklusive ihrer Verknüpfungen zum Bestandteil eines objektivierte Projektmeilensteines wird.

### Zusammenfassung

Hersteller und Anwender von Anforderungsmanagement-Werkzeugen haben bereits langjährige Erfahrungen in der Anwendung der Traceability gemacht und versucht, den Zeitas-

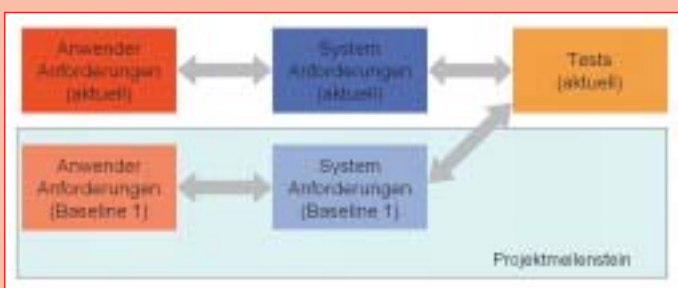
pekt zu berücksichtigen. Bei dem Versuch ist es oft geblieben, und man vertraute darauf, dieses Problem per 'Handarbeit' kontrollieren zu können. Die meisten der heutigen Projekte sind unglücklicherweise zu groß und auch nicht mehr 'isolierte' Entwicklungen einer Version eines Produktes, ohne Bezug auf etwaige Vorgängervarianten.

Die intelligente Traceability ist mächtig genug, parallele Entwicklung von Teams an verschiedenen Iterationen des entstehenden Systems zu unterstützen, indem sie Verknüpfungen zu verschiedenen Versionen der Objekte erlaubt, egal ob sie aktuell oder bereits in einer Baseline eingefroren sind. Baseline-Sets können iterativ zusammengestellt werden, indem weitere Baselines hinzugefügt werden, wenn die dafür erforderlichen Arbeiten abgeschlossen sind.

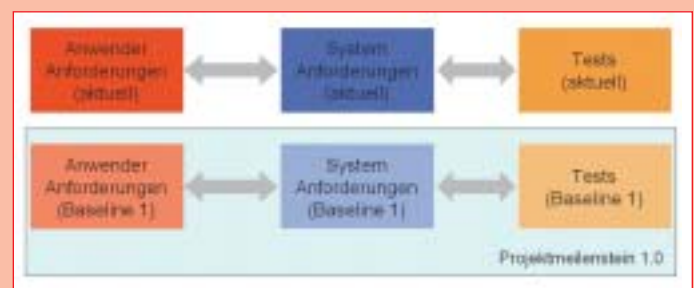
Diese Funktionalität ist in Telelogic DOORS implementiert. Dieses Werkzeug für Anforderungsmanagement unterstützt die intelligente Traceability wie hier beschrieben und ermöglicht damit die systematische Kontrolle über die Realisierung von Anforderungen auch in iterativen Projekten, in denen parallel an mehreren Versionen gearbeitet wird. Die Erreichung von Meilensteinen wird durch Baseline-Sets dokumentiert und kann zweifelsfrei verifiziert werden. Komplette Auditierungsreports geben Aufschluss über alle älteren Versionen der Objekte und wie diese miteinander verknüpft sind. So entsteht ein umfassendes Bild, das nicht nur enthält, woran in der Vergangenheit gearbeitet wurde, sondern auch warum. ■

Weiterführende Infos auf [www.duv24.net](http://www.duv24.net)

more @ click DV064801 >



Ein Baseline-Set dokumentiert den Projektmeilenstein zu dessen Erreichung die Baseline 1 der Tests noch nicht vorliegen muss



Projektmeilenstein 1.0 mit allen Baselines und Verknüpfungen

Anzeige