Modellbasierte Dokumentation mit dem V-Modell® XT - ein Erfahrungsbericht

Marc Sihling
4Soft GmbH
Mittererstraße 3
D-80336 München
sihling@4soft.de

Zusammenfassung

Das V-Modell XT ist der neue deutsche Standard für Entwicklungs- und Wartungsprojekte des Bundes. Veröffentlicht und eingeführt am 4. Februar 2005 erfährt dieses Vorgehensmodell derzeit ein großes Interesse sowohl bei Behörden, als auch bei kleinen wie großen Unternehmen.

Eine technische Besonderheit des V-Modell XT liegt in seinem Fundament, einem präzisen Metamodell, das die grundlegenden Konzepte definiert und sie zueinander in Beziehung setzt. Der Standard wurde entsprechend der Vorgaben des Metamodells spezifiziert und liegt vollständig im XML-Format vor. Erst dieser technische Ansatz ermöglichte den Aufbau einer höchst leistungsfähigen technischen Infrastruktur, die über kontinuierliche Überprüfungen auf Vollständigkeit, Korrektheit und Konsistenz den Erstellungsprozess deutlich vereinfachte und frühzeitig Aussagen über die Qualität des immerhin mehr als 500 Seiten umfassenden Dokuments zuließ.

Dieser Bericht beschreibt die Erfahrungen bei der Ausweitung dieses bewährten Verfahrens und der zugrunde liegenden Infrastruktur auf die Dokumentation eines Entwicklungsprozesses. Dies erweist sich als besonders erfolgreich bei Dokumenten, die – wie das V-Modell XT selbst – eine starke Strukturierung aufzeigen. Im Folgenden wird die Handhabung solcher „modellbasierter Dokumente" anhand eines Beispielprojekts dargestellt und ihr Nutzen für die Anwendbarkeit von Vorgehensmodellen im Generellen und für die Projektdurchführung im Speziellen aufgezeigt.

1 Motivation

Die Notwendigkeit zur methodischen Herangehensweisen in der Softwareentwicklung ist gerade bei größeren Projekten anerkannt. Zu wenig koordiniert verläuft ansonsten die Entwicklung, zu schnell werden die Anforderungen aus dem Auge verloren und zu spät wird die Qualität begutachtet. Jedoch zeigt sich auch in Projekten, die explizit nach einem definierten Vorgehen arbeiten, oftmals nicht die gewünschte Verbesserung im „magischen" Dreieck von Kosten, Qualität und Zeit. In solchen Projekten können meistens die folgenden zwei Symptome beobachtet werden:

- Das Vorgehensmodell wird nicht *gelebt*: insbesondere unter Projektdruck wird immer stärker von den Vorgaben des Vorgehensmodells abgewichen mit der Folge, dass Dokumente mehr und mehr veralten und inkonsistent werden. Die Gründe liegen auch darin, dass die Projektbeteiligten das Vorgehensmodell nicht als gewinnbringend erachten sondern als zusätzliche Bürokratie, die sie an der Durchführung vermeintlich wichtigerer Tätigkeiten hindert. Manchmal wird der Nutzen auch als zu gering im Verhältnis zum Aufwand gesehen: wer beispielsweise eher selten die Möglichkeit zur Nachverfolgung von Anforderungen benötigt, scheut sicherlich eher den Aufwand die entsprechenden Informationen abzulegen und aktuell zu halten.
- Das Vorgehensmodell ist nicht *lebendig*: Diskrepanzen zwischen Entwicklungsprozess und tatsächlichem Prozess können auf die Starrheit des dokumentierten Verfahrens zurückzuführen sein. Obwohl ein eigentlich besseres Verfahren angewendet wird, wird das Vorgehensmodell nicht angepasst und verliert weiter an Überzeugungskraft und Nutzen.

Mit der Erstellung des neuen V-Modell XT wurde darauf geachtet, diesen Symptomen entgegenzuwirken. So ist die Einstiegshürde deutlich niedriger als bei anderen Standards – Projektbeteiligte finden sich somit schneller in den Zielen und Mechanismen des Vorgehensmodells zurecht. Weiterhin wurde bei der Erstellung auf weitestgehende Flexibilität und Erweiterbarkeit geachtet, so dass die Vorgaben einfach angepasst oder ergänzt werden können.

Darüber hinaus erkannten die Autoren frühzeitig die Bedeutung einer bedarfsgerechten, projektbe-

gleitenden Werkzeugunterstützung, um das Team von automatisierbaren Tätigkeiten zu entlasten und wichtige Informationen komfortabel zugänglich zu machen. Im Lieferumfang des V-Modell XT befinden sich zwei Werkzeuge, die beide unter einer Open-Source Lizenz verfügbar gemacht wurden. Mit dem V-Modell XT Editor kann die XML-Datei des V-Modells angepasst und erweitert werden und mit dem V-Modell XT Projektassistenten steht ein komfortables Werkzeug für das initiale Tailoring zur Verfügung. Mit Tailoring wird die Anpassung, also das „Maßschneidern“ des Vorgehensmodells an die konkreten Bedürfnisse eines Projekts bezeichnet. Das V-Modell XT definiert etwa einhundert unterschiedliche Produkte, die in den meisten Fällen in Form von Dokumenten während des Projektablaufs erstellt werden müssen – im Verlauf der Projektinitiierung wird dabei festgelegt, welche Dokumente zu welchem Zeitpunkt erstellt werden. Über eine geeignete Werkzeugunterstützung, die die einfache und komfortable Erstellung und Pflege dieser teils umfangreichen Dokumente ermöglicht, macht das V-Modell keine Aussagen.

Im Rahmen des vom BMBF geförderten Forschungsvorhaben NOW erarbeitete die 4Soft GmbH den Ansatz der „modellbasierten Dokumentation“, der nun mit der Zielsetzung erweitert wurde, die bestehenden V-Modell Werkzeuge auch für die Erstellung und Pflege der mannigfaltigen Produkte einzuführen und somit die Anwendbarkeit des Vorgehensmodells zu verbessern und langfristig oben aufgeführten Symptomen entgegenzuwirken.

Dieser Bericht beschreibt die Erfahrungen mit dem Ansatz in einem praxisnahen Beispielprojekt. Hierfür werden zuerst im nachfolgenden Kapitel die Basiskonzepte des V-Modell XT dargestellt. Anschließend beschreibt Kapitel 3 den Ansatz der modellbasierten Dokumentation, der in einem Beispielprojekt angewendet wurde. Die resultierenden Erfahrungen werden in Kapitel 4 beschrieben. Eine Zusammenfassung schließt den Bericht ab.

2 Übersicht über das V-Modell XT

Das V-Modell XT ist der Nachfolger des Standards V-Modell 97, der zwischenzeitlich nicht nur in Entwicklungsprojekten der öffentlichen Hand, sondern gleichfalls in Vorhaben der Industrie starke Verbreitung findet. Die Anpassung war notwendig, da sich neue, moderne Technologien und Herangehensweisen (beispielsweise komponentenorientierte oder auch agile Methoden) nicht geeignet abbilden ließen. Weiterhin wurde das V-Modell 97 oftmals als zu bürokratisch und die entsprechende Lernphase als zu lang erachtet. Und schließlich deckt der Standard lediglich den Entwicklungsanteil des Software-Lifecycles ab – eine Phase, die in den meisten Fällen nur den kleineren Teil der Gesamtkosten ausmacht.

In einem über einem Jahr dauernden Projekt haben bis zu 30 Fachleute aus fünf unterschiedlichen Organisationen den neuen Standard V-Modell XT entwickelt und dokumentiert. Sie beschränkten sich hierbei auf eine Handvoll elementarer Basiskonzepte:

- Wie jedes Vorgehensmodell basiert auch das V-Modell XT auf *Rollen*, *Aktivitäten* und *Produkte*n. Jedoch beschreibt es ein stark produktzentriertes Vorgehen, bei dem die Frage nach dem Ergebnis bedeutsamer als die Frage nach der Erstellung des Ergebnisses ist.
- Rollen, Aktivitäten und Produkte, die zu einem gemeinsamen Themenkomplex gehören werden in so genannten *Vorgehensbausteinen* zusammengefasst. So existieren für Teilprozesse wie Projektmanagement, Qualitätssicherung oder auch Problem- und Änderungsmanagement separate Vorgehensbausteine.
- Der Projektablauf ist in Teilabschnitte gegliedert, die stets mit einem konkreten Zwischenergebnis abschließen. Die Begutachtung dieses Zwischenstands und die Entscheidung über den weitern Projektverlauf erfolgt im Rahmen von *Entscheidungspunkten*, die in etwa dem bekannten Konzept der „quality gates“ entsprechen.
- In welcher Reihenfolge die einzelnen Entscheidungspunkte durchlaufen werden, hängt vornehmlich von der ausgewählten *Projektdurchführungsstrategie* ab. Von diesen sind acht im V-Modell definiert und jede einzelne deckt einen bestimmten Vorgehenstypus ab, wie beispielsweise im Fall der agilen, der komponentenorientierten oder der inkrementellen Vorgehensweise.

Die Modularisierung der Basiskonzepte in Vorgehensbausteinen vereinfacht das Tailoring deutlich: anstelle aus der Gesamtmenge aller Produkte, Rollen und Aktivitäten die herausschneiden zu müssen, die im konkreten Projekt nicht benötigt werden, nimmt der Projektleiter von Anfang an nur jene Bausteine, die er benötigt. Dass trotzdem am Ende des Anpassungsvorgangs ein in sich stimmiges,

konsistentes Vorgehensmodell steht, ist die Aufgabe des Werkzeugs „Projektassistent“, das sich hierzu der klaren Vorgaben aus dem Metamodell bedient.
Folgendes Beispiel verdeutlicht diesen Zusammenhang: das V-Modell XT definiert einen Vorgehensbaustein „Kaufmännisches Projektmanagement“ mit der Rolle des „Projektkaufmanns“. In der allgemeinen Beschreibung zum Produkt „Projekthandbuch“ ist dokumentiert, dass der Projektkaufmann an der Erstellung und Fortschreibung des Projekthandbuchs beteiligt ist. Sind jedoch in einem Projekt beispielsweise die Kosten nicht der entscheidende Erfolgsfaktor, so kann sich der Projektleiter im Rahmen des Tailorings entschließen, das Projekt ohne kaufmännisches Projektmanagement durchzuführen und somit den zugehörigen Vorgehensbaustein abzuwählen. Das Werkzeug „Projektassistent“ generiert in diesem Fall eine angepasste Dokumentation zum Vorgehensmodell, ein so genanntes „projektspezifisches V-Modell XT“, in dem der Projektkaufmann nicht mehr in der Beschreibung des Projekthandbuchs auftritt.

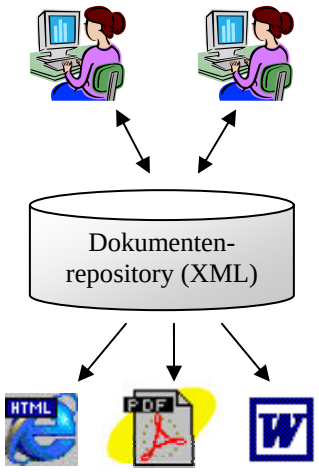

Abbildung 1: Nutzungsszenario

Das Werkzeug profitiert hierbei bereits von einer Infrastruktur, die im Rahmen des Entwicklungsprojekts initiiert und umgesetzt wurde (siehe Abbildung 1):

- Die XML-Datei zum V-Modell XT wird durch ein zentrales Konfigurationsmanagement verwaltet.
- Einzelne Bearbeiter nutzen den „V-Modell XT Editor“, um eine aktuelle Version zu beziehen, diese nach Wunsch zu bearbeiten und dann wieder an den Server zurückzuschicken.
- Bevor der Server eine neue Datei annimmt, wird diese auf Konformität zum zugehörigen XML-Schema untersucht. Zudem wird die Einhaltung von insgesamt fünfzehn Konsistenzbedingungen überprüft. Alle Tests kann der Anwender bei Bedarf auch in seiner lokalen Umgebung starten.
- Der Server führt eventuell Versionen mehrerer Bearbeiter zusammen und bereinigt entstehende Konflikte.
- Aus der jeweils aktuellen XML-Datei werden serverseitig täglich eine Reihe von Dokumenten in unterschiedlichen Formaten (PDF, HTML, etc.) generiert und zugänglich gemacht. Auf diese Weise ist die Weiterentwicklung für das Team transparent und nachvollziehbar.

Dieses Nutzungsszenario hat sich im Verlauf der Erstellung des V-Modells XT bewährt: von insgesamt über 2.500 Versionsständen mussten lediglich einige wenige von Hand überarbeitet werden.
Die dargestellte Infrastruktur basiert auf dem Open-Source Editorframework 4Ever, das im vom BMBF geförderten Forschungsvorhaben NOW entwickelt wurde. Es stellt insbesondere eine gemeinsame Basis für die beiden Werkzeuge dar:

- der *V-Modell XT Editor* dient zur Bearbeitung der XML-Datei, zum Überprüfen auf Konsistenz und zum Generieren bedarfsgerechter Sichten in Form von Dokumenten.
- Der *V-Modell XT Projektassistent* dient zur Anpassung des Standards an die konkreten Bedürfnisse und Rahmenbedingungen eines Projekts (Tailoring). Als Ergebnis können aus dem reduzierten Modell die gewohnten Dokumente generiert werden.

Die Zielsetzung ist hierbei, noch weitere Werkzeuge zu erstellen, die auf einem gemeinsamen Fundament, etwa einer V-Modell-Plattform aufbauen könnten. Aus diesem Grund und auch in der Hoffnung, dass sich eine entsprechende Community bildet, wurden beide Werkzeuge als Open-Source Software verfügbar gemacht (siehe die fourever Homepage http://fourever.sf.net).

3 Modellbasierte Dokumentation

Das Kernproblem der Verfahrenstechnik besteht in der Sammlung, Verarbeitung und Auswertung von Informationen. Alle Daten zu dem zu erstellenden Produkt aber auch über den jeweils praktizierten Prozess gilt es vorrätig zu halten, um sie bei Bedarf

einzusetzen, um die Fragen zu beantworten, die über Wohl und Wehe eines Entwicklungsprozesses entscheiden können. Beispielsweise: ist eine bestimmte Anforderung vollständig umgesetzt? Welche Auswirkungen auf ein System haben Veränderungen einer Anforderung? Oder: welchen Einfluss hat die Entscheidung für eine bestimmte, technische Architektur auf die Performanz des Systems?
Aus den umfangreichen Erfahrungen mit der Erstellung des V-Modells XT sind zwei Punkte direkt aus dem letzten Kapitel motiviert:

- Die Erarbeitung umfangreicher, komplexer Dokumente in einem großen, zeitlich und räumlich verteilten Team ist mit klassischer Textverarbeitung nicht mehr zu handhaben oder nur mit vergleichsweise hohem Aufwand für Fehlererkennung und -beseitigung.

- Die Erarbeitung von Dokumenten in einem entsprechenden XML-Format ermöglicht eine technische Unterstützung des Entwicklungsprozesses, ganz ähnlich zu der gängigen Praxis bei der Programmierung von Software: die Zusammenführung mehrerer Bearbeitungsversionen erfolgt in den meisten Fällen automatisch und was aus der Software-Entwicklung als nightly-build und nightly-test bekannt ist, hat seine Entsprechung in der Überprüfung der Konsistenzbedingungen mit anschließender Generierung der gewünschten Ausgabeformate.

Eine Reihe von Dokumenten des Entwicklungsprozesses können heute bereits elektronisch aus vorhandenen Informationspools (z.B. eben Quelltexte mit JavaDoc) abgeleitet und generiert werden. Lediglich die Dokumente, die nach wie vor in Handarbeit mit Textverarbeitungen erstellt werden, stellen weiße Bereiche auf der Landkarte des Entwicklungsprozesses dar: die hier abgelegten Informationen können nicht automatisch extrahiert geschweige denn auf Vollständigkeit oder Konsistenz hin überprüft werden – es verbleibt auch hier mühsame Handarbeit, um zum Ziel zu gelangen.
Mit der modellbasierten Dokumentation ist daher die Vison verbunden, zukünftig die entwicklungsbegleitenden Informationsbestände noch besser miteinander verknüpfen zu können. In stark strukturierten Dokumenten können beispielsweise Informationen aus einem zentralen UML-Repository mit Elementen aus der Implementierungsebene, also Quelltexten, zusammengeführt werden. Durch die Eigenschaft, dass somit weitere Informationen elektronisch verarbeitbar sind, ergeben sich eine Reihe von Vorteilen:

- Informationen werden stärker miteinander verwoben. So können Anforderungen auf Elemente von Design und Implementierung verweisen oder Rollenbelegungen im Team auf die Personen in der Organisation.
- Monotone Tätigkeiten sind leichter automatisierbar, wodurch die Aktualität der Informationen verbessert wird.
- Die Repräsentation der Information ist von ihrer Darstellung entkoppelt. So können einfacher bedarfsgerechte Sichten abgeleitet werden.
- Die Überprüfung des Informationsbestands ist automatisierbar, wodurch sich einfacher Aussagen über Vollständigkeit und Konsistenz feststellen lassen.

Insgesamt kann auf diese Weise der Entwicklungsprozess deutlich besser unterstützt und der mit der Informationspflege verbundene Aufwand merklich reduziert werden. Die bestätigen unsere Erfahrungen aus der Pilotierung.

4 Konzeption und Infrastruktur

Im Folgenden werden die Basiskonzepte der modellbasierten Dokumentation dargestellt, wie sie in der Weiterentwicklung der Werkzeuge zum V-Modell XT prototypisch umgesetzt und pilotiert wurden. Grundlage ist die Erweiterung des zentralen CVS-Servers als Dokumenten-Repository um ein Webportal, in dem die Inhalte der Dokumente aufbereitet und in Beziehung gesetzt werden. Die Funktionalität der Infrastruktur umfasst die Bereiche „Bearbeitung“, „Koordination“ und „Auswertung“ (siehe Abbildung 2).

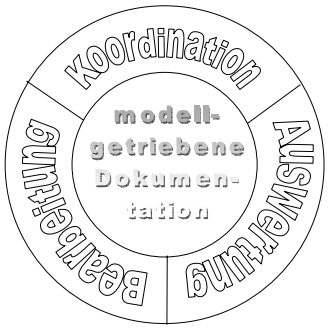

Abbildung 2: Bereiche der Infrastruktur

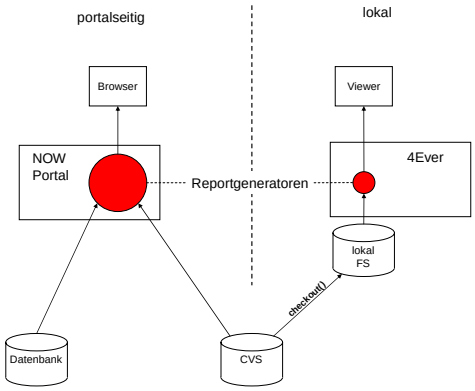

Abbildung 3: Dokumentenauswertung

Die *Dokumentenbearbeitung* umfasst alle Aktivitäten, die ein vorhandenes Dokument ändern. Das Anlegen neuer Dokumente ist in der Dokumentenbearbeitung nicht enthalten. Es können also nur vorhandene Dokumente und Templates bearbeitet werden. Templates sind Dokumente, die noch keinen Inhalt haben, in ihrer Struktur aber schon einem Dokument entsprechen.
Das Konzept sieht vor, dass die Dokumente in einem Dokumentationsordner des CVS-Repositories verwaltet werden. Projektmitarbeiter können das Dokument auschecken, lokal mit dem V-Modell XT Editor bearbeiten und wieder einchecken. Dieses Vorgehen entspricht der klassischen Bearbeitung von Quellcode, der in einem Versionsverwaltungssystem abgelegt ist. Die Vorteile sind eine schnelle und dezentrale Bearbeitung. Die Bearbeitung kann sogar „offline“ stattfinden.
Die Verwendung des V-Modell Editors bietet sich hier an, da dieser beliebige XML-Dokumente bearbeiten kann und zudem einen geeigneten Mechanismus für Referenzen innerhalb und zwischen mehreren Dokumenten vorsieht.
Die *Dokumentenauswertung* ist im Wesentlichen die Aufbereitung und Visualisierung von Informationen, die in den Dokumenten enthalten ist. Die Aufbereitung der Informationen eines oder mehrerer Dokumente findet in Form von Reporten statt, die durch einen Browser dargestellt werden können. Es werden Basisreporte und Individualreporte unterschieden. Basisreporte sind Reporte, die auf alle XML-Dokumente anwendbar sind. Individualreporte sind hingegen auf ganz bestimmte Dokumenttypen zugeschnitten. Das Konzept sieht vor, dass sich die Reporte sowohl lokal als auch portalseitig generieren lassen. Für diesen Zweck gibt es generische Reportgeneratoren, die detailliert konfiguriert werden können. Ein portalseitiger Reportgenerator ist jedoch prinzipiell mächtiger, weil zudem die Benutzerdatenbank des Portals zur Verfügung steht. Um lokal Reporte erstellen zu können, müssen die benötigten Dokumentdateien aus dem CVS-Repository in das lokale Dateisystem ausgecheckt werden (siehe Abbildung 3).
Die *Koordination* der Dokumentenerstellung umfasst sowohl die Initialisierung als auch die Pflege der Dokumentenstruktur eines Projektes. Dazu gehört

- das Erstellen einer Dokumentationsstruktur nach einem vorgegebenen Muster
- die Auswahl der später möglichen Reporte
- die Konfiguration der Reporte sowie
- die Verteilung von Rollen

Das Festlegen einer Dokumentationsstruktur erfolgt durch die Bearbeitung eines Schemas, das die Dokumentationsstruktur beschreibt. Nur der Administrator verfügt über Schreibberechtigung auf der Schemadatei, wodurch eine Veränderung der Dokumentationsstruktur von anderen Beteiligten ausgeschlossen ist. Das gleiche gilt für Reportkonfigurationsdateien.

5 Erfahrungen im Pilotprojekt

Es gibt viele verschiedene Aspekte in einem Projekt, die dokumentiert werden müssen. Dementsprechend vielfältig sind die einzelnen Dokumentarten. Im Rahmen der Weiterentwicklung der V-Modell XT Werkzeuge wurde der vorgestellte Ansatz modellbasierter Dokumentation wie folgt umgesetzt (siehe Abbildung 4):

- Es wurde ein generisches Metamodell für alle Dokumente eines Projekts erarbeitet. Hierdurch sind Metainformationen festgelegt, die jedes Dokument unabhängig von seinem Inhalt mit sich führt. Dazu gehört der Titel, eine kurze Beschreibung des Inhalts, das Erstellungsdatum, die Versionsnummer, eine (eventuell leere) Änderungshistorie und schließlich eine Reihe von Kapiteln. Jedem Kapitel sind ebenfalls Informationen zugeordnet und zwar eine Überschrift, eine Beschreibung, das Datum der letzten Bearbeitung, der eigentliche Inhalt, ein oder mehrere Reviewkommentare und schließlich ein oder mehrere Unterkapitel.
- Ausgehend vom allgemeinen Metamodell wurde ein spezielles für die Erhebung und Dokumentation von Anforderungen an die Werkzeuge erarbeitet. Hierfür wurden verfeinerte

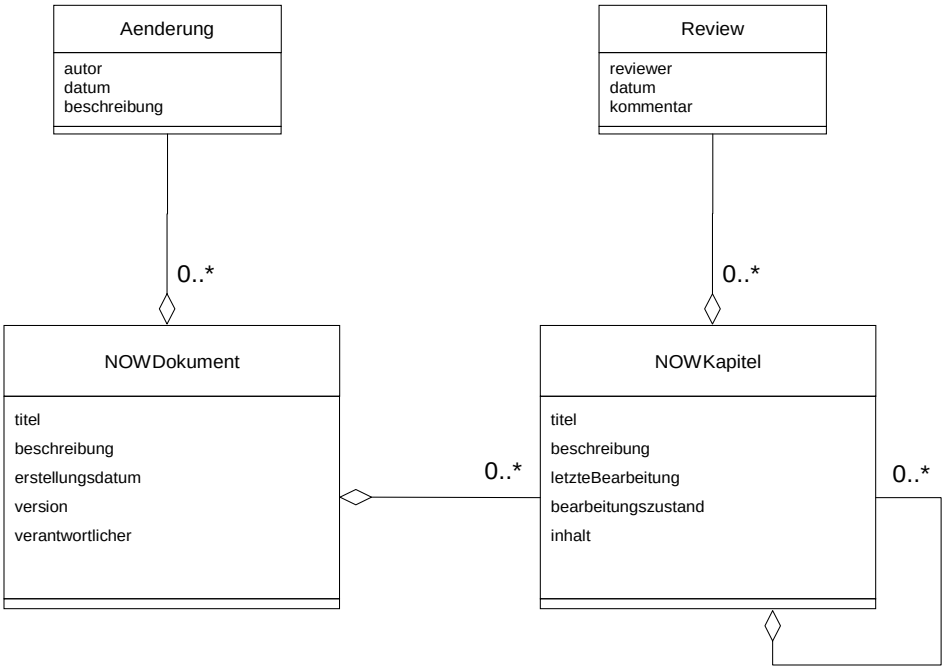

Abbildung 4: Generisches Metamodell für Projektdokumente

Unterkapitel eingeführt, die in strukturierter Form die relevanten Informationen für eine bestimmte Anforderung enthalten (laufende Nummer, Beschreibung, Priorität, Erfüllungsgrad und weitere). Anforderungsdokumente entsprechen somit gleichzeitig diesem wie dem generischen Schema.

Auf diese Weise ließen sich bereits sinnvolle Auswertungen erzeugen, auch wenn noch nicht alle Dokumente detailliert modelliert waren. Nebenstehende Abbildung 5 zeigt die Dokumentenstruktur im Projekt und zudem die Fähigkeit des Editors, eine Gruppe von Dokumenten gleichzeitig bearbeiten zu können. Auf diese Weise ist es insbesondere möglich, Referenzen zwischen mehreren Dokumenten herzustellen (beispielsweise von den Anforderungen zum Design).

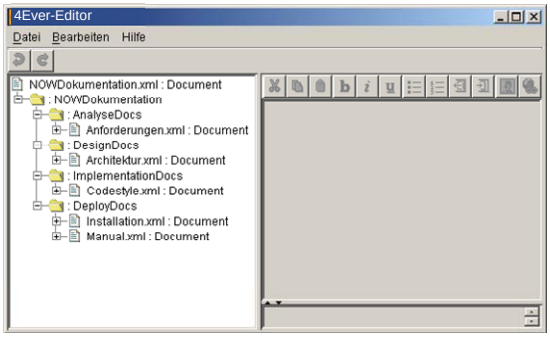

Abbildung 5: Bearbeitung mit dem Editor

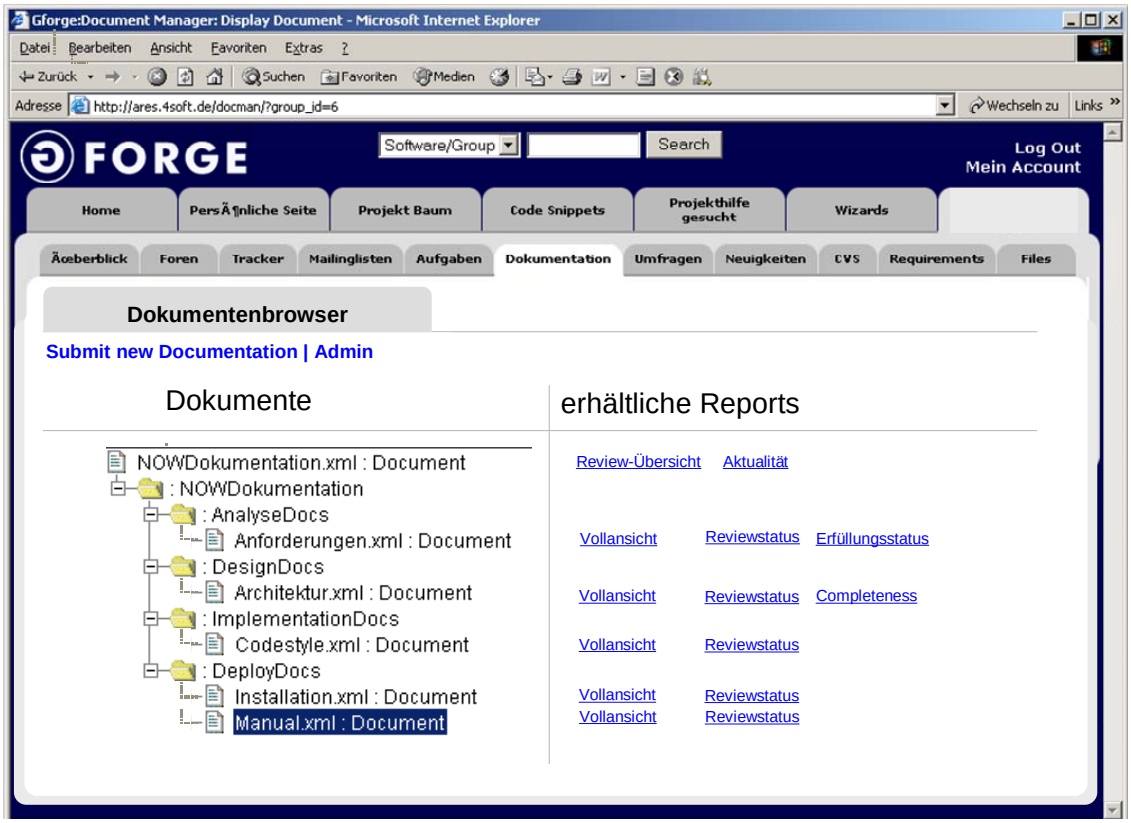

Abbildung 6: Dokumentenübersicht im Entwicklungsportal

Sobald die veränderten Dokumente im zentralen Repository eingecheckt sind, kann der aktuelle Bestand im Dokumentenbrowser des Entwicklungsportals abgerufen werden. In Abbildung 6 ist eine dem entsprechende Erweiterung des Open-Source Entwicklungsportals „GForge“ abgebildet.

Der Dokumentenbrowser zeigt Struktur und Dokumente des Projekts und die jeweils möglichen Reports. So kann der Inhalt jedes einzelnen Dokuments in aufbereiteter Form dargestellt werden (*Detailsicht*) – ein Anforderungsdokument beispielsweise in Form einer tabellarischen Übersicht.

Auswertungen, die die Meta-Informationen der XML-Dokumente betreffen, werden für jedes Dokument angeboten. So ist es möglich, bei jedem Dokument den Reviewstatus anzusehen oder diese Statusmeldungen für alle Dokumente in einem Bericht zusammenzufassen. Der Qualitätsverantwortliche sieht so auf einen Blick, in welchem Zustand die einzelnen Dokumente sind (siehe auch Abbildung 7 für eine Übersicht der Navigationsmöglichkeiten im Portal).

Der Prototyp des Reportgenerators ist so flexibel, dass weitere Berichte auf einfache Art und Weise hinzugefügt werden können. So ist angedacht, einen Bericht für die Nachverfolgung von Anforderungen einzubinden, sobald die diversen Designdokumente mit entsprechenden Schemata hinterlegt sind.

In der Pilotierungsphase des beschriebenen Ansatzes im Rahmen der Weiterentwicklung der V-Modell Werkzeuge musste unglücklicherweise mit zwei Portalen gearbeitet werden: auf der einen Seite das bekannte Sourceforge-Portal, auf dem die Quellen der Werkzeuge verwaltet werden sowie darüber hinausgehende Daten (z.B. eine Bug- und eine Feature-Request-Liste). Auf der anderen Seite das interne Portal mit der prototypischen Erweiterung. Dennoch waren die Erfahrungen durchaus positiv: speziell im fokussierten Teilprozess der Anforderungsergebung und –verwaltung ergab sich eine spürbare Erleichterung. Dies liegt sowohl an der strukturierten Erfassung der Anforderungen aber gleichfalls auch an den komfortablen Berichten, die erzeugt werden. So gestaltete sich der Reviewprozess besonders einfach.

6 Technische Umsetzung

Da die Werkzeuge zum V-Modell XT (siehe http://fourever.sf.net/) ebenso wie das GForge Entwicklungsportal (siehe http://www.gforge.org/) unter Open-Source Lizenzen verfügbar sind, können Interessierte mit vertretbarem Aufwand* eine eigene Lösung realisieren. Das Zusammenspiel der beteiligten Komponenten wird im Folgenden skizziert.

Das GForge Entwicklerportal basiert auf einer recht frühen Version der Sourceforge-Portalsoftware. Grundlage ist eine SQL-Datenbank, ein Webserver und eine inzwischen sehr umfangreiche Bibliothek an Skripten im PHP-Format. Weiterhin ist ein CVS-Server vorgesehen für die Verwaltung und Versionierung von Quelltexten und Dokumenten. Die Integration in die Oberfläche des Portals ist jedoch unzureichend – es ist lediglich eine statistische Auswertung der CVS-Aktivitäten vorgesehen und selbst diese konnte im Pilotprojekt nicht zum laufen gebracht werden (Grundlage des erstellten Prototypen war die GForge-Software in der Version 2.x – zwischenzeitlich ist die Entwicklung bereits bei Version 4.x angekommen).

Auch die Erweiterungsmöglichkeiten von GForge sind nicht gerade herausragend: letztlich wurden für den Piloten bestehende PHP-Skripten so angepasst, dass auf der Bedienoberfläche ein neuer Reiter für die Projektdokumentation erscheint, für dessen Seite ein eigenes Skript verantwortlich ist. Für die Ausführung dieses Skripts wurde dann basierend auf der Apache-Erweiterung „Cocoon“ (siehe http://cocoon.apache.org) eine Pipes-and-Filter-Architektur konzipiert und umgesetzt. Auf diese Weise werden nachfolgende Skripte zu einem passenden Datenfluss gekoppelt, der letztlich in der Ausgabe der Reports resultiert:

- *Eingabeskripte* binden unterschiedliche Datenquellen an, beispielsweise die relationale GForge-Datenbank mit Informationen zu Benutzern und Projekten oder eben auch das CVS-Repository mit den eigentlichen XML-Dokumenten bzw. Dateien zur Projektkonfiguration (gleichfalls als XML).
- *Aufbereitungsskripte* sind für die Weiterverarbeitung der Daten verantwortlich. So ist es beispielsweise möglich, dem Anwender eine spezifische Sicht auf die Dokumente anzubieten („welche Dokumente habe ich gereviewed?“ oder „wurde ein Dokument verändert, nachdem ich es in Bearbeitung hatte?“).
- *Ausgabeskripte* kümmern sich schließlich um die Sammlung der für einen Report notwendigen Informationen und ihre Ausgabe.

Die Kombination der diversen Skripte ist nun einfach zu konfigurieren und kann im Prototyp durch den Projektleiter selbst vorgenommen werden. Dazu ist im CVS-Verzeichnis des Projekts eine XML-Datei festgelegt, in der folgende Informationen abgelegt sind:

- Die Dokumentenstruktur, also welche Dokumente wo abgelegt sind.
- Die Menge der Dokumenttypen, für die weitergehende Reports verfügbar sind (während die Basisreporte ja lediglich die Koordination von Review ermöglichen).

Die sinnfällige Kombination der Skripten auf dem Entwicklerportal sorgt nun dafür, dass erst die projektspezifische Konfiguration ausgelesen wird, dann die benötigten Dokumente aus dem CVS bezogen werden und schließlich die konfigurierten Reports ausgegeben werden.

Prinzipiell können dieselben Skripte auch clientseitig Verwendung finden (sofern kein Zugriff auf die gForge-Datenbank nötig ist). Gerade für Reporte von Einzeldokumenten ist es vorteilhaft, diese ausgeben lassen zu können, ohne zuvor das Dokument in das zentrale Repository einspeichern zu müssen.

* Aus unseren Erfahrungen heraus lässt sich eine einfache Lösung bereits mit einem Aufwand von etwa 20 Personentagen umsetzen.

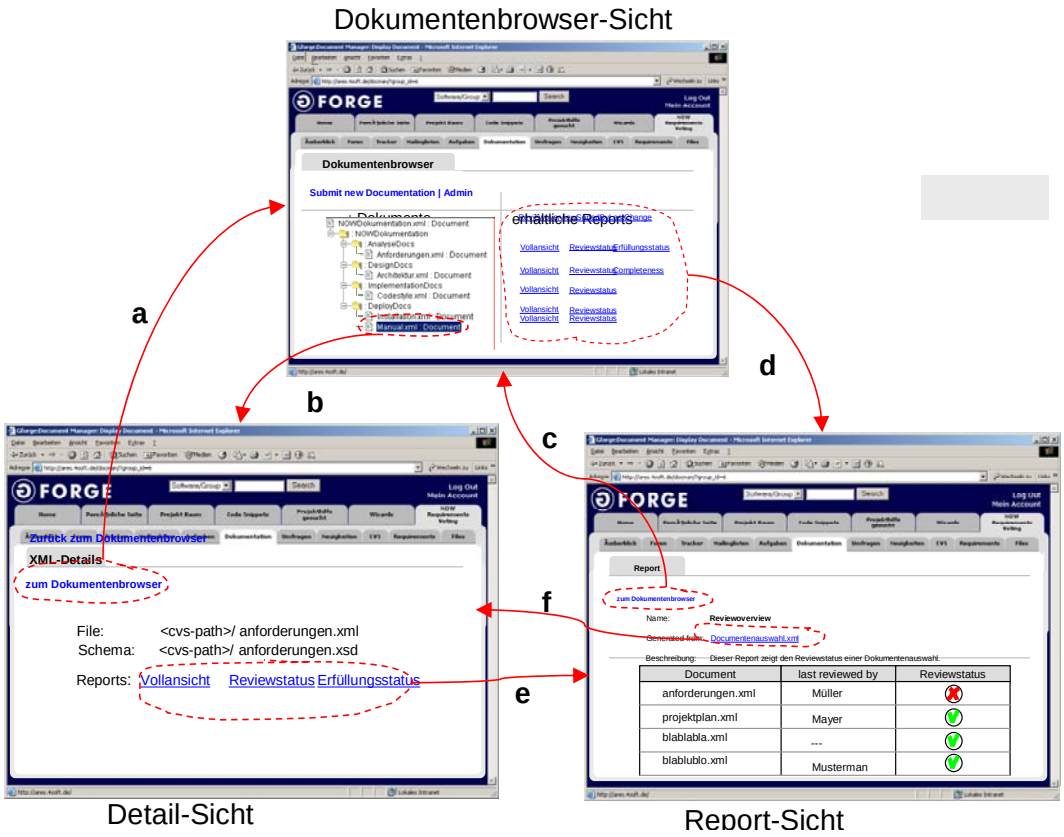

Abbildung 7: Dokumentenübersicht im Entwicklungsportal

7 Zusammenfassung und Ausblick

Dieser Bericht beschreibt den Ansatz der modellbasierten Dokumentation mit dem Ziel, zukünftig die gewohnte Dokumentation mit Textverarbeitungssystemen so umzustellen, dass die Informationen elektronisch weiterzverarbeitbar vorliegen.

Auf diese Weise vereinfacht sich nicht nur Erstellung und Pflege der unterschiedlichen Dokumente, es bieten sich zudem vielfältige Möglichkeiten, den Anwender im Rahmen des Entwicklungsprozesses komfortabel zu unterstützen. Über automatisierte Prüfungen kann der Informationsbestand mit einfachen Mitteln hinsichtlich Vollständigkeit und Konsistenz bewertet werden.

Eine für den Ansatz geeignete Infrastruktur wurde basierend auf dem Open-Source Entwicklungsportal GForge prototypisch realisiert und im Rahmen einer Pilotierungsphase in der Werkzeugentwicklung für den neuen, deutschen Standard V-Modell XT evaluiert. Die hierbei gewonnenen Erfahrungen sind durchweg Erfolg versprechend und weisen den Weg für geeignete Ergänzungen:

- Das generische Metamodell sieht eine Kapitelstruktur für Dokumente vor. Für alle Dokumente des V-Modell XT sind die entsprechenden Kapitel bereits definiert – die passenden XML-Dokumente könnten somit im Rahmen des Tailorings automatisch generiert werden.
- Da alle Dokumente dem System bekannt gemacht werden müssen, kann der Anwender derzeit keine neuen Dokumente anlegen. Eine entsprechende Erweiterung wäre mit vertretbarem Aufwand zu realisieren.
- Derzeit können innerhalb der Dokumente nur Informationen referenziert werden, die sich im selben oder in anderen Dokumenten wiederfinden. Hier ist geplant, peu à peu weitere Informationsbestände zugänglich zu machen. Hierzu zählen UML Modellrepositories, Quelltexte der Implementierung aber auch Datenbestände des Entwicklungsportals (beispielsweise die Bugtrackingliste von Sourceforge).
- Derzeit werden Berichte lediglich in HTML-Format erzeugt. Hier ist es insbesondere wün-

schenswert, zu jedem Dokument eine PDF-Variante anzubieten, die einfacher verschickt und mit besserem Resultat ausgedruckt werden kann.

Zusammenfassend gelingt es, mit bereits bestehnden Werkzeugen den bewähten Ansatz aus der Entwicklungszeit des V-Modell XT auf beliebige Entwicklungsdokumente auszudehnen und auf diese Weise den Erarbeitungs- und Pflegeprozess der Dokumentation merklich zu vereinfachen.