TUM

INSTITUT FÜR INFORMATIK

Werkzeugspezifisches Tailoring für das V-Modell XT

Marco Kuhrmann, Georg Kalus

TUM-I0804
Februar 08

TECHNISCHE UNIVERSITÄT MÜNCHEN

TUM-INFO-02-I0804-0/1.-FI

Druck: Institut für Informatik der
Technischen Universität München

Werkzeugspezifisches Tailoring für das V-Modell XT

CollabXT-SP, CollabXT-TFS – Projekt- und Ergbnisbericht

Marco Kuhrmann, Georg Kalus
Technische Universität München
Institut für Informatik, Software & Systems Engineering
Boltzmannstr. 3
85748 Garching, Germany
{kuhrmann|kalus}@in.tum.de

Zusammenfassung

Im Rahmen des Projekts CollabXT wurde ein Verfahren entwickelt und erprobt, das eine (teilweise) automatische Übersetzung und Integration eines projektspezifisch angepassten V-Modell XT vornimmt und dieses in die Portalsoftware SharePoint Server 2007 sowie die Entwicklungsumgebung Visual Studio Team System mit Team Foundation Server integriert. Zur Anwendung kommen hier ein Meta-Template- und ein Generator-basierter Ansatz, der V-Modell XT Inhalte in die jeweiligen Zielumgebungen integriert. Diese Dokumentation stellt die erarbeiteten Konzepte, Methoden und Werkzeuge vor.

Keywords

Sofware Engineering, Process Engineering, Software Development Process,
V-Modell XT, Tailoring, Visual Studio 2005, Team Foundation Server, Microsoft Office SharePoint Server 2007

CR-Classification: D.2

Inhaltsverzeichnis

1. Einleitung **1**
- 1.1. Das V-Modell XT . . . 2
- 1.2. Microsoft Office SharePoint Server 2007 . . . 2
- 1.3. Microsoft Visual Studio Team Foundation Server . . . 2
- 1.4. Beitrag und Aufbau des Berichts . . . 2

2. Operationalisierung des V-Modell XT **5**
- 2.1. Optionen, Verfahren und Auswahl . . . 5
- 2.2. Abbildung des V-Modell XT . . . 7
  - 2.2.1. Klassifikation abzubildender V-Modell-Elemente . . . 7
  - 2.2.2. Produktmodell des V-Modell XT . . . 8
  - 2.2.3. Aktivitätsmodell des V-Modell XT . . . 9
  - 2.2.4. Ablaufmodell des V-Modell XT . . . 10
- 2.3. Abzubildende Prozesse . . . 12
- 2.4. Weiteres und Besonderheiten . . . 13
  - 2.4.1. Behandlung des Rollenmodells . . . 13
  - 2.4.2. Anpassung des V-Modells für Organisationen . . . 13
  - 2.4.3. Weitergehende Anpassung des V-Modells . . . 15

3. CollabXT – SharePoint **19**
- 3.1. SharePoint – Einführung und Überblick . . . 19
- 3.2. Abbildung des V-Modell XT in SharePoint . . . 20
  - 3.2.1. Abbildung: Produkte . . . 22
  - 3.2.2. Abbildung: Aktivitäten . . . 25
  - 3.2.3. Abbildung: Rollen . . . 25
  - 3.2.4. Abbildung: Entscheidungspunkte . . . 25
  - 3.2.5. Abbildung: Projektplan (Meileinsteine) . . . 26
  - 3.2.6. Abbildung: Dokumentation . . . 26
- 3.3. Werkzeug . . . 27
  - 3.3.1. Konzept und Architektur des Werkzeugs . . . 27
  - 3.3.2. Installation und Randbedingungen . . . 28
  - 3.3.3. Beschreibung und Anwendung des Generators . . . 29

4. CollabXT – Team Foundation Server **33**
- 4.1. Team Foundation Server – Einführung und Überblick . . . 33
  - 4.1.1. Architektur . . . 33
  - 4.1.2. Process Templates . . . 34
- 4.2. Work Item Design . . . 36
  - 4.2.1. Work Item Typen – Grundlagen . . . 36
  - 4.2.2. Work Item: Produkt . . . 38
  - 4.2.3. Work Item: Aktivität . . . 40
  - 4.2.4. Work Item: Entscheidungspunkt . . . 41
  - 4.2.5. Work Item: Arbeitsauftrag . . . 42
  - 4.2.6. Work Item: Risiko und Maßnahme . . . 44
  - 4.2.7. Work Item: PÄM . . . 47
- 4.3. Sonstige Process Template Elemente . . . 49
  - 4.3.1. Work Item Queries . . . 49
  - 4.3.2. Reports . . . 49
  - 4.3.3. Abbildung von Rollen . . . 50
  - 4.3.4. Abbildung von Projektstrukturen . . . 51
- 4.4. Werkzeug . . . 53
  - 4.4.1. Konzept und Architektur des Werkzeugs . . . 54

4.4.2. Installation und Randbedingungen . . . 55
4.4.3. Beschreibung und Anwendung des Generators . . . 56

5. Zusammenfassung **61**

A. Referenzprofile für CollabXT-TFS **63**
A.1. Profil: Systementwicklung (AN) . . . 63
A.1.1. Produkt- und Aktivitätslisten AN . . . 63
A.1.2. Inkrementelle Systementwicklung (AN) . . . 64
A.1.3. Agile Systementwicklung (AN) . . . 64
A.1.4. Komponentenorientierte Systementwicklung (AN) . . . 65
A.1.5. Wartung und Pflege von Systemen (AN) . . . 65
A.2. Profil: Systementwicklung (AG/AN) . . . 66
A.2.1. Produkt- und Aktivitätslisten AG/AN . . . 66
A.2.2. Inkrementelle Systementwicklung (AG/AN) . . . 67
A.2.3. Agile Systementwicklung (AG/AN) . . . 68
A.2.4. Komponentenorientierte Systementwicklung (AG/AN) . . . 68
A.2.5. Wartung und Pflege von Systemen (AG/AN) . . . 69

B. Work Item Types – Erweiterte Erläuterungen und Modellierungen **71**
B.1. Regeln für Work Items . . . 71
B.2. Formular-Design . . . 72
B.2.1. Work Item: Produkt . . . 72
B.2.2. Work Item: Aktivität . . . 75
B.2.3. Work Item: Entscheidungspunkt . . . 76
B.2.4. Work Item: Arbeitsauftrag . . . 78
B.2.5. Work Item: Risiko und Maßnahme . . . 82
B.2.6. Work Item: PÄM . . . 87
B.2.7. Empfehlung für das Work Item Formular-Design . . . 92
B.3. Übersetzung für englische Systeme . . . 92
B.3.1. Anpassung der Work Items . . . 93
B.3.2. Anpassung der Queries und Reports . . . 93
B.3.3. Anpassung des Metatemplates . . . 93

Abbildungsverzeichnis

2.1. Werkzeugspezifische Anpassung als Forsetzung des Tailorings 6
2.2. Metamodell des V-Modell XT, Sicht: Produkt 9
2.3. Metamodell des V-Modell XT, Sicht: Aktivität 10
2.4. Metamodell des V-Modell XT, Sicht: Abläufe und Projektdurchführungsstrategien . . 11
2.5. Position der Rollen im V-Modell-Metamodell 13
2.6. Anpassungsprozess des V-Modell XT 14
2.7. Anpassung für Dokumente und Exportvorlage des V-Modell XT 15
2.8. Anpassungsprozess des MSF auf Basis eines TFS Process Template 17

3.1. Grundlegende Architektur, inhaltlicher Aufbau eines SharePoint-Portals 19
3.2. XML-Schema eines SharePoint-Portals 21
3.3. V-Modell XT im SharePoint-Portal 23
3.4. V-Modell XT Produktzustandsmodell und seine Abbildung im SharePoint-Portal . . . 24
3.5. V-Modell XT im SharePoint-Portal 26
3.6. Konzeptmodell für die V-Modell-Integration in ein SharePoint-Portal 27
3.7. Objektmodell für den SharePoint-Portal-Generator 28
3.8. Start des SharePoint-Portal-Generators 29
3.9. Eingabe der Daten für den SharePoint-Portal-Generator 29
3.10. Vollständiger Datensatz für den SharePoint-Portal-Generator 30
3.11. Laufender Generierungsprozess mit Statusanzeigen 30

4.1. Technische Architektur des Visual Studio Team Foundation Server 33
4.2. XML-Schema eines Process Templates 34
4.3. Gefülltes Process Template mit zugeordneten Gruppen 35
4.4. Vorgehensbaustein Projektmanagement 36
4.5. XML-Schema der Work Item Types 37
4.6. XML-Schema der Work Item Types – Verfeinerung für Workflows 38
4.7. Produktzustandsautomat des V-Modell XT 39
4.8. Workflow für Produkt-Work Items 40
4.9. Workflow für Aktivitäts-Work Items 41
4.10. Workflow für Entscheidungspunkt-Work Items 42
4.11. Workflow für Arbeitsauftrag-Work Items 44
4.12. Workflow für Risiko-Work Items 45
4.13. Workflow für Maßnahme-Work Items 46
4.14. Vorgehensbaustein Problem- und Änderungsmanagement 47
4.15. Workflow für PÄM-Work Items 48
4.16. Vorgehensbaustein Messung und Analyse 50
4.17. Architektur des TFS-Report Subsystems 50
4.18. Generatorschnittmuster für AN-Projektdurchführungsstrategien 51
4.19. Generatorschnittmuster für AG/AN-Projektdurchführungsstrategien 53
4.20. Abbildung von PDS-Abläufen 54
4.21. Ein- und Ausgaben von CollabXT-TFS und Position in der Werkzeugkette 55
4.22. Generator und Visual Studio Startseite für ein V-Modell-Projekt 56
4.23. Schritt 1: Auswählen des V-Modell-Exportverzeichnisses 57
4.24. Schritt 2: Auswählen des projektspezifischen V-Modells in XML 58
4.25. Schritt 3: Auswählen des TFS Meta-Process Template 59
4.26. Generiertes Process Template im PT-Editor (Visual Studio) 60
4.27. Workflow des Produkt Work Item Typs im PT-Editor 60

A.1. Inkrementelle Systementwicklung (AN) 64
A.2. Agile Systementwicklung (AN) 65
A.3. Komponentenorientierte Systementwicklung (AN) 65

A.4. Wartung und Pflege von Systemen (AN) 66
A.5. Inkrementelle Systementwicklung (AN) 67
A.6. Agile Systementwicklung (AG/AN) 68
A.7. Komponentenorientierte Systementwicklung (AG/AN) 69
A.8. Wartung und Pflege von Systemen (AG/AN) 69

B.1. Regelmenge der Work Items 71
B.2. Formlayout für das Work Item Produkt 72
B.3. Formlayout für das Work Item Aktivität 75
B.4. Formlayout für das Work Item Entscheidungspunkt 77
B.5. Formlayout für das Work Item Arbeitsauftrag 79
B.6. Formlayout für das Work Item Risiko 82
B.7. Formlayout für das Work Item Risiko 83
B.8. Formlayout für das Work Item Maßnahme 84
B.9. Formlayout für das Work Item PÄM 1 88
B.10. Formlayout für das Work Item PÄM 2 89
B.11. Formlayout für das Work Item PÄM 3 90

Tabellenverzeichnis

3.1. Abbildungsumfang des V-Modell XT auf SharePoint 2007 22

4.1. Datenstruktur für Produkt-Work Items 39
4.2. Datenstruktur für Aktivität-Work Items 40
4.3. Datenstruktur für Entscheidungspunkt-Work Items 41
4.4. Datenstruktur für Arbeitsauftrag-Work Items 43
4.5. Datenstruktur für Risiko-Work Items 45
4.6. Datenstruktur für Maßnahme-Work Items 46
4.7. Datenstruktur für PÄM-Work Items 48

A.1. Referenzprofil für Systementwicklung (AN) 63
A.2. Referenzprofil für Systementwicklung (AG/AN) 67

B.1. Datenstruktur (DE/EN) für Produkt-Work Items 93
B.2. Datenstruktur (DE/EN) für Aktivität-Work Items 94
B.3. Datenstruktur (DE/EN) für Entscheidungspunkt-Work Items 94
B.4. Datenstruktur (DE/EN) für Arbeitsauftrag-Work Items 95
B.5. Datenstruktur (DE/EN) für Risiko-Work Items 96
B.6. Datenstruktur (DE/EN) für Maßnahme-Work Items 96
B.7. Datenstruktur (DE/EN) für PÄM-Work Items 97

1. Einleitung

Vorgehensmodellen sind oft umfangreiche Werkzeuglandschaften beiseite gestellt, die verschiedene Bereiche der Erstellung und Anwendung von Vorgehensmodellen unterstützen. Oftmals ist jedoch festzustellen, dass Vorgehensmodelle hauptsächlich als Leitfaden bzw. Dokumentation verstanden werden. Die Überführung und Umsetzung im Projekttagesgeschäft gestaltet sich daher meist schwierig, da Vorgehensmodelle durch den ihnen anheftenden Dokumentationscharkter für Entwickler keinen unmittelbare Mehrwert bieten. Auch Projektleiter, insbesondere in kleinen und kurz laufenden Projekten, sehen in strukturierten Vorgehensmodellen nur beschränkten Nutzen. Dabei weisen verschiedene Studien (z. B. [BEJ06]) in verschiedener Weise darauf hin, dass Vorgehensmodelle, oder Prozesse im Allgemeinen, nutzbringend – insbesondere hinsichtlich der Qualität – sein können. Ein weiterer Punkt: Selbst wenn ein Vorgehensmodell für die Einführung in Betracht gezogen wird, ist es mit der einfachen Proklamation nicht getan. Armbrust [AEH+07] zeigt am Beispiel einer vollständig begleiteten Prozessanpassung und Einführung den Umfang und die Tragweite. Angefangen bei der Analyse der Ist-Prozesse über die Anpassung, Pilotierung sowie die Überarbeitung und Aktualisierung der Werkzeuglandschaft sind vielfältige Aufgaben wahrzunehmen.

Das Projekt CollabXT

CollabXT[1] adressiert die Problematik der werkzeugunterstützten Prozesseinführung für den Kontext V-Modell XT. Im Zentrum der Aufmerksamkeit steht dabei aber nicht die organisationsspezifische Anpassung des V-Modells, sondern dessen pragmatische, projektorientierte Anwendung.

Das V-Modell XT ist die erste Version des V-Modells, das Referenzwerkzeuge im Standardpaket umfasst. Diese Werkzeuge unterstützen die organisationsspezifische und die projektspezifische Anpassung des V-Modell XT. Nach dem Tailoring endet die Unterstützung auf der Ebene des Standards jedoch und die Anwender sind entweder darauf angewiesen, auf Werkzeuge Dritter auszuweichen oder auf der Grundlage der bereits verfügbaren Werkzeuge zu improvisieren. Allzu groß ist hier die Gefahr, dass die Vorteile, die das V-Modell XT für große und verteilte Projekte bietet, ungenutzt bleiben, weil es durch die Nichteinbindung in die Arbeitsumgebung schlichtweg ignoriert wird.

Die Lücke zwischen Prozess und Projekt

Es existiert eine Lücke zwischen den Referenzwerkzeugen und den Werkzeugen, die im Projekttagesgeschäft eingesetzt werden. Projektleiter und Entwickler reden aneinander vorbei, wenn der Bruch zwischen Prozess und Projekt nicht beseitigt wird.

An dieser Stelle setzt das Projekt CollabXT auf. Das Ziel von CollabXT ist es, das V-Modell XT nicht nur projektspezifisch, sondern darauf aufbauend auch *werkzeugspezifisch* anzupassen – und das mit minimalem Aufwand für die Anwender.

Das Konzept von CollabXT beinhaltet Abbildungsvorschriften, wie Elemente des V-Modells werkzeugspezifisch umgestaltet werden können. Es beschreibt Modelltransformationen, bei der das V-Modell XT das Ausgangsmodell ist und als Ergebnis ein für konkrete Werkzeuge angepasstes Modell vorliegt.

Microsoft Office SharePoint Server 2007 (MOSS) und Visual Studio Team Foundation Server (TFS) stellen in der aktuellen Version von CollabXT die Zielwerkzeuge dar. In Kooperation mit verschiedenen Partnern sind hierbei zwei Open Source Werkzeuge entstanden, die unmittelbar nach der projektspezifischen Anpassung ansetzen und diese erweitern. Der vorliegende Bericht beschreibt die zugrunde liegenden Konzepte und geht auf die Werkzeuge ein.

1 Weitere Informationen: `www.collabxt.de`

1.1. Das V-Modell XT

Das V-Modell XT[2] (kurz V-Modell) ist als verbindlicher deutscher Entwicklungsstandard für IT-Vorhaben in der öffentlichen Hand ein Vorgehensmodell, das einer weitreichenden Werkzeugunterstützung bedarf. Es fasst moderne Konzepte und bewährtes Wissen in einem integrierten, generischen Standard zusammen, der sowohl für umsetzende Institutionen bzw. Organisationen als auch auf einzelne Projekte noch weit gehend angepasst werden kann. Das V-Modell basiert auf einigen formalen Basiskonzepten, die in einem Metamodell hinterlegt sind. Diese Basiskonzepte sind u.a.:

Vorgehensbausteine integrieren Produkt-, Rollen- und Aktivitätsmodelle, womit sie den inhaltlichen Rahmen für das V-Modell aufspannen und beschreiben. Vorgehensbausteine sind sowohl Tailroing- als auch Bearbeitungseinheiten des V-Modells.

Projektdurchführungsstrategien stellen prinzipielle Vorgehensweisen zur Durchführung von Projekten bereit und stellen die Grundlage der Plangenerierung dar.

Tailoring Definiert einen konsistenzerhaltenden Anpassungsmechanismus auf der Projektebene.

In einem technischen Bericht [Kuh07], haben wir bereits die grundlegenden Ideen und Konzepte zum Projekt CollabXT motiviert. Mit dem vorliegenden Bericht verfeinern wir die originalen Konzepte und geben detailliert Einblick in die Konzeption und Umsetzung. In Kapitel 2 vertiefen wir das V-Modell XT und gehen noch einmal auf bereits vorliegende Vorarbeiten ein.

1.2. Microsoft Office SharePoint Server 2007

Das erste Anwendungsszenario, mit dem wir uns befassen, bezieht sich auf den Microsoft Office SharePoint Server 2007 (MOSS). Dieser ist eine Portalsoftware, die internetbasierte Kollaboration unterstützt. Dafür bietet SharePoint gemeinsame Dokumentenbibliotheken (Repositories), Aufgabenlisten, Kalender, Wikis sowie einen einheitlichen Browser-basierten Zugang und eine enge Integration in Microsoft Office. Der Portalserver setzt auf den Windows SharePoint Services auf, die als kostenloses Add-On für Windows Server 2003 verfügbar sind. Prinzipiell genügen auch diese schon für unsere Zwecke. Bereits in [KK07] haben wir einen ersten Prototyp demonstriert. Dieser Bericht bezieht sich in Kapitel 3 auf eine weiter verfeinerte Umsetzung dieses Konzepts.

1.3. Microsoft Visual Studio Team Foundation Server

Mit dem *Visual Studio Team Foundation Server* (TFS) stellt Microsoft erstmalig ein integriertes Werkzeugsystem [GP06] vor, das neben dem reinem Entwickeln von Anwendungen auch Prozesse der Projektorganisation und des Managements unterstützt.

TFS basiert dabei auf einer Template Engine, in der verschiedene Prozessvorlagen (Process Templates) verwaltet und für verschiedene Projekte instanziiert werden können. Den Projekten steht mit der Instanziierung eine Umgebung zur Verfügung, in der verschiedenartig ausgeprägte Techniken, wie z. B. Task- oder Risikomanagement, unterstützt werden. Den Mitarbeitern im Projekt stehen Vorlagen und Hilfen zur Verfügung (je nach Ausprägung des konkreten Prozesses), ebenso wie Quellcode- und Dokumentenverwaltung. Auch ein Reporting System ist integriert.

TFS ist eine sehr komplexe Infrastruktur, die sich aus verschiedenen Softwarekomponenten zusammensetzt und sowohl in der Installation als auch im Betrieb hohe Ansprüche stellt. Im Kapitel 4 gehen wir darauf detaillierter ein.

Basierend auf dem ursprünglichen Konzept [KT06, Kuh07], auf dem CollabXT basiert, stellt TFS nur eine von vielen möglichen Zielplattformen dar. Wir nehmen daher zu Schluss des vorliegenden Berichts in Kapitel 5 eine entsprechende Einordnung in das Gesamtkonzept vor.

1.4. Beitrag und Aufbau des Berichts

Dieser Bericht gibt wesentliche Inhalte des Projekts CollabXT wieder. Er zeigt, wie für einen generischen Vorgehensstandard (zum Teil/vollständig) automatisiert *Arbeitsumgebungen erzeugt* wer-

2 Weitere Informationen: `http://www.v-modell-xt.de`

den können. Einen signifikanten Beitrag leistet der vorliegende Bericht im Rahmen der Analysen die zur Ermittlung der unmittelbaren Operationalisierbarkeit eines Vorgehensmodells stattfanden. Vorgehensmodelle werden hier nicht mehr nur als Dokumentation und Leitfaden betrachtet, sondern als aktive (ggf. auch reaktive) Komponenten in einer integrierten Werkzeugumgebung. Im Rahmen verschiedener Bestrebungen und Arbeiten aus dem Bereich der Projektleitstände, Projekt-Cockpits und kollaborativer Zusammenarbeit [bAB05, Gna05, Ber06, Fri06, Kuh06, KMR06, AEH+07, KHTS07, Kuh07, BULL07] finden sich eine Reihe verwandter Ansätze und Ideen.

Einen weiteren Beitrag liefern wir im Arbeitsgebiet der *Prozessintegrationsverfahren*. Hierfür legen wir mit dem vorliegenden Bericht einen Grundstein im Bereich der dynamischen Prozess- und Methodenintegration. Da insbesondere im Kontext des V-Modells und des TFS bereits eine inhaltserhaltende (und somit den Anforderungen der *konstruktiven V-Modell-Konformität* weitgehend genügende) Übersetzung von Prozessstrukturen stattgefunden hat, ist der nächste konsequente Schritt die weitergehende Anpassung und Individualisierung des Generats. Da das V-Modell einen eher groben organisatorischen Rahmen darstellt, ist die Integration fein granularer Prozesse, z. B. konkreter Entwicklungsprozesse für die Programmierung oder das Testen, der nächste Schritt. Verfahren zur Integration der durch die (verschiedenen) Templates vorliegenden Prozessbestandteile müssen auf dieser Grundlage entwickelt werden (siehe dazu auch Kapitel 5).

Als Ergebnis diesen Projekts, über das wir berichten, liegt ein *Werkzeugset* vor, das ein mithilfe der V-Modell-Referenzwerkzeuge projektspezifisch angepasstes V-Modell in ein vollständiges SharePoint Team-Portal bzw. ein valides TFS Process Template übersetzt. Wir stellen die Ergebnisse vor und erläutern die Abbildungen am direkten Beispiel.

Aufbau des Berichts

Dieser Bericht ist wie folgt aufgebaut: Im Kapitel 2 diskutieren wir die Optionen zur Operationalisierung des V-Modell XT und gehen dabei auf bereits geleistete Vorarbeiten ein. Wir führen weiterhin eine entsprechende Auswahl der gewählten Verfahren durch. Kapitel 3 geht auf das Teilprojekt *CollabXT-SP* ein. Kapitel 4 stellt kompakt den Team Foundation Server als Infrastruktur vor und beschreibt die Architektur der Process Templates, die unser Zielformat definieren. Weiterhin beschreibt es die Abbildung des V-Modells auf die Struktur der Process Templates. Darüber hinaus beschreiben wir wesentlichen Punkte und Besonderheiten der Abbildung sowie bereits vordefiniert Prozesse, die also Workflows zur Verfügung gestellt werden. Die Kapitel 3 und 4 enthalten jeweils eine kurze Anleitung und Anwendungsbeschreibung der entstandenen Werkzeuge. Eine Zusammenfassung und ein Ausblick befindet sich im Kapitel 5

2. Operationalisierung des V-Modell XT

In diesem Abschnitt gehen wir auf die verschiedenen Optionen zur Operationalisierung[1] des V-Modells ein. Wir beleuchten dabei mögliche Wege durch die mit den Referenzwerkzeugen begonnene Werkzeugkette des V-Modells und zeigen mögliche Aufsetzpunkte für eine weitergehende Operationalisierung. Im Anschluss diskutieren wir die relevanten Anteile des V-Modell XT Metamodells und gehen auf abzubildende Strukturen, Prozesse und Besonderheiten ein.

Vorarbeiten

Zum Themenkomplex der Operationalisierung wurden bereits an vielen Stellen Vorarbeiten geleistet. Insbesondere im Projekt *CollabXT* (und dessen Vorarbeiten) sowie im Projekt *WEIT* wurden vielfältige Arbeiten hinsichtlich der Anpassbarkeit und der Werkzeugintegration geleistet. In [KT06, Kuh07] wurde das V-Modell unter dem Blickpunkt der Erweiterbarkeit betrachtet.

MSF-V-Modell-Integration als Prototyp. Insbesondere bei der Betrachtung des *Microsoft Solutions Frameworks* (MSF, [Tur06]) als konkrete Methode für die Systementwicklungsanteile des V-Modells wurde erstmalig die Integration eines vollständig operationalisierten Prozesses relevant. Beachtenswert dabei war neben der Angleichung der Philosophien der zu integrierenden Modelle zu berücksichtigen, wie hoch Informationsverluste oder -verschiebungen bei der Methodenintegration sind. MSF wurde dazu mit den Mitteln des V-Modells formalisiert. Bereits modellierbare Elemente (Metamodellelemente des MSF) wurden nach Entwicklung einer geeigneten Abbildungsvorschrift in das V-Modell überführt. werkzeugrelevante Anteile des MSF, die nicht Teil des Modells, sondern der implementierenden Werkzeuge sind, mussten in die hier durch das V-Modell bereitgestellten Sub-Modelle integriert werden. Dies hatte zwar den Verlust der hoch integrierten Werkzeugunterstützung zur Folge, schaffte jedoch ein grundlegendes Verständnis für die Operationalisierungsoptionen – auch für das V-Modell XT.

Die automatische, Metamodell-basierte Prozessintegration auf der Basis eines projektspezifisch angepassten V-Modells findet seine Ursache im identifizierten Bedarf [NK05] an unmittelbarer Unterstützung der Anwender im Anschluss an das Tailoring. Mit [KK07] wurde hierfür ein integriertes, an Projektleitständen [KMR06] orientiertes Konzept erstmalig prototypisch umgesetzt. Im Projekt CollabXT fließen diese Vorarbeiten mit ein. Dabei steht immer im Vordergrund, wie mit minimalen Aufwänden und möglichst wenigen Eingriffen in existierende Systemlandschaften die Werkzeugkette des V-Modells transparent erweitert werden kann; andererseits wie gleichzeitig das V-Modell für verschiedene Zielsysteme konfiguriert und bereitgestellt werden kann.

2.1. Optionen, Verfahren und Auswahl

Für die Operationalisierung des V-Modell XT im Kontext einer integrierten Werkzeugumgebung existiert eine Vielzahl von Optionen. Wir fokussieren eine *Generator-basierte Methode*, die auf den Quellen des V-Modells (Metamodell[2] und Modell) aufsetzt und für verschiedene Plattformen Ausgaben erzeugen kann. In Abbildung 2.1 ist die Erweiterung der Referenzwerkzeugkette des V-Modells bis hin zur Projektdurchführungsumgebung skizziert. Sie zeigt die Überbrückung des Werkzeugbruchs zwischen den V-Modell Referenzwerkzeugen und den Werkzeugen der Projektinfrastruktur. Durch eine zusätzliche Generatorstufe, die der projektspezifischen Anpassung des V-Modells mit dem Projektassistenten folgt, wird die Verbindung zur Projektinfrastruktur

1 Unter der Operationalisierung des V-Modells verstehen wir hierbei und allgemein das „in die Anwendung"-bringen des V-Modells. Dazu gehört eine über das Tailoring (projektspezifische Anpassen) hinaus gehende Werkzeugunterstützung, die Tailoring-Resultate unmittelbar in entsprechend definierte Werkzeuge einspeist und den Prozess somit lebbar macht.

2 Das Metamodell des V-Modells befindet sich zurzeit in Überarbeitung. Ein (nicht mehr ganz aktuelle) Dokumentation ist jedoch hier [Gna06] zu finden.

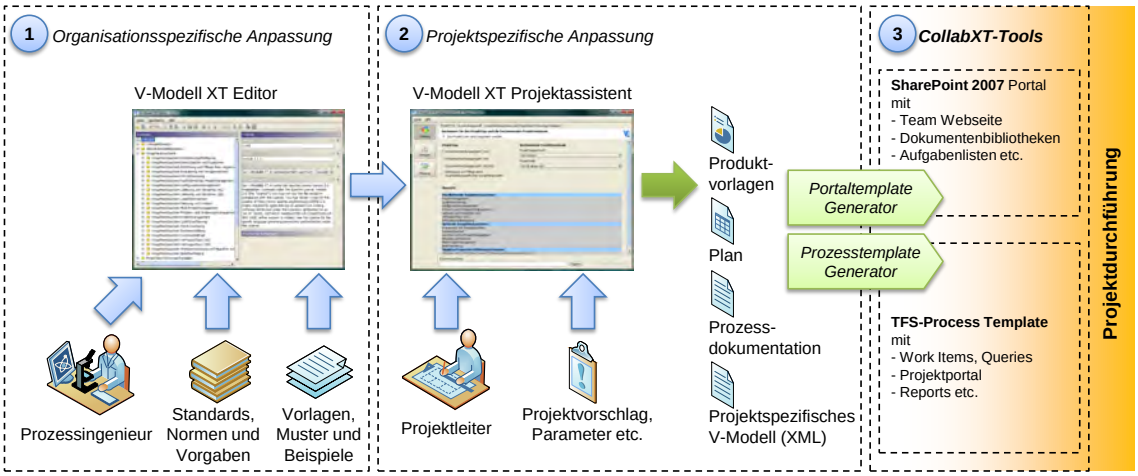


Abbildung 2.1.: Werkzeugspezifische Anpassung als Forsetzung des Tailorings

hergestellt. Es ist zu sehen, dass verschiedene Zielumgebungen zu erwarten sind. Für diese Umgebungen sind angepasste Übersetzungsvorschriften erforderlich.

Übersetzungsvorschriften. Die Übersetzungsvorschriften zwischen Prozess- und Werkzeugmodell lassen sich zunächst generisch beschreiben. Auf der Seite des V-Modells sind die Aussagen durch das Metamodell (vgl. Abschnitt 2.2) sehr präzise. Auf der anderen Seite sind die Zielumgebungen als nicht konstante Größen zu finden. Die Miteinbeziehung verschiedener Ziele (Werkzeuge) erfordert auf der Ausgabeseite des Generators eine Spezifikation, die eine Menge zu erfüllender Anforderungen enthält und beschreibt.

Als Verfahren, das für diesen Fall infrage kommt, wird ein Übersetzungsmodell erstellt, dass anhand einer Menge von definierten V-Modell XT Elementen (Abschnitte 2.2 und 2.3 sowie die Kapitel 3 und 4) einen Regelsatz beschreibt, der die Anforderungen an die Zielumgebungen definiert. Die Regelsätze sind dann für die jeweilige Zielumgebung zu spezialisieren, womit vom Grundkonzept ein parametrisierbares Regelwerk entsteht.

Generierungsverfahren. Auf der Basis der geleisteten Vorarbeiten lassen sich mindestens zwei verschiedene Übersetzungs- bzw. Generierungsverfahren angeben.

1. Ein vollständig generierender Ansatz nimmt ein angepasstes V-Modell entgegen und generiert alle Inhalte mitsamt der Strukturen der Plattform. Dies entspricht einer (prozessbezogen) vollständigen Modellierung der Zielplattform[3].
2. Ein partiell generierender Ansatz setzt auf einen vorlagenbasierten Mechanismus, in dem gültige Teilinformationen der Zielplattform als Template hinterlegt sind. Dies ist für Teile von vergleichsweise hoher zeitlicher Stabilität möglich, die somit einen hohen Vorfertigungsanteil bieten. Variable Anteile werden problembezogen durch einen Generator erstellt.

Für Szenarios, in denen eine homogene Integration (nur ein Werkzeug ist das Ziel) vorzunehmen ist, bietet sich das erste Verfahren an. Hier kann ein Werkzeugt relativ präzise und weitgehend spezifiziert und das durch das Werkzeug angebotene API optimal adressiert werden. Für den Anwendungsfall, der durch die SharePoint-Implementierung (Kapitel 3) adressiert wird, ist diese Verfahrensweise angewendet worden.

Für Szenarios, in denen eine komplexe Infrastruktur aus Werkzeugen zu unterstützen ist, ist ein vollständig generierender Ansatz i. d. R. zu aufwändig. Für dieses Szenario, das mit TFS (siehe Kapitel 4) adressiert wird, wird ein Ansatz mit partieller Vorfertigung gewählt umgesetzt.

3 Die vollständige Modellierung von Quell- und Zielumgebungen wird vielfach im Bereich der *Model-driven Architecture* (MDA) gefunden. Diese Bestrebungen bauen im Wesentlichen auf den UML-bezogenen Standards [OMG07a, OMG07b, OMG06a, OMG06b] auf. Andere aber ähnlich motivierte Ansätze sind z. B. bei Greenfield und Short [GS04] zu finden.

2.2. Abbildung des V-Modell XT

In diesem Kapitel fokussieren wir alle relevanten Anteile des V-Modells, die für die Abbildung erforderlich sind. Es sind sehr detaillierte Betrachtungen zu führen, da den Integrationspartnern (Prozess und Werkzeug) i. d. R. unterschiedliche Philosophien zugrunde liegen. Somit können nicht alle Elemente 1:1 abgebildet werden. Weiterhin sind bereits im Vorfeld Entscheidungen zu treffen, wie sich die Unterstützung des V-Modells durch eine integrierte Werkzeugumgebung in der Konkretisierung des Prozesses niederschlägt. Um hier eine grundlegende Vereinbarung hinsichtlich der Abbildungsvolumina zu treffen, legen wir zunächst die Elemente fest, die bei einer Generierung berücksichtigt werden müssen. Im Anschluss unterlegen wir diese Elemente – soweit erforderlich – mit wesentlichen Prozessanteilen für die Projektplanung und -führung.

2.2.1. Klassifikation abzubildender V-Modell-Elemente

Die im Rahmen der Abbildung relevanten Prozesselemente des V-Modells klassifizieren wir zunächst in die Kategorien *Steuerung* und *Inhalt*. Bezogen auf das V-Modell identifizieren wir Steuerungselemente in den Entscheidungspunkten; Inhaltselemente finden wir beispielsweise in Teilprozessen.

Da das V-Modell vollständig durch ein formales Metamodell beschrieben ist, ist dieses der Ausgangspunkt für die Ermittlung der Elemente. Jedoch sind nicht alle Elemente erforderlich, da bestimmte (technische) Aspekte durch die Zielumgebungen abgedeckt werden. Wir reduzieren die Abbildung anhand der gerade benannten Klassifizierung:

Steuerungselemente finden wir dort, wo Projektfortschritte erfasst, gemessen und bewertet werden müssen.

- Entscheidungspunkte (Planung)
- Aktivitäten (Planung)
- Produkte (Status)

Inhaltselemente (Teilprozesse) sind insbesondere im Bereich der Projektergebnisse zu finden.

- Arbeitsauftrag (Prozess Task Management)
- Risiko (Prozess Risikomanagement)
- PÄM (Prozess Problem- und Änderungsmanagement)
- Produkte (Artefakt)

Besonderheiten von Produkten. Produkte ordnen wir in unserer Modellierung nicht ausschließlich den Inhaltselementen zu. In unserer Modellierung sind Produkte im Wesentlichen *statustragende Repräsentanten* von Produkten in einer Produktbibliothek. Im Kontext des TFS z. B. ist für das Work Item Tracking System eine Messgröße für Produkte einzuführen, damit der Status im Rahmen der Projektfortschrittsbestimmung erfasst werden kann. Hierzu wird für jedes Produktexemplar ein Work Item im System gepflegt, das im Wesentlichen die Statusmaschine des V-Modell-Produktsystems abbildet. Für SharePoint hingegen verstehen wir Produkte auch als wirkliche Artefakte, die in einer Dokumentenbibliothek abgelegt werden können.

Entscheidungspunkte, Produkte und Aktivitäten beziehen wir in unserer Modellierung für die Projektsteuerung mit ein. Jedes dieser Metamodellelemente findet in der Zielplattform eine angemessene Entsprechung. Inhaltselemente unterfüttern wir mit kompakten Basisprozessen des V-Modells, wie z. B. Risiko-, Aufgaben- sowie Problem- und Änderungsmanagement. Das V-Modell stellt hierfür verschiedene Produkttypen bereit, die diese Prozesse „versteckt" enthalten und in den einzelnen Produkt- und Aktivitätsbeschreibungen ansatzweise[4] ausgestalten.

Abbildungsrelevante Elemente. Weitere Elemente, die wir in unsere Betrachtung mit aufnehmen sind die Prozessdokumentation und die erzeugte Produktbibliothek. Folgende Elemente/-Teile des V-Modells werden für die Abbildung herangezogen:

4 Das V-Modell stellt an einigen Stellen so genannte beispielhafte Produktgestaltungen zur Verfügung, die wir bei der Modellierung in geeigneter Weise berücksichtigen. Wir explizieren sie dann und stellen sie entsprechend der Zielumgebung zur Verfügung.

Entscheidungspunkt Entscheidungspunkte werden als Planungs- und Messgrößen für die Projektsteuerung herangezogen. Entscheidungspunkte sind als Metamodellelemente des V-Modells definiert. Sie sind in einem Tailoring-Profil projekttypspezifisch (umfangsmäßig) reduziert.

Entscheidungspunkte liegen für die Ausgestaltung bereits teilweise in instanziierter Form vor. Der V-Modell XT Projektassistent liefert im Rahmen der initialen Planung eine entsprechende, konsistente und geordnete Menge. Weitere Entscheidungspunktinstanzen sind ggf. im laufenden Projekt zu definieren.

Produkt Das Produkt als zentrales V-Modell Element ist als Metamodellelement definiert und wird entweder als Repräsentant für ein Produktexemplar oder als Artefakt in einer Dokumentenablage erfasst. Die Messgrößen werden über das *Produktstatusmodell* erfasst, gemessen und ausgewertet.

Bestimmte Produkttypen werden im Kontext von CollabXT als werkzeugspezifische Prozesselemente, z. B. in Form von Work Item Typen modelliert.

Aktivität Die Aktivität ist als Metamodellelement hinterlegt und muss im Rahmen der Modellierung als Planungsgröße herangezogen werden. Die Menge der Aktivitäten ist durch das Tailoring bestimmt. Initial sind maximal so viele Aktivitätstypen wie Produkttypen definiert. Aktivitäten und Produkte werden im Kontext der Entscheidungspunkte miteinander verknüpft.

Arbeitsauftrag Wird auf Basis des Produkts *Arbeitsauftrag* und der assoziierten Aktivitäten erstellt und als Prozess abgebildet. Die Abbildung erfolgt werkzeugspezifisch.

Risiko Wird auf der Basis des Produkts *Risikoliste* und der assoziierten Aktivitäten erstellt und als Prozess abgebildet. Die Abbildung erfolgt werkzeugspezifisch.

PÄM Wird auf Basis der Produkte *Problemmeldung/Änderungsforderung*, *Problem-/Änderungsbewertung*, *Änderungsstatusliste* sowie *Änderungsentscheidung* und der assoziierten Aktivitäten erstellt und zu einem Prozess integriert. Die Abbildung erfolgt werkzeugspezifisch.

Prozessdokumentation Die Prozessdokumentation ist auf der Modellebene im V-Modell selbst hinterlegt. Ein Export z. B. in das HTML-Format stellt nur eine spezialisierte Sicht zur Verfügung. Die Prozessdokumentation kann in verschiedener Weise verwendet werden.

Produktbibliothek Die Produktbibliothek wird beim Tailoring ebenfalls standardmäßig exportiert. Sie wird dabei gleichzeitig spezialisiert und (dynamisch) angepasst.

Sie kann, unter der Voraussetzung, dass ein entsprechendes Konfigurationsmanagementsystem (Anforderung an die Zielplattform) verfügbar ist, 1:1 überführt werden. Weitere Anpassungen sind hierbei dann nicht mehr vorgesehen. Die Produktbibliothek liefert auf der Grundlage des Tailorings die für das Projekt relevanten Vorlagentypen. Diese werden in das KM-System eingespeist und dort zweckabhängig verwendet. Unterschieden werden muss dabei zwischen *initialen* und *abhängigen* Produkten (vgl. Abschnitt 2.2.2). Sie unterscheiden sich hinsichtlich des späteren Instaziierungsverhaltens bei und nach der Generierung.

Bei den gerade aufgeführten Elementen finden sich zum Teil bereits Metamodellelemente des V-Modells wieder. Die Typen *Produkt*, *Aktivität* und *Entscheidungspunkt* entstammen dabei den V-Modell Sub-Metamodellen: Produkt-, Aktivitäts- und Ablaufmodell. Diese Sub-Metamodelle besprechen wir nun.

2.2.2. Produktmodell des V-Modell XT

Das V-Modell XT ist ein *produktorientiertes* bzw. *ergebnisorientiertes* Vorgehensmodell, d.h. nicht Aktivitäten, sondern Projektergebnisse (in Form von Produkten) stehen im Zentrum des Prozesses. Abbildung 2.2 zeigt den betreffenden Ausschnitt des Metamodells für das Produktmodell des V-Modells.

Das Produkt als Typ besitzt nur zwei für uns relevante Beziehungstypen zu anderen Elementen. Einmal ist dies die *Rolle*, also ein Verweis der die Verantwortung[5] für ein Produkt beschreibt. Der zweite Aspekt wird durch den *Entscheidungspunkt* gegeben, der Vereinbarungen hinsichtlich der Fertigstellung von Produkten trifft. Der Entscheidungspunkt ist dabei gleichzeitig Teil des Ablaufmodells (Abschnitt 2.2.4).

5 Die Feststellung der Verantwortlichkeit für ein Produkt ist u.a. eines der Koformitätskriterien des V-Modell XT.

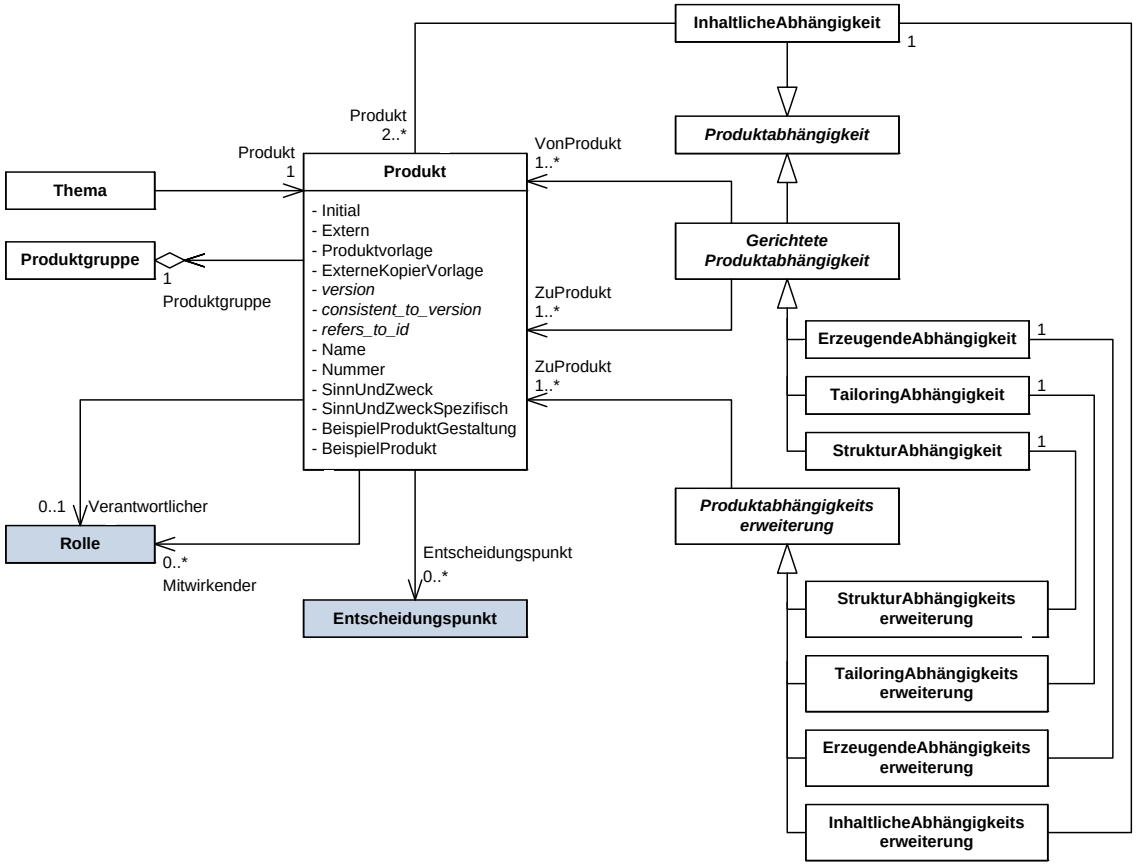


Abbildung 2.2.: Metamodell des V-Modell XT, Sicht: Produkt

Produktabhängigkeiten. Weitere Elemente, die noch zu berücksichtigen sind, sind die *Produktabhängigkeiten*. Diese dienen im V-Modell dazu, Beziehungen und Abhängigkeiten zwischen Produktexemplaren auszudrücken. Sie stellen ein wesentliches Mittel des Prozesses dar, um Erzeugnisstrukturen des V-Modells verstehen und umsetzen zu können. Da das V-Modell seine Ergebnisse nicht durch eine Sequenz von Aktivitäten und deren Ausgaben beschreibt, dienen Produktabhängigkeiten dazu, insbesondere die *erzeugenden Produktabhängigkeiten*, die Notwendigkeit der Erstellung eines Produktexemplars zu regeln.

Beispiel: Ein intuitives Beispiel für eine erzeugende Produktabhängigkeit finden wir in der Abhängigkeit *Produktumfang für das Projektmanagement* des V-Modells. In dieser sind das *Projekthandbuch* und der *Projektplan* als *erzeugende Produkte* definiert, die die Erstellung aller weiteren Managementprodukte motivieren. So z. B. ist im *Projekthandbuch* in den Vorgaben zum Berichtswesen geregelt, *dass* Berichte vom Typ *Projektstatusbericht* erstellt werden. Der genaue Zeitpunkt wird dann durch den *Projektplan* festgelegt. Die Kombination *Projekthandbuch* und *Projektplan* sorgt also dafür, dass zu definierten Zeitpunkten im Projekt ein *Projektstatusbericht* erstellt wird.

Strukturelle Produktabhängigkeiten bilden die Basis der Systemstruktur im Entwicklungsanteil des V-Modells. Hier werden *Teile-Ganzes*-Beziehungen zwischen System, Teilsystemen und sonstigen Komponenten und Modulen hergestellt. Diese Strukturen finden sich im V-Modell auch bei den Projektdurchführungsstrategien wider. Dort – insbesondere in den Entwicklungsanteilen der AN- und AG/AN-Strategien – sind durch die Entscheidungspunkte Ergebnisse angefordert, die sich passend in der Systemstruktur wiederfinden.

2.2.3. Aktivitätsmodell des V-Modell XT

Da das V-Modell ein produktzentriertes Vorgehensmodell ist, ist das Aktivitätsmodell vergleichsweise undetailliert ausgearbeitet. Vielmehr findet sich in den Aktivitäten eine eher allgemein gehaltene, grobgranulare Beschreibung von Aktionen, die zur Produkterstellung durchgeführt werden müssen. Aktivitäten im V-Modell dienen nur der Fertigstellung eines einzigen Produkts (sie-

he Abbildung 2.3, Assoziation *stellt_fertig* 0..1 - 1). Die kontinuierliche Bearbeitung wird durch das V-Modell nicht beschrieben. Im Rahmen der Abbildung auf ein unterstützendes und prozessführendes Werkzeug muss daher das Aktivitätsmodell für die Kopplung mit Planungswerkzeugen erweitert und ausgestaltet [Gna05] werden.

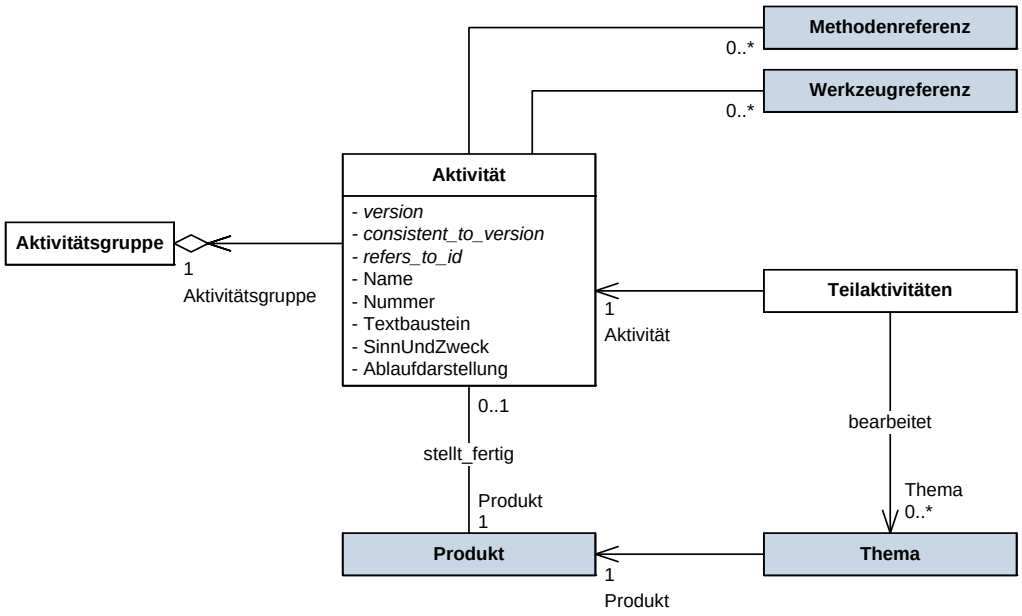


Abbildung 2.3.: Metamodell des V-Modell XT, Sicht: Aktivität

Unter der Aktivität ist im V-Modell XT noch die *Teilaktivität* zu finden. Diese strukturiert eine Aktivität weiter und beschreibt wesentliche Arbeitsschritte zu deren Durchführung. Ähnlich zu Produkten definiert eine Teilaktivität somit quasi Anforderungen an eine Aktivität. Anders als bei den Produkten, ist es hier sinnvoll, die Teilaktivitäten weiter zu berücksichtigen. Da Aktivitäten nur grobgranular beschrieben sind, haben wir hier die Möglichkeit durch eine Menge von Teilaktivitäten eine entsprechende Verfeinerung zu erhalten, mit deren Hilfe wir sogar eine Integration weiterer Prozesse und Methoden realisieren können.

Abbildungsvereinbarung von Teilaktivitäten

Aktivitäten betrachten wir im Folgenden als Planungsgrößen. Dies hat zur Folge, dass wir sie mit entsprechenden Informationen anreichern müssen, um sie unmittelbar für Planungswerkzeuge verfügbar zu machen. Da Aktiviäten Teilaktivitäten enthalten, kann entweder rekursiv eine (beliebig tiefe) Hierarchie über Aktivitäten erstellt werden, oder das Konzept Teilaktivität explizit modelliert werden oder das Konzept Teilaktivität wird zugunsten eines alternativen Konzepts fallen gelassen. Wir entscheiden uns für die letzte Variante und ersetzen Teilaktivitäten vollständig durch die modellierten Elemente des Aufgabenmanagements (vgl. Abschnitt 2.3).

2.2.4. Ablaufmodell des V-Modell XT

Das Ablaufmodell des V-Modell XT (Abbildung 2.4) dient der groben Strukturierung des Projekts und dessen Einteilung in verschiedene Projektfortschrittstufen. Zentrale Elemente sind *Entscheidungspunkte*, die *Projektdurchführungsstrategien* und wieder die Produkte.

Referenzverfahren für die Planermittlung. Der in Abbildung 2.4 gezeigte Ausschnitt des Metamodells zeigt die Einordnung dieser Elemente im V-Modell Metamodell. Über diesen Modellausschnitt wird aber nicht nur die (logische, zeitliche) Projektstrukturierung vorgenommen, sondern gleichzeitig die Planungskomponente des V-Modell XT Projektassistenten [Ber06] unterstützt. Diese greift auf die vorliegenden Daten zu und nutzt sie zur Erstellung des initialen Projektplans. Dabei verwendet er im Wesentlichen folgendes Verfahren:

- Eine Projektdurchführungsstrategie fasst mehrere Entscheidungspunkte zusammen und bringt diese in eine mögliche Ordnung.
- Bei der initialen Planung des Projekts werden die Entscheidungspunkte dann gemäß den Vorgaben verschieden oft instanziiert und entsprechend der Projektdurchführungsstrategie und der durch sie definierten Pfade zwischen den einzelnen Entscheidungspunkten geordnet.
- Entscheidungspunkte wiederum werden durch eine Reihe von Produkten referenziert, was nichts anderes heißt, als dass ein Produkt, das einen Entscheidungspunkt referenziert, zu einer Instanz dieses Entscheidungspunktes fertig gestellt und zur Qualitätssicherung vorgelegt werden muss.

Entscheidungspunkte regeln somit die Ergebnismengen in bestimmten Projektfortschrittstufen. Sie verweisen weiterhin auf so genannte *Ablaufentscheidungspunkte*, die dann für Planungen herangezogen werden. Ein Ablaufentscheidungspunkt referenziert einen Ablaufbaustein und kann über eine Nachfolgerbeziehung weiter verzweigen (auch in Parallelabläufe). Der Ablaufbaustein mit Ablaufentscheidungspunkten und Parallelabläufen regelt somit die Erstellungsreihenfolge der Ergebnisse. Ein Ablaufbaustein ist dann wiederum einer Projektdurchführungsstrategie zugeordnet.

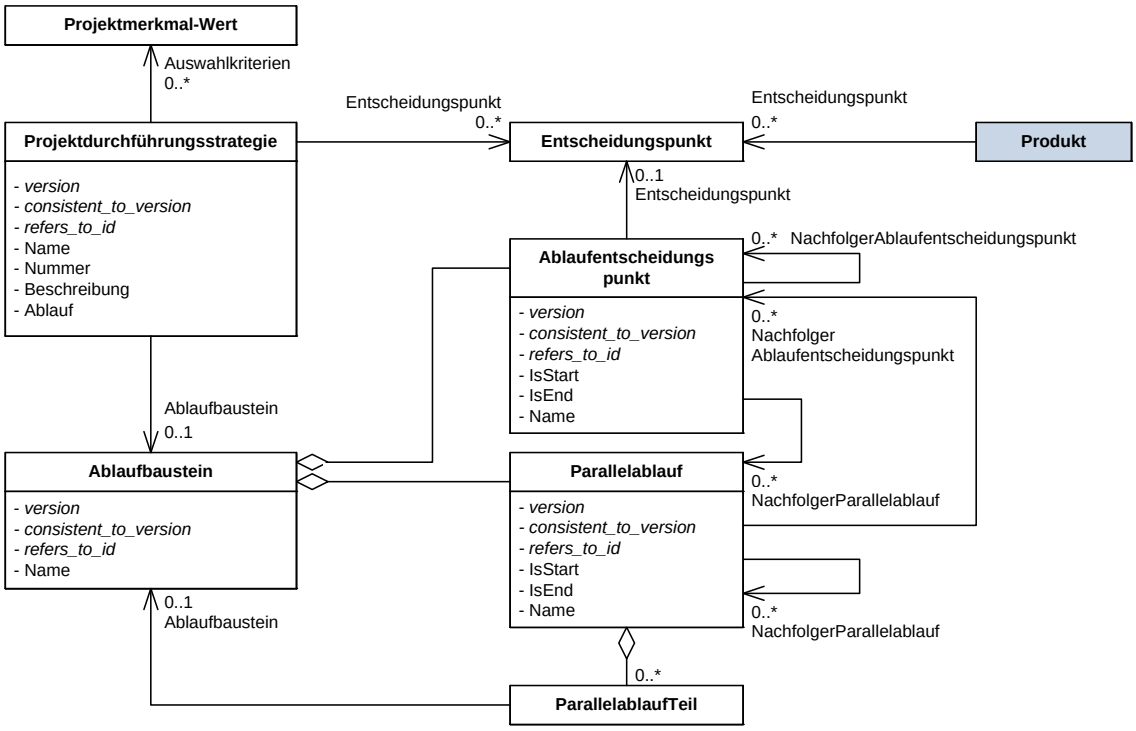


Abbildung 2.4.: Metamodell des V-Modell XT, Sicht: Abläufe und Projektdurchführungsstrategien

Abbildungsvereinbarung zu Entscheidungspunkten

Entscheidungspunkte werden im Projektplan üblicherweise als Meilensteine abgebildet, zu denen bestimmte Vorgänge fällig werden. Diese Umsetzung wird dem Entscheidungspunkt nicht vollständig gerecht, da er nicht nur ein Zieldatum für Aktivitäten/Vorgänge definiert, sondern gleichzeitig zu prüfende Zwischenergebnisse. Er enthält somit in der notwendigen Ausgestaltung sowohl Eigenschaften eines Meilensteins als auch die eine Quality Gates.

Achtung: Das hier beschriebene Verfahren, inklusive der technischen und konzeptionellen Grundlagen ist zurzeit Gegenstand einer Überarbeitung. Im Rahmen der Überarbeitung des Metamodells des V-Modells wird das Konzept der Projektdurchführungsstrategie mitsamt der darunter liegenden Konzepte besser in das Tailoring integriert. Mit Hinblick auf die Erweiterbarkeit und die Standardisierung des V-Modells werden weiterhin Schnittstellen geschaffen, die definierte

Erweiterungen und Anpassungen von Projektdurchführungsstrategien sowie die Nachnutzung einzelner Ablaufbausteine vereinfachen.

2.3. Abzubildende Prozesse

Das V-Modell XT ist ein Vorgehensmodell, das neben der Produktzentrierung gleichzeitig versucht, eine hohe Wiederverwendbarkeit zu erreichen, indem es verschiedene Verfahren beschreibt und für die Anwendung empfiehlt, jedoch keine konkreten Methoden vorgibt. Das V-Modell ist *methodenneutral*. Für die Anwendung im Kontext konkreter Werkzeuge ist es jedoch erforderlich, an einigen Stellen Methoden auszugestalten, vorzufertigen und verbindlich vorzugeben. Diese vorgefertigten Methoden (ausgestaltete Prozesse) dienen dann im Weiteren als Schnittstellen für die bereits existierenden Prozesse der Zielumgebung, die passend und sinnvoll mit dem V-Modell integriert werden müssen. Wir haben zu Beginn dieses Kapitels bereits drei passende Elemente bzw. Prozesse identifiziert:

- Arbeitsauftrag (Prozess Aufgaben- bzw. Taskmanagement)
- Risiko (Prozess Risikomanagement)
- PÄM (Prozess Problem- und Änderungsmanagement)

Aufgabenmanagement

Das Aufgabenmanagement ist ein elementarer Prozess, der notwendig ist, um den Prozess V-Modell „zum Leben zu erwecken". Es wird durch das Produkt *Arbeitsauftrag* im V-Modell erfasst. Elementare Aufgaben können im Projekt vergeben und überwacht werden. Eine einfache Statusüberwachung von Aufgaben wird ermöglicht. Unter Berücksichtigung der Modellierung der Metamodellelemente Produkt, Aktivität und Entscheidungspunkt sind mit Hinzunahme eines Aufgabenmanagements bereits alle elementaren Bestandteile für eine einfache Projektsteuerung vorhanden.

Abgrenzen müssen wir die Konzepte *Arbeitsauftrag*, *Aktivität* und *Teilaktivität*, die wir im Kontext dieser Abbildung parallel vorhalten. Aktivitäten sind messbare Repräsentanten des Metamodellkonzepts aus dem V-Modell. Für konkrete Aufgaben, bzw. für eine konkrete Leistungserbringung, sind Aktivitäten zu grobgranular. Außerdem können sie kontinuierliche Arbeiten an einem Produkt nicht abdecken. Aktivitäten sind weiterhin ausschließlich einem Produkt zu geordnet. Diese Beschränkungen gelten nicht für den Arbeitsauftrag. Dieser ist ein Konzept, das eine konkrete Handlung verursacht. Aus diesem Grund modellieren wir bspw. auch keine Teilaktivitäten, sondern erfassen dieses Konzept sofort mit dem Arbeitsauftrag.

Konzeption: Aktivität/Arbeitsauftrag

Wir bilden auf der Konzeptebene die Kombination Aktivität/Teilaktivität auf Aktivität/Arbeitsauftrag ab.

Risikomanagement

Das Risikomanagement im Rahmen des V-Modells wird umgesetzt durch das Produkt *Risikoliste* und die korrespondierende Aktivität. Das Produkt Risikoliste definiert als Themenstruktur die identifizierten Risiken und einen Maßnahmenplan zur Minderung und Behandlung von Risiken. Wir nehmen auch hier – soweit möglich – eine Ersetzung des V-Modell-Produkts vor. Im Falle der Ersetzung müssen wir einen Prozess bzw. Workflow definieren, der wesentlich konkreter ist als die Vorgaben des V-Modells.

Problem- und Änderungsmanagement

Das Problem- und Änderungsmanagement ist ebenfalls wie das Aufgaben- und das Risikomanagement eine der zu unterstützenden Kerndisziplinen des V-Modells. Anders bei den anderen beiden betrachteten Disziplinen betrachten wir hier nicht nur einen Produkttyp, der in einem der Kernbausteine integriert ist, sondern einen auch im V-Modell separat hinterlegten Teilprozess. Diesen müssen wir komplett fertigen und sinnvoll abbilden.

2.4. Weiteres und Besonderheiten

Es gibt eine Reihe von Disziplinen und Elementen im V-Modell, die geeignet wären, um z. B. in Form von TFS-Work Items definiert zu werden. Andere Elemente im V-Modell entfallen bei genauerer Betrachtung der Zielplattformen und werden entsprechend der Gegebenheiten abgebildet. Beispielhaft sei hier das Konfigurationsmanagement (KM) zu nennen, das in wesentlichen Teilen durch die Infrastrukturen der Werkzeuge erbracht werden kann. Wir gehen in diesem Abschnitt auf noch verbleibende Elemente ein und betrachten dabei das Rollenmodell und die auch Anpassung des V-Modells.

2.4.1. Behandlung des Rollenmodells

Das V-Modell beinhaltet ein Rollenmodell, das anders als wie bei vergleichbaren Vorgehensmodellen nicht aufgabenorientiert, sondern *verantwortlichkeitsbasiert* aufgebaut ist.

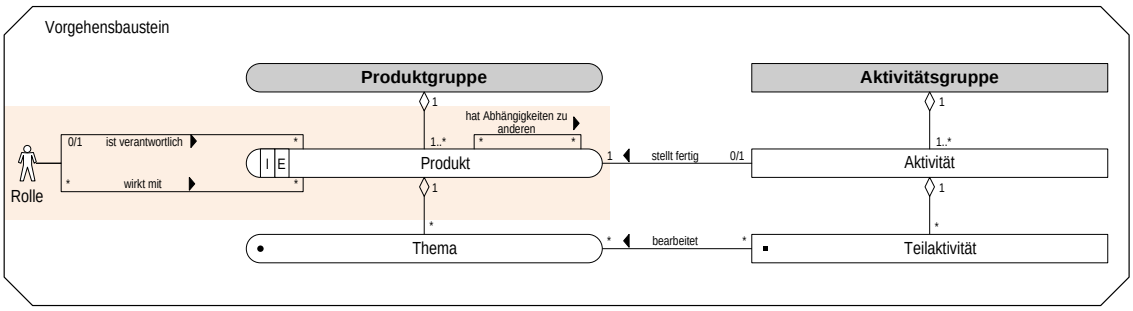


Abbildung 2.5.: Position der Rollen im V-Modell-Metamodell

Abbildung 2.5 zeigt einen Ausschnitt des V-Modell-Metamodells[6], der die Rolleneinbindung zeigt. Rollen im V-Modell beziehen sich auf Produkte. Für ein Produkt ist immer maximal eine Rolle *verantwortlich*. Weiterhin können für ein Produkt mehrere Rollen *mitwirkend* angegeben werden. Die Verantwortlichkeit von Rollen orientiert sich an der Qualität der fertigzustellenden Produkte. Die Rollen des V-Modells sind *nicht* an Aktivitäten gebunden.

Dies hat Konsequenzen für die implementierenden Werkzeuge, die Rollen i. d. R. sicherheitsbezogen sehen. Rollen sind üblicherweise mit Berechtigungen verbunden. Somit sind Rollen im Spannungsfeld von Verantwortlichkeit für Produkte, Ergebnisbearbeitung und Sicherheit zu finden. Da das V-Modell ein hierarchieloses, sehr einfaches Rollenmodell enthält, sind Abbildungen auf ein Sicherheitsmodell sehr stark individualisierbar. Eine Abbildung kann bspw. erfolgen, indem Rollen auf Sicherheitsobjekte abgebildet werden. Über die im Metamodell des V-Modells hinterlegten Assoziationen können Berechtigungssätze errechnet werden.

2.4.2. Anpassung des V-Modells für Organisationen

Wir definieren unser Konzept und somit unsere Abbildungen auf der Basis des Standard V-Modells 1.2.1. Üblicherweise wird das V-Modell in dieser reinen Form nicht verwendet, sondern *organisationsspezifisch* angepasst (siehe Abbildung 2.1, Stufe 1). Diese Anpassungsspezifika erfassen wir nicht, sondern definieren eine Abbildung, die das V-Modell trotz Integration in eine konkrete Entwicklungsplattform, noch weitgehend allgemein hält. Ziel dieses Vorgehens ist es im Sinne von Abschnitt 2.1, die Werkzeugkette transparent zu erweitern und dem Tailoring nachgelagert eine *inhaltserhaltende* Anpassung auf spezifische Werkzeugkontexte anzuschließen.

Anpassungsprozess des V-Modell XT

Die organisationsspezifische Anpassung sehen wir trotzdem nicht nur als Option an, sondern als *Notwendigkeit*. Wir beschreiben diesen Prozess im folgenden kurz. Als Orientierung verwen-

6 Die hier gezeigte Positionierung des Rollenmodells basiert auf der V-Modell-Version 1.2.1 und wird sich in den Metamodellüberarbeitungen verändern.

den wir Abbildung 2.6, die sowohl die organisationsspezifische als auch die projektspezifische Anpassung skizziert. Relevant ist dabei für diesen Kontext der Bezug zum nachgelagerten Generierungsprozess für die Werkzeuge.

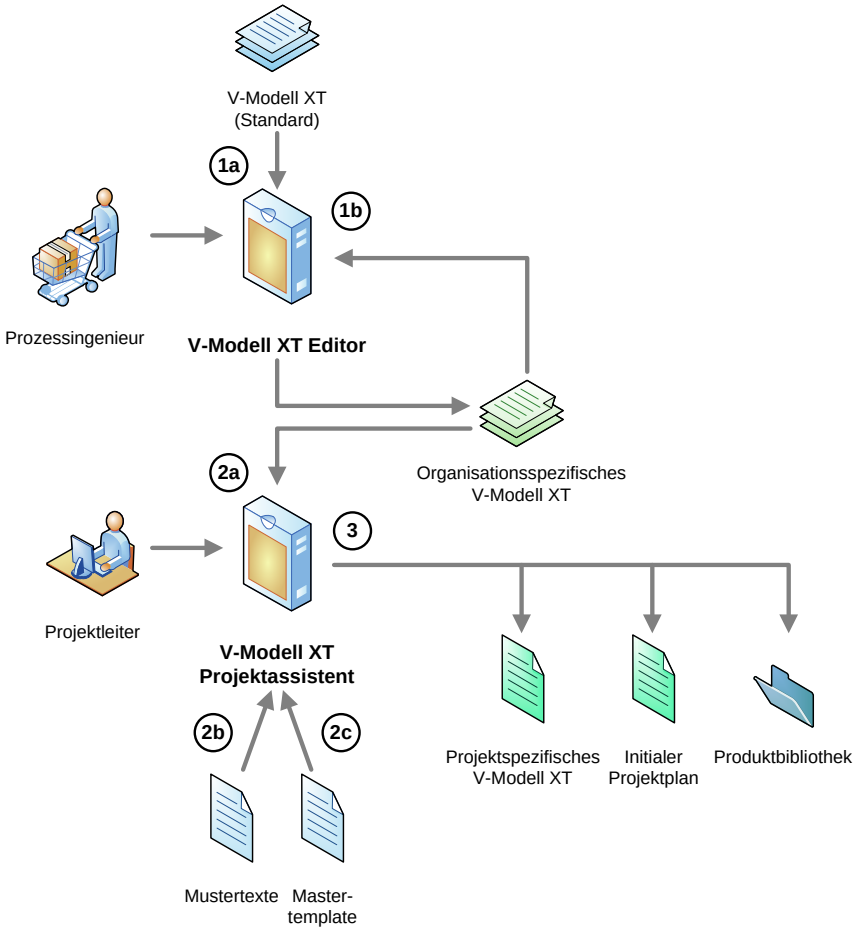


Abbildung 2.6.: Anpassungsprozess des V-Modell XT

Schritt 1a beschreibt die organisationsspezifische Anpassung des V-Modells [KHTS07] mit dem V-Modell XT Editor. Eingabe für den Editor ist ein V-Modell-Derivat. Ausgabe ist ebenfalls ein V-Modell-Derivat (Schritte 1a und 1b). Solange das V-Modell XT-Metamodell als Grundlage für diesen Arbeitsschritt dient, ist die sogenannte *konstruktive Konformität* sicher gestellt[7].

Der aktuell gültige[8] Anpassungsprozess für das V-Modell XT orientiert sich an den Referenzwerkzeugen und den Vorgaben, die das V-Modell durch das Metamodell macht. Ausgegangen wird vom Standard V-Modell XT, das durch einen Prozessingenieur an die jeweilige Organisation angepasst wird. Die Anpassung umfasst dabei neben der Anpassung der Prozessbeschreibungen und Bilder auch die Anpassung von Exportstrukturen und Vorlagen (vgl. Abbildung 2.7), sowie ggf. die Erstellung und Bereitstellung von Beispielen und ähnlichem. Dieser Vorgang ist durch den *V-Modell XT Editor* werkzeugunterstützt. Als Ergebnis liegt ein organisationsspezifisches V-Modell vor, das weiter gepflegt oder dem *V-Modell XT Projektassistent* als Eingabe (Schritt 2a) zugeführt werden kann. Der Projektassistent nimmt das organisationsspezifische V-Modell, mischt Mustertexte (Schritt 2b) bei und stellt ein Mastertemplate (Schritt 2c) für die Vorlagengenerierung bereit. Mit diesen Eingaben führt er das Tailoring durch und produziert seine Ausgaben für die weitere Verwendung. Die Ausgaben werden mithilfe der Werkzeuge aus CollabXT für die kon-

7 Im Rahmen des Projekts WEIT-IV/2 wurde hierfür ein detailliertes Konzept erarbeitet, das die genauen Anforderungen detailliert beschreibt. Für den aktuellen Kontext können wir folgende Aussage als gültig annehmen: „Konstruktiv konform ist ein Vorgehensmodell, das auf der Basis des V-Modell-Referenzmodells und mit den Mitteln des V-Modells erstellt wurde.

8 Ein verbindlicher, prozesslinienorientierter Prozess zur standardkonformen Entwicklung und Anpassung des V-Modell XT ist zurzeit noch nicht verfügbar und wird im Rahmen des Projekts WEIT-IV/2 und V-Bench entwickelt.

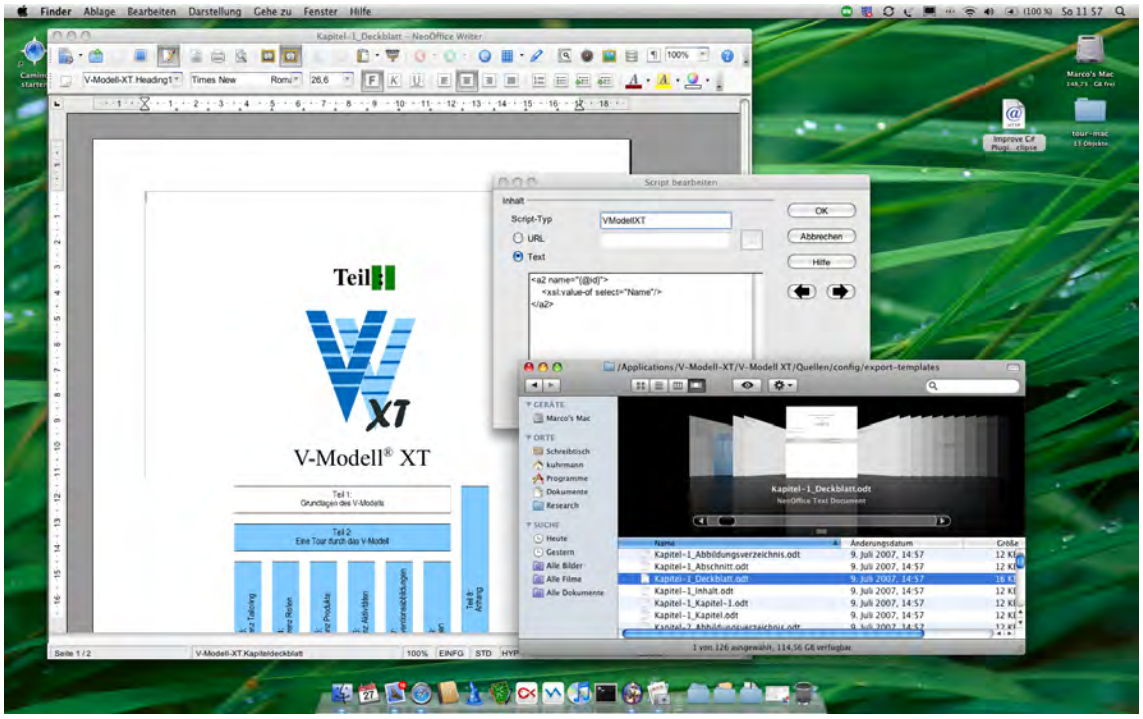


Abbildung 2.7.: Anpassung für Dokumente und Exportvorlage des V-Modell XT im Rahmen einer organisationsspezifischen Anpassung

krete Zielplattform transformiert. Wie bereits in Abbildung 2.1 gezeigt, ergänzen die Werkzeuge von CollabXT diese Kette von Referenzwerkzeugen.

2.4.3. Weitergehende Anpassung des V-Modells

Aber auch nach der organisationsspezifsichen Anpassung des V-Modells sind i. d. R. noch weitere Anpassungsstufen erforderlich. In [Kuh06] haben wir dieses Thema bereits erörtert und konkrete Szenarios vorgeschlagen, wie nach einer organisationsspezifischen und einer projektspezifischen Anpassung das V-Modell effizient in die Praxis überführt werden kann. Diese Fragestellung ist en antreibendes Moment hinter CollabXT. Es reicht nicht, ein gegebenes V-Modell *einfach nur* in ein CVS einzuspielen. Es kommt darauf an, Anwendern einen Mehrwert zu bieten, der mit der Anwendung und Einführung des V-Modells verbunden ist. Um dies zu erreichen, sind zwei Felder zu berücksichtigen:

- Inhaltliche, projektbezogene Anpassungen
- Technische, organisations-/projektspezifische Anpassungen

Inhaltliche Anpassungen

Inhaltliche Anpassungen treten üblicherweise in konkreten Projektkontexten auf, wo Ad-hoc Entscheidungen treffen und Verfahren anzuwenden sind. Derartige inhaltliche Anpassungen sind nur schwer konstruktiv zu erfassen; lediglich die Optionen hierfür sind vorzusehen. Im Kontext des V-Modell XT sind derartige Anpassungen gestattet und erwünscht. das V-Modell definiert mit den Themen *Abweichungen vom V-Modell* und *Organisation und Vorgaben zu…* das *Projekthandbuchs* definierte Stellen zur Dokumentation konkreter, nicht unmittelbar erfasster Prozesse. Dokumentierte Abweichungen gefährden dabei grundsätzlich nicht die (analytische) Konformität zum V-Modell.

Einige inhaltliche Anpassungen lassen sich aber eingeschränkt vorfertigen. In Abschnitt 2.3 sind wir auf Standardprozesse eingegangen, die sich für eine vorgelagerte Ausgestaltung anbieten. Die Ansatzpunkte für die Ausgestaltung sind:

- Einführung, Anpassung und Dokumentation der ausgestalteten Verfahren in einer organisationsspezifisch angepassten V-Modell-Variante. V-Modell-Elemente hierfür sind z. B.: Produkte, Rollen, Aktivitäten, Teilaktivitäten etc.
- Dokumentation des angepassten Verfahrens in Form von Mustertexten, die beim Tailoring für die inhaltliche Generierung der Produktvorlagen zur Verfügung stehen.
- Erstellung einer separaten Anwendungsvorschrift/eines Anwendungsleitfadens als Ergänzung zu einer verwendeten V-Modell-Variante, die alle relevanten Prozesse und Ergebnisse beschreibt.

Insbesondere die Mustertexte eignen sich zur Beschreibung konkreter, projektbezogener Prozesse. Sie werden zum spätest möglichen Zeitpunkt des Tailorings (im Rahmen der Generierung der Produktvorlagen) erfasst und verarbeitet, sodass sie relativ unabhängig von konkreten Vorgaben eingebracht werden können. Mustertexte eignen sich weiterhin sehr gut für kleinere Anpassungen und Ausgestaltungen, die eine umfassende organisationsspezifische Anpassung des gesamten V-Modells nicht rechtfertigen. Sie werden getrennt vom eigentlichen V-Modell definiert und separat abgelegt[9].

Technische Anpassung

Die zweite Option zur weitergehenden Anpassung des V-Modells besteht in der Anpassung hinsichtlich der in einer Organisation verwendeten Werkzeuge und Werkzeuginfrastrukturen. Inhalte des V-Modells sowie grundlegende Prozessstrukturen bleiben hierbei erhalten. Lediglich auf der Basis des dem Modell zugrunde liegenden Metamodells werden Anpassungen vorgenommen, sodass das Prozessmetamodell mit dem Werkzeugobjektmodell verträglich umgeformt wird. Diesen Weg beschreiten wir CollabXT.

Prozesslebenszyklus von Zielwerkzeugen. Sofern eine Übersetzung des V-Modells für eine konkrete Zielplattform stattgefunden hat, ist eine weitere Problematik unbedingt zu berücksichtigen: Unterstützt die Zielplattform ein eigenes Prozessmodell, so verfügt dieses auch über ein eigenes *Lebenszyklusmodell*, in dem Erstellung, Anpassung, Verteilung und Instanziierung definiert sind. Beispielhaft beziehen wir uns hier auf das Ziel *Team Foundation Server*, der eine eigene Werkzeugunterstützung für die Anpassung der Process Templates zur Verfügung stellt. Anhand von Abbildung 2.8 skizzieren wir den Lebenszyklus am Beispiel des *Microsoft Solutions Framework* [Tur06], das von Mircosoft als Standardprozess für den TFS mit ausgeliefert wird.

Ein Prozessingenieur muss zunächst alle relevanten Inhalte zusammentragen und diese dann mithilfe des *Process Template Editor* sinnvoll zusammenstellen[10] (Schritt 1a in Abbildung 2.8). Das Ergebnis, ein TFS-basiertes Process Template z. B. *MSF for Agile Software Development*, muss in einen TFS übertragen werden (Schritt 2). Hierzu dient der Process Template Manager, der neben der Bereitstellung der Templates gleichzeitig auch einen Export für weitere Anpassungen gestattet (Schritt 1b). Ist ein Process Template einmal im Process Template Manager installiert, kann es im Folgenden für jedes neue Projekt als Vorlage herangezogen werden. Legt ein Projektleiter ein neues Projekt an (Schritt 3), muss er ein Template auswählen, das dann für die Grundkonfiguration des Projekts verwendet wird. Aber auch ein bereits instanziiertes Template (hier also ein konkretes Vorgehen) kann im laufenden Betrieb eines Projekts noch (in Grenzen) angepasst werden (Schritt 1c). Da Microsoft lediglich Anpassungsoptionen und Randbedingungen nicht jedoch einen vollständigen Lifecycle für TFS-basierte Prozess definiert, sind alle Operationen im Bereich der Prozessverbesserung [Kne06] zunächst (technisch) risikobehaftet.

Kombinierter Lebenszyklus. Gehen wir zurück auf die Intention hinter CollabXT, so sollten Inhalte des V-Modells auf verschiedene Werkzeuginfrastrukturen, z. B. TFS, überführt werden. Mit CollabXT werden also mehrere Lebenszyklusmodelle miteinander kombiniert. Problematisch an dieser Kombination ist den technischen Risiken, dass Entwicklungen am Prozess parallel verlaufen können und die kombinierten Lebenszyklusmodelle eigene/differierende Anpassungen vorsehen. Die nachlaufende Anpassung eines aus einem V-Modell generierten Process Templates

9 Dieses Verfahren steht zurzeit auf dem Prüfstand, ob es als reguläres Feature in das V-Modell XT integriert wird. Aktuell sind Mustertexte und deren Integration eine Funktion des Projektassistenten.

10 Diese Aufgabe wird durch die Werkzeuge aus CollabXT automatisiert, die die Inhalte aus dem V-Modell XT entnehmen und auf eine gültige Templatestruktur übersetzen.

stellt hier einen *besonders kritischen Punkt* dar, da Anpassungen im Template nicht auf das ursprünglich verwendete V-Modell wirken. Das Ergebnis einer solchen Anpassung ist durch den Generator (Kapitel 4) nicht wiederherstellbar, womit die beiden Prozesse nicht mehr *konsistent* sind. Im schlimmsten Fall geht die Konformität zum V-Modell XT verloren. Pauschal kann hierzu jedoch keine Aussage getroffen werden – dies ist im Zweifelsfall durch ein individuelles Assessment zu klären.

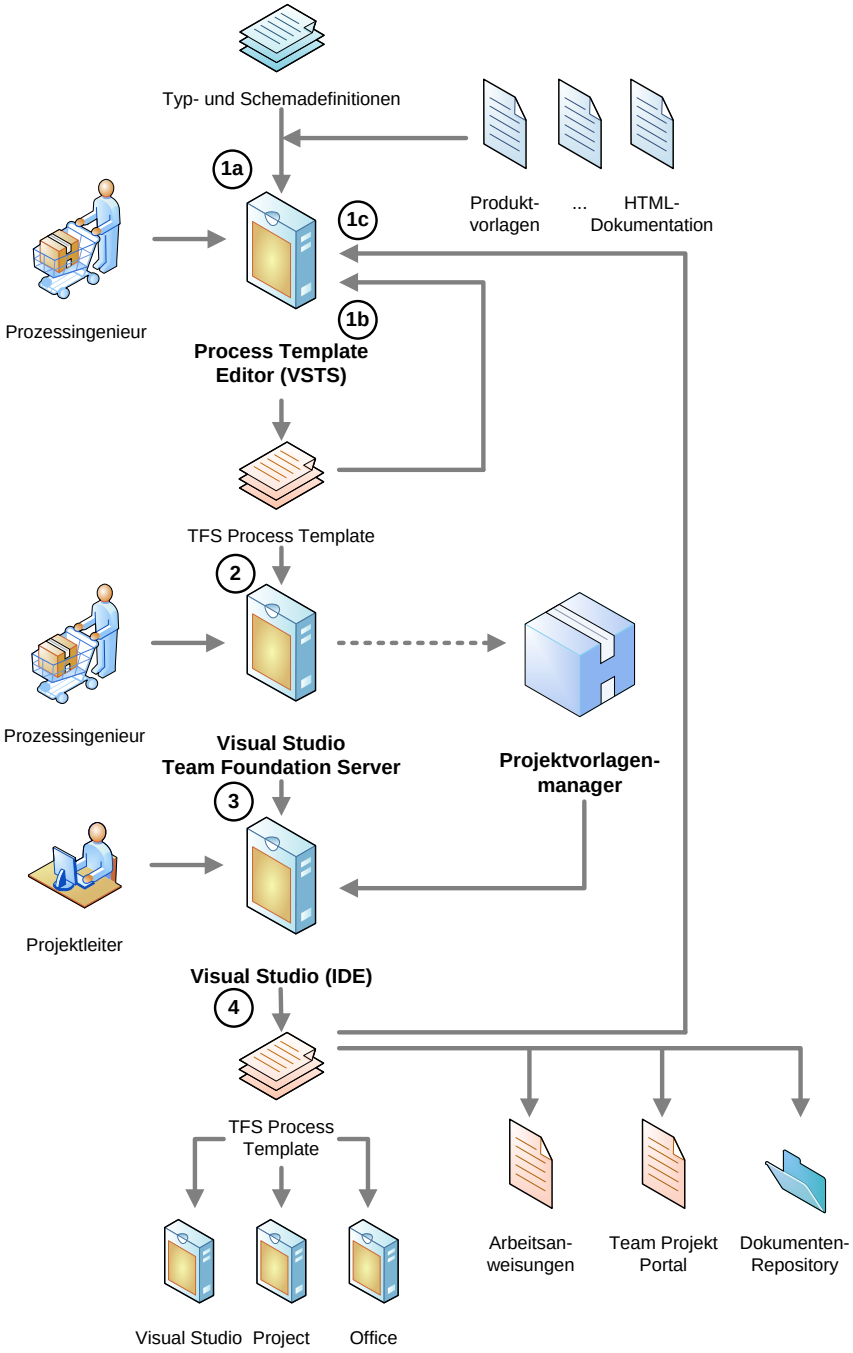


Abbildung 2.8.: Anpassungsprozess des MSF auf Basis eines TFS Process Template

Zusammenfassung

Die diesem Abschnitt haben wir das V-Modell XT unter Berücksichtigung der Abbildbarkeit auf verschiedene Werkzeug-Infrastrukturen betrachtet. Wir haben dabei in Abschnitt 2.2 die relevanten Metamodellelemente des V-Modells beschrieben und haben darauf aufbauend im Ab-

schnitt 2.3 Prozesse identifiziert, die wir für eine Abbildung im Sinne einer Operationalisierung in Betracht ziehen können. Im Abschnitt 2.4 sind wir auf Eigenschaften eingegangen, die bei der Anpassung und Überführung des V-Modells berücksichtigt werden müssen. Dazu zählt die angemessene Abbildung des Rollenmodells ebenso wie die Berücksichtigung des/der Lebenszyklusmodelle der betrachteten Vorgehensmodelle.

3. CollabXT – SharePoint

In diesem Kapitel stellen wir die CollabXT-Implementierung für den *Microsoft Office SharePoint Server* (MOSS) vor – CollabXT-SP. Diese wurde an der TU München entwickelt. Die Generator-Software ist als Open Source unter `http://www.codeplex.com/collabxt` verfügbar. Wir stellen nun die Identifikation der benötigen V-Modell-Anteile und deren grundlegende Modellierung im Kontext des MOSS vor.

Einige Bemerkungen vorweg...

Anpassungen des V-Modells müssen im aktuellen Status der Software zu Beginn der Generierung *vollständig abgeschlossen* sein. Der CollabXT-SP Generator sieht aktuell keine Möglichkeit vor, organisationsspezifische oder projektspezifische Anpassung in der Zielplattform selbst durchzuführen. Da sich CollabXT-SP zur Generierung des Zielportals allerdings ausschließlich auf das V-Modell-Metamodell, nicht aber auf Inhalte, abstützt, ist es gegenüber Anpassungen sehr tolerant. Das trifft z. B. auch auf die englische Version des V-Modells zu.

3.1. SharePoint – Einführung und Überblick

Zur eingehenden Beschreibung von CollabXT-SharePoint charakterisieren wir zunächst die Zielplattform: Microsoft Office SharePoint 2007 Server. Aus den von dieser Software zur Verfügung gestellten Bausteinen wird sich die Zielplattform zusammensetzen. Die Grundelemente zur Gestaltung eines SharePoint 2007 Portals sind in Abbildung 3.1 skizziert.

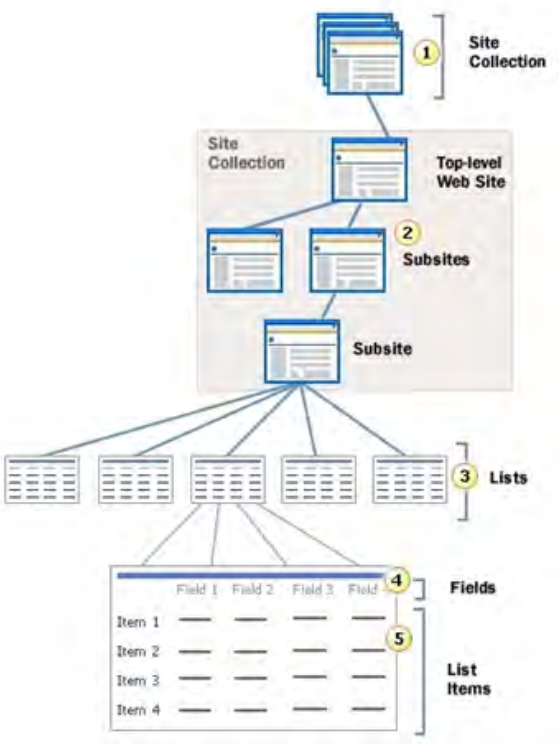


Abbildung 3.1.: Grundlegende Architektur, inhaltlicher Aufbau eines SharePoint-Portals

Site Collections. Die primäre Nutzerschnittstelle zu allen in SharePoint hinterlegten Inhalten ist ein Webbrowser. Damit ist der Grundbaustein eines SharePoint Portals die Webseite. Die Strukturierung und Organisation von Inhalten in SharePoint folgt der Organisation der Webseiten. Ein Baum von Webseiten wird in der Terminologie der Zielplattform als Site Collection bezeichnet. Ein Server kann prinzipiell mehrere Site Collections beheimaten, z. B. für unterschiedliche Unternehmensbereiche.

Webseiten. Die Erzeugung von Webseiten innerhalb einer Webseitenhierarchie kann durch Angabe einer entsprechenden Webseitenvorlage parametrisiert werden. Die Vorlagen geben neben dem Aussehen einer Webseite an, welche Listen und Webparts sich initial in der erzeugten Seite befinden sollen. Zudem kann in einer Webseitenvorlage beispielsweise angegeben werden, ob die Seite im Navigationsmenü der Hauptseite auftauchen soll oder nicht.

Listen. Innerhalb einer Webseite ist die Liste das vorherrschende Strukturelement. Beispielsweise gibt es in SharePoint *Aufgabenlisten*, *Hyperlink*-Listen, frei benutzerdefinierbare Listen, *Dokumentbibliotheken* und viele mehr. Auch das Verzeichnis von Unterwebseiten in einer Webseite ist als Liste verwaltet.

Die Art und das Verhalten von Listen können durch Angabe eines entsprechenden Listentyps spezifiziert werden. SharePoint 2007 bietet dazu eine Reihe von vorgefertigten Listentypen an, wie z. B. Aufgabenliste, Kontaktliste, Ereignisse und Dokumentbibliothek. Listentypen legen fest, welche Felder (also Spalten) eine Liste initial haben soll, was für Ansichten auf dieser Liste möglich sind und optional, welche Inhalte die neu angelegte Liste haben soll.

Listentypen lassen sich verändern und anpassen und selbst frei definieren. Auf diese Weise lässt sich z. B. die von SharePoint 2007 vorgegebene Standard Dokumentbibliothek um weitere Felder erweitern. Die Listen einer Website können mit verschiedenen Berechtigungsstufen versehen werden. Listen können vor Anwendern komplett versteckt werden, wenn sie lediglich interne Informationen zur Anwendung beinhalten. Zudem kann die Bearbeitung der Liste verboten werden.

Felder beschreiben einzelne Spalten einer Liste. Neben einem Datentyp beinhalten Felder[1] weitere Informationen zur entsprechenden Spalte, z. B. ob das Datum dieser Spalte erforderlich für einen gültigen Listeneintrag ist oder einen Voreinstellungswert, falls der Benutzer keinen Wert für die Spalte angibt. Analog zu Listen können Felder in einer Liste vor Benutzern der Seite bzw. der Liste versteckt werden.

Webparts. Webparts werden nicht direkt zur Abbildung von V-Modell-Entitäten verwendet. Da es sich bei Webparts aber um frei programmierbare Teile des SharePoint 2007 Portals handelt, werden sie zur Darstellung und Aufbereitung von V-Modell-Inhalten verwendet. Nicht zuletzt die Oberfläche des Generators für die V-Modell-Portalseite ist unter anderem als Webpart realisiert. SharePoint 2007 verwendet Webparts exzessiv zur Darstellung von Listen und ihren Elementen. Je nach Typ stellt SharePoint 2007 für eine Liste verschiedene Ansichten zur Verfügung. Beispielsweise gibt es für den Listentyp Bildbibliothek eine Ansicht, welche kleine Vorschaubilder (Thumbnails) der beinhalteten Bilder anzeigt. All diese Ansichten wie auch die von SharePoint 2007 zur Verfügung gestellten Standard-Bearbeitungsformulare für Listeneinträge sind in Webparts realisiert.

Workflows. Im SharePoint verwendete Workflows basieren auf der *Windows Workflow Foundation* (WF), die als Bestandteil von .NET 3.x Teil der Laufzeitumgebung ist. In der aktuellen Version nutzen wir, obwohl sich vielfältige Anwendungsgebiete finden lassen, die WF nicht.

3.2. Abbildung des V-Modell XT in SharePoint

Nachdem wir nun die Möglichkeiten und die Bausteine der Zielplattform erfasst haben, wenden wir uns der Abbildung des V-Modells zu. Wie schon in Kapitel 2 angedeutet, wird sich nicht

1 Die wichtigsten Datentypen für Felder sind: Attachments, Boolean, Calculated, Choice, Computed, Counter, Currency, DateTime, File, Guid (Globally Unique Identifier), Integer, Lookup, ModStat (Moderation Status), MultiChoice, Note, Number, Text, URL, User. Eigenschaften der V-Modell XT Entitäten werden auf Felder obiger Typen abgebildet.

das ganze Metamodell in der Zielumgebung wiederfinden. Die technische Grundlage für unsere Zieldefinition bildet das SharePoint-Portal (Abbildung 3.2).

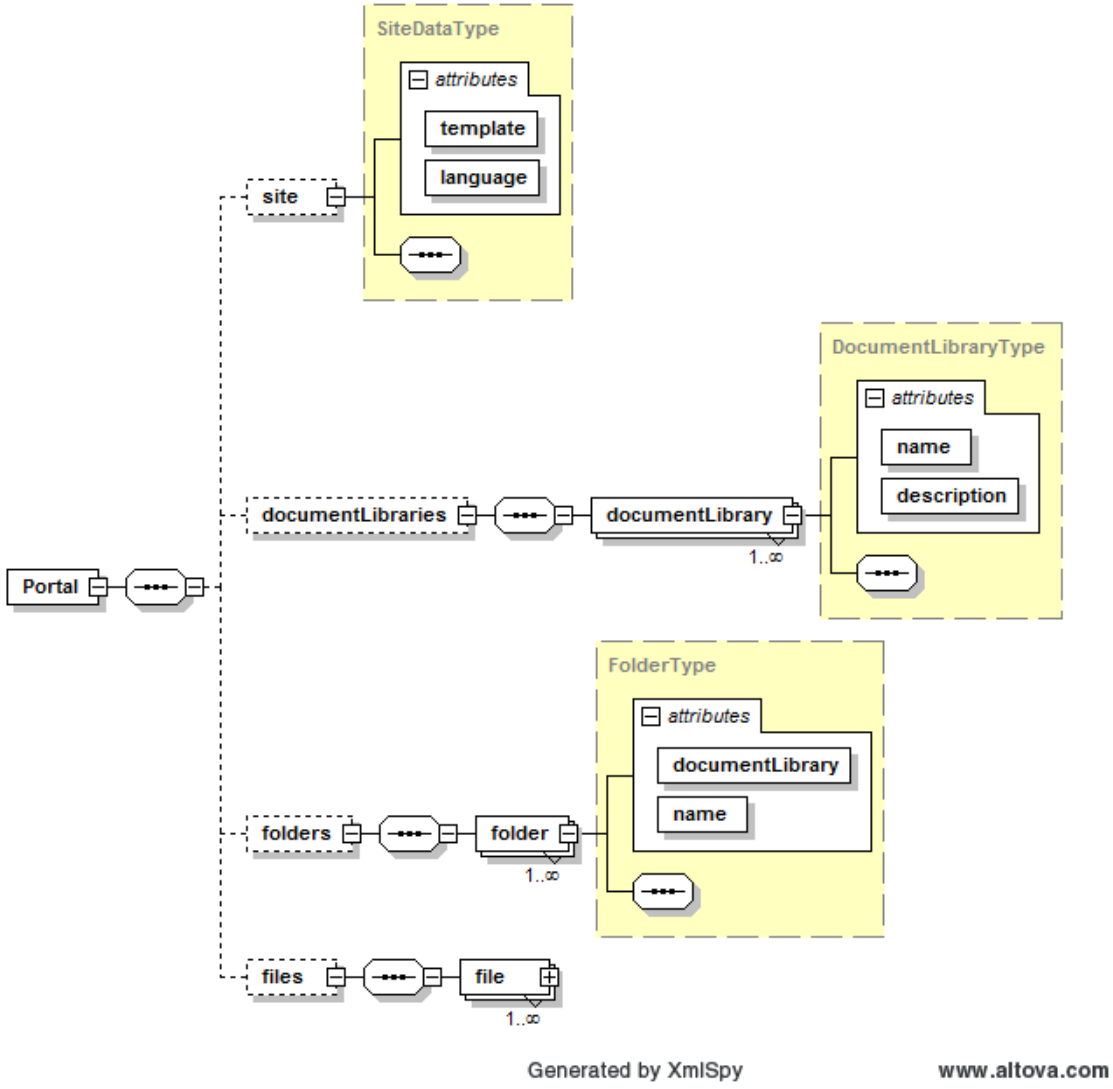


Abbildung 3.2.: XML-Schema eines SharePoint-Portals

Einschränkungen bei der Abbildung. Bevor wir die genauen Submodelle (vgl. Abschnitt 2.2) betrachten, stellen wir noch einige generelle Überlegungen zur Abbildung an. Betrachten wir zuerst die Produkte. Produkte sind im V-Modell-Metamodell in Themen untergliedert. Für die Abbildung wird ein Produkt stets als Einheit betrachtet und dessen Themen nicht in SharePoint dargestellt. Im Produkt selbst sind diese natürlich trotzdem von Bedeutung[2]. Auch die Darstellung der Teilaktivitäten, die diese Themen bearbeiten, entfällt damit. Andere Inhalte des V-Modells sind für die Projektdurchführung nicht von Bedeutung – und damit auch nicht für CollabXT-SP. So macht beispielsweise die Abbildung von Tailoringabhängigkeiten keinen Sinn, da diese nach dem Tailoring keine Relevanz mehr haben (CollabXT-SP knüpft erst an ein abgeschlossenes Tailoring an). Ebenso verhält es sich mit Vorgehensbausteinen und Projektmerkmalen. Diese haben ihre Bedeutung in der Anpassung des V-Modells an die Projektgegebenheiten und sind bei der Durchführung des Projekts irrelevant. Ähnlich verhält es sich mit den Konventionsabbildungen um das V-Modell XT mit anderen Vorgehensmodellen in Beziehung zu setzen. Bei der Projektdurchführung ist die Abbildung des V-Modells auf andere Modelle nicht von zentraler Bedeutung. Dies führt dazu, dass die Elemente:

2 Da die Produkte für die SharePoint-Dokumentenbibliotheken auch vom Projektassistenten übernommen werden, sind die aus dem Tailoring resultierenden Themenstrukturen in den Produktvorlagen selbstverständlich enthalten.

- Vorgehensbaustein
- Projektmerkmal
- Tailoringabhängigkeit
- Konventionsabbildung
- Werkzeug- und Methodenreferenzen

in der Abbildung nicht erfasst werden. Für andere Inhalte ist SharePoint 2007 nur bedingt geeignet. Siehe dazu den Abschnitt 3.2.1 über Produkte. Ziel von CollabXT-SP ist es, die Arbeit mit dem V-Modell XT zu unterstützen und zu vereinfachen. Aus diesem Grund werden gewisse Teilaspekte des Metamodells als Einheit betrachtet. Beispielhaft seien hier die Teilprozesse *Risikomanagement* und *Aufgabenmanagement* (siehe auch Abschnitt 2.3) genannt. In Tabelle 3.1 sind alle abbildungsrelevanten V-Modell-Elemente zusammengefasst.

V-Modell-Element	Erläuterung
Dokumentation	Siehe Abschnitt 3.2.6
Produkte	Siehe Abschnitt 3.2.1
Themen	Siehe Abschnitt 3.2.1
Rollen	Siehe Abschnitt 3.2.3
Aktivitäten	Siehe Abschnitt 3.2.2
Teilaktivitäten	Siehe Abschnitt 3.2.2
Entscheidungspunkte	Siehe Abschnitt 3.2.4
Vorgehensbausteine	Werden nicht abgebildet. Siehe im Text oben.
Produktgruppen	Siehe Abschnitt 3.2.1
Aktivitätsgruppen	Siehe Abschnitt 3.2.2
Tailoringabhängigkeiten	Werden nicht abgebildet. Siehe im Text oben.
Erzeugende PAH	Siehe Abschnitt 3.2.1
Inhaltliche PAH	Siehe Abschnitt 3.2.1
Strukturabhängigkeiten	Siehe Abschnitt 3.2.1
Projektdurchführungsstrategien	Siehe Abschnitt 3.2.4
Ablaufbausteine	Siehe Abschnitt 3.2.4
Projektmerkmale	Werden nicht abgebildet. Siehe im Text oben.
Konventionsabbildungen	Werden nicht abgebildet. Siehe im Text oben.
Methodenreferenzen	Siehe Abschnitt 3.2.6
Werkzeugreferenzen	Siehe Abschnitt 3.2.6

Tabelle 3.1.: Abbildungsumfang des V-Modell XT auf SharePoint 2007

3.2.1. Abbildung: Produkte

Abbildung 2.2 stellt das V-Modell-Metamodell in einem Auszug rund um Produkte dar. Jedes Produkt ist Teil einer Produktgruppe. Diese werden in SharePoint in Form von *Unterwebseiten* (Subwebs) unterhalb der Homepage des Gesamtprojekts dargestellt. Das Zielportal wird also entlang der V-Modell-Produktgruppen organisiert. Jede Produktgruppe ist in einer eigenen SharePoint 2007 Dokumentbibliothek verwaltet. Wir vereinbaren daher für die Abbildung von Produktgruppen: *Produktgruppe* $\Rightarrow$ *Unterwebseite*. Ein beispielhaftes Ergebnis der Generierung einer Produktgruppe ist in Abbildung 3.3 dargestellt.

Wie bereits erwähnt, werden die in einem Produkt zu bearbeitenden Themen nicht explizit in der Zielplattform dargestellt. Sie werden allerdings vom Projektassistenten in die Produktvorlagen generiert und sind damit nicht verloren. Themen sind somit bereits Inhalte der erzeugten Produktvorlagen (ggf. auch bereits mit Mustertexten des Projektassistenten), womit wir sie auf feineren Abbildungsebenen nicht mehr berücksichtigten. Wir nehmen für die Abbildung *Produkte* als atomare Bausteine an. Zur Abbildung verantwortlicher und mitwirkender Rollen siehe Abschnitt 3.2.3.

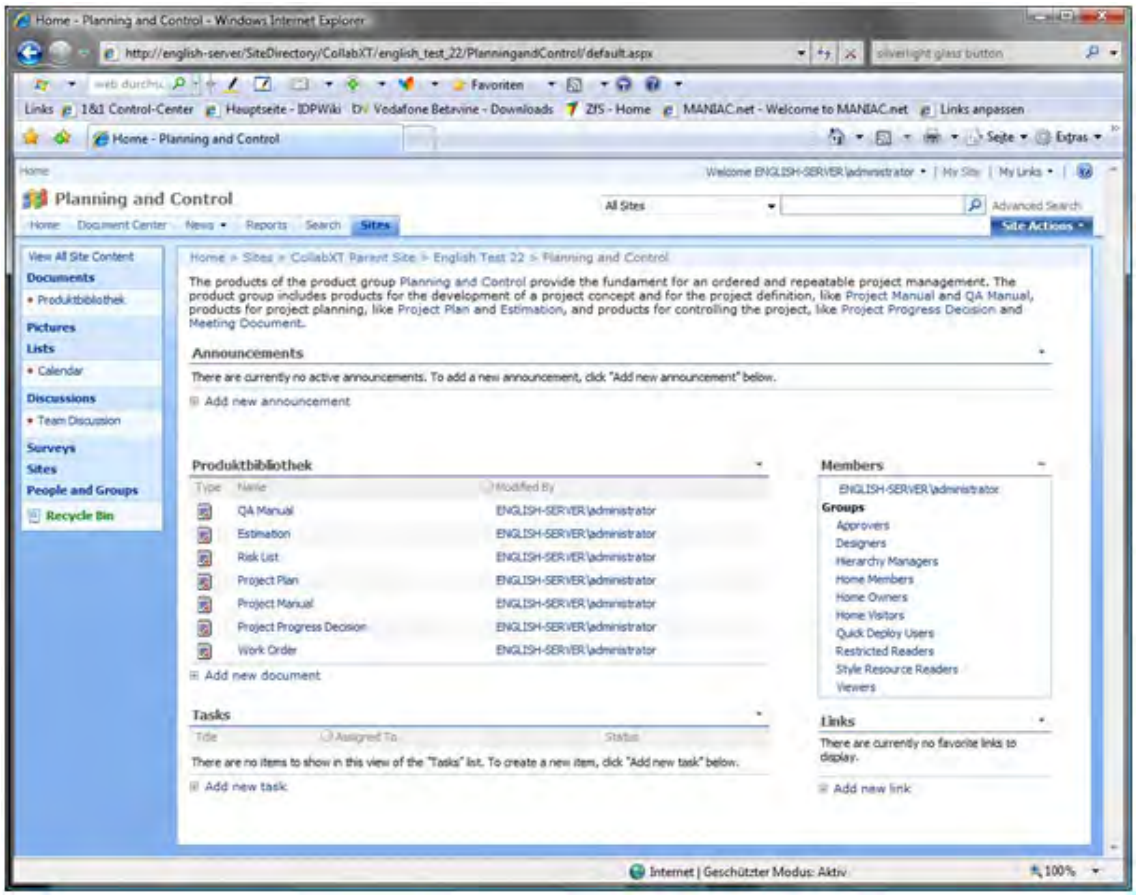


Abbildung 3.3.: V-Modell XT im SharePoint-Portal

Produkte und Entscheidungspunkte. Geht das Produkt in einen Entscheidungspunkt ein, so hat das Produkt im Portal einen Verweis darauf. Damit wären z. B. entsprechende, in SharePoint 2007 hinterlegte Workflows in der Lage, über den Bearbeitungszustand des Produkts die Erreichung eines Entscheidungspunktes automatisch zu überprüfen. Siehe dazu auch Abschnitt 3.2.4. Erforderlich ist es hingegen, die dafür benötigten Datenstrukturen für SharePoint bekannt zumachen. Daher wird folgende Ergänzung in der Abbildung vorgenommen: *Produkt.Bearbeitungszustand ⇒ vom Generator hinzugefügte Spalte in Dokumentbibliothek*. Die hinzugefügte Spalte erweiterte hier als die Basisdatenstruktur der SharePoint-Listen (vgl. Abbildung 3.1).

Produktabhängigkeiten (PAH). Im V-Modell-Metamodell können Produkte durch Produktabhängigkeiten zu zahlreichen anderen Produkten in Beziehung stehen. In der Zielplattform werden davon nur *Erzeugende Produktabhängigkeiten* berücksichtigt. Diese haben eine entscheidende Bedeutung bei der Instanziierung von Produktexemplaren. So kann das Anlegen eines neuen Produktexemplars die automatische Erstellung weiterer Produkte, die vom ursprünglich angelegten Produkt abhängen, nach sich ziehen. Tailoringabhängigkeiten werden von der Abbildung ignoriert. Inhaltliche und strukturelle Abhängigkeiten werden vom Projektassistenten in die Produktvorlagen generiert. Diese stellen für SharePoint 2007 allerdings eine „Black Box" dar. Die Abhängigkeiten sind für die Bearbeiter eines Produkts in der Zielplattform nicht verloren aber sie sind nicht explizit darin modelliert.

Auf inhaltlicher Ebene lassen sich im V-Modell grob drei Arten von Produkten unterscheiden:

- Modellierungs-/Entwicklungsbezogene Produkttypen
- Managementprodukttypen
- Sonstiges

Modellierungs-/Entwicklungsbezogene Produkttypen haben im V-Modell typischerweise keine Produktvorlage. Aufgrund ihrer Natur (diese Produkte bestehen oft zu großen Teilen aus Diagram-

men und/oder Quellcode) können sie in der Dokumentenverwaltung von SharePoint nicht angemessen verwaltet werden. Zum Management dieser *Auftragnehmer-Produkte* ist die mit CollabXT-TFS beschriebene Abbildung auf den Microsoft Team Foundation Server vorgesehen (siehe Kapitel 4). Produkte ohne Produktvorlagenattribut im V-Modell schließen wir daher im Generierungsprozess aus.

Die zweite Gruppe lässt sich bezeichnen als *Managementprodukte*. Hierzu zählen wir alles, was in den Bereichen Berichtswesen, Aufgaben- oder Risikomanagement, oder auch Anforderungsmanagement etc. zu finden ist. Diese Produkte finden in klassischen (Word-)Dokumenten, Listen und an geeigneter Stelle in Formularen ihre geeignete Repräsentation. Für diese Art von Produkten eignet sich die Dokumentenverwaltung von SharePoint, weshalb der Fokus von CollabXT-SP auf Auftraggeber- und Prozesseinführungsprojekten, die zumeist keine Entwicklungsanteile haben und somit i. d. R. nur Managementprodukte erzeugen, liegt. In diesen Projekten ist der Anteil an Managementprodukten höher. Auf der technischen Ebene bedeutet dies, dass nur V-Modell-Produkte mit einer Produktvorlage nach SharePoint 2007 abgebildet werden.

Produktzustand. Produkte mit Vorlage werden inklusive des vom V-Modell vorgesehenen Produktzustandsmodells abgebildet. Ein in SharePoint hinterlegter Workflow sorgt für die Einhaltung und die Konsistenz des Produktzustands. Vom Projektassistenten wird dabei stets nur die Produktvorlage bereitgestellt. Diese ist nicht zu verwechseln mit der konkreten Instanz[3] eines Produkts. Insbesondere kann ein Produkt auch mehrfach instanziiert sein (beispielsweise regelmäßig zu erstellende Berichte). Abbildung 3.4 zeigt die Auswertung des Produktzustands durch das SharePoint-Portal. Gemäß [KMR06] stehen mit den Daten des V-Modells genügend Informationen bereit, um z. B. umfassendere Projektleitstände zu betreiben.

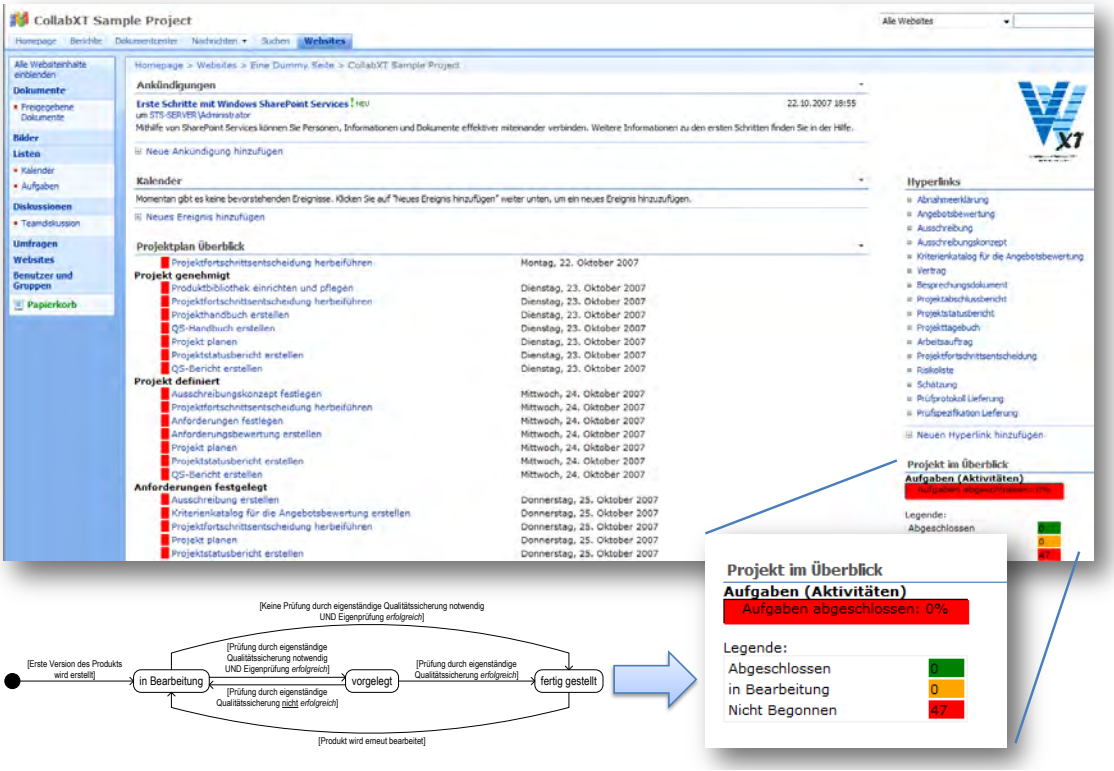


Abbildung 3.4.: V-Modell XT Produktzustandsmodell und seine Abbildung im SharePoint-Portal

3 Dieser Betrachtung liegt die Art der Produkterzeugung des V-Modell XT zugrunde. das V-Modell XT unterscheidet *initiale*, *abhängig erzeugte* und *externe* Produkte. Der Projektassistent erzeugt aber für alle Produkttypen einheitlich Vorlagen. Der Unterschied ist dabei der folgende: initiale Produkte werden nur *einmal* erzeugt und dann kontinuierlich weiter gepflegt. Abhängige Produkte können beliebig oft instanziiert werden. In CollabXT setzen wir zurzeit direkt auf dem Projektassistenten auf und unterscheiden daher nicht zwischen den Instanziierungsmodi.

3.2.2. Abbildung: Aktivitäten

Zur Darstellung von Aktivitäten wird die von SharePoint standardmäßig zur Verfügung gestellte Aufgabenliste verwendet und erweitert. Damit können die bereits eingebauten Mechanismen zur Auswertung weiterverwendet werden. Auch hier verweisen wir auf Abbildung 3.4, in der wir neben der Ampel-Darstellung, basierend auf dem Produktzustandsmodell, auch die Stati der Aufgaben erfassen. Das Zustandsmodell für Produkte wurde auf Aktivitäten überführt, womit nun auch Aufgaben der SharePoint-Tasklisten einen Indikator besitzen.

Analog zu Produkten sind Aktivitäten im V-Modell-Metamodell in Gruppen zusammengefasst. Um allerdings die Auswertungsmechanismen von SharePoint voll nutzen zu können, müssen alle Aufgaben in einer SharePoint Aufgabenliste (vgl. Abschnitt 3.1) gehalten werden. Davon machen wir Gebrauch, weshalb Aktivitätsgruppen nicht in der Zielplattform abgebildet werden. Jedem Thema eines Produkts ist mindestens eine Teilaktivität zugeordnet, die sich auf die Aktivität beziehen, die für die Erstellung des Produkts zuständig ist (vgl. Abschnitt 2.2.3). Wie auch die Themen eines Produkts finden diese sich im beschreibenden Text der Aktivität wieder.

Abbildungsoption

Obwohl wir bereits Aktivitäten auf Aufgaben von Sharepoint abbilden, besteht auch die Möglichkeit, Teilaktivitäten auf Aufgaben abzubilden. Für den Fall, dass ein fein granulares Projektmanagement erforderlich ist, z. B. bei der Einbindung eines *Project Server*, wird dies auch eine Anforderung sein. Die Generierung ist durch das Metamodell nicht wesentlich aufwändiger. Zurzeit wird sie jedoch von uns nicht umgesetzt.

Für einige Aktivitäten sieht das V-Modell XT eine Ablaufdarstellung vor. In dieser wird die Sequenzierung der Teilaktivitäten festgelegt. Wie die Teilaktivitäten selbst wird auch diese nicht explizit in der Zielplattform dargestellt. An dieser Stelle müssen wir noch einen weiteren Punkt deutlich machen: In der aktuellen Version des Generators werden ausschließlich Texte übernommen. Bildinformationen und Grafiken werden ignoriert. Dies ist zurzeit für alle integrierten Inhalte so.

3.2.3. Abbildung: Rollen

Die Abbildung von Rollen aus dem V-Modell XT ist in der vorliegenden prototypischen Umsetzung nicht bis zu SharePoint 2007 durchgezogen[4]. Mögliche Bausteine zur Umsetzung des Rollenkonzepts sind:

- CrossSiteGroups
- Roles (nicht zu verwechseln mit dem ansonsten in diesem Papier verwendeten Rollenbegriff)
- das Konzept der Target Audience (nur bedingt)

3.2.4. Abbildung: Entscheidungspunkte

Der Ablauf eines V-Modell XT Projekts wird durch Entscheidungspunkte strukturiert. Zur Erreichung eines Entscheidungspunkts müssen sich jeweils gewisse Produktexemplare im Zustand fertig gestellt befinden. Diese Abhängigkeit wird von der Abbildung in der Zielplattform dargestellt (siehe Abbildung 3.4). Zusammen mit dem Zustandsmodell für V-Modell XT Produkte ist die Zielplattform die vom V-Modell XT geforderten Kriterien zur Erreichung eines Entscheidungspunktes zu überwachen und deren Einhaltung zu erzwingen.

Entscheidungspunkte werden im Projektassistenten durch die Wahl einer Projektdurchführungsstrategie sequenziert. Da die Wahl einer Projektdurchführungsstrategie bereits im Projektassistenten erfolgt, werden Projektdurchführungsstrategien in der Zielplattform nicht mehr dargestellt. Auch die zur Sequenzierung herangezogenen Ablaufbausteine sind während der Projektdurchführung nicht mehr von Relevanz.

4 Grundsätzliche Überlegungen zur Abbildung von V-Modell-Rollen sind in Kapitel 2.4.1 zu finden.

3.2.5. Abbildung: Projektplan (Meileinsteine)

Mit den Planungsinformationen, die durch den Export aus dem Projektassistenten zur Verfügung stehen, können initiale Projektpläne erstellt werden. Für den aktuellen Stand des Generators nutzen wir dieses Fähigkeiten jedoch noch nicht aus. Optionen, die hier denkbar sind, sind z. B. die Bereitstellung eines *echten* Projektplans, eine direkte Ansteuerung einer SharePoint-integrierten Planungskomponente oder die Verwendung/Ansteuerung des *Project Servers*. Eine weiter fortgeschrittene Interpretation der Planungsanteile des V-Modell-Exports findet sich in CollabXT-TFS (Kapitel 4.3.4).

3.2.6. Abbildung: Dokumentation

Die Dokumentation der Prozesselemente steht im V-Modell XT nicht neben der Prozessdefinition im Metamodell, sondern ist Teil derselben. In der Zielumgebung findet sich diese an unterschiedlichen Stellen wieder: Ein Großteil der Dokumentation wird vom Projektassistenten in die Produktvorlagen generiert. Sie dient hier als Hilfe zur tatsächlichen Ausgestaltung des Produkts.

Für viele Elemente des V-Modells gibt es einen das Element beschreibenden Text *Sinn und Zweck*. Für Produktgruppen wird dieser z. B. in die Webseite der Produktgruppe mit generiert (vgl. Abbildung 3.3). Ähnlich verhält es sich mit dem beschreibenden Text zu einer Aktivität. Auch dieser wird in die jeweiligen Instanzen der Aktivitäten generiert. Ein Beispiel dafür findet sich in Abbildung 3.5.

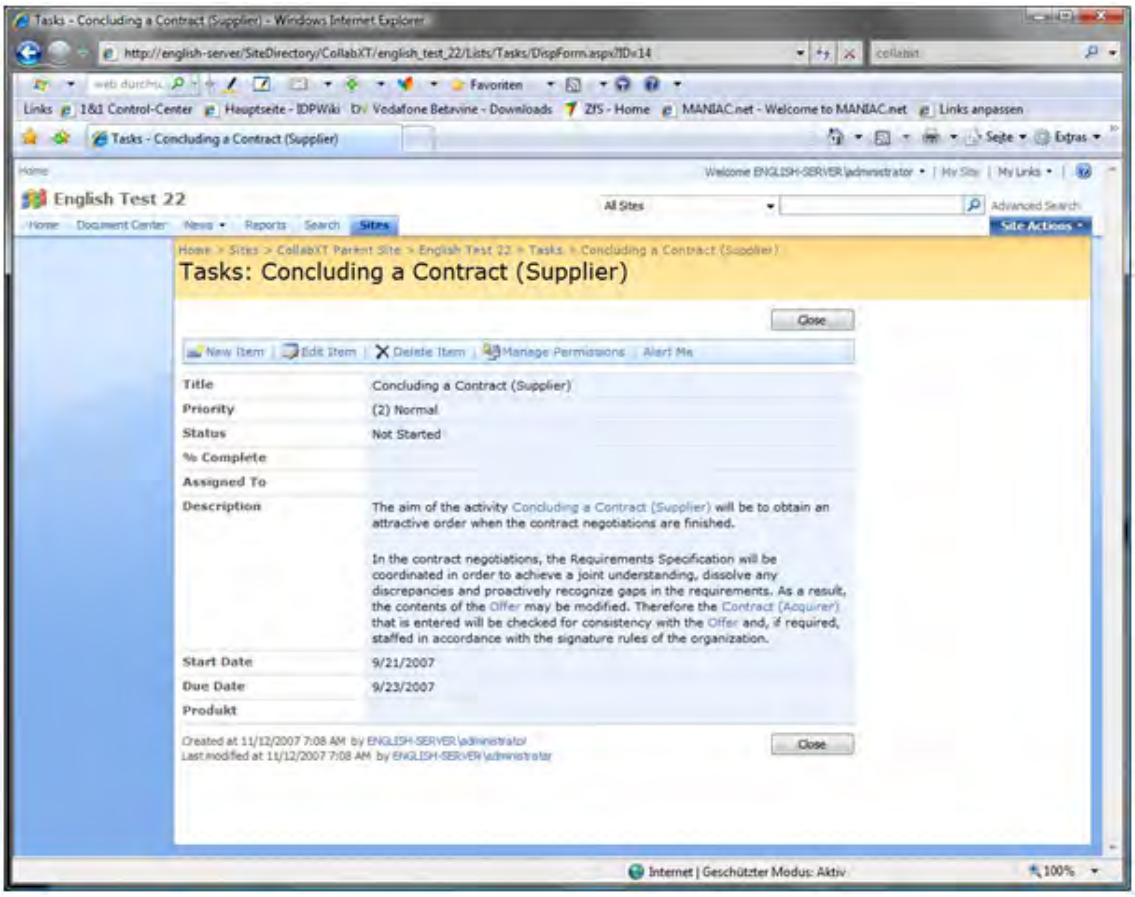


Abbildung 3.5.: V-Modell XT im SharePoint-Portal

Auch bei der Dokumentation gibt es wieder Teile, die nicht in der Zielplattform dargestellt werden. Hier seien die Methodenreferenzen und Werkzeugreferenzen erwähnt.

3.3. Werkzeug

Für CollabXT-SP wurde über die Konzeption von Prozess-/Toolintegrationsverfahren hinaus eine Werkzeugentwicklung durchgeführt. Die Anforderungen an das Werkzeug beschreiben ein generatorbasiertes System, das die Kette der Referenzwerkzeuge des V-Modells weiterführt und eine weitere Stufe so einführt (Abschnitt 2.1), dass Ergebnisse des Tailorings direkt in andere Umgebungen zu überführen und zu integrieren.

3.3.1. Konzept und Architektur des Werkzeugs

Für SharePoint ist dafür ein generatorbasiertes System gewählt worden, das das auf den Eingaben des V-Modells basiert und das Portal vollständig generiert.

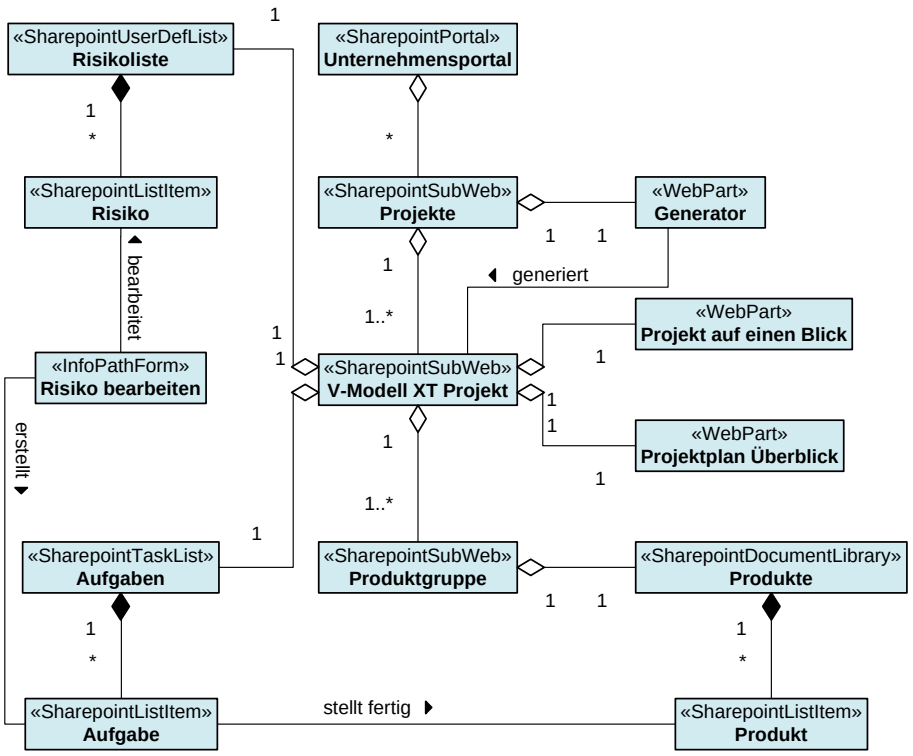


Abbildung 3.6.: Konzeptmodell für die V-Modell-Integration in ein SharePoint-Portal

Abbildung 3.6 zeigt das dafür verwendete Analysemodell, das auf der Architekturskizze von Abbildung 3.1 basiert. Das Modell berücksichtigt neben der Struktur des Portals auch die Elemente, die durch den Generator verwendet, bzw. neu ausprogrammiert werden. Ein neu ausprogrammiertes Element haben wir in [KK07] gezeigt: Die Abbildung des Risikomanagement-Prozesses auf eine *InfoPath*-basierte, Workflow-unterstützte Formularstruktur. Ansonsten sind in diesem Modell noch die neuen Webparts für die Aufgabenliste und den Projektüberblick zu sehen. Im unteren Teil der Abbildung sind die Elemente gruppiert, die durch den Generator aus einem gegebenen V-Modell extrahiert und in SharePoint-Strukturen überführt werden. In Abbildung 3.7 verfeinern wir diesen Part. Sie zeigt die SharePoint-relevanten Größen, die der Generator erfasst. Im Wesentlichen sind das die für die Abbildung verwendeten Elemente, die wir in Abschnitt 3.2 zusammengestellt haben. Damit unterstützt der Generator nur eine Teilmenge der Objekte des V-Modells.

Adressaten. CollabXT-SP ist von uns für folgende Anwendungsgebiete konzipiert worden. In Projekten, in denen keine Entwicklungsanteile lokalisiert sind, die ein ausgefeiltes Quellcode- oder allgemein Konfigurationsmanagement benötigen, sind dennoch Dokumente und Aufgaben

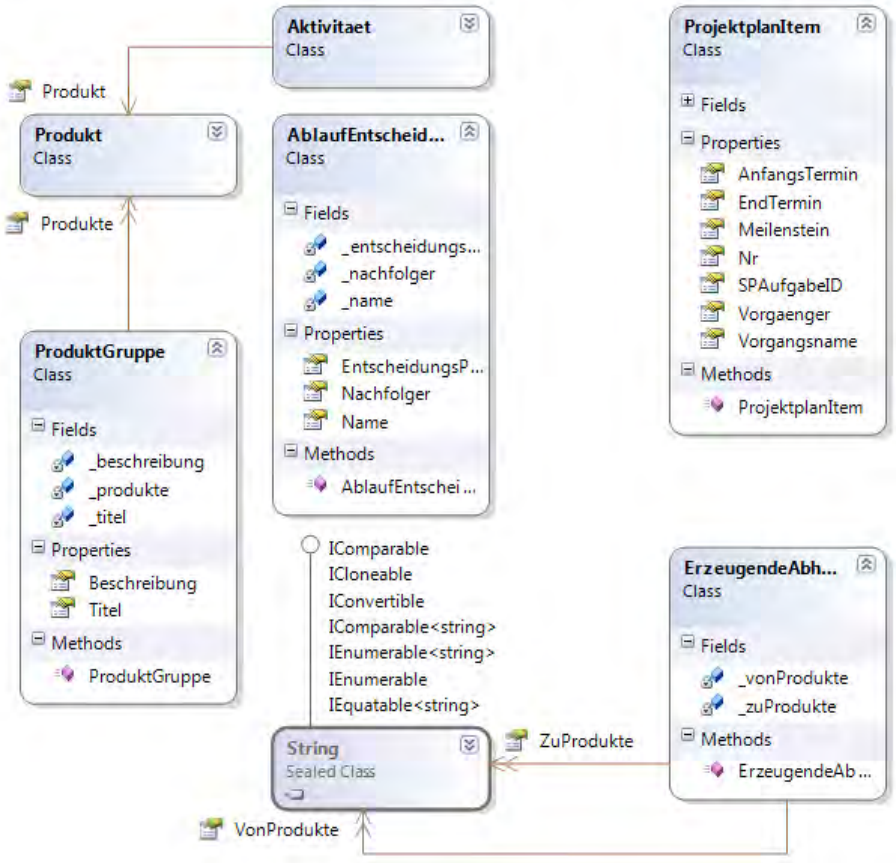


Abbildung 3.7.: Objektmodell für den SharePoint-Portal-Generator

zu verwalten. Typische Szenarios, in denen wir uns hier bewegen sind Vergabe- und Beschaffungsprojekte der öffentlichen Hand (Projekttyp: *Systementwicklungsprojekt (AG)*) oder Prozesseinführungsprojekte (Projekttyp: *Einführung und Pflege eines organisationsspezifischen Vorgehensmodells*). Auch bislang nicht adäquat durch das V-Modell unterstützte Projekttypen, wie z. B. Dienstleistungs- oder Beratungsprojekte können von CollabXT relativ aufwandsneutral erfasst werden. Der Generator ist konzipiert, dass er keinen Projekttypen explizit ausschließt. Das heißt, dass der Generator alle Projekttypen des aktuellen V-Modells (in allen verfügbaren Sprachversionen) und alle V-Modell-Anpassungen unterstützt, die auf dem V-Modell-Metamodell aufbauen (sofern verfügbar also z. B. auch das V-Modell Witt [AEH+07]).

3.3.2. Installation und Randbedingungen

CollabXT-SP ist so konzipiert, dass keine Eingriffe im System erforderlich sind. Es sind lediglich einmalig serverseitige Konfigurationen[5] vorzunehmen, um die Sicherheitsstufen für die verwendeten Web Parts vorzunehmen, sowie das SharePoint-Seitentemplate einzuspielen. Folgende Software wird benötigt:

- Die V-Modell XT Werkzeuge (erhältlich via `http://www.v-modell-xt.de`) erfordern eine Java Laufzeitumgebung (1.5 oder besser).
- Für die CollabXT-SP-Werkzeuge ist (erhältlich via `http://www.codeplex.com/collabxt`) eine .NET Laufzeitumgebung (2.0 oder besser) erforderlich.

Die Software unterstützt sowohl die englische als auch die deutsche Variante des V-Modells auf deutschen und englischen SharePoint-Installationen. Hierfür sind für den SharePoint-Server jeweils nur (sprachlich) angepasste Seitentemplates bereitzustellen.

5 Eine detaillierte Anleitung und Beschreibung findet sich auf CodePlex: `http://www.codeplex.com/collabxt`.

3.3.3. Beschreibung und Anwendung des Generators

Um mit den Werkzeugen von CollabXT-SP zu arbeiten, wird ein projektspezifisches V-Modell XT benötigt. Das Tailoring muss mit dem Projektassistenten ausgeführt werden. Der Generator verfügt über eine einfache, an einem Assistenten angelehnte, Oberfläche (Abbildung 3.8).

Abbildung 3.8.: Start des SharePoint-Portal-Generators

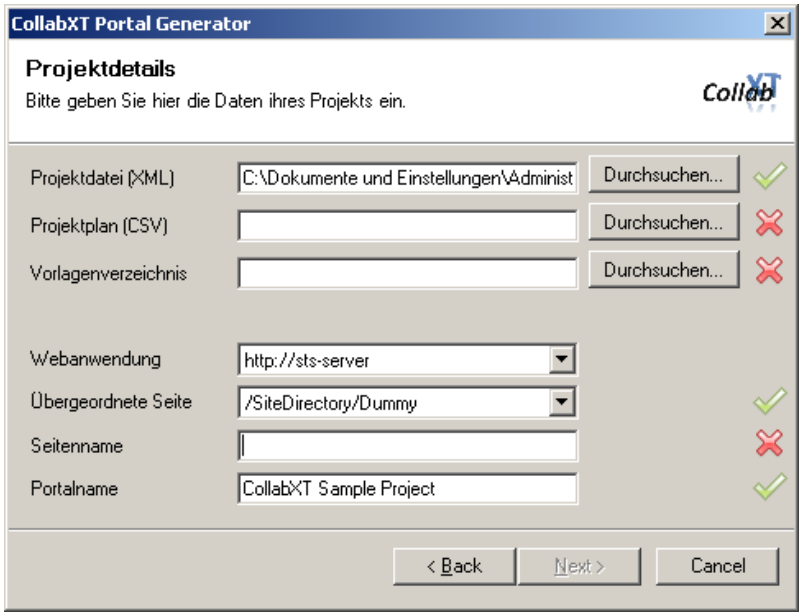


Abbildung 3.9.: Eingabe der Daten für den SharePoint-Portal-Generator

Der Generator erwartet vom Anwender die Eingabe folgender Informationen (Abbildung 3.9):

Projektdatei (XML) Damit ist das projektspezifische V-Modell gemeint, das der Generator dazu verwendet, um Beschreibungstexte und Strukturen zu extrahieren und für den Export nach SharePoint zu ermitteln.

Projektplan (CSV) Der CollabXT-SP-Generator baut anders als CollabXT-TFS nicht auf der Projektdatei des Projektassistenten auf, sondern verarbeitet den im CSV-Format exportierten

initialen Projektplan[6].

Vorlagenverzeichnis Im Vorlagenverzeichnis erwartet der Assistent alle exportierten Produktvorlagen gemäß der Exportstruktur des Projektassistenten.

Der Generator macht in der aktuellen Fassung keine weiteren Annahmen über untergeordnete Strukturen und Verzeichnisse. Er fragt lediglich bestimmte Dateien ab, die ihm direkt zu übergeben sind. Dies entkoppelt den Generator weitgehend von Vorgaben und sonstigen Exportstrukturen. Die Eingabe der Daten vom Anwender wird geprüft. Über ein visuelles Feedback wird dem Anwender mitgeteilt, welche Eingaben noch fehlen (Abbildungen 3.9 und 3.10).

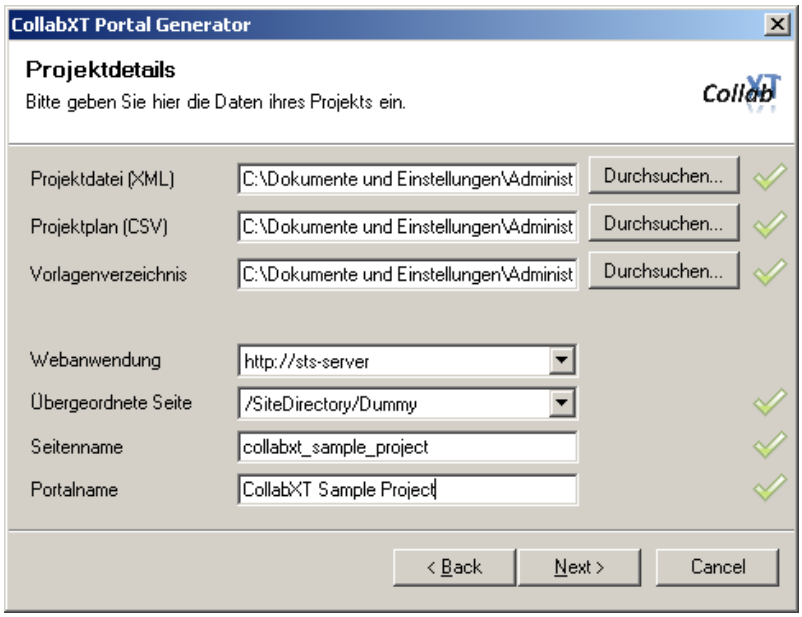


Abbildung 3.10.: Vollständiger Datensatz für den SharePoint-Portal-Generator

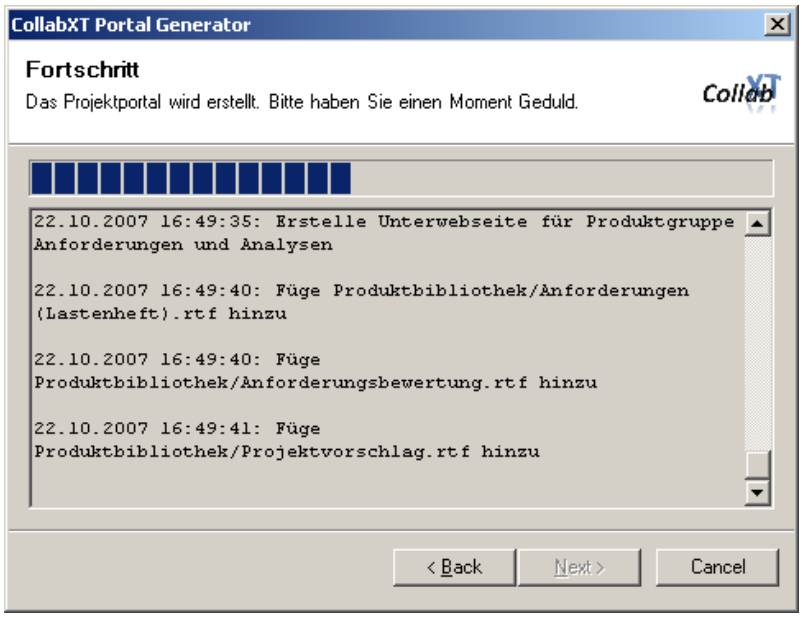


Abbildung 3.11.: Laufender Generierungsprozess mit Statusanzeigen

6 Das gibt dem Projektleiter noch die Chance, außerhalb der V-Modell-Werkzeuge notwendige Anpassungen am Plan zu tätigen.

Ein vollständiger Datensatz (Abbildung 3.10) für den Generator besteht zusätzlich zu den Eingaben aus dem V-Modell noch aus Informationen zum Zielserver:

Webanwendung Das ist der Webserver, auf dem das Portal gehostet werden soll.

Übergeordnete Seite Unter dieser Seite werden die neuen Projektwebseiten erstellt und in die Seitenhierarchie des Portals integriert.

Seitenname Das ist der physische Name der Webseite (Adresse).

Portalname Das ist der Name des generierten Portals.

Mit diesen Informationen kann der Generierungsprozess gestartet werden. In der aktuellen Version des Systems empfiehlt es sich jedoch, dieses direkt auf dem Server und mit Administrator-Rechten zu tun. Dies ist keine konzeptionelle Frage, sondern ein Implementierungsaspekt, der gerade in der Überprüfung ist. Eine vollständig serverseitige Lösung (Tailoring, Generierung etc.) ist zurzeit in der Konzeption und hat diese Beschränkungen nicht mehr.

Ergebnisse der Generierung

In Abbildung 3.11 ist der laufende Generierungsprozess zu sehen. Dieser erstellt auch ein Log, das in möglichen Fehlerfällen ausgewertet werden kann. Als Ergebnisse erzeugt der Generator eine vollständige SharePoint Webseite mit Aufgabenlisten (basierend auf dem initialen Projektplan) und Dokumentbibliotheken (basierend auf der exportierten Produktbibliothek des V-Modells). Die Abbildungen 3.3, 3.4 und 3.5. Letztere zeigt insbesondere auch noch einmal die Integration der Prozessdokumentation in die entsprechenden Portalinhalte.

4. CollabXT – Team Foundation Server

In diesem Kapitel stellen wir die CollabXT-Implementierung für den *Visual Studio Team Foundation Server* (TFS) vor. Diese wurde im Rahmen eines Verbundprojekts zwischen der TU München, Microsoft Deutschland und der Zühlke Engineering GmbH entwickelt. Die Generator-Software ist als Open Source unter `http://www.codeplex.com/VModellXTTFS` verfügbar. Wir stellen nun die Identifikation der benötigen V-Modell-Anteile und deren grundlegende Modellierung im Kontext des TFS vor.

4.1. Team Foundation Server – Einführung und Überblick

Der Visual Studio Team Foundation Server ist die Backend-Komponente des *Visual Studio Team System* [GP06]. Er stellt der Microsoft-Entwicklungsplattform Dokumentenablagen, Quellcodeverwaltung und Reporting zur Verfügung. TFS ist im Wesentlichen eine generische Prozessplattform, die verschiedene Prozesse unterstützt. Auf den Microsoft-Webseiten sind z. B. schon Scrum oder RUP-Derivate zu finden. Um die Vielfalt möglicher Prozesse zu unterstützen, definiert TFS ein *Vorlagenformat* – so genannte Prozess-Templates – in dem Prozesse im TFS beschrieben und implementiert werden müssen.

4.1.1. Architektur

Team Foundation Server ist eine integrierte Serverlösung, die auf der Basis von Windows Server 2003 verschiedene Softwarekomponenten zusammenfasst und somit eine Plattform für serverseitige, prozessbasierte Projektunterstützung realisiert.

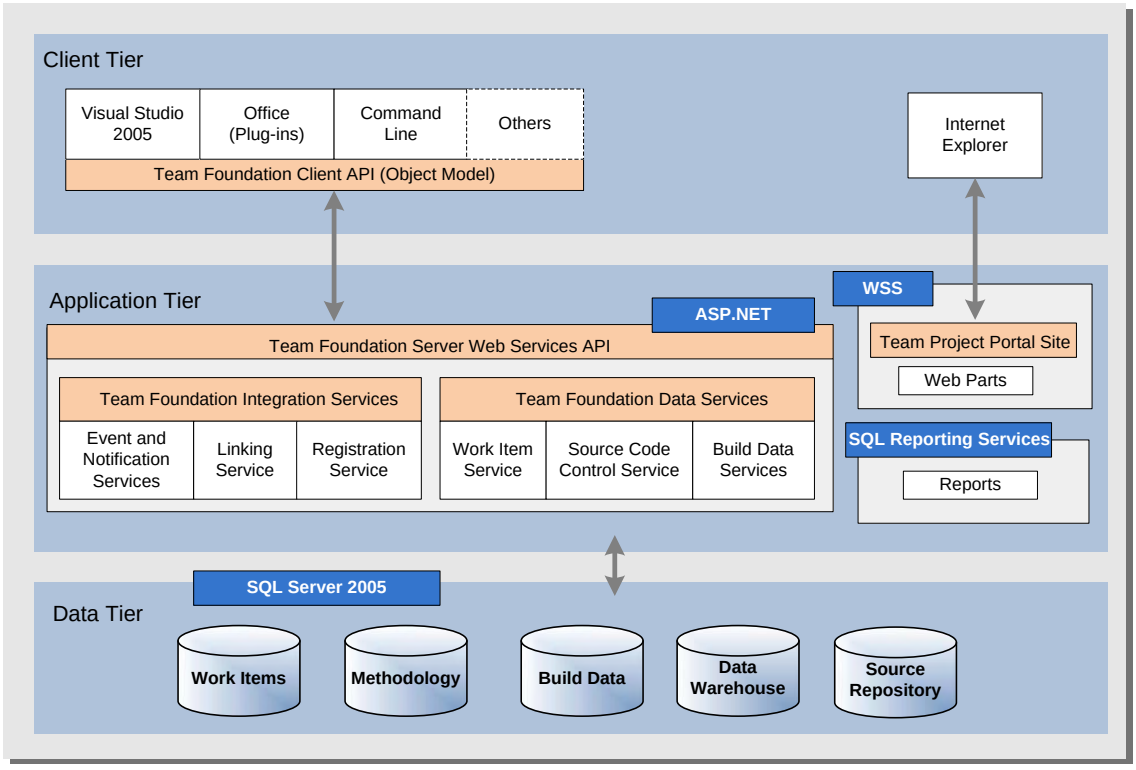


Abbildung 4.1.: Technische Architektur des Visual Studio Team Foundation Server

Abbildung 4.1 zeigt die technische Architektur der TFS-Umgebung nach [Mic07]. Über der Datenschicht, bestehend aus verschiedenen Datenbank-Servern, liegt die eigentliche TFS-Applikation. TFS ist im Wesentlichen eine Webanwendung, die auf dieser Schicht aus drei Teilen besteht:

- Der Anbindung an die SQL-Reporting Services. Diese stehen den anderen Clients über entsprechende Schnittstellen auch zur Verfügung, sodass z. B. im Visual Studio die Statistiken ebenso zu sehen sind, wie im Webportal.
- Das SharePoint-basierte Webportal ist der zweite Teil der TFS-Anwendung. SharePoint dient dem TFS dazu, Dokumente und Artefakte abzulegen, die nicht Code sind (dafür steht eine eigene Codeverwaltung zur Verfügung).
- Der dritte Applikationsteil ist unter dem *Team Foundation Server Web Services API* zusammengefasst. Hier sind die Komponenten und Schnittstellen für die Prozesse und die Integration von Drittsystemen zu finden. Die *Data Services* umfassen hierbei das *Work Item* System, das Build- und das Quellcodeverwaltungssystem. Die *Integration Services* stellen das API für die Konnektoren z. B. für Office oder den Visual Studio Team Explorer bereit.

4.1.2. Process Templates

Daten- und Integrationsdienste sind hoch integriert, was sich auf die Komplexität der Plattform auswirkt. Um die Prozessführung, bzw. die Vorbereitung der Prozessunterstützung dennoch beherrschbar zu gestalten, definiert TFS ein Vorlagenformat für die unterstützten Prozesse; so genannte *Process Templates*. Abbildung 4.2 zeigt das XML-Schema eines Process Templates. Zentrale

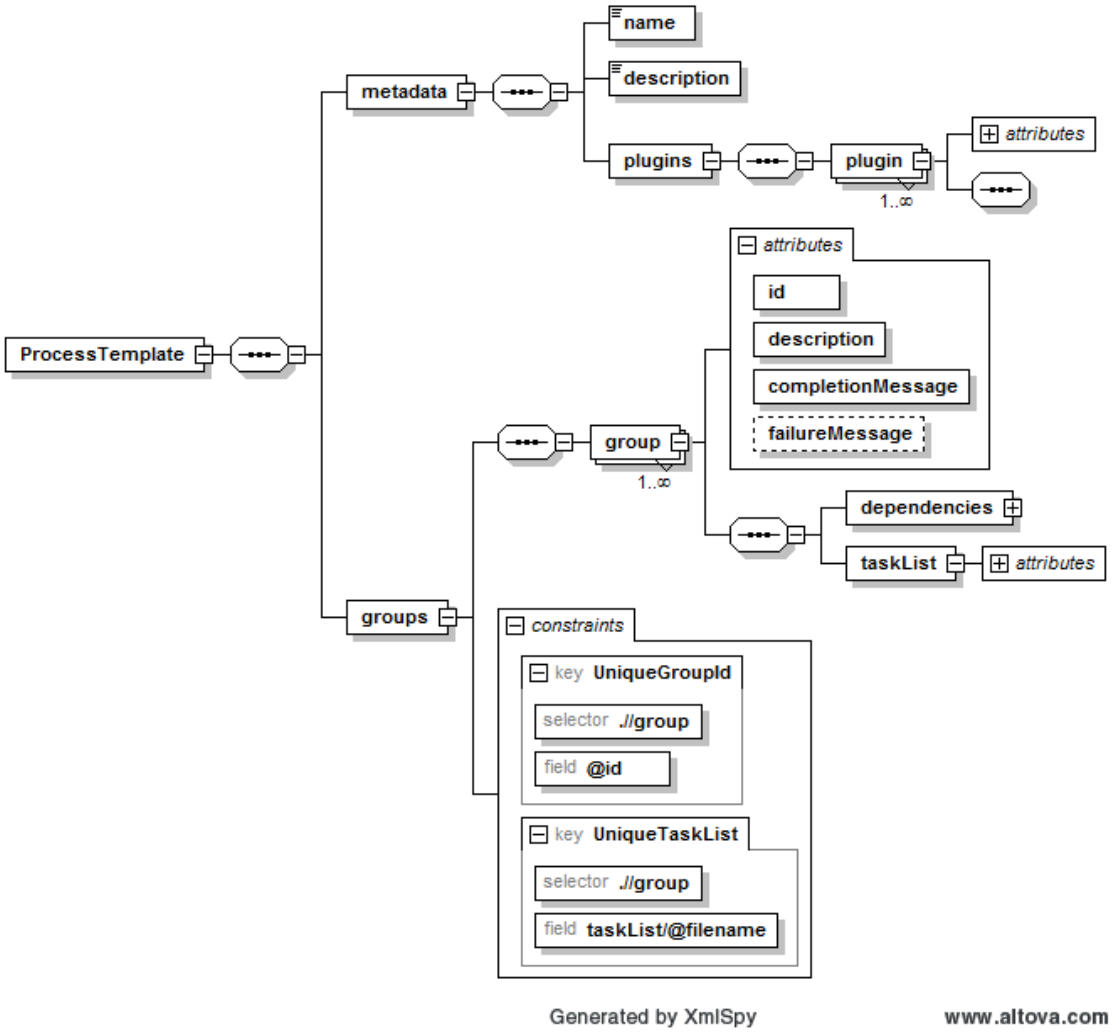


Abbildung 4.2.: XML-Schema eines Process Templates

Elemente des Schemas sind die Metadaten (`metadata`) und die Gruppen (`groups`). Erstere enthalten neben einigen Beschreibungstexten Referenzen auf Plug-Ins, die das Instanziieren des Process Template gestalten. So ist z. B. die Ansteuerung der Quellcodeverwaltung als Plug-in realisiert. In einem Process Template werden die Plug-Ins entweder aus einem Standardpool des TFS bezogen. Alternativ können hier auch eigene Plug-Ins verwendet werden.

Das zweite zentrale Element stellen die Gruppen dar. Diese referenzieren die Inhalte des Templates und seine Strukturen. Abbildung 4.3 zeigt ein Beispiel eines konkreten Process Templates, das die Gruppen: *Classification*, *Groups*, *Portal*, *Reporting*, *WorkItemTracking* sowie *VersionControl* enthält. In diesen Gruppen sind jeweils Referenzen enthalten, auf inhaltlich Komponenten des Process Templates in den jeweiligen Unterverzeichnissen.

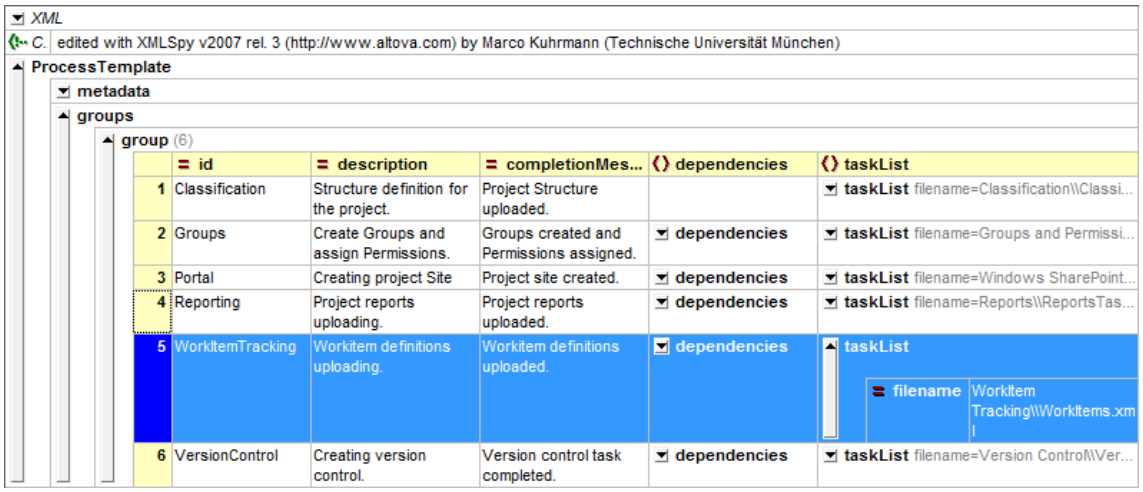


Abbildung 4.3.: Gefülltes Process Template mit zugeordneten Gruppen

Das zentrale Element des Modells ist das Process Template, das als *ProcessTemplate.xml* (Abbildung 4.3) vorliegt und auf einem XML-Schema (Abbildung 4.2) basiert. Die entsprechenden Elemente sind in Form von Complex Types hinterlegt und enthalten jeweils Referenzen auf die Elemente in den Unterverzeichnissen des Templates. Wichtigste Größe hierbei sind die *Work Item Type* Definitionen. Auf Work Items gehen wir im Abschnitt 4.2 detailliert ein. Die verbleibenden Elemente der einzelnen Gruppen beschreiben wir nun:

Classification dient der Beschreibung der Projektorganisationsstruktur. Strukturierungselemente sind *Bereiche* (Areas) und *Iterationen* (Iterations). Die Classification dient unter anderem der Sicherheitseinstufung von verschiedenen Projektbereichen. Hier können rollebezogen Rechte vergeben werden. Iterationen kennzeichnen definierte Entwicklungsabschnitte. Work Items sind sowohl Bereichen als auch Iterationen zugeordnet. Darüber können beispielsweise *Abfragen* (Queries) ausgeführt werden (z. B. Alle Aufgeben in Iteration 2).

Reports sind SQL-Server-basierte Datenbankabfragen über dem TFS zugrunde liegenden Data Warehouse. Reports geben hier vielfältige Möglichkeiten der komplexeren Auswertung eines Projekts, wie z. B. einer Trendanalyse hinsichtlich Aufgaben, Anforderungen etc. oder Build- und Teststatistiken (vgl. auch Abschnitt 4.3.2).

Portal enthält alle Dokumente und Daten für das Webportal. SharePoint dient im TFS zum Ablegen der Dokumentbibliothek[1]. Dies können z. B. Anforderungsdokumente sein. Weiterhin stellt TFS über das SharePoint-Portal eine integrierte *Prozessdokumentation* in Form einer HTML-basierten Dokumentation bereit (Ordner: Process Guidance). Dies ist z. B. die nach HTML exportierte Sicht auf ein projektspezifisches V-Modell.

Groups dient der Festlegung von Rechten und Rollen. Hier werden Benutzergruppen definiert, die entsprechend zu bestimmten Bereichten und Aufgaben im TFS Schreib- und/oder Leserechte zugewiesen bekommen.

1 Dokumente legt TFS nicht im Versionskontrollsystem Source Safe ab, sondern nutzt dafür die Dokumentbibliotheken des SharePoint. Codes liegen jedoch im Source Safe-System oder alternativen SCMs.

Process Templates anpassen. Im Umfang des Visual Studio SDK ist der *Process Template Editor* (PTE) enthalten. Dieser unterstützt die Anpassung eines gegebenen Process Templates und kann auch eingeschränkt Prüfungen vornehmen. Die Anpassung eines Process Templates erfolgt nach dem Muster aus Abbildung 2.8.

4.2. Work Item Design

In diesem Abschnitt stellen wir das Work Item Design für den TFS vor. Mit den identifizierten Work Item Typen decken wir einen **grundlegenden** Umfang an zu überwachenden Anteilen des V-Modells ab. Gemäß unseren Überlegungen aus Kapitel 2 und insbesondere Kapitel 2.3, konzentrieren wir uns schwerpunktmäßig auf generische V-Modell-Prozesse. Als Grundlage stützen wir uns zuerst auf die zentralen Managementfähigkeiten des V-Modells ab, die im Vorgehensbaustein *Projektmanagement* (Abbildung 4.4) definiert sind. Wie wir schon ausgeführt haben, besteht bei der Abbildung des V-Modells auf eine konkrete Werkzeuginfrastrukur *kein* Dokumentenzwang. In Abbildung 4.4 haben wir daher diejenigen Produkttypen hervorgehoben, die wir nicht als eigenständiges Dokument verstehen. Diese Typen beziehen sich unter anderem auch auf die abzubildenden V-Modell-Prozesse aus Kapitel 2.3.

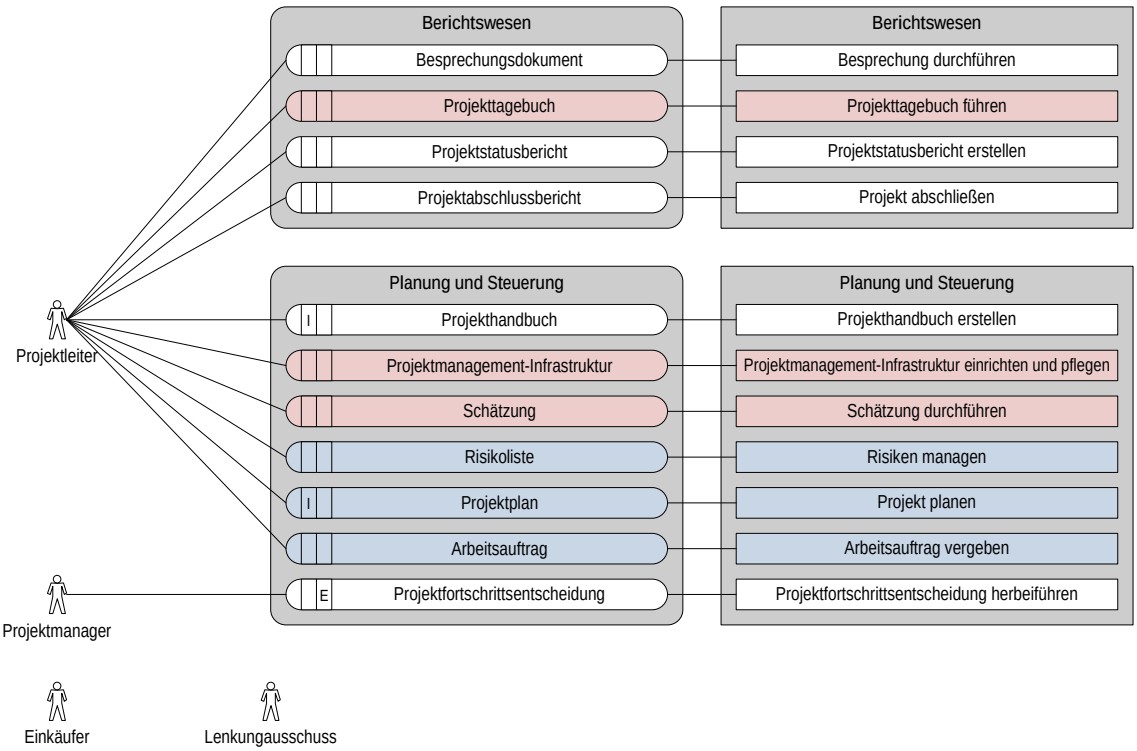


Abbildung 4.4.: Vorgehensbaustein Projektmanagement

Die Produkttypen *Risikoliste*, *Projektplan* und *Arbeitsauuftrag* werden wir als eigenständige Work Items modellieren. Hierbei kommt der Projektplanung noch eine besondere Rolle zu, da TFS eine Integration in MS Project anbietet. Daher wird hier eine Verbindung zwischen den Werkzeugen hergestellt, sodass ein initialer Projektplan aus dem V-Modell direkt in Project verarbeitet werden kann. Analog verfahren wir mit den anderen hervorgehobenen Produkttypen. Diese integrieren wir entweder in andere Work Item Typen, wie z. B. die Schätzung, die in vielen Work Items als Planungsgröße hinterlegt wird. So keine Integration in andere Work Item Typen erfolgt, werden die hervorgehobenen Produktypen direkt auf Komponenten der TFS-Plattform abgebildet.

4.2.1. Work Item Typen – Grundlagen

Bevor wir uns mit der inhaltlichen Modellierung der konkreten Work Item Typen für das V-Modell XT befassen, gehen wir noch auf die grundlegenden Strukturen der Work Items ein.

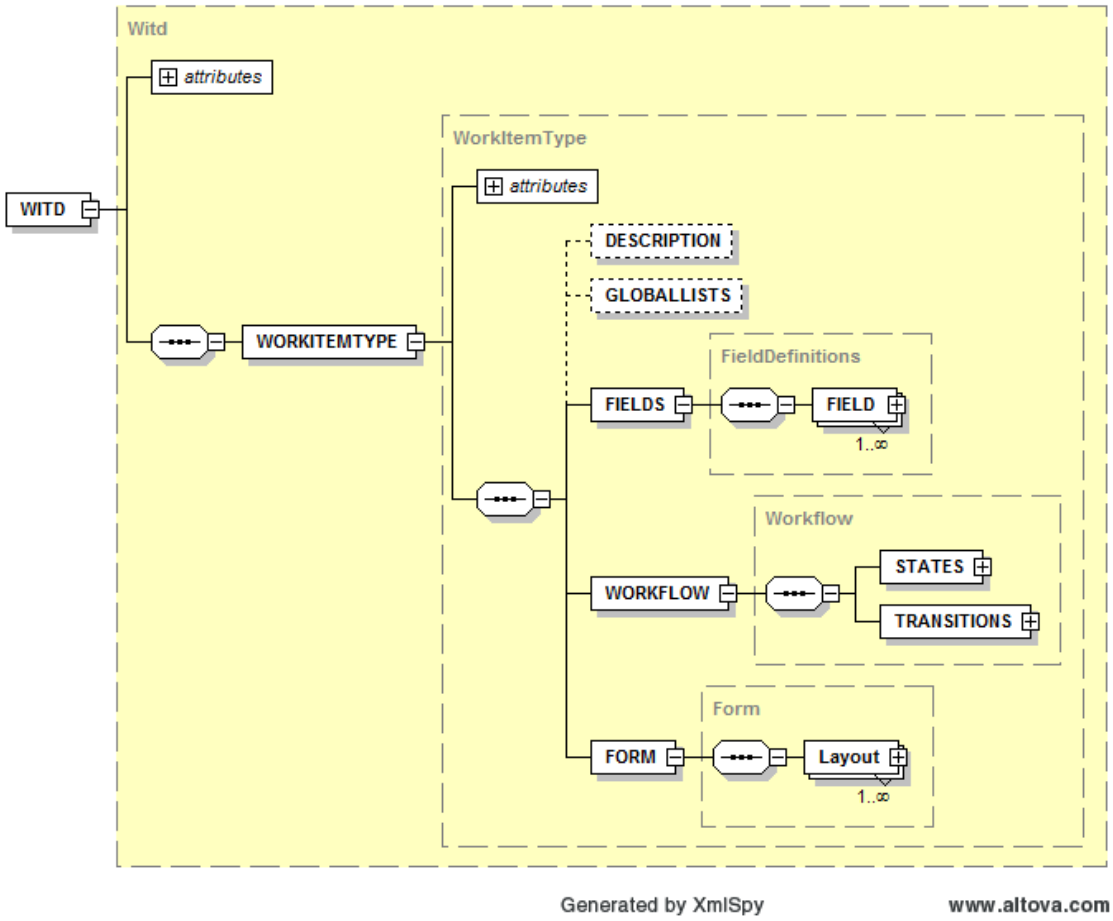


Abbildung 4.5.: XML-Schema der Work Item Types

Abbildung 4.5 zeigt dazu das XML-Schema, das allen Work Items zugrunde liegt. Work Item Typen bestehen im Wesentlichen aus Datenstrukturen (`Fields`), einem Workflow (`Workflow`) sowie einer Benutzerschnittstelle (`Form`), die für die Interaktion mit dem Anwender erforderlich ist. Alle diese Komponenten sind untereinander verknüpft. So setzen z. B. einzelne Statusübergänge eines Workflows Felder auf Standardwerte. Formen dienen ausschließlich dazu, den Daten der Felder eine grafische Repräsentation an der Benutzerschnittstelle zu geben.

TFS Mikro-Workflows. Abbildung 4.6 zeigt die Verfeinerung für die Workflow-Anteile des Schemas. Diese sind die interessantesten für die spätere Prozessmodellierung. Diese Workflows sind direkt in die Work Item Typdefinitionen eingebettet und somit nicht mit den komplexen, in der Geschäftsprozessmodellierung verwendeten, von z. B. Biztalk Server zu vergleichen. Es handelt sich hier im Wesentlichen um *Mikro-Workflows*, die Anwender bei alltäglichen Aufgaben unterstützen und führen.

Die Workflows orientieren sich an Zustandsautomaten und umfassen daher Zustände (`States`) und Zustandsübergänge (Transitionen, `Transitions`).

- Einem Zustand können mehrere Felder zugeordnet werden, die anhand von Regeln entweder automatisch Werte zugewiesen bekommen (z. B. vom Server) oder die als verpflichtend auszufüllen gekennzeichnet werden.
- Das zweite Element eines Workflows sind die Transitionen. Auch sie können auf Felder wirken, z. B. Änderungsdaten setzen. Weiterhin sind vergleichbar mit der UML 2 [JRH+03] *Reasons* (UML: Trigger) und *Actions* vorgesehen. Aktionen entsprechenden dabei Funktionsaufrufen auf dem TFS-API, z. B. das Einchecken von Code. Reasons (Gründe, Trigger) zeigen an, warum von einem Zustand S_1 nach S_2 gewechselt wird. Hier ist es möglich, verschiedene Gründe anzugeben, um mögliche Pfade im Workflow zu modellieren. Mindestens ein Grund, der sogenannte *Default Reason*, muss jedoch angegeben werden.

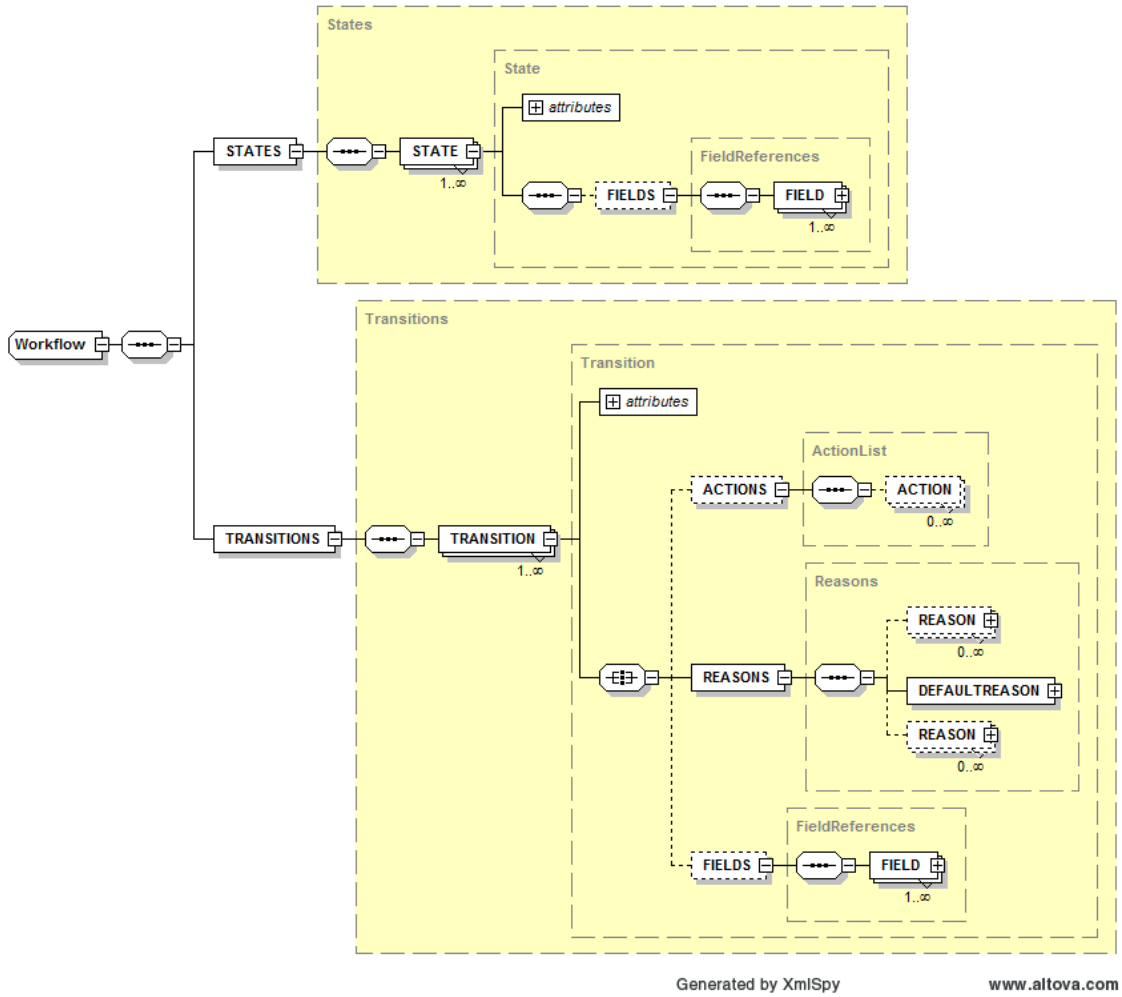


Abbildung 4.6.: XML-Schema der Work Item Types – Verfeinerung für Workflows

Die Workflows der TFS-Work Items eignen sich sehr gut dazu, kleinere Prozesse wie z. B. das Aufgabenmanagement oder das Risikomanagement zu modellieren. Die entsprechenden Modellierungen im Kontext des V-Modell XT nehmen wir in den folgenden Abschnitten vor. Ergänzende Informationen, wie z. B. das Layout der `Form`-Anteile, sind Anhang B zu entnehmen.

4.2.2. Work Item: Produkt

Der Work Item Typ *Produkt* entspricht dem Metamodellelement Produkt des V-Modell XT Metamodells (vgl. Abbildung 2.2). Es dient als Repräsentant für ein Produktexemplar, das in der Dokumentenverwaltung des TFS (genauer: des hinter dem TFS liegenden SharePoint Servers) aufbewahrt wird. Gleichzeitig bildet dieser Work Item Typ das *Produktzustandsmodell* (Abbildung 4.7) des V-Modells ab. Produkte sind somit nach den Regeln des V-Modells mit verschiedenen Stati versehen. Änderungen am Status sind so nach zu verfolgen.

Das Zustandsmodell für Produkte. Das Zustandsmodell für den Produktstatus hat einige Konsequenzen für die Work Items der Typen *Produkt* und *Aktivität*. Einerseits muss das Zustandsmodell für den Aktivitäts-Work Item Typ anhand dieses Modells entworfen werden, da die Aktivitäten im Kontext des V-Modells primär dazu dienen, Produkte fertig zu stellen. Diese Aufgabe muss mindestens beim Entwurf berücksichtigt werden. In der Modellierung des passenden Zustandsautomaten sind wir jedoch vergleichsweise frei, da das V-Modell hierzu keine Angaben macht. Anderseits ist das Zustandsmodell entscheidend für das Handling der einzelnen Produkttypen im Kontext der Qualitätssicherung. Technisch lässt sich die Zuordnung zur eigenständigen

Qualitätssicherung in Form von Abfragen (Queries, siehe Abschnitt 4.3.2) über den Work Items eines Projekts realisieren. Der Zustandsautomat des V-Modell-Produkts wird in einem Workflow modelliert.

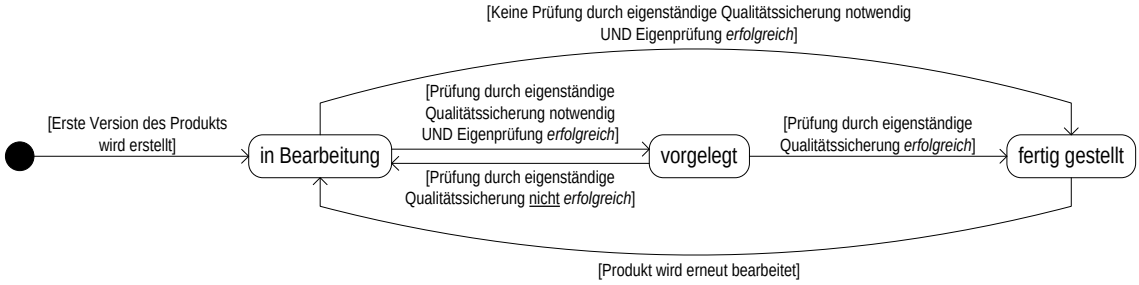


Abbildung 4.7.: Produktzustandsautomat des V-Modell XT

Produkterzeugung. V-Modell Produkte besitzen ein Erzeugungsattribut, welches aussagt, wie Exemplare eines Produkts im Rahmen eines Projekts erstellt werden. Das V-Modell unterscheidet *initiale*, *abhängige* und *externe* Produkte. Abhängig und extern erzeugte Produkte sind unspektakulär, da sie nicht unmittelbar erzeugt werden, sondern zunächst nur Planungsgrößen sind und dann bei „Bedarf" erstellt bzw. dem Projekt von außen zugeführt werden. Die genauen Abhängigkeiten und Erzeugungspfade sind im V-Modell durch die *erzeugenden Produktabhängigkeiten* geregelt. Initiale Produkte werden hingegen *genau einmal* in einem Projekt erzeugt. Im Kontext eines Work Items heißt das, dass es nur genau eine Instanz eines solchen Produkt Work Items gibt, auf das sich alle geplanten Aktivitäten beziehen. So z. B. ist in verschiedenen Entscheidungspunkten das Projekthandbuch vorzulegen, weshalb hier ggf. Aktivitäten eingeplant sind. Alle diese Aktivitäten beziehen sich immer auf dasselbe Projekthandbuch.

Folgendes Datenmodell legen wir dem Work Item Typ *Produkt* zugrunde:

Name	Reference Name
Titel	System.Title
Zustand	System.State
Grund	System.Reason
Iterationspfad	System.IterationPath
Zugewiesen an	System.AssignedTo
Beschreibung	System.Description
Verlauf	System.History
Bereichspfad	System.AreaPath
Geschlossen von	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Microsoft.VSTS.Common.ClosedDate
Verbleibende Arbeit	Microsoft.VSTS.Scheduling.RemainingWork
Startdatum	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Microsoft.VSTS.Scheduling.FinishDate
Produkttyp (VMXT)	VMXT.ProduktTyp
Erzeugung (VMXT)	VMXT.Erzeugung
vmxt_ref (VMXT)	VMXT.id
Bemerkungen (VMXT)	VMXT.Log
Prüfung (VMXT)	VMXT.Assessment

Tabelle 4.1.: Datenstruktur für Produkt-Work Items

Für das Feld *VMXT.Erzeugung* greift der Wertebereich des V-Modell XT Produktmodells: {initial, extern, abhängig}, um die Art der Produkterzeugung zu zeigen. Folgende Modellierung ergibt sich für den Workflow des Produkt Work Item Typs auf der Grundlage des Produktzustandsmodells des V-Modells:

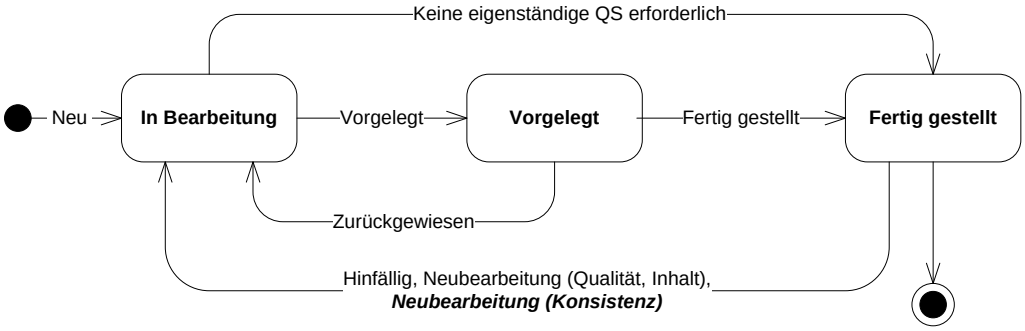


Abbildung 4.8.: Workflow für Produkt-Work Items

Konsistenzbedingungen. Für Produkt und deren Einplaung oder Erstellung im Projekt sind folgende Konsistenzbedingungen zu beachten. Ein Produkt darf nur in den Zustand fertig gestellt überführt werden,

- wenn die Aktivität, die das Produkt erstellt *abgeschlossen* ist und
- alle definierten QS-Maßnahmen erfolgreich ausgeführt wurden.

4.2.3. Work Item: Aktivität

Aktivitäten sind im Metamodell des V-Modells als Elemente enthalten, jedoch sind sie im V-Modell vergleichsweise undetailliert ausgearbeitet. Sind Aktivitäten definiert, so beziehen sie sich immer nur auf genau ein Produkt, können jedoch beliebig viele Teileaktivitäten enthalten (Teilaktivitäten verweisen auf eine Aktivität zu der sie gehören, Abbildung 2.3). Aktivitäten verfügen im V-Modell nur über deskriptiven Charakter. Sie enthalten keinerlei planungsrelevanten Informationen (Daten, Plan, Ist, Aufwände etc.). Weitere Informationen erhalten Aktivitäten jedoch nicht, sodass wir sowohl Daten als auch Verhalten ergänzen bzw. neu definieren müssen.

Folgende Daten liegen der Modellierung der Aktivitäten zugrunde (Ergänzungen sind hervorgehoben):

Name	Reference Name
Titel	System.Title
Zustand	System.State
Grund	System.Reason
Iterationspfad	System.IterationPath
Zugewiesen an	System.AssignedTo
Beschreibung	System.Description
Verlauf	System.History
Bereichspfad	System.AreaPath
Geschlossen von	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Microsoft.VSTS.Common.ClosedDate
Verbleibende Arbeit	Microsoft.VSTS.Scheduling.RemainingWork
Startdatum	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Microsoft.VSTS.Scheduling.FinishDate
p_ref (VMXT)	VMXT.PRef
ep_ref (VMXT)	VMXT.EPRef
vmxt_ref (VMXT)	VMXT.id
Bemerkungen (VMXT)	VMXT.Log

Tabelle 4.2.: Datenstruktur für Aktivität-Work Items

Im Rahmen des Tailorings werden die Aktivitäten instanziiert und mit Endedaten und Vorgänger-/Nachfolgerbeziehungen angereichert. Die Auswertung von Teilaktivitäten, im Rahmen der stan-

dardmäßigen Planexporte, ist nicht vorgesehen, weshalb wir keine explizite Modellierung vornehmen, sondern uns direkt unterhalb der Aktivitäten auf das Aufgabenmanagement abstützen (siehe Abschnitt 4.2.5). Die unscharfe Behandlung der Aktivitäten hat auch zur Folge, dass keinerlei Informationen verfügbar sind, die für die Ableitung eines Zustandsmodells herangezogen werden können. Wir orientieren uns daher am Work Item für Produkte und definieren einen einfachen Zustandsautomaten. Folgende Modellierung liegt der Definition des Workflows zugrunde:

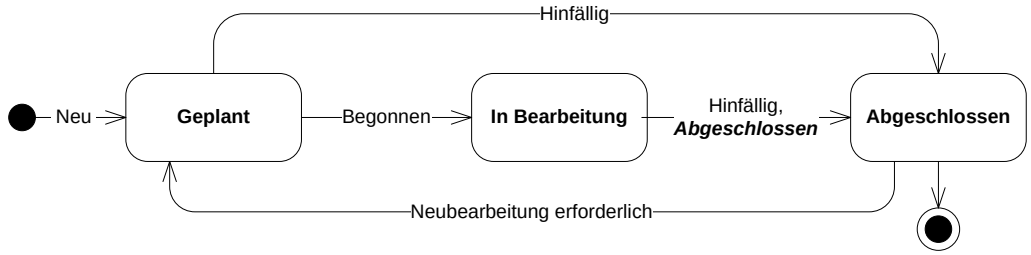


Abbildung 4.9.: Workflow für Aktivitäts-Work Items

Konsistenzbedingungen. Wir nehmen keine weitere Verfeinerung des Aktivitätskonzepts vor, sondern stützen uns für fein granulare Aufgaben auf die *Arbeitsaufträge* ab. Analog zu den Produkten sind aber auch hier Konsistenzbedingungen zu formulieren. Eine Aktivität darf nur abgeschlossen werden,

- wenn alle referenzierten *Arbeitsaufträge* oder *Aktivitäten* abgeschlossen sind.
- Wird eine Aktivität abgeschlossen, *muss* sich der Zustand referenzierter Produkte ändern.
- Eine Aktivität referenziert *max. ein Produkt*.

4.2.4. Work Item: Entscheidungspunkt

Der Entscheidungspunkt hat die Aufgabe, Produkte zusammenzufassen, die im Rahmen eines *Quality Gates* die Bestimmung des Projektfortschritts zulassen. Für den Entscheidungspunkt definiert das V-Modell keine im Rahmen des Work Item Trackings verwertbaren Daten. Folgende Daten legen wir daher der Modellierung des Work Item Typs *Entscheidungspunkt* zugrunde:

Name	Reference Name
Titel	System.Title
Zustand	System.State
Grund	System.Reason
Iterationspfad	System.IterationPath
Zugewiesen an	System.AssignedTo
Beschreibung	System.Description
Verlauf	System.History
Bereichspfad	System.AreaPath
Geschlossen von	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Microsoft.VSTS.Common.ClosedDate
Verbleibende Arbeit	Microsoft.VSTS.Scheduling.RemainingWork
Startdatum	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Microsoft.VSTS.Scheduling.FinishDate
Entscheidungspunkttyp (VMXT)	VMXT.Entscheidungspunkttyp
Prüfung (VMXT)	VMXT.Assessment
vmxt_ref (VMXT)	VMXT.id
Bemerkungen (VMXT)	VMXT.Log

Tabelle 4.3.: Datenstruktur für Entscheidungspunkt-Work Items

Das Feld *VMXT.Entscheidungspunkttyp* beinhaltet als vorgegebene Werte die Bezeichner der durch das V-Modell XT vorgegebenen Entscheidungspunkte:

- Abnahme erfolgt
- Anforderungen festgelegt
- Angebot abgegeben
- Feinentwurf abgeschlossen
- Gesamtprojekt aufgeteilt
- Gesamtprojektfortschritt überprüft
- Iteration geplant
- Lieferung durchgeführt
- Projekt abgeschlossen
- Projekt ausgeschrieben
- Projekt beauftragt
- Projekt definiert
- Projektfortschritt überprüft
- Projekt genehmigt
- Systemelemente realisiert
- System entworfen
- System integriert
- System spezifiziert
- Nicht zugeordnet (entspricht dem freien Meilenstein des V-Modell XT Projektassistenten)

Im Entscheidungspunkt sind alle vorzulegenden Produkte im Zustand *fertig gestellt* vorzuhalten. Dies hat zur Folge, dass alle hier geplanten Aktivitäten abgeschlossen sein müssen. Dies zeigt den Meilensteincharakter des Entscheidungspunkts. Im Projektplan wird er deshalb als solcher instanziiert. Folgende Modellierung legen wir dem Workflow des Work Item Typs Entscheidungspunkt zugrunde:

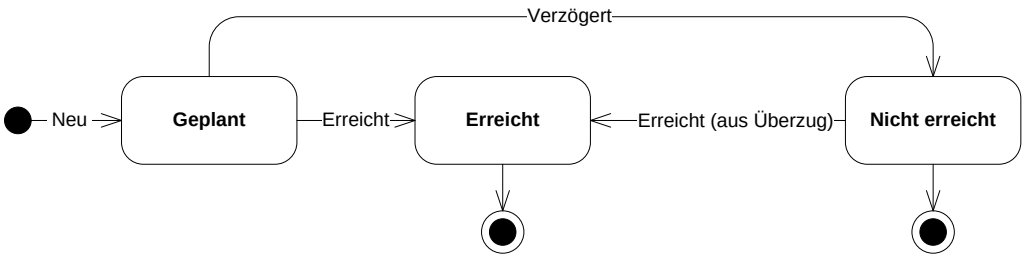


Abbildung 4.10.: Workflow für Entscheidungspunkt-Work Items

Konsistenzbedingungen. Ein Entscheidungspunkt darf nur dann erfolgreich abgeschlossen (erreicht) werden, wenn:

- Alle Produkte, die zum Entscheidungspunkt geplant und im Rahmen der eigenständigen Qualitätssicherung zu prüfen sind, im Zustand *fertig gestellt* vorliegen.
- Alle Aktivitäten, die im Entscheidungspunkt mit Fälligkeit eingeplant sind, im Zustand *abgeschlossen* sind.

4.2.5. Work Item: Arbeitsauftrag

Der Work Item Typ *Arbeitsauftrag* ist ein fachlicher Work Item Typ, der keine V-Modell-bezogenen Statusinformationen enthält, sondern einen im V-Modell integrieren Prozess abdeckt. Das Aufgabenmanagement wird im V-Modell durch ein Produkt vom Typ *Arbeitsauftrag* repräsentiert. Die korrespondierende Aktivität ist *Arbeitsauftrag erteilen* (vgl. Abbildung 4.4). Im Rahmen unserer

Abbildung auf TFS stellen wir einen Aufgabenmanagementprozess auf der Basis des Arbeitsauftrags bereit, der gleichzeitig die Teilaktivitäten[2] ersetzen soll. Eine Aktivität nach V-Modell-Interpretation für den TFS ist somit ein Container für spezifische Arbeitsaufträge. Arbeitsaufträge können auch ohne Bezug zu einer Aktivität im Projekt vergeben werden, wodurch sie nicht unter „Produktzwang" stehen und daher relativ einfach das Projekttagesgeschäft unterstützen können. Der Modellierung des Work Item Typs *Arbeitsauftrag* legen wir folgende Daten zugrunde[3]:

Name	Reference Name
Titel	System.Title
Zustand	System.State
Grund	System.Reason
Iterationspfad	System.IterationPath
Zugewiesen an	System.AssignedTo
Beschreibung	System.Description
Verlauf	System.History
Bereichspfad	System.AreaPath
Geschlossen von	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Microsoft.VSTS.Common.ClosedDate
Priorität	Microsoft.VSTS.Common.Priority
Schweregrad	Microsoft.VSTS.Common.Severity
Verbleibende Arbeit	Microsoft.VSTS.Scheduling.RemainingWork
Abgeschlossene Arbeit	Microsoft.VSTS.Scheduling.CompletedWork
Startdatum	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Microsoft.VSTS.Scheduling.FinishDate
Aufgabenhierarchie	Microsoft.VSTS.Scheduling.TaskHierarchy
Disziplin (VMXT)	VMXT.Disciplin
Schätzung (VMXT)	VMXT.Estimate
Blockiert (VMXT)	VMXT.Blocked
Prüfung (VMXT)	VMXT.Assessment
Bemerkungen (VMXT)	VMXT.Log

Tabelle 4.4.: Datenstruktur für Arbeitsauftrag-Work Items

Im Arbeitsauftrag sind verschiedene Felder mit definierten Wertebereichen hinterlegt.

- Priorität (*Microsoft.VSTS.Common.Priority*) hat den Wertebereich: {1, 2, 3}.
- Schweregrad (*Microsoft.VSTS.Common.Severity*): {kritisch, hoch, mittel, niedrig}.
- Disziplin (VMXT) (*VMXT.Discipline*), um eine nähere Klassifikation durchzuführen:
  - Entwicklung
  - Qualitätssicherung
  - Projektmanagement
  - Anforderungsfestlegung
  - Architektur, Entwurf, Spezifikation
  - Konfigurationsmanagement
  - Ausschreibungs- und Vertragswesen
  - Problem- und Änderungsmanagement.
- Blockiert (VMXT) (*VMXT.Blocked*): {ja, nein}.

2 Wobei wir eine automatische Auswertung und Zuordnung im Rahmen des Generierungsprozesses zunächst nicht vorsehen.

3 Das V-Modell definiert zu verschiedenen Produkten so genannte *beispielhafte Produktgestaltungen*. Wir orientieren uns weitgehend an diesen, stellen jedoch im Zweifelsfall die Einfachheit und Klarheit vornan. Weitere Ausgestaltungen, insbesondere im Rahmen von organisationsspezifischen Anpassungen sind aber möglich und durch das V-Modell erwünscht. In wie fern eine stark abweichende, inhaltlich Ausgestaltung den Regeln der V-Modell Konformität widerspricht, muss ggf. durch ein separates Assessment geklärt werden. Sofern die Anpassung/Ausgestaltung im Rahmen einer organisationsspezifischen Anpassung nach V-Modell XT stattgefunden hat, ist jedoch nicht mit Problemen zu rechnen. Ggf. muss hier ein angepasster Work Item Typ erstellt werden.

In der Modellierung des Arbeitsauftrags orientieren wir uns technisch an den Work Item Typen, die durch das Visual Studio SDK und die von TFS mitgelieferten Process Templates vorgefertigt werden. Inhaltlich, d. h. für die Ableitung von Transitionen und Zuständen orientieren wir uns an den Gegebenheiten des WEIT-Projekts, wo mit einfachsten Mitteln ein räumlich verteiltes Team koordiniert wird. Folgende Modellierung liegt dem Workflow des Work Item Typs Arbeitsauftrag zugrunde:

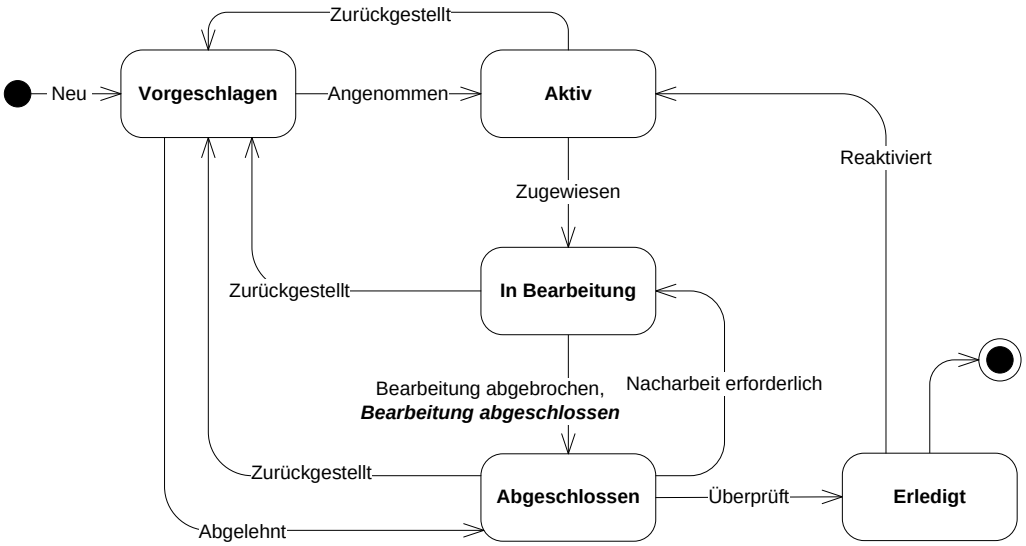


Abbildung 4.11.: Workflow für Arbeitsauftrag-Work Items

4.2.6. Work Item: Risiko und Maßnahme

Die Work Item Typen *Risiko* und *Maßnahme* sind ebenfalls fachliche Typen, die einen komplexen Teilprozess des Projektmanagements – das Risikomanagement – adressieren.

Work Item Typ: (Projekt-)Risiko

Das V-Modell berücksichtigt das Risikomanagement durch den Produkttyp *Risikoliste* (vgl. Abbildung 4.4), der alle erkannten Risiken und einen korrespondierenden Maßnahmenplan enthält. Analog zum Arbeitsauftrag sind hier verschiedene Ausgestaltungen möglich. Tabelle 4.5 zeigt die Datenstruktur, die wir dem Work Item Typ *Risiko*[4] zugrunde legen. Auch für das Risiko Work Item sind verschiedene Standardwertebereiche definiert:

- Wahrscheinlichkeit (VMXT) (*VMXT.Likelihood*) hat den Wertebereich: {10, 30, 50, 70, 90} zur groben Einstufung der Eintrittswahrscheinlichkeiten.
- Die Felder Auswirkung auf Zeit (VMXT) (*VMXT.ImpactTime*), Auswirkung auf Qualität (VMXT) (*VMXT.ImpactQuality*), Auswirkung auf Budget (VMXT) (*VMXT.ImpactBudget*), Auswirkung auf Umfang (VMXT) (*VMXT.ImpactScope*) haben jeweils die Wertebereiche: {kritisch, hoch, mittel, gering, keine} zur genaueren Einstufung von Risiken.
- Strategietyp (VMXT) (*VMXT.StrategyType*): {vermeidung, transfer, akzeptanz}.
- Risikoklasse (VMXT) (*VMXT.RiskClass*): {katastrophal, kritisch, uner-wünscht, tolerierbar}.

Die Risikoliste ist ein Produkttyp , für den es eine *beispielhafte Produktgestaltung* gibt. Diese orientiert sich an den Inhalten der zu erstellenden Produkte und nicht am dahinter liegenden Prozess. Aus diesem Grund haben wir analog zum *Arbeitsauftrag* bei der Modellierung auf die Prozesse aus dem WEIT-Projekt sowie auf die bereits verfügbaren Work Item Typen zurückgeriffen. Abbildung 4.12 zeigt die Modellierung für den Workflow des Work Item Typs Risiko.

4 Wir verwenden einheitlich den Terminus *Risiko*, auch wenn der konkrete Work Item Typ *Projektrisiko* heißt. Als Realisierungsentscheidung liegt dieses in einer aufgesplitteten Form vor, in der Risiken und Maßnahmen getrennt sind. Eine alternative Konstruktion, in dem ein integriertes Risiko-Work Item alle Informationen hält, ist aber ebenfalls möglich und widerspricht nicht dem V-Modell XT.

Name	Reference Name
Titel	System.Title
Zustand	System.State
Grund	System.Reason
Iterationspfad	System.IterationPath
Zugewiesen an	System.AssignedTo
Beschreibung	System.Description
Verlauf	System.History
Bereichspfad	System.AreaPath
Erstellungsdatum	System.CreatedDate
Geschlossen von	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Microsoft.VSTS.Common.ClosedDate
Wahrscheinlichkeit (VMXT)	VMXT.Likelihood
Auswirkung auf Zeit (VMXT)	VMXT.ImpactTime
Auswirkung auf Qualität (VMXT)	VMXT.ImpactQuality
Auswirkung auf Budget (VMXT)	VMXT.ImpactBudget
Auswirkung auf Umfang (VMXT)	VMXT.ImpactScope
Risikoschaden (VMXT)	VMXT.Severity
Strategietyp (VMXT)	VMXT.StrategyType
Strategiebeschreibung (VMXT)	VMXT.StrategyDescription
Maßnahmenbeschreibung (VMXT)	VMXT.MeasureComment
Risikomaß (VMXT)	VMXT.RiskImpact
Risikoklasse (VMXT)	VMXT.RiskClass
Prüfung (VMXT)	VMXT.Assessment
Bemerkungen (VMXT)	VMXT.Log
Autor (VMXT)	VMXT.Autor

Tabelle 4.5.: Datenstruktur für Risiko-Work Items

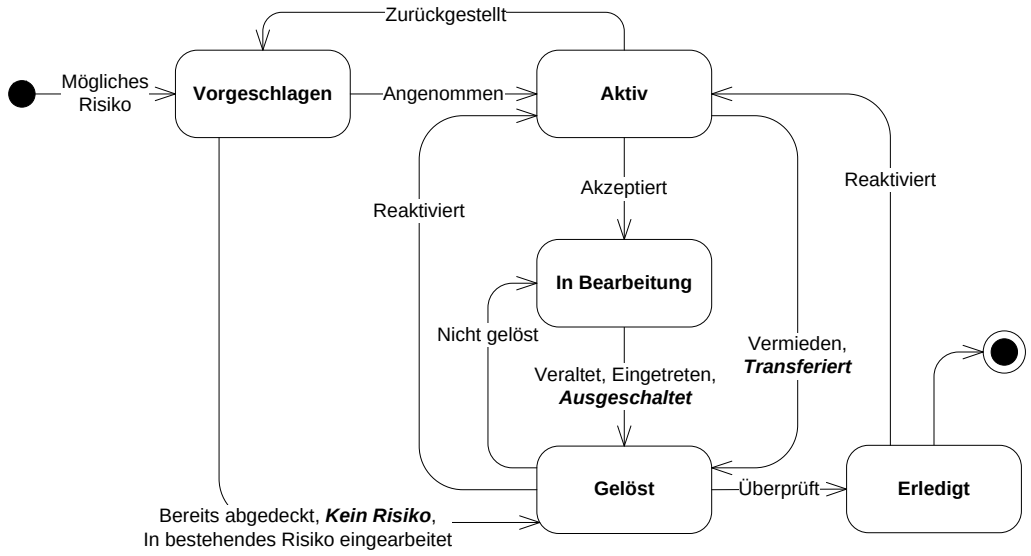


Abbildung 4.12.: Workflow für Risiko-Work Items

Work Item Typ: Maßnahme

Es ist sinnvoll, Risiken mit anderen Work Item Typen zu verknüpfen. Dies ist nicht nur sinnvoll sondern sogar erforderlich, wenn Risiken schwer wiegend und intensiv zu bewerten sind. Dann ist es erforderlich, *Arbeitsaufträge* zu vergeben, um bspw. Risiken zu bewerten oder Maßnahmen zu deren Minderung einzuleiten. Risiken können weiterhin Bezug zu *Produkten*, *Aktivitäten* und *Entscheidungspunkten* haben, sofern diese durch das Eintreten eines Risikos direkt betroffen wären.

Im Rahmen der Abbildung ist hier insbesondere für Maßnahmen ein eigener Work Item Typ *Maßnahme* als beispielhafte Ausgestaltung vorgesehen. Folgende Datenstruktur legen wir dem Work Item Typ *Maßnahme* zugrunde:

Name	Reference Name
Titel	System.Title
Zustand	System.State
Grund	System.Reason
Iterationspfad	System.IterationPath
Zugewiesen an	System.AssignedTo
Beschreibung	System.Description
Verlauf	System.History
Bereichspfad	System.AreaPath
Geschlossen von	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Microsoft.VSTS.Common.ClosedDate
Abschlussdatum	Microsoft.VSTS.Scheduling.FinishDate
Prüfung (VMXT)	VMXT.Assessment
Bemerkungen (VMXT)	VMXT.Log
Trigger (VMXT)	VMXT.Trigger

Tabelle 4.6.: Datenstruktur für Maßnahme-Work Items

Folgende Modellierung orientiert sich am Arbeitsauftrag und liegt dem Workflow des Work Item Typs *Maßnahme* zugrunde:

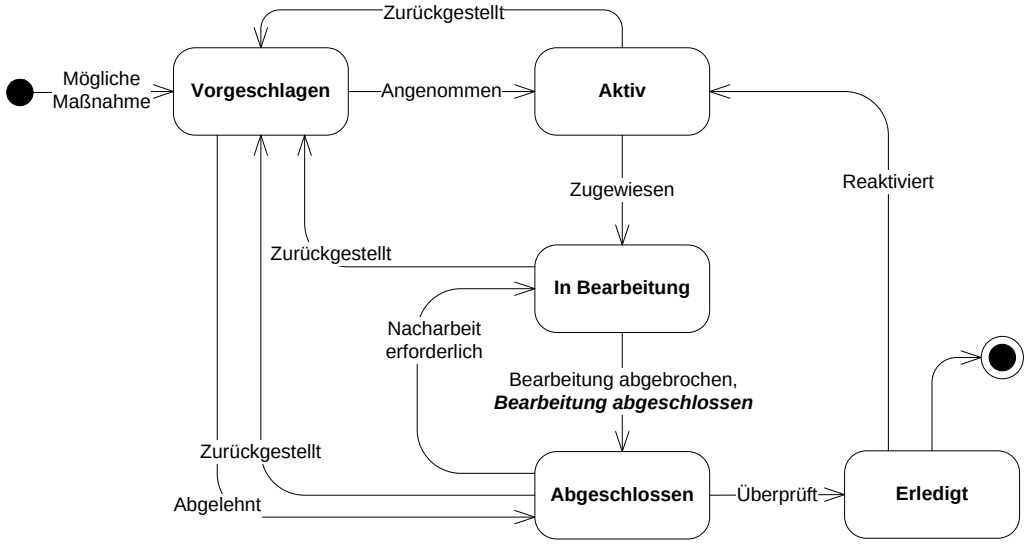


Abbildung 4.13.: Workflow für Maßnahme-Work Items

4.2.7. Work Item: PÄM

Das Problem- und Änderungsmanagement ist ein komplexer Prozess den wir im Kontext dieser Interpretation so weit wie möglich vereinfacht haben. Auch hier sind selbstverständlich Anpassungen möglich. Ausgestaltungen sind insbesondere bei diesem Work Item Typ notwendig, da wir mit unserer Überführung des V-Modells nur den Managementrahmen, jedoch keine Software-Entwicklungsanteile berücksichtigen. Diese Abbildung leistet das *nicht*, um den generischen Rahmen des V-Modells zu erhalten.

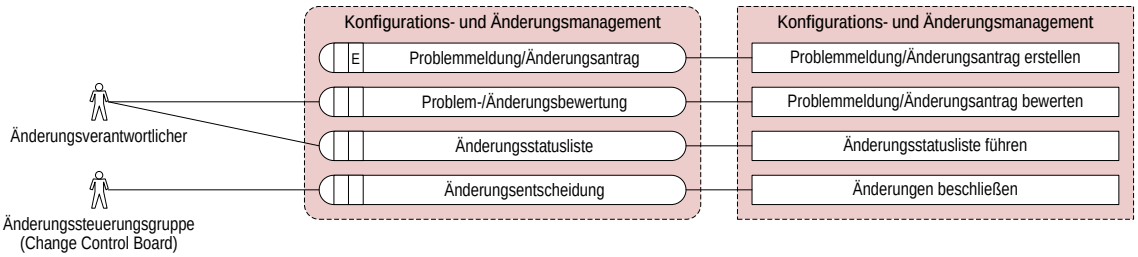


Abbildung 4.14.: Vorgehensbaustein Problem- und Änderungsmanagement

Problem- und Änderungsmanagement á la WEIT. Problem- und Änderungsmanagement ist im V-Modell als eine der Kerndisziplinen in einem eigenen Vorgehensbaustein untergebracht. Dieser Vorgehensbaustein *Problem- und Änderungsmanagement* (Abbildung 4.14) definiert eine Reihe von Produkten und korrespondierenden Aktivitäten, die wir geeignet zu einem integrierten und kompakten Prozess zusammenfassen. Dies hat natürlich zur Folge, dass wir in unserer Modellierung nicht nur einzelne Produkttypen in geeignete Work Item Typen überführen, sondern den gesamten Vorgehensbaustein ablösen. Die in Abbildung 4.14 zu sehenden Produkttypen:

- Problemmeldung/Änderungsantrag
- Problem-/Änderungsbewertung
- Änderungsstatusliste
- Änderungsentscheidung

werden gänzlich in einen kompakten Prozess integriert. Kernelement unseres Prozesses ist eine integrierte *Problem- und Änderungsmeldung* (PÄM), die wir als Work Item Typ modellieren. Wir orientieren uns hier jedoch nicht weiter an den Daten und Datenstrukturen die das V-Modell im Rahmen seiner Dokumentation bereitstellt, sondern bauen auf dem Änderungsmanagementsystem des V-Modell XT Entwicklungsprojekts WEIT auf. Das System setzt auf Mantis[5] auf, einem Open Source Bug Trackingsystem, das über die Webseiten der KBSt (V-Modell XT Portal) öffentlich zugänglich ist und somit auch den Anwendern des V-Modells offen steht, um Fehler am V-Modell zu melden, Änderungsforderungen einzustellen oder Feature Requests zu starten.

Tabellen 4.7 legt die Daten für die Modellierung des Work Item Typs *PÄM* fest. Die Standardwertebereiche für ausgewählte Felder sind:

- Auswirkungen auf Projektziel (VMXT) (*VMXT.ImpactOnProjectPromise*) hat den Wertebereich: {kritisch, hoch, mittel, niedrig}.
- Kategorie (VMXT) (*VMXT.Kategorie*): {änderungsantrag, problemmeldung}.
- Reproduzierbar (VMXT) (*VMXT.Reproduzierbar*): {ja, nein, nicht relevant}.
- Dringlichkeit (VMXT) (*VMXT.Urgency*): {kritisch, sehr wichtig, wichtig, wünschenswert}.
- Empfehlung (VMXT) (*VMXT.Recommendation*): {ablehnen, annehmen, zurückstellen}.

Im Vorgehensbaustein *Problem- und Änderungsmanagement* sind aber auch einige Aktivitäten definiert, die wir in die Modellierung des Workflows einfließen lassen müssen. Hier gehen wir einen gänzlich anderen Weg in der Modellierung und orientieren uns an einem existierenden Workflow aus dem *MSF for CMMI* [GP06]. Der Hintergrund ist die angestrebte Einfachheit des PÄM-Prozesses. Der hinter dem Änderungssystem des V-Modells liegende Prozess ist dafür zu

5 Quelle zu Mantis: `http://www.mantisbt.org`

Name	Reference Name
Titel	System.Title
Zustand	System.State
Grund	System.Reason
Iterationspfad	System.IterationPath
Zugewiesen an	System.AssignedTo
Beschreibung	System.Description
Verlauf	System.History
Bereichspfad	System.AreaPath
Geschlossen von	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Microsoft.VSTS.Common.ClosedDate
Verbleibende Arbeit	Microsoft.VSTS.Scheduling.RemainingWork
Startdatum	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Microsoft.VSTS.Scheduling.FinishDate
Prüfung (VMXT)	VMXT.Assessment
Bemerkungen (VMXT)	VMXT.Log
Empfehlung (VMXT)	VMXT.Recommendation
Entscheid (VMXT)	VMXT.Decision
Analyse (VMXT)	VMXT.Analysis
Auswirkungen auf Projektziel (VMXT)	VMXT.ImpactOnProjectPromise
Tatsächliche Lösung (VMXT)	VMXT.CorrectiveActionActualSolution
Gewünschter Fertigstellungszeitpunkt (VMXT)	VMXT.TargetDate
Kategorie (VMXT)	VMXT.Kategorie
Reproduzierbar (VMXT)	VMXT.Reproduzierbar
Autor (VMXT)	VMXT.Autor
Dringlichkeit (VMXT)	VMXT.Urgency
Gegenstand (VMXT)	VMXT.Item
Version (VMXT)	VMXT.ItemVersion

Tabelle 4.7.: Datenstruktur für PÄM-Work Items

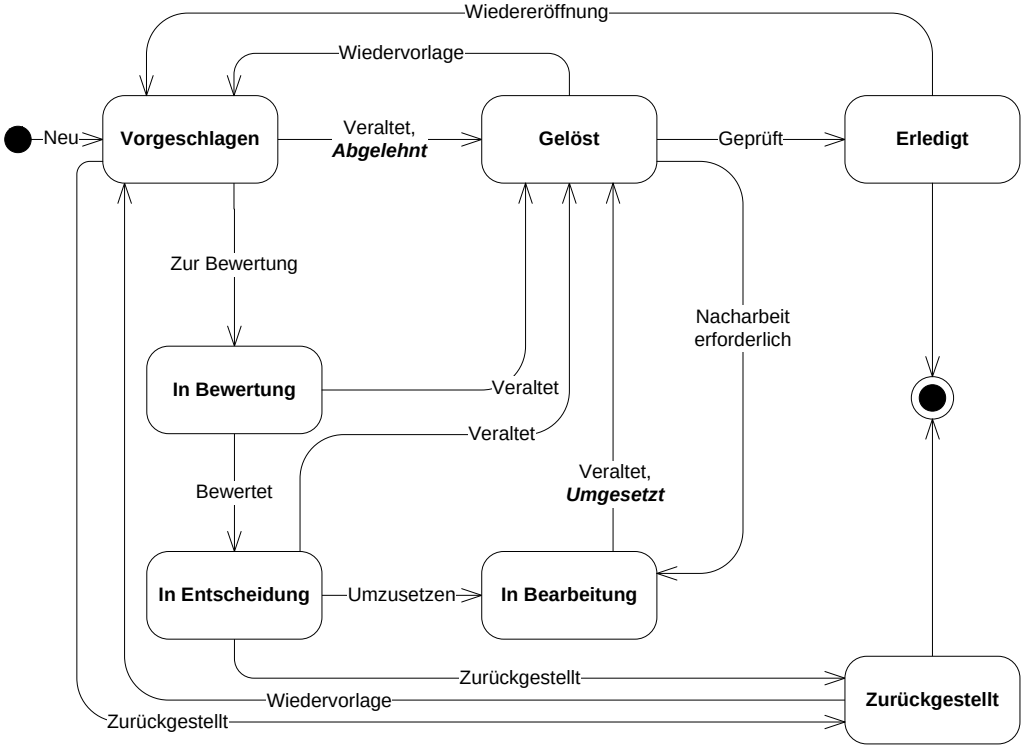


Abbildung 4.15.: Workflow für PÄM-Work Items

komplex und vielschichtig. Die in den problem- und änderungsbezogenen Aktivitäten beschriebenen Pfade sind zu allgemein und wenig integriert. Die Definitionen des MSF for CMMI bilden hingegen eine schlanke und elegante Grundlage, auf der wir aufbauen. Die Modellierung aus Abbildung 4.15 liegt dem Workflow des Work Item Typs *PÄM* zugrunde.

Mögliche Verknüpfungen und Strukturierungen. Auch für Problemmeldungen ist es erforderlich, andere Work Items sinnvoll anzubinden und zu verknüpfen. Insbesondere betrifft das wieder Arbeitsaufträge, die ausgelöst werden, um PÄMs zu bewerten, einzuplanen oder umzusetzen. Weiterhin sind auch Risiken zu berücksichtigen, die durch auftretende PÄMs motiviert oder ausgelöst werden können. Dem entsprechend sind auch Produkte zu betrachten, die durch PÄMs direkt oder indirekt betroffen sind.

4.3. Sonstige Process Template Elemente

Neben den Work Items definieren Process Templates noch weitere Elemente (siehe Abschnitt 4.1.2). In den folgenden Abschnitten gehen wir auf diese Elemente näher ein. Im Speziellen betrachten wir aber in Abschnitt 4.3.4 die Art, wie Projektdurchführungsstrategien des V-Modells auf die Projektstrukturen des TFS (Areas und Iterations) abgebildet werden.

4.3.1. Work Item Queries

Im Rahmen eines Process Template sind auch einfache Abfragemechanismen umsetzbar . Diese können dazu verwendet werden, schnell einen Überblick über das Work Item Tracking System des TFS erhalten. In einem TFS-Projekt gibt es verschiedene Bereiche für die Abfragen, nämlich so genannte *Teamabfragen* (Team Queries) und *eigene Abfragen* (My Queries). Bei der Generierung eines Process Templates aus einem projektspezifisch angepassten V-Modell können die Teamabfragen bereits für die neuen V-Modell-bezogenen Work Item Typen vorgefertigt werden. Die Abfragen beziehen sich somit auf:

- Produkte
- Aktivitäten
- Entscheidungspunkte
- Arbeitsaufträge
- Risiken
- PÄMs

Wir geben diese Abfragen im Folgenden an:

- Query: Alle
- Query: Alle Entscheidungspunkte
- Query: Alle Produkte
- Query: Alle Aktivitäten
- Query: Alle PÄM
- Query: Alle Arbeitsaufträge
- Query: Alle Risiken
- Query: Alle offenen Produkte
- Query: Alle offenen Aktivitäten
- Query: Alle offenen PÄM
- Query: Alle offenen Arbeitsaufträge
- Query: Alle offenen Risiken

4.3.2. Reports

Das V-Modell XT sieht in seinem Vorgehensbaustein *Messung und Analyse* (Abbildung 4.16) bereits die Erfassung von quantitativen Projektkennzahlen vor. Die Erfassung solcher Zahlen kann

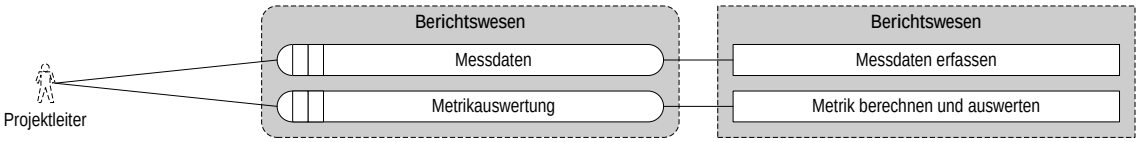


Abbildung 4.16.: Vorgehensbaustein Messung und Analyse

dazu verwendet werde, um z. B. Projekt mit einander vergleichbar zu machen oder Trends zu errechnen.

TFS baut unter anderem auch auf den SQL-Reporting Services auf (Abschnitt 4.1.1). Abbildung 4.17 zeigt einen Ausschnitt des Report-Subsystems nach [Mic07]. TFS kann somit verschiedene interne und externe Datenbanken zu einem Data Warehouse zusammenfassen und über diesen Strukturen detaillierte Reports generieren. Dabei sind die Daten in den Tabellen die Entsprechungen für den V-Modell-Produkttyp *Messdaten* (Abbildung 4.16). Die generierten Reports entsprechend dann dem Produkttyp *Metrikauswertung*. Je nach Reporttyp können über das Reporting auch weitreichende Standardberichte für ein Projekt erfasst werden. Diese unterstützen dann die Automatisierung des Berichtswesens und können z. B. als Teil des *Projektstatusberichts* verwendet werden. Für das V-Modell XT-Template sind einige Standardreports enthalten.

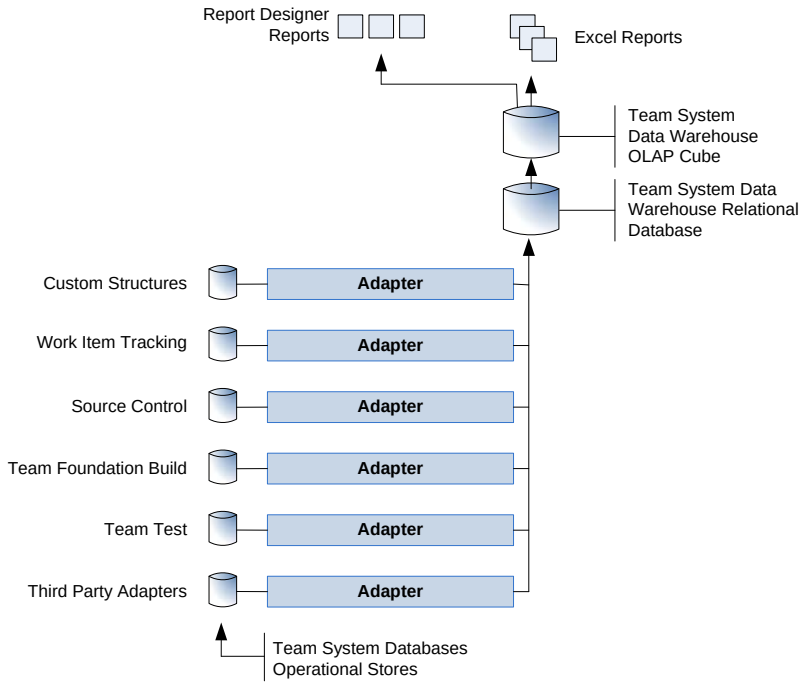


Abbildung 4.17.: Architektur des TFS-Report Subsystems

4.3.3. Abbildung von Rollen

Die Rollenkonzepte und das Verständnis Rollen im V-Modell und im TFS unterscheiden sich grundlegend. Das V-Modell versteht Rollen als Konzept, das Befähigungen und Zuständigkeiten beschreibt. TFS koppelt sein Rollenverständnis an das Sicherheits- und Berechtigungssystem des Hostsystems, sodass hier Berechtigungen im Sinne von Lesen und/oder Schreiben im Dateisystem vergeben werden. Diese beiden vollständig gegensätzlichen Ansätze lassen sich zum aktuellen Zeitpunkt nicht in einer automatischen Abbildung ineinander überführen. Als Zwischenlösung werden jedoch, da nach dem Tailoring des V-Modells alle erforderlichen Rollen ermittelt wurden, die Rollen des V-Modells als Gruppen im Process Template hinterlegt. Beim Import des Templates und der Instanziierung werden die Rollen somit als Gruppen im System hinterlegt.

4.3.4. Abbildung von Projektstrukturen

Eine der aufwändigen Aufgaben bei der Übersetzung der V-Modell-Strukturen auf den TFS ist die Auswertung der Projektdurchführungsstrategien. Diese dienen im V-Modell als Vorlage für eine grob granulare, initiale Projektplanung. Im Rahmen des Tailorings mit dem Projektassistenten kann auf der Basis einer oder mehrerer Projektdurchführungsstrategie(n) ein initialer aber bereits sehr komplexer Projektplan generiert werden [Ber06, KHTS07, HKST07]. Dieser spiegeln in groben Zügen die Organisation eines Projekts (Teilprojekte, Parallelentwicklungen etc.) wider. Die Projektdurchführungsstrategien legen neben den Entscheidungspunkten auch die Pfade zwischen den Entscheidungspunkten fest. Das V-Modell unterstützt dabei interatives Vorgehen bei gleichzeitiger Auswahl- und Wechselmöglichkeit des Vorgehens.

Beispiel: Als Beispiel wollen wir ein typisches Entwicklungsprojekt aufführen, das in mehreren Iterationen erstellt wird. Ein gewisser Anteil der Iteration dient der Erstellung einer Benutzerschnittstelle. Diese wird üblicherweise nicht in einem formalen inkrementell/iterativen Ansatz erstellt, wie sie die PDS *Inkrementelle Systementwicklung (AN)* des V-Modells beschreibt. Vielmehr sind hier oft *Prototyping*-Ansätze zu finden, in denen GUI-Prototypen in enger Zusammenarbeit mit Kunden abgestimmt werden. Das V-Modell sieht hierfür die *Agile Systementwicklung (AN)* vor. Um nach der Entwicklung und Abnahme der GUI nicht zwangsweise Prototyp-basiert weiterarbeiten zu müssen, gestattet das V-Modell zu ausgewählten Punkten im Projekt einen Wechsel der Projektdurchführungsstrategie, womit verschiedene Entwicklungsansätze auch gemischt werden können.

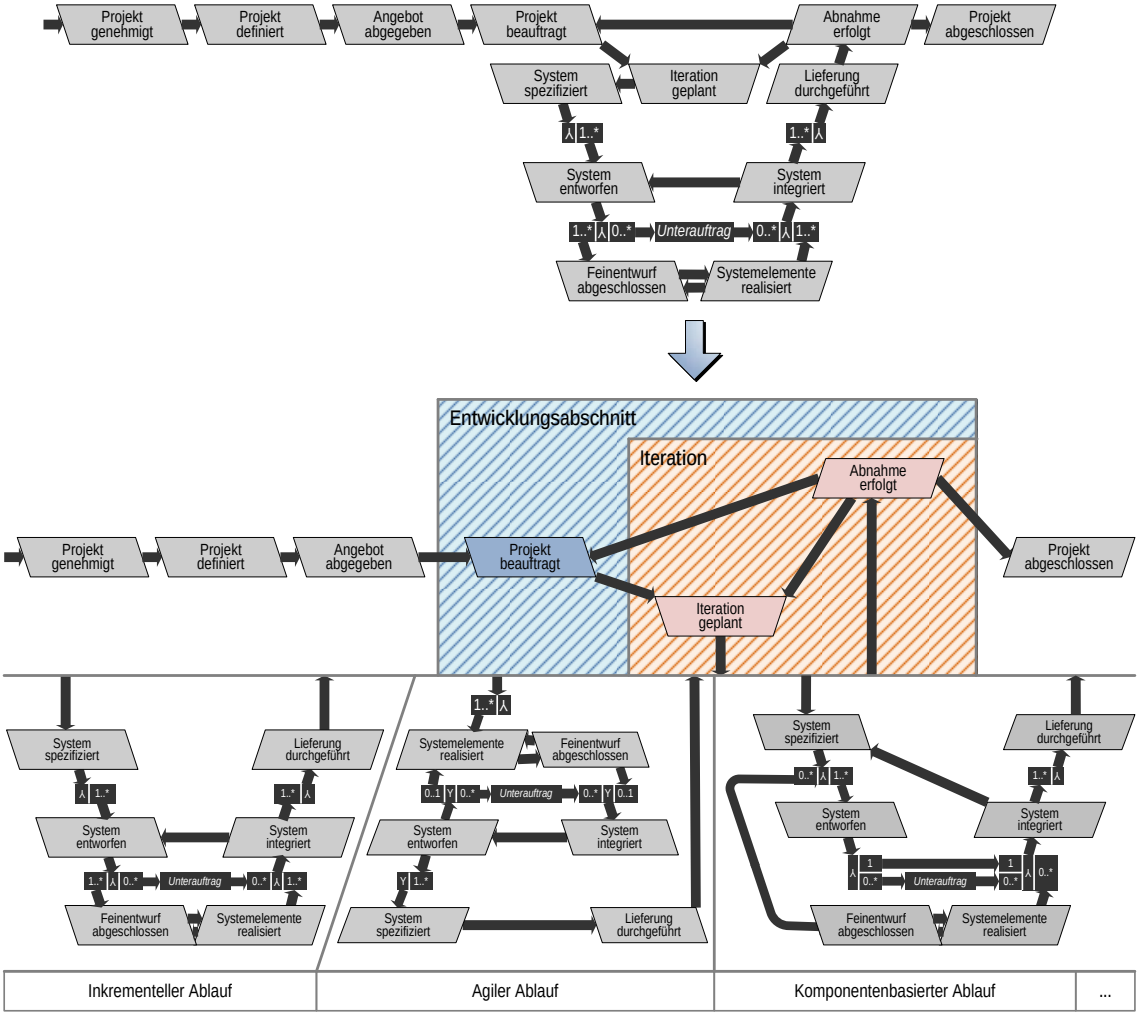


Abbildung 4.18.: Generatorschnittmuster für AN-Projektdurchführungsstrategien

Der Entscheidungspunkt Iteration geplant

Der Wechsel einer Vorgehensweise ist immer zum Entscheidungspunkt *Iteration geplant* möglich. Dieser bietet einen definierten Aufhänger, um Entwicklungsabläufe zu kombinieren. Ein V-Modell-Iteration beginnt in unserer Modellierung an diesem Entscheidungspunkt *Iteration geplant* und ist am Entscheidungspunkt *Abnahme erfolgt* beendet. Abbildung 4.19 zeigt dies am Beispiel für Projektdurchführungsstrategien der Auftragnehmer-Projekttypen. Abläufe, Entscheidungspunkte und Pfade zwischen diesen beiden Entscheidungspunkten sind für uns transparent.

Dies hat zur Folge, dass wir mit einem generischen Konzept die *Iterationen* des TFS adressieren können. Eine TFS-Iteration wird dann begonnen, wenn in einem geplanten V-Modell-Ablauf der Entscheidungspunkt *Iteration geplant* erreicht wird. Sie ist abgeschlossen, wenn der Entscheidungspunkt *Abnahme erfolgt* erreicht wird. Da die konkreten Abläufe zwischen diesen beiden Entscheidungspunkten transparent sind, können hier auch die verschiedenen Entwicklungsabläufe verwendet werden. Abbildung 4.19 zeigt dies beispielhaft für die Inkrementelle, die Agile und die Komponentenbasierte Systementwicklung (AN).

Der Entscheidungspunkt Projekt beauftragt

Nicht alle Iterationen im V-Modell beginnen mit dem Entscheidungspunkt *Iteration geplant*. Das V-Modell sieht z. B. auch Szenarios vor, in denen eine stufenweise Entwicklung (z. B. Losverfahren der Bundeswehr) vorkommt. In solchen Fällen werden einzelne *Entwicklungsabschnitte* separat beauftragt. Diese Entwicklungsabschnitte können dann wiederum beliebig viele Iterationen enthalten. Ein Entwicklungsabschnitt liegt in unserer Modellierung genau dann vor, wenn der Entscheidungspunkt *Projekt beauftragt* erreicht wird. Somit hat der Entwicklungsanteil eines V-Modell-Projekts im TFS nach unserer Modellierung immer mindestens einen Entwicklungsabschnitt, der mindestens eine Iteration enthält.

Der Entscheidungspunkt Anforderungen festgelegt

Mit TFS adressieren wir auch das zweite Entwicklungsszenario, das vom V-Modell unterstützt wird: den Projekttyp *Auftraggeber/Auftragnehmer* (siehe auch Anhang A.2). Im Gegensatz zum reinen Auftragnehmerprojekt sieht dieser Projekttyp keine Auftragsvergabe und Beauftragung vor, sondern beinhaltet eine integrierte Anforderungsfeststellung. In den Projektdurchführungsstrategien spiegelt sich dies wider. Abbildung 4.19 zeigt im oberen Anteil die Projektdurchführungsstrategie für die *Inkrementelle Systementwicklung (AG/AN)*. Wir wenden auch hier wieder der Iterationen-/Entwicklungsabschnittmodell an. Die einzelnen Entwicklungsabläufe, z. B. für inkrementelle oder agile Abläufe entsprechen denen der reinen Auftragnehmerprojekte (Abbildung 4.19). Jedoch unterscheidet sich die Bruchstelle für die Entwicklungsabschnitte.In AG/AN-Projekten wird ein neuer Entwicklungsabschnitt dann betreten, wenn der Entscheidungspunkt *Anforderungen festgelegt* erreicht wird. Das Verhalten und die weiter gehende Strukturierung entspricht aber wieder dem, wie es in AN-Projekten am Entscheidungspunkt *Projekt beauftragt* vorliegt.

In Abbildung 4.20 ist ein Beispiel zu sehen, wie die iterationsorientierten Entwicklungsabläufe des V-Modells auf eine Struktur bestehend aus Entwicklungsabschnitten und Iterationen für den TFS abgebildet werden. Ein Entwicklungsabschnitt im Kontext des TFS ist dabei selbst wieder eine Iteration, die hierarchisch geordnet weitere Iterationen enthält. Im Beispiel in Abbildung 4.20 haben wir dabei sowohl die AN- als auch die AG/AN-Interpretation aufgeführt. Beide Abläufe in der Abbildung beschreiben eine Struktur bestehend aus zwei Entwicklungsabschnitten, wobei der erste Abschnitt zwei Iterationen und der zweite Abschnitt drei Iterationen enthält. Die Projektstrukturierung sieht keine weitere Hierarchisierung, z. B. für einzelne Entscheidungspunkte vor. Entscheidungspunkte modellieren wir als Work Items (Abschnitt 4.2.4). Über die Felder `System.AreaPath` und `System.IterationPath` können wir dann einzelne Entscheidungspunkt Work Items den jeweiligen Iterationen zuordnen.

Anmerkung: Prinzipiell erfasst unsere Modellierung auch Unteraufträge, die Auftraggeberabläufe auch in Auftragnehmerprojekten gestatten. Aufgrund der AG/AN-Schnittstelle sich die Zusammenhänge zwischen den Entscheidungspunkten gleich, wobei aufgrund der Modellierung des V-Modells sogar dieselben Entscheidungspunkttypen verwendet werden. Unsere Modellierung adressiert dieses Szenario zurzeit nicht explizit, sondern adressiert nur grob granulare Ite-

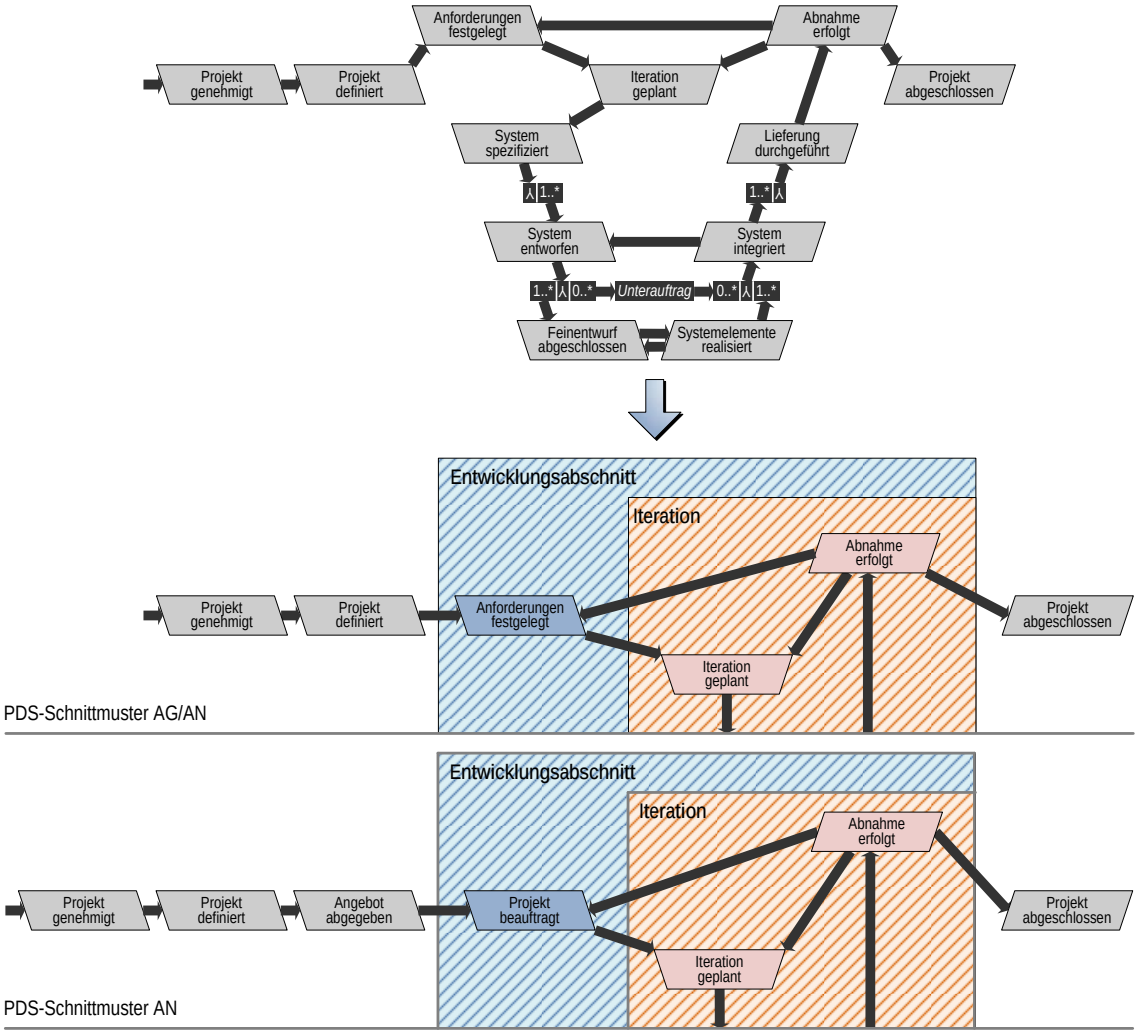


Abbildung 4.19.: Generatorschnittmuster für AG/AN-Projektdurchführungsstrategien

rationen, sodass wir an dieser Stelle die Empfehlung geben, Unteraufträge nicht in der initialen Planung vorzusehen, sondern diese direkt im TFS einzuplanen.

Projektstart und Projektabschluss

Neben den Entwicklungsabschnitten mit ihren Iterationen beinhaltet ein V-Modell-Projekt auch einen Projektstart und einen Projektabschluss. Analog zu den Entwicklungsabschnitten modellieren wir eine TFS-Iteration für den Projektstart, die als Zuweisungsziel für die Entscheidungspunkt Work Items der Entscheidungspunkte *Projekt genehmigt* und *Projekt definiert* dient. Analog verfahren wir mit dem Projektabschluss, der als Zuweisungsziel für das Entscheidungspunkt Work Item des Entscheidungspunkts *Projekt abgschlossen* dient. Je Projekt gibt es nur einen Projektstart und ein Projektende. Dazwischen wird mindestens ein Entwicklungsabschnitt mit mindestens einer Iteration erzeugt.

4.4. Werkzeug

Für CollabXT-TFS wurde über die Konzeption von Prozess-/Toolintegrationsverfahren hinaus eine Werkzeugentwicklung durchgeführt. Die Anforderungen an das Werkzeug beschreiben ein generatorbasiertes System, das die Kette der Referenzwerkzeuge des V-Modells weiterführt und eine weitere Stufe so einführt (Abschnitt 2.1), dass Ergebnisse des Tailorings direkt in andere

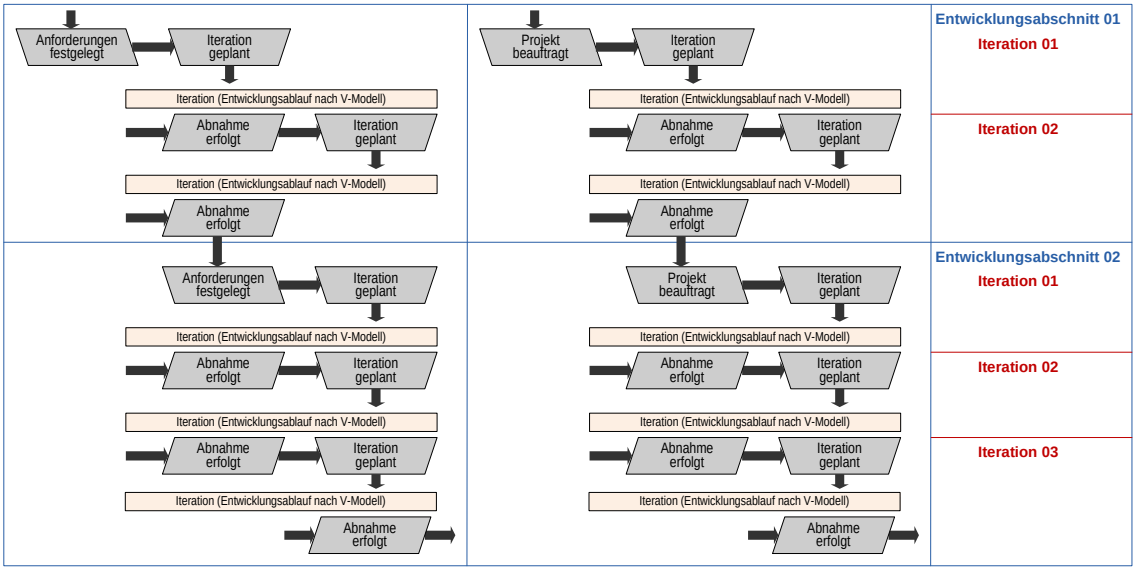


Abbildung 4.20.: Beispielhafte Abbildung von konkreten PDS-Abläufen auf Entwicklungsabschnitte und Iterationen

Umgebungen zu überführen und zu integrieren. Für TFS ist dafür ein template- und generatorbasiertes System gewählt worden, das das einerseits auf den Eingaben des V-Modells basiert, andererseits jedoch auch Daten mit einer zeitlich höheren Stabilität in einem TFS *Meta-Process Template* ablegt.

4.4.1. Konzept und Architektur des Werkzeugs

In Abbildung 4.21 ist das Grundkonzept des Generatorsystems in der V-Modell-Werkzeugkette zu sehen. Sie zeigt die Eingaben des Systems, die aus dem *Meta-Process Template* und den *V-Modell Inhalten* aus dem Tailoring bestehen. Der Generator füllt das Meta-Process Template mit den V-Modell Inhalten zu einem validen TFS-kompatiblen Process Template für ein V-Modell Projekt.

Achtung: An dieser Stelle ist es wichtig, noch einmal sprachlich präzise zu werden und das Ergebnis des Generators genauer zu charakterisieren. Genau genommen, erzeugt der Generator ein projektspezifisches Template, sondern ein *projekttypspezifisches Template*. Damit handelt es sich noch nicht um ein Projektmodell, sondern genau genommen immer noch um ein *organisationsspezifisches Vorgehensmodell*. Das V-Modell beschreibt mit seinem Variantenraum lediglich eine diskrete Einschränkung des Prozesses sowohl in Umfang als auch Struktur. Durch den CollabXT-Generator werden die Inhalte übernommen und spezifisch interpretiert. Das ändert jedoch nichts daran, dass auch weiterhin ein Rest Generik erhalten bleibt, da es sich hier im Wesentlichen um eine Modelltransformation auf einer Abstraktionsebene handelt.

Durch die Aufbewahrung im Process Template Manager finden wir hier ein Konzept, das sich nicht sehr von einem V-Modell-Projekttyp unterscheidet. Präzise gesprochen wird also aus einem TFS Process Template ein V-Modell-basiertes Projekt vom Typ des Templates instanziiert.

Adressaten. Der CollabXT-TFS-Generator ist ein Werkzeug, das von Prozessingenieuren und Projektleitern eingesetzt werden kann. Erstere können mithilfe dieses Werkzeugs ein V-Modell-basiertes Vorgehen in eine TFS-Struktur überführen und dort ggf. Anpassungen und Ausgestaltungen[6] vornehmen. Letztere können mithilfe des Werkzeugs ein projektspezifisch angepasstes V-Modell unmittelbar in den TFS einspielen und sofort instanziieren. Dabei werden die Anwender durch ein grafisches Werkzeug unterstützt, das in seiner ersten Stufe die Prozessstruktur anpasst und weiterhin in einer zweiten Stufe die Verknüpfung der Elemente des im TFS instanziierten Projekts vornimmt. Aufgrund der Ausrichtung auf das Visual Studio als Zielplattform gibt

6 Zu beachten bei der Ausgestaltung des Prozesses außerhalb der methodischen Möglichkeiten des V-Modells und seiner Standardwerkzeuge ist, dass die V-Modell-Konformität des resultierenden Vorgehens **nicht** konstruktiv sichergestellt werden kann.

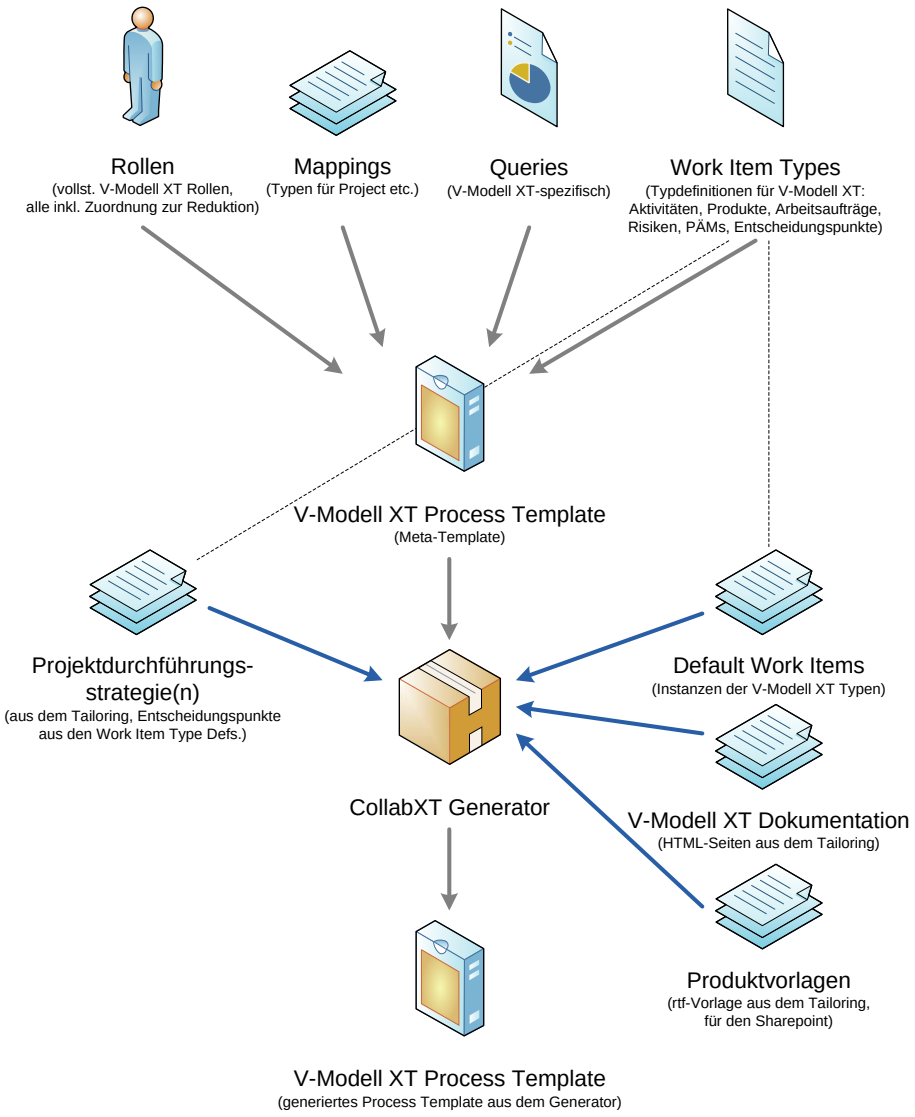


Abbildung 4.21.: Ein- und Ausgaben von CollabXT-TFS und Position in der Werkzeugkette

es Beschränkungen hinsichtlich der unterstützten Einsatzszenarios. Der CollabXT-TFS-Generator unterstützt nur Projekte mit Entwicklungsanteilen. Das sind V-Modell-Projekte, die auf den Projekttypen *Auftragnehmer (AN)* oder *Auftraggeber/Auftragnehmer (AG/AN)* basieren. Reine Auftraggeber (AG) Projekte oder Projekte zur Einführung und Anpassung eines Vorgehensmodells (Org) werden explizit nicht unterstützt. Hierfür sehen wir die SharePoint-Implementierung (Kapitel 3) als besser geeignet an. Der CollabXT-Generator unterstützt somit:

- Systementwicklungsprojekte AN
- Systementwicklungsprojekte AG/AN
- Angepasste V-Modell XT Derivate, die AN bzw. AG/AN erweitern.

Der Generator arbeitet für inhaltliche Anpassungen des V-Modells ebenso und unterstützt somit die Variationen des Tailorings transparent. Im Anhang A befinden sich die Referenzprofile für das Tailoring des Standard V-Modell XT, die durch den Generator unterstützt werden mitsamt der Ergebnisstrukturen für den Vergleich und die Abstimmung/Feststellung der Konformität des Generats zum originalen V-Modell.

4.4.2. Installation und Randbedingungen

Für die Installation der Software sind im Wesentlichen die Anforderungen der beteiligten Komponenten maßgebend. Prinzipiell ist das Gesamtsystem so entworfen, dass keine Eingriffe in beste-

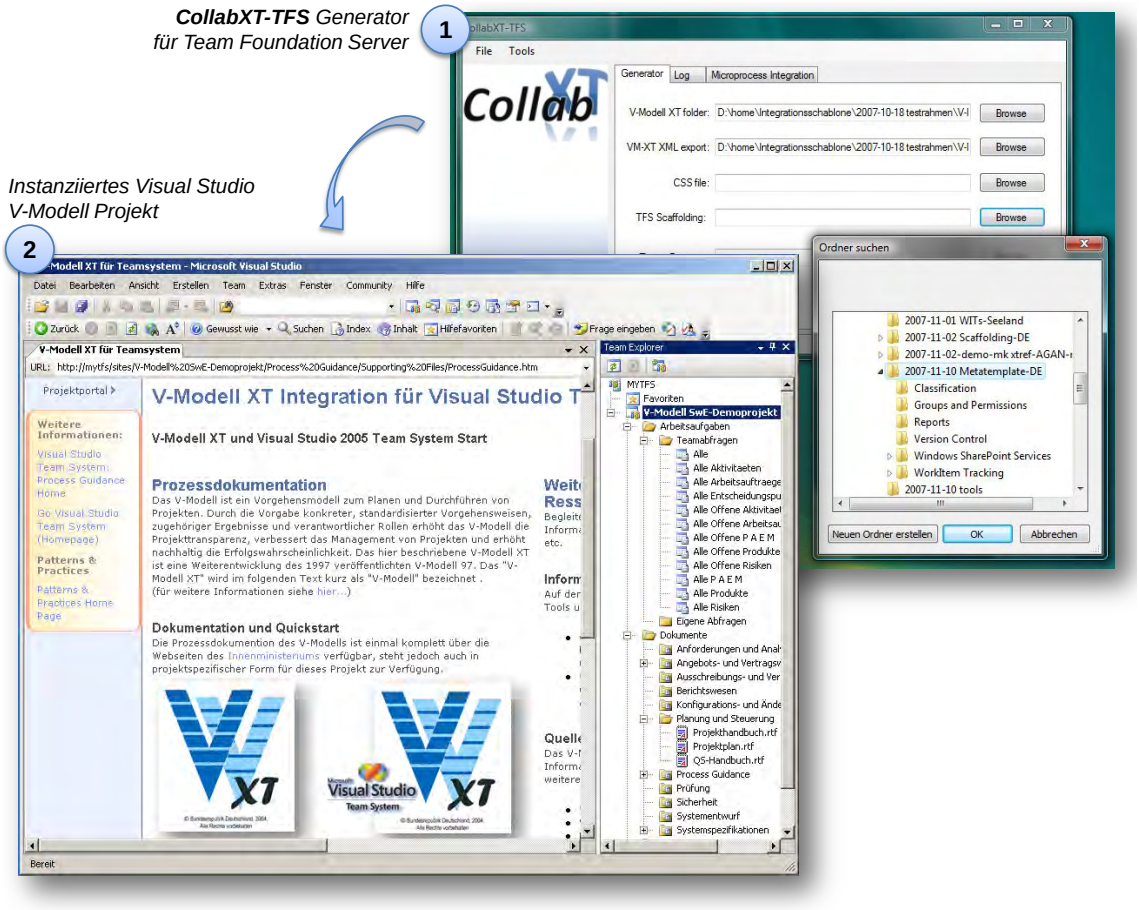


Abbildung 4.22.: Generator (1) und Visual Studio Startseite (2) für ein V-Modell-Projekt

hende Infrastrukturen notwendig sind. Da kaum eine Organisation die unkontrollierte Installation von Software auf Servern zulässt, ist die installationslose Verwendung eine der wichtigen Fähigkeiten der Werkzeuge. Sofern eine Installation von Serverkomponenten erforderlich ist, kann diese einmalig durch einen Administrator mit vergleichsweise hoher zeitlicher Stabilität erfolgen. Kurzfristige Softwarebereitstellungen sind also nur auf der Seite es Clients erforderlich. Folgende Software wird benötigt:

- Die V-Modell XT Werkzeuge (erhältlich via `http://www.v-modell-xt.de`) erfordern eine Java Laufzeitumgebung (1.5 oder besser).
- Für die CollabXT-TFS-Werkzeuge ist (erhältlich via `http://www.codeplex.com/VModellXTTFS`) eine .NET Laufzeitumgebung (2.0 oder besser) erforderlich.
- Ein Team Foundation Server ist installiert und im Netzwerk verfügbar (Sprache des Systems: Deutsch, Installationsstatus: Neu).

Die aktuell vorliegende Version der CollabXT-TFS-Werkzeuge adressiert ausschließlich Szenarios in denen ein deutschsprachiges V-Modell XT-Derivat in Verbindung mit deutschsprachigen TFS-Installationen. Die Verwendung anderer Sprachversionen ist möglich, erfordert ggf. aber Anpassungen hinsichtlich Im- und Exportskripten z. B. für die Installation des Sharepoint-Templates.

Achtung: Um Schwierigkeiten mit bereits etablierten Work Item Typen aus dem Weg zu gehen, empfehlen wir einen „frischen" Server zu installieren.

4.4.3. Beschreibung und Anwendung des Generators

Um mit den Werkzeugen von CollabXT-TFS zu arbeiten, wird ein projektspezifisches V-Modell XT benötigt. Dieses muss den Referenzprofilen genügen (Anhang A) oder eine organisationsspezifische Anpassung auf der Grundlage der Referenzprofile darstellen. Das Tailoring muss mit dem

Projektassistenten ausgeführt werden. Der Generator verfügt über eine einfache Oberfläche. Im Wesentlichen dient sie dazu, die Pfade zu den Ausgaben des Projektassistenten zu erfassen. Die einzugebenden Felder sind:

V-Modell XT folder Das ist der Ordner, der das projektspezifische V-Modell mit allen Exporten beinhaltet. Weitere Informationen hierzu im Anschluss.

VM-XT XML export Das ist das projektspezifsiche V-Modell im XML-Format.

CSS file Hier besteht die Möglichkeit, das Ausgabeformat für die Prozessdokumentation zu variieren und in Form einer CSS-Datei zu hinterlegen. Dieses wird dann im TFS für dieses Template mit hinterlegt.

TFS Scaffolding Hiermit ist das TFS Meta-Template gemeint, das z. B. die V-Modell XT Work Item Typen, die Queries, die Reports etc. enthält.

Target Directory Das ist das Ausgabeverzeichnis für das V-Modell-TFS Process Template.

Schritte 1 – 3

Der Generator durchläuft im Wesentlichen drei Schritte bis ein projektspezifisches V-Modell in ein gültiges TFS-Template überführt wurde. In Schritt 1 (in Abbildung 4.23 gezeigt) muss dem Generator das projektspezifische V-Modell-Exportverzeichnis bekannt gegeben werden.

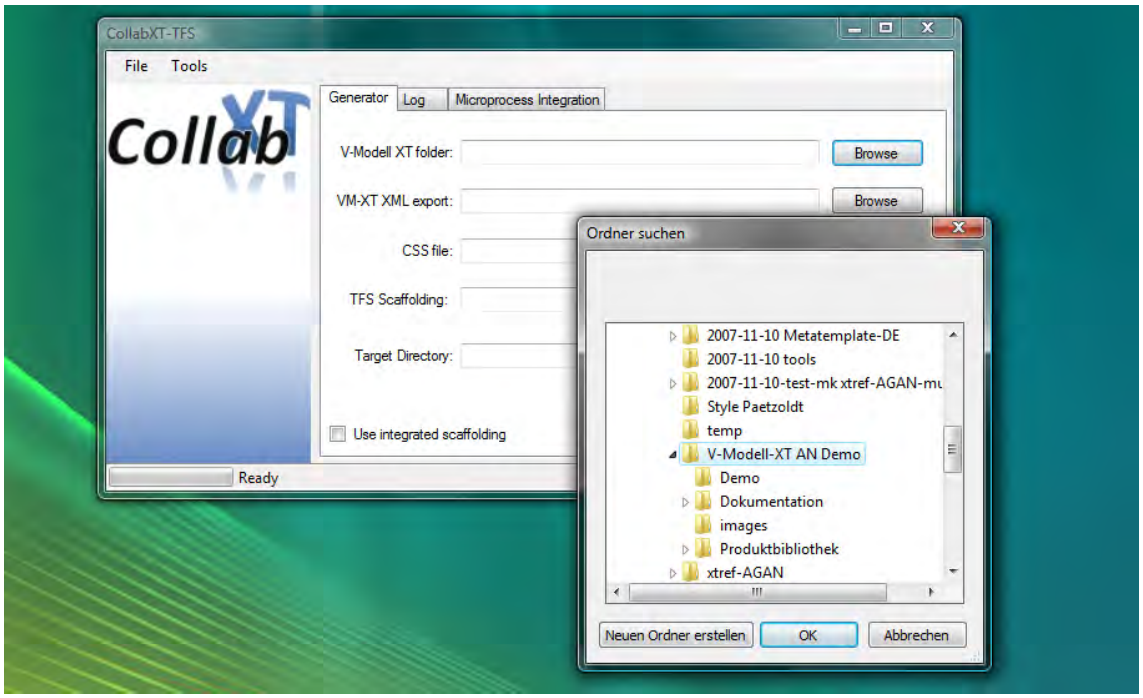


Abbildung 4.23.: Schritt 1: Auswählen des V-Modell-Exportverzeichnisses

Um das projektspezifische V-Modell gesammelt verarbeiten zu können, haben wir eine Verzeichnisexportstruktur definiert, die der Generator voraussetzt:

```
Projektverzeichnis\
        Dokumentation\
        Produktbibliothek\
        Projekt-V-Modell-XT.xml
```

Unter dem Verzeichnis `Projektverzeichnis` erwartet der Generator die Unterverzeichnisse `Dokumentation` und `Produktbibliothek`. In Ersterem wird der HTML-Export der Prozessdokumentation des projektspezifischen V-Modells abgelegt. Im anderen Verzeichnis wird die Produktvorlagenbibliothek des V-Modells exportiert. Die Inhalte dieser beiden Verzeichnisse übernimmt der Generator für die Erstellung der Inhalte für den TFS. Weiterhin empfehlen wir unter Projektverzeichnis auch den XML-Export des projektspezifischen V-Modells anzulegen. Neben der

XML-Datei erzeugt der Projektassistent daraufhin auch alle notwendigen Schemainformationen und kopiert alle benötigen Bilder in ein entsprechendes Unterverzeichnis (siehe Abbildung 4.24).

Achtung: Die Verzeichnisstruktur muss durch den Anwender von Hand angelegt werden. Die Auswertung erfolgt automatisch.

Achtung: Nachdem sämtliche Planungen im Projektassistenten abgeschlossen sind, muss die *Projektdatei* des Projektassistenten gespeichert werden. Diese und das projektspezifische V-Modell im XML-Format sind die Komponenten, die durch den TFS-Generator zur Ermittlung des Projektplans benötigt werden. Ein expliziter Export des Projektplans durch den Projektassistenten ist nicht nötig.

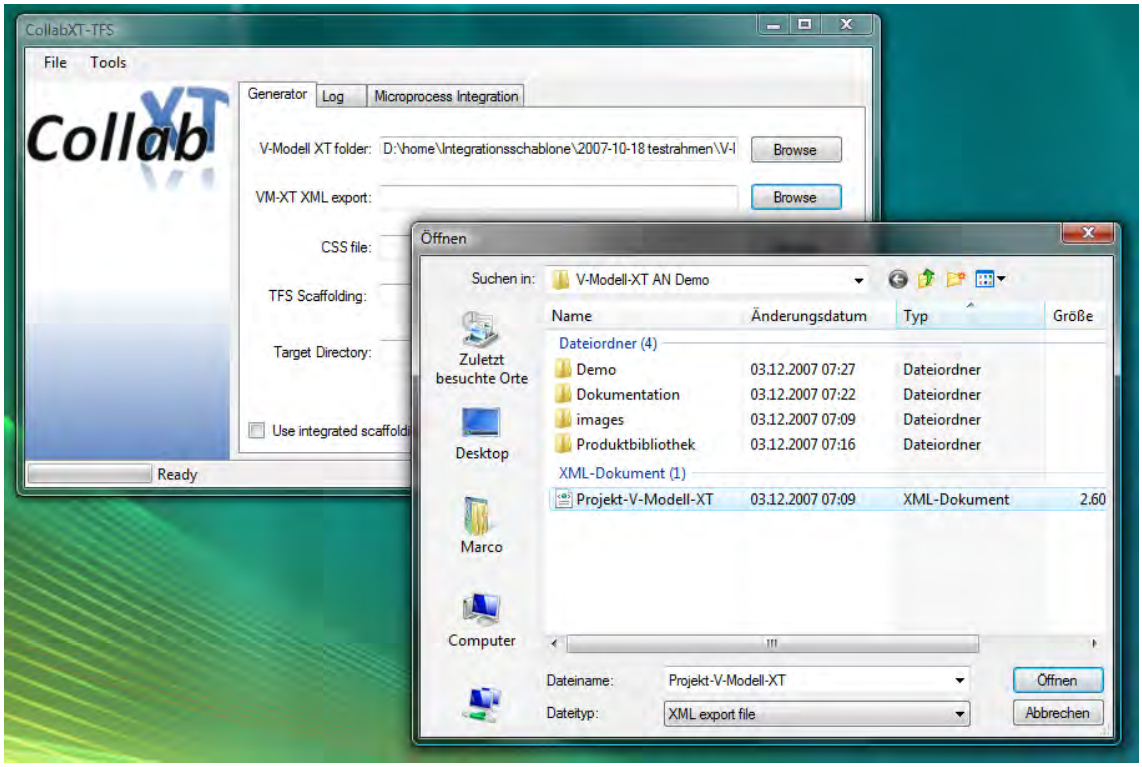


Abbildung 4.24.: Schritt 2: Auswählen des projektspezifischen V-Modells in XML

Das TFS Meta-Template liegt analog zum V-Modell als Verzeichnisstruktur vor. Das Meta-Template (gelegentlich als *Scaffolding* bezeichnet) muss verfügbar sein, da der Generator die Inhalte mit dem projektspezifischen V-Modell verrechnet (vgl. Abbildung 4.21). Optional ist dabei jedoch die (separate) Angabe einer CSS-Datei. Sofern keine explizite CSS-Datei angegeben wird, wird ein Standardformat verwendet. Dieses unterscheidet sich in sofern vom Standardformat des V-Modells durch geänderte und für die Bildschirmausgabe optimierte Schriftstile.

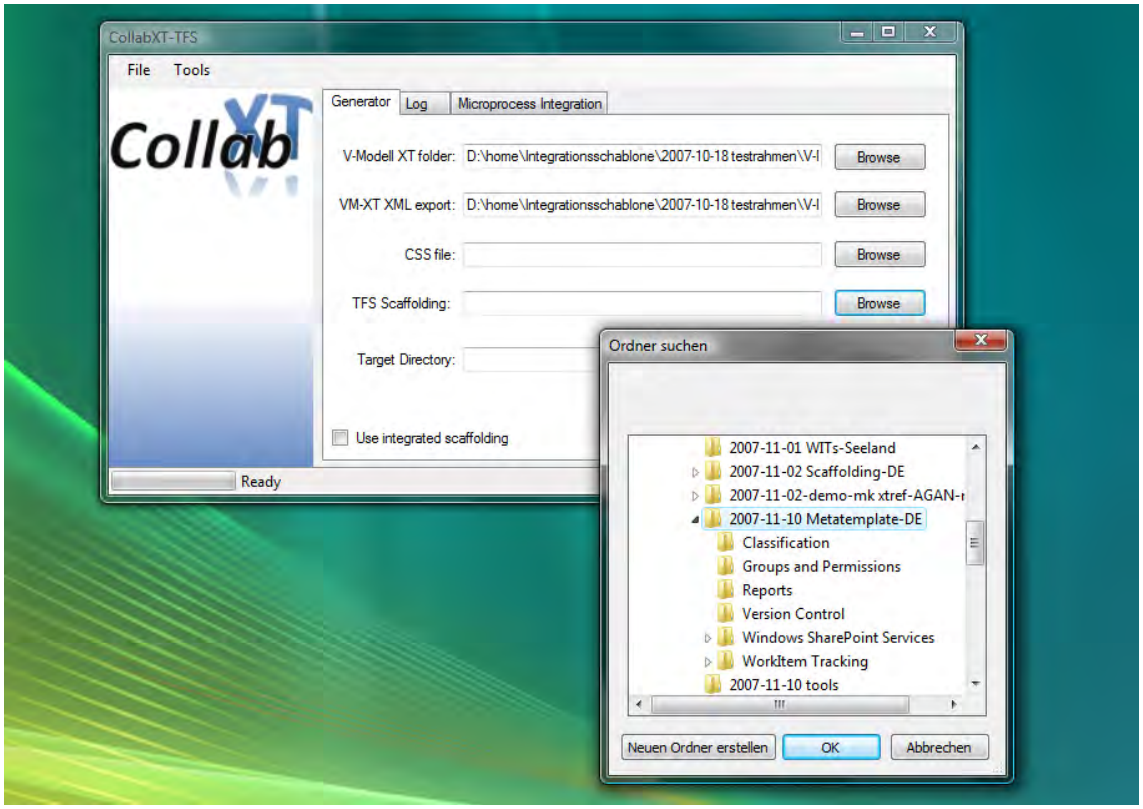


Abbildung 4.25.: Schritt 3: Auswählen des TFS Meta-Process Template

Ergebnisse des Generators im Überblick

Nachdem alle Informationen und Eingabedaten vorliegen, kann der Generator gestartet werden. Der Vorgang kann ein wenig Zeit in Anspruch nehmen – je nachdem, wie komplex das zu verarbeitende Tailoring ist. Als Ergebnis wird durch den Generator ein valides TFS-Template auf der Basis des projektspezifischen V-Modells erzeugt. Dieses Template kann im *Process Template Editor* des Visual Studio (Abbildung 4.26, als Teil der TFS Power Toys/des SDK) betrachtet werden. Für einen ausführlicheren Rundgang verweisen wir auf die Demonstration im Web[7]. Wir betrachten nur kurz die grundlegende Struktur.

Durch den Generator wird das Work Item Tracking System angesteuert. Hierzu werden auf der Basis der durch das Metatemplate bereitgestellten Work Item Types und der verfügbaren Planungsinformationen aus dem V-Modell initiale (Default) Work Items erstellt. Dies umfasst im Wesentlichen:

- Entscheidungspunkte
- Aktivitäten
- Produkte (teilw.)

Auf der Basis des initialen Projektplans wird die Projektstruktur im TFS auch entsprechend Abschnitt 4.3.4 konfiguriert. Abbildung 4.26 zeigt die Startseite mit der Strukturansicht des generierten Process Template im linken Teil der Abbildung. Hierbei handelt es sich um die Sektionen und Gruppen, die wir in Abschnitt 4.1.2 beschrieben haben.

Wir vertiefen einen Aspekt: Wählt man in der Strukturansicht unter dem Knoten *Work Item Tracking* einen der aufgeführten Work Item Typen aus, wird dieser durch den PT-Editor zur Bearbeitung geöffnet. Abbildung 4.27 zeigt den Workflow des Work Item Typs *Produkt* (vgl. Abschnitt 4.2.2, Abbildung 4.8). Die modellierten Zustandsautomaten werden im Workflow-Designer mithilfe von *States* und *Transitions* grafisch modelliert.

7 CollabXT-Projektseite: `http://www.collabxt.de`

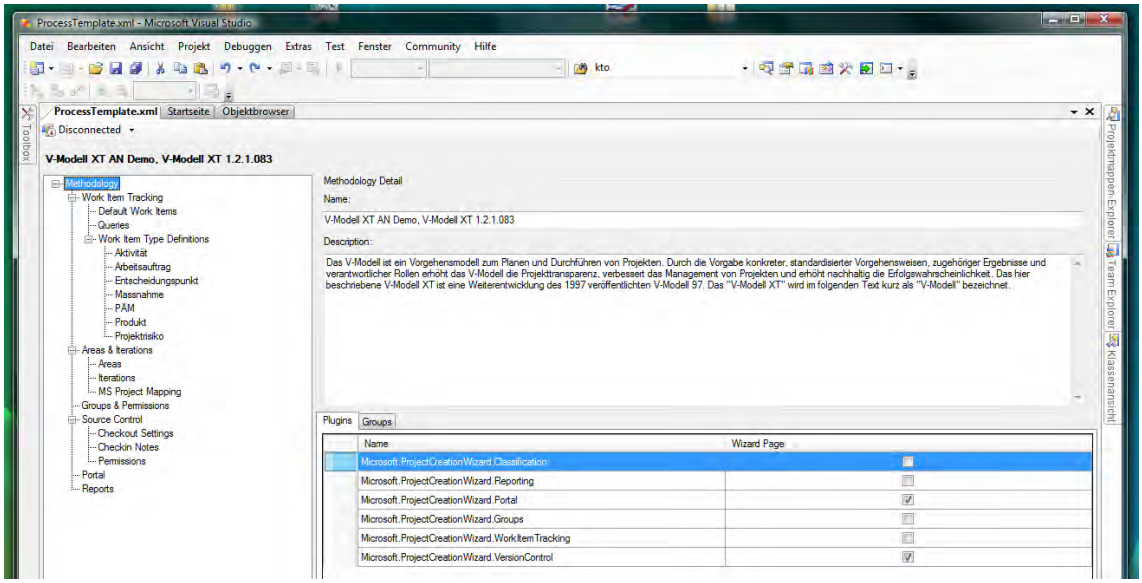


Abbildung 4.26.: Generiertes Process Template im PT-Editor (Visual Studio)

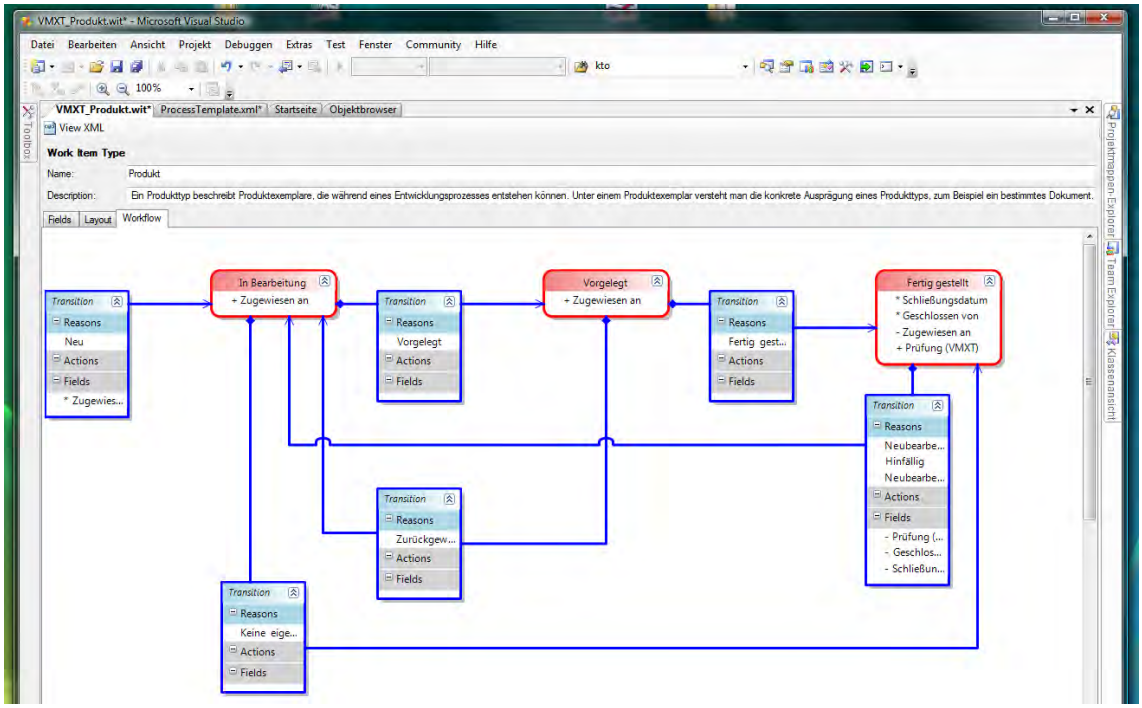


Abbildung 4.27.: Workflow des Produkt Work Item Typs im PT-Editor

5. Zusammenfassung

In diesem Bericht haben wir die Ergebnisse des Projekts CollabXT zusammengefasst. Wir haben dabei beide Teilprojekte erfasst und alle wesentlichen Entscheidungen und Modellierungen dokumentiert. Im Kapitel 2 haben wir grundlegende Überlegungen zur Operationalisierung des V-Modell XT angestellt. Kapitel 3 geht detailliert auf das erste Teilprojekt *CollabXT-SP* ein; Kapitel 4 auf das zweite Teilprojekt *CollabXT-TFS*. Die Motivation für dieses Projekt ist die Notwendigkeit der Operationalisierung, also der Ausführungsunterstützung, von Vorgehensmodellen im Allgemeinen und dem V-Modell XT im Speziellen. Mit den gerade genannten und in diesem Bericht vorgestellten Werkzeugen adressieren wir diese Problematik und stellen Lösungskonzepte vor, mit denen sich die Lücke zwischen Prozessen und konkreten Projekten überbrücken lässt.

Aktueller Stand und offene Punkte

Voraussetzung für die Anwendung des Konzepts und der Tools ist ein Metamodell-basiertes Vorgehensmodell – in diesem Stand das V-Modell XT. Als zweite Voraussetzung ist ein programmierbares Werkzeug erforderlich. Mit *CollabXT* wird eine Brücke zwischen Prozessmodell und Werkzeugen geschaffen. Mit den aktuellen, vorliegenden Versionen der Werkzeuge sind neben dem prinzipiellen Umsetzbarkeitsnachweis zum Teil auch produktionsnahe Versionen entstanden. Dennoch sind viele offene Punkte, auch für zukünftige Vorhaben, zu benennen:

Philosophie: Die Operationalisierung eines Prozesses erfordert die detaillierte Betrachtung von Aktivtäts- und Vorgangsstrukturen. Das V-Modell XT verfolgt eine *produktzentrierte* Sichtweise auf ein Projekt. Damit unterscheidet sich das V-Modell XT von anderen etablierten Prozessen. Für *CollabXT* ist dies zu berücksichtigen. Dies hat weit reichende Konsequenzen hinsichtlich Strukturen, die von den Werkzeugen erwartet, vom V-Modell XT jedoch nicht definiert werden.

Verfahren mit abstrakten Prozessen: Aktivitäts- und Prozessbeschreibungen sind im V-Modell XT – sofern vorhanden – nur generisch und sehr unscharf hinterlegt. Umfassendere Prozesse sind darüber hinaus auch auf mehrere V-Modell-Elemente verteilt. Für eine konkrete Werkzeugunterstützung müssen verschiedene Prozesse aufgenommen, konkretisiert und ausgestaltet werden. Abstrakte und/oder allgemeine Prozesse sind hier zu interpretieren.

Einhaltung von Konformitätsanforderungen: Insbesondere im Kontext zertifizierter Prozesse sind Konformitätsanforderungen wichtig. Für das V-Modell XT ist zum Zeitpunkt der Erstellung dieses Berichts ein entsprechendes Programm inklusive Zertifizierungsverfahren in Arbeit. Das Konformitätsprogramm unterscheidet dabei eine *konstruktive* und eine *analytische* Konformität. Letztere wird im Rahmen eines Assessments bescheinigt (vgl. CMMI, [Kne06]). Konstruktive Konformität hingegen beschreibt eine Prozessmodelleigenschaft. Prozesse, die mit den Mitteln und auf der Grundlage eines V-Modell-Referenzmodells erstellt wurden, sind per Definition (konstruktiv) konform zum V-Modell XT. Mit *CollabXT* nehmen wir ebenfalls eine Anpassung des V-Modell XT vor. Dem entsprechend müssen wir uns auch zur Konformität positionieren:

- **CollabXT-SP:** Für CollabXT-SP gelten prinzipiell die Anforderungen der konstruktiven Konformität. Gründe hierfür sind, dass ausschließlich das V-Modell XT inklusive seines Metamodells verwendet werden und alle V-Modell-Derivate verwendet werden können. Eine Anpassung des V-Modell XT, das der Projektassistent erzeugt, ist hier nicht vorgesehen. Alle Informationen des Modells mit dem Ergebnisstand Post-Tailoring bleiben erhalten. Die Strukturen des SharePoint-Portals werden auf der Grundlage des projektspezifischen V-Modells erstellt und gefüllt.
- **CollabXT-TFS (1):** Für CollabXT-TFS ist eine Fallunterscheidung durchzuführen. Fall 1 beschreibt einen änderungsfreien Durchlauf. Das bedeutet, dass die Eingabe für den Generator direkt dem Projektassistenten entstammt und unmittelbar nach der Erzeugung in einen TFS überspielt wird. Gründe: Hierfür gilt das Lebenszyklusmodell des V-Modell XT, das heißt, Anpassungen des Modells werden ausschließlich mithilfe des

V-Modell XT Editors erstellt. In diesem Fall sind die Vorbedingungen für die konstruktive Konformität erfüllt. Alle Informationen bleiben hierbei konsistent erhalten. Das Projekt wird anhand des projektspezifischen V-Modells strukturiert.

CollabXT-TFS (2): Der zweite Fall für CollabXT-TFS liegt dann vor, wenn (wie in Kapitel 2.4.3 beschrieben) nach oder während des Generierens des Process Template das projektspezifische V-Modell im Rahmen des TFS-Lebenszyklusmodells weiter angepasst wird. Gründe: Als Eingabe für den Generator stehen die Exporte des Projektassistenten zur Verfügung. Weitere Anpassungen nach oder während der Generierung verletzen, sofern sie nicht in das Ausgangsmodell zurückgerechnet werden, die Konsistenz des Modells. Die Konformität zum V-Modell XT ist weiterhin möglich, kann aber *ausschließlich* analytisch per Assessment erfolgen.

Lokalisierung: Das V-Modell XT existiert sowohl in einer deutschen wie auch in einer englischen Version. Die Unterstützung beider Sprachen ist noch nicht abschließend und erschöpfend gelöst. Für CollabXT-SP ist die Sprache des verwendeten V-Modells transparent. Lediglich der verwendete SharePoint Server sollte synchron zur verwendeten Sprache sein (oder zumindest ein entsprechendes Template anbieten), um ein abgestimmtes Gesamtbild zu geben. Ansonsten erhält der Anwender aus einem deutschen V-Modell XT ein deutsches Portal und aus einem englischen V-Modell XT ein englisches Portal.

Für TFS ist eine Transparenz hinsichtlich der verwendeten Sprache nicht automatisch gegeben. Hier spielen verschiedene Randbedingungen der TFS-Plattform eine Rolle. Im Anhang B.3 sind wir darauf bereit kurz eingegangen.

Abbildung der Prozessstruktur: Das V-Modell XT bildet deklarativ verschiedene Prozessstrukturen ab. Diese sind zurzeit nur aufwändig abzubilden und auch nur zum Zeitpunkt der Generierung. Zur Laufzeit des Projekts sind diese Strukturinformationen nicht mehr direkt abrufbar und benötigen (sowohl für SharePoint als auch TFS) separate Assistenzsysteme.

Beim TFS stellt sich diese Situation bereits zum Zeitpunkt der Generierung als kompliziert dar: Konkrete Elementverknüpfungen kann der TFS erst zur Laufzeit herstellen. Deklarativ können solche Informationen nicht ausgewertet werden. Für die Verknüpfung sind daher in den Work Items separate Metadaten zu hinterlegen (vgl. Kapitel 4.2).

Ausblick

Die aktuell vorliegenden Ergebnisse aus *CollabXT* motivieren weitere Arbeiten in folgenden Bereichen:

Prozessassistenzsysteme: Die Einhaltung der Vorgaben des Prozesses zur Laufzeit und nach der Generierung erfordert weitere, zusätzliche Logik auf der Ebene der Werkzeuge. Moderne Assistenzsysteme sind hier für die weitergehende Unterstützung erforderlich.

Andere Vorgehensmodelle: Neben dem V-Modell XT gibt es noch weitere Vorgehensmodelle, die auf einem Metamodell (SPEM/UMA etc.) aufbauen. Auch diese Prozesse sind geeignet, um in *CollabXT* verwendet zu werden.

Andere Werkzeuge: Neben SharePoint Server und Team Foundation Server sind auch weitere Werkzeuge relevant im Kontext *CollabXT*. Optionen sind z. B. IBM Jazz, Polarion etc.

Transformationskalkül: Trotz der beiden Werkzeuge hat *CollabXT* noch viele Eigenschaften einer Individuallösung. Die Entwicklung eines Transformationskalküls zur Übersetzung von Vorgehensmodellen auf Werkzeuge erscheint hier erforderlich. Ein passendes System vereinfacht unter anderem auch die Erarbeitung von Lösungen für die beiden gerade aufgezählten Punkte der Variation hinsichtlich Vorgehensmodellen und Werkzeugen.

Zusammenfassend stellt *CollabXT* eine einfache Option dar, insbesondere die Werkzeugkette des V-Modell XT transparent für den Anwender zu erweitern. Als zusätzliche Tailoringstufe wird das V-Modell XT auf ein Werkzeug überführt und stellt Anwendern eine V-Modell XTUnterstützung in ihrer gewohnten Umgebung zur Verfügung.

A. Referenzprofile für CollabXT-TFS

In diesem Anhang sind die Referenzprofile für die Abbildung des V-Modell XT auf die Visual Studio Umgebung. Die Referenzprofile umfassen die Tailoring- bzw. Anwendungsprofile für den Projektassistenten. Die Profile beschreiben dabei einen „Maximalausbau", d. h. Teilmengen von Vorgehensbausteinen und Projektdurchführungsstrategien werden ebenfalls unterstützt.

A.1. Profil: Systementwicklung (AN)

Das Profil für (einfache) Auftragnehmer berücksichtigt ausschließlich Entwicklungsprojekte ohne eigenständige Anforderungsfestlegung. Die Vergabe von Unteraufträgen wird aber durch dieses Profil mit abgedeckt.

Eigenschaft	Wert(e)
Bestimmende Projektmerkmale	Projekttyp: Systementwicklungsprojekt (AN) Projektgegenstand: SW-System Projektrolle: AN mit Unterauftragnehmern
Verpflichtende Vorgehensbausteine	Projektmanagement Qualitätssicherung Konfigurationsmanagement Problem- und Änderungsmanagement Lieferung und Abnahme (AN) Vertragsschluss (AN) Systemerstellung
Optionale Vorgehensbausteine (gewählt)	Lieferung und Abnahme (AG) Vertragsschluss (AG) Evaluierung von Fertigprodukten Systemsicherheit Kaufmännisches Projektmanagement Messung und Analyse SW-Entwicklung Benutzbarkeit und Ergonomie Weiterentwicklung und Migration von Altsystemen
Mögliche Projektdurchführungsstrategien	Inkrementelle Systementwicklung (AN) Agile Systementwicklung (AN) Komponentenorientierte Systementwicklung (AN) Wartung und Pflege von Systemen (AN)

Tabelle A.1.: Referenzprofil für Systementwicklung (AN)

A.1.1. Produkt- und Aktivitätslisten AN

Das Tailoring erzeugt folgende Produkt- und Aktivitätslisten. Die Produkttypen beschreiben die erwarteten Ergebnistypen des projektspezifischen V-Modells. Die Aktivitätslisten beschreiben die zu erbringenden Vorgängen. Die Produkttypen werden für die Erzeugung der Vorlagen in der Produktbibliothek benötigt; die Aktivitäten zur Erzeugung von Planungsgrößen zur Ableitung der Projektpläne. Die genaue Beschreibung der vorliegenden Produkt- und Aktivitätstypen ist der V-Modell XT Dokumentation zu entnehmen.

Neben den Vorgehensbausteinen und den daraus resultierenden Produkt- und Aktivitätslisten sind folgende Projektdurchführungsstrategien möglich, wobei drei davon ausgezeichnete Entwicklungsstrategien für die Neuentwicklung von Softwaresystemen sind:

- Inkrementelle Systementwicklung (AN)
- Agile Systementwicklung (AN)
- Komponentenorientierte Systementwicklung (AN)
- Wartung und Pflege von Systemen (AN)

Die Projektdurchführungsstrategien für den Projekttyp dieses Szenarios sind miteinander kombinierbar. Die Kopplung erfolgt über einen „gemeinsamen" Entscheidungspunkt *Iteration geplant*, der eine definierte Schnittstelle zwischen den verschiedenen Vorgehensweisen beschreibt.

A.1.2. Inkrementelle Systementwicklung (AN)

Die inkrementelle Systementwicklung ist das bereits aus dem V-Modell 97 bekannte Standardvorgehen. Ausgehend von einer Iterationsplanung wird über schrittweise Verfeinerung ein Gesamtsystem über mehrere Kompositionsstufen auf seine Komponenten und Module herunter gebrochen. Das Projekt strukturiert sich hier in die drei Phasen:

- Initialisierung, Angebot und Beauftragung
- Systementwicklung
- Projektabschluss

Die inkrementelle Systementwicklung sieht die Vergabe von Unteraufträgen vor. Diese können entweder auf ausgelagerte Entwicklungsprojekte oder sonstige Einkaufs- und Dienstleistungsanteile abgebildet werden. Die Möglichkeit zur Unterbeauftragung ist in allen verfügbaren Projektdurchführungsstrategien gegeben und durch das spezifizierte Szenario unterstützt.

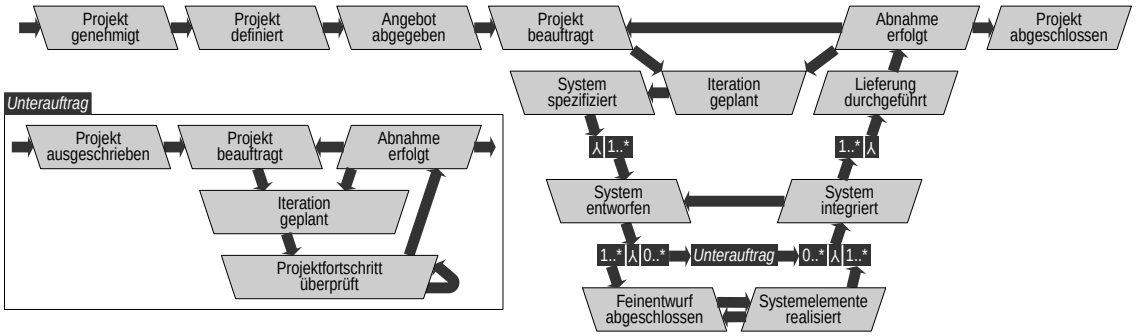


Abbildung A.1.: Inkrementelle Systementwicklung (AN)

A.1.3. Agile Systementwicklung (AN)

Die agile Projektdurchführungsstrategie bildet einen Prototyping-basierten Ansatz ab. Das Projekt strukturiert sich auch hier in die drei Phasen:

- Initialisierung, Angebot und Beauftragung
- Systementwicklung
- Projektabschluss

Diese Projektdurchführungsstrategie eignet sich bspw. für UI-lastige Anwendungen, in denen Anwender schnell Feedback geben können, ohne dass seitens der Entwicklung unverhältnismäßig hohe Dokumentations- oder Spezifikationsaufwände entstehen.

Die agile Systementwicklung sieht die Vergabe von Unteraufträgen vor. Diese können entweder auf ausge-lagerte Entwicklungsprojekte oder sonstige Einkaufs- und Dienstleistungsanteile abgebildet werden. Die Möglichkeit zur Unterbeauftragung ist in allen verfügbaren Projektdurchführungsstrategien gegeben und durch das spezifizierte Szenario unterstützt.

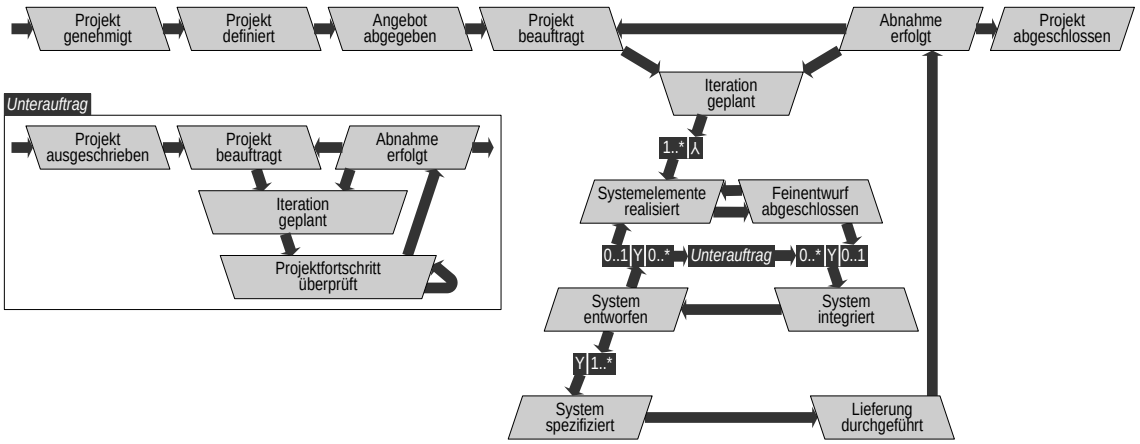


Abbildung A.2.: Agile Systementwicklung (AN)

A.1.4. Komponentenorientierte Systementwicklung (AN)

Die komponentenorientierte Systementwicklung entspricht von der Grundstruktur im Wesentlichen der Inkrementellen Systementwicklung (AN) (Abschnitt A.1.2). Das Projekt strukturiert sich auch hier in die drei Phasen:

- Initialisierung, Angebot und Beauftragung
- Systementwicklung
- Projektabschluss

Anders als die inkrementelle Projektdurchführungsstrategie wird hier die explizite Entwicklung – das Coden – in der Regel auf das Erstellen von Gluecode für vorhandene Komponenten abgebildet. Das Vorgehen setzt auf das Vorhandensein von Komponenten, die zu einem System integriert werden. Komponenten sind hier grob granulare Einheiten, die entweder schon erstellt und in einem Repository vorhanden sind oder durch den Zukauf von Fertigkomponenten im System verfügbar gemacht werden.

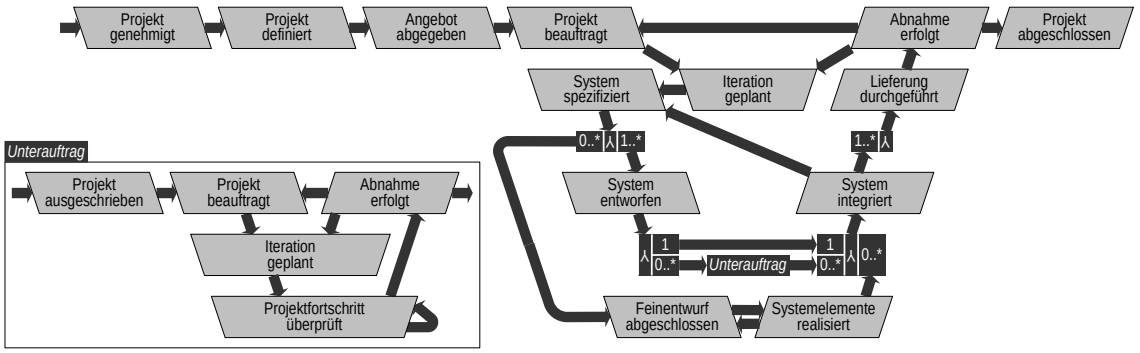


Abbildung A.3.: Komponentenorientierte Systementwicklung (AN)

Die komponentenorientierte Systementwicklung sieht die Vergabe von Unteraufträgen vor. Diese können entweder auf ausgelagerte Entwicklungsprojekte oder sonstige Einkaufs- und Dienstleistungsanteile abgebildet werden. Die Möglichkeit zur Unterbeauftragung ist in allen verfügbaren Projektdurchführungsstrategien gegeben und durch das spezifizierte Szenario unterstützt.

A.1.5. Wartung und Pflege von Systemen (AN)

Wartung und Pflege ist nicht den eigentlichen Entwicklungsvorgehensweisen zuzuordnen. Hier sind nach der Iterationsplanung verschiedene Aufsetzpunkte möglich, je nach Größe, Umfang

oder Tragweite der umzusetzenden Änderung. Das Projekt strukturiert sich auch hier in die drei Phasen:

- Initialisierung, Angebot und Beauftragung
- (punktuelle, änderungsgetriebene) Systementwicklung
- Projektabschluss

Wartung und Pflege sieht die Vergabe von Unteraufträgen vor. Diese können entweder auf ausgelagerte Entwicklungsprojekte oder sonstige Einkaufs- und Dienstleistungsanteile abgebildet werden. Die Möglichkeit zur Unterbeauftragung ist in allen verfügbaren Projektdurchführungsstrategien gegeben und durch das spezifizierte Szenario unterstützt.

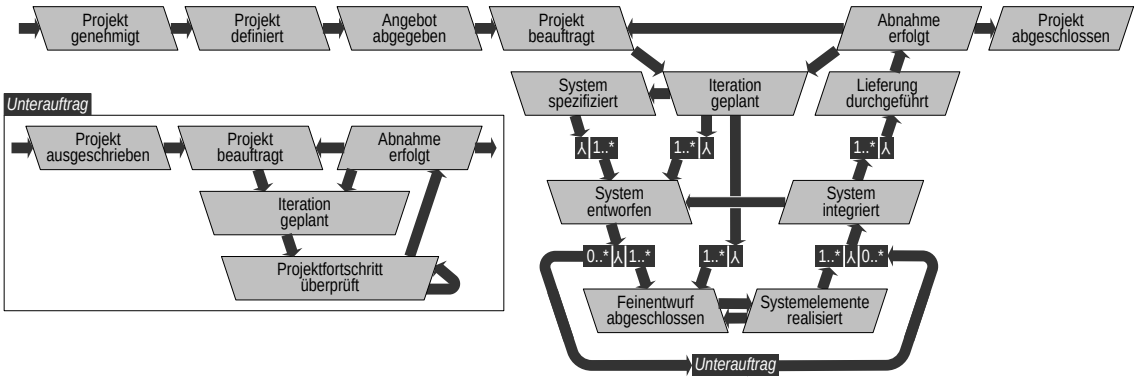


Abbildung A.4.: Wartung und Pflege von Systemen (AN)

A.2. Profil: Systementwicklung (AG/AN)

Das Profil für Auftraggeber/Auftragnehmerprojekttypen deckt analog zum reinen Auftragnehmerprojekt im Wesentlichen Entwicklungsprojekte ab. Anders als beim reinen Auftragnehmerprojekt sind hier jedoch keine Ausschreibungs- und Vergabeprozesse zu durchlaufen. Dafür findet eine integrierte Anforderungsfeststellung statt. Auch hier ist die Einbindung von Unterauftragnehmern vorgesehen.

A.2.1. Produkt- und Aktivitätslisten AG/AN

Das Tailoring erzeugt folgende Produkt- und Aktivitätslisten. Die Produkttypen beschreiben die erwarteten Ergebnistypen des projektspezifischen V-Modells. Die Aktivitätslisten beschreiben die zu erbringenden Vorgängen. Die Produkttypen werden für die Erzeugung der Vorlagen in der Produktbibliothek benötigt; die Aktivitäten zur Erzeugung von Planungsgrößen zur Ableitung der Projektpläne. Die genaue Beschreibung der vorliegenden Produkt- und Aktivitätstypen ist der V-Modell XT Dokumentation zu entnehmen.

[[Tabelle mit Produkten und Aktivitäten aus Excel-Importieren]]

Neben den Vorgehensbausteinen und den daraus resultierenden Produkt- und Aktivitätslisten sind folgende Projektdurchführungsstrategien möglich, wobei drei davon ausgezeichnete Entwicklungsstrategien für die Neuentwicklung von Softwaresystemen sind:

- Inkrementelle Systementwicklung (AG/AN)
- Agile Systementwicklung (AG/AN)
- Komponentenorientierte Systementwicklung (AG/AN)
- Wartung und Pflege von Systemen (AG/AN)

Die Projektdurchführungsstrategien für den Projekttyp dieses Szenarios sind miteinander kombinierbar. Die Kopplung erfolgt über einen „gemeinsamen" Entscheidungspunkt *Iteration geplant*, der eine definierte Schnittstelle zwischen den verschiedenen Vorgehensweisen beschreibt.

Eigenschaft	Wert(e)
Bestimmende Projektmerkmale	Projekttyp: Systementwicklungsprojekt (AG/AN) Projektgegenstand: SW-System Projektrolle: AG/AN mit Unterauftragnehmern
Verpflichtende Vorgehensbausteine	Projektmanagement Qualitätssicherung Konfigurationsmanagement Problem- und Änderungsmanagement Lieferung und Abnahme (AN) Lieferung und Abnahme (AG) Anforderungsfestlegung Systemerstellung
Optionale Vorgehensbausteine (gewählt)	Vertragsschluss (AG) Evaluierung von Fertigprodukten Systemsicherheit Kaufmännisches Projektmanagement Messung und Analyse SW-Entwicklung Benutzbarkeit und Ergonomie Weiterentwicklung und Migration von Altsystemen
Mögliche Projektdurchführungsstrategien	Inkrementelle Systementwicklung (AG/AN) Agile Systementwicklung (AG/AN) Komponentenorientierte Systementwicklung (AG/AN) Wartung und Pflege von Systemen (AG/AN)

Tabelle A.2.: Referenzprofil für Systementwicklung (AG/AN)

A.2.2. Inkrementelle Systementwicklung (AG/AN)

Die inkrementelle Systementwicklung ist das bereits aus dem V-Modell 97 bekannte Standardvorgehen. Ausgehend von einer Iterationsplanung wird über schrittweise Verfeinerung ein Gesamtsystem über mehrere Kompositionsstufen auf seine Komponenten und Module herunter gebrochen. Das Projekt strukturiert sich hier in die drei Phasen:

- Initialisierung und Anforderungsfestlegung
- Systementwicklung
- Projektabschluss

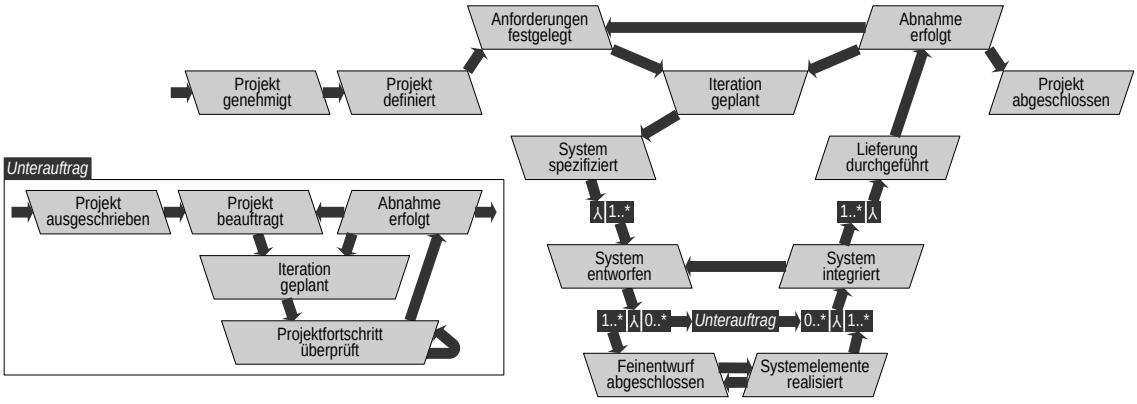


Abbildung A.5.: Inkrementelle Systementwicklung (AN)

Die inkrementelle Systementwicklung sieht die Vergabe von Unteraufträgen vor. Diese können entweder auf ausgelagerte Entwicklungsprojekte oder sonstige Einkaufs- und Dienstleistungsan-

teile abgebildet werden. Die Möglichkeit zur Unterbeauftragung ist in allen verfügbaren Projektdurchführungsstrategien gegeben und durch das spezifizierte Szenario unterstützt.

A.2.3. Agile Systementwicklung (AG/AN)

Die agile Projektdurchführungsstrategie bildet einen Prototyping-basierten Ansatz ab. Das Projekt strukturiert sich auch hier in die drei Phasen:

- Initialisierung und Anforderungsfestlegung
- Systementwicklung
- Projektabschluss

Diese Projektdurchführungsstrategie eignet sich bspw. für UI-lastige Anwendungen, in denen Anwender schnell Feedback geben können, ohne dass seitens der Entwicklung unverhältnismäßig hohe Dokumentations- oder Spezifikationsaufwände entstehen.

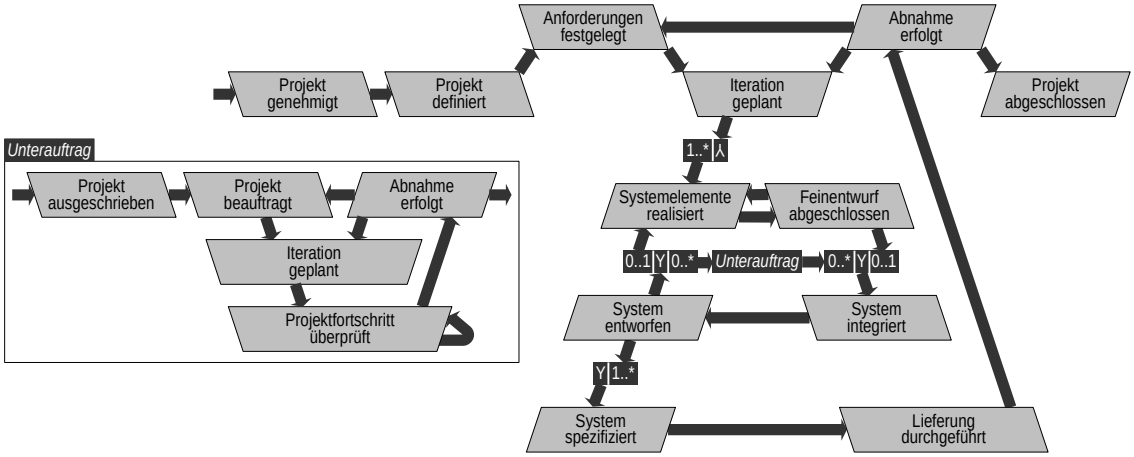


Abbildung A.6.: Agile Systementwicklung (AG/AN)

Die agile Systementwicklung sieht die Vergabe von Unteraufträgen vor. Diese können entweder auf ausgelagerte Entwicklungsprojekte oder sonstige Einkaufs- und Dienstleistungsanteile abgebildet werden. Die Möglichkeit zur Unterbeauftragung ist in allen verfügbaren Projektdurchführungsstrategien gegeben und durch das spezifizierte Szenario unterstützt.

A.2.4. Komponentenorientierte Systementwicklung (AG/AN)

Die komponentenorientierte Systementwicklung entspricht von der Grundstruktur im Wesentlichen der Inkrementellen Systementwicklung (AG/AN) (Abschnitt A.2.2). Das Projekt strukturiert sich auch hier in die drei Phasen:

- Initialisierung und Anforderungsfestlegung
- Systementwicklung
- Projektabschluss

Anders als die inkrementelle Projektdurchführungsstrategie wird hier die explizite Entwicklung – das Coden – in der Regel auf das Erstellen von Gluecode für vorhandene Komponenten abgebildet. Das Vorgehen setzt auf das Vorhandensein von Komponenten, die zu einem System integriert werden. Komponenten sind hier grob granulare Einheiten, die entweder schon erstellt und in einem Repository vorhanden sind oder durch den Zukauf von Fertigkomponenten im System verfügbar gemacht werden.

Die komponentenorientierte Systementwicklung sieht die Vergabe von Unteraufträgen vor. Diese können entweder auf ausgelagerte Entwicklungsprojekte oder sonstige Einkaufs- und Dienstleistungsanteile abgebildet werden. Die Möglichkeit zur Unterbeauftragung ist in allen verfügbaren Projektdurchführungsstrategien gegeben und durch das spezifizierte Szenario unterstützt.

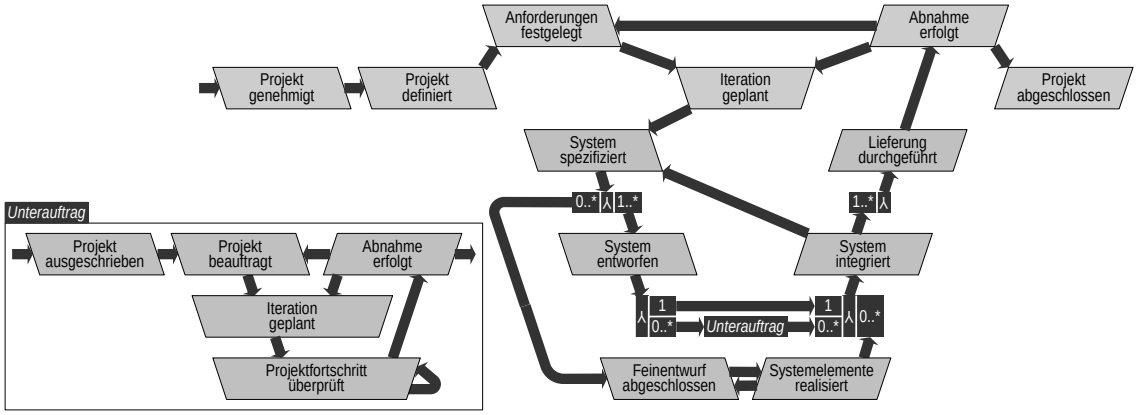


Abbildung A.7.: Komponentenorientierte Systementwicklung (AG/AN)

A.2.5. Wartung und Pflege von Systemen (AG/AN)

Wartung und Pflege ist nicht den eigentlichen Entwicklungsvorgehensweisen zuzuordnen. Hier sind nach der Iterationsplanung verschiedene Aufsetzpunkte möglich, je nach Größe, Umfang oder Tragweite der umzusetzenden Änderung. Das Projekt strukturiert sich auch hier in die drei Phasen:

- Initialisierung und Anforderungsfestlegung
- (punktuelle, änderungsgetriebene) Systementwicklung
- Projektabschluss

Wartung und Pflege sieht die Vergabe von Unteraufträgen vor. Diese können entweder auf ausgelagerte Entwicklungsprojekte oder sonstige Einkaufs- und Dienstleistungsanteile abgebildet werden. Die Möglichkeit zur Unterbeauftragung ist in allen verfügbaren Projektdurchführungsstrategien gegeben und durch das spezifizierte Szenario unterstützt.

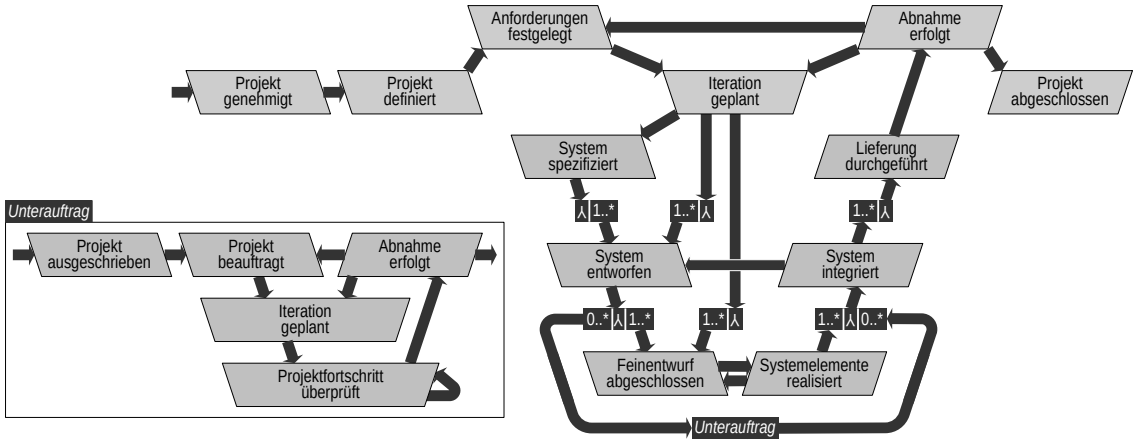


Abbildung A.8.: Wartung und Pflege von Systemen (AG/AN)

B. Work Item Types – Erweiterte Erläuterungen und Modellierungen

Dieser Anhang detailliert das den Work Item Typen aus CollabXT-TFS (Kapitel 4) zugrunde liegende Design. Betrachtet werden hierbei die Formulare der Work Items sowie die Regeln über den Daten und Transitionen. Weiterhin stellen wir hier Übersetzungshinweise bereit, um die deutschen Work Item Typen auch auf englischsprachigen Systemen betreiben zu können.

B.1. Regeln für Work Items

Regeln dienen dazu, Wertebelegungen einzelner Datenfelder zu steuern. Regeln sind z. B. unterhalb er `Field`-Definitionen (Abbildung 4.5) eingehängt. Abbildung B.1 zeigt die Regeln, die in Process Templates verfügbar sind. Beispiele für intuitiv verständliche Regeln sind z. B. `Required` oder `AllowedValues`. Neben Regeln gibt es auch noch Bedingungen (`Conditions`), die die Steuerung seitens TFS unterstützen.

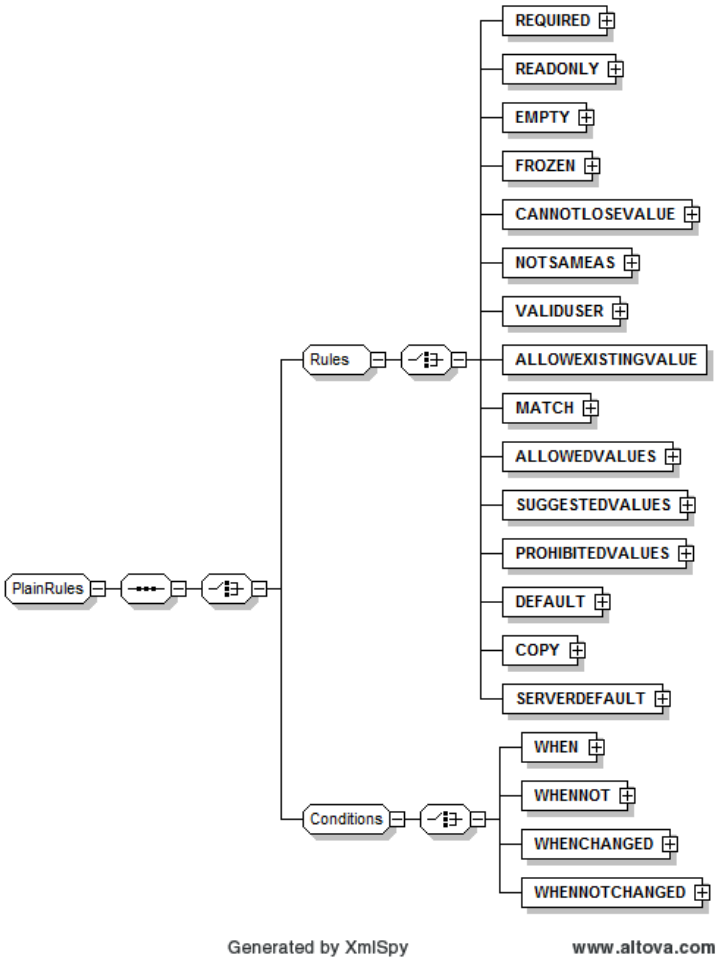


Abbildung B.1.: Regelmenge der Work Items

B.2. Formular-Design

Im Folgenden gehen wir detailliert auf den Formularentwurf der einzelnen Work Item Typen ein. Wir gehen dabei nicht auf die XML-Spezifikation ein, sondern zeigen die Masken, die der Process Template Editor erzeugt. Signifikante Elemente beschreiben wir detaillierter. Weiterhin nutzen wir die Gelegenheit, die Regeln und Konditionen zu einzelnen Work Item Typen noch einmal aufzugreifen und detailliert darzustellen.

B.2.1. Work Item: Produkt

Der Work Item Typ für Produkte ist als *Repräsentant* entworfen. Konkret bedeutet das, dass die V-Modell-Produkte, im Sinne von Dokumenten, in SharePoint Dokumentenbibliotheken liegen. Damit sind sie aber außerhalb des Zugriffsbereichs für die TFS Work Items. Der bereitgestellte Work Item Typ referenziert ein Produkt und hält im Wesentlichen Daten, die für Planung und Statuskontrolle erforderlich sind.

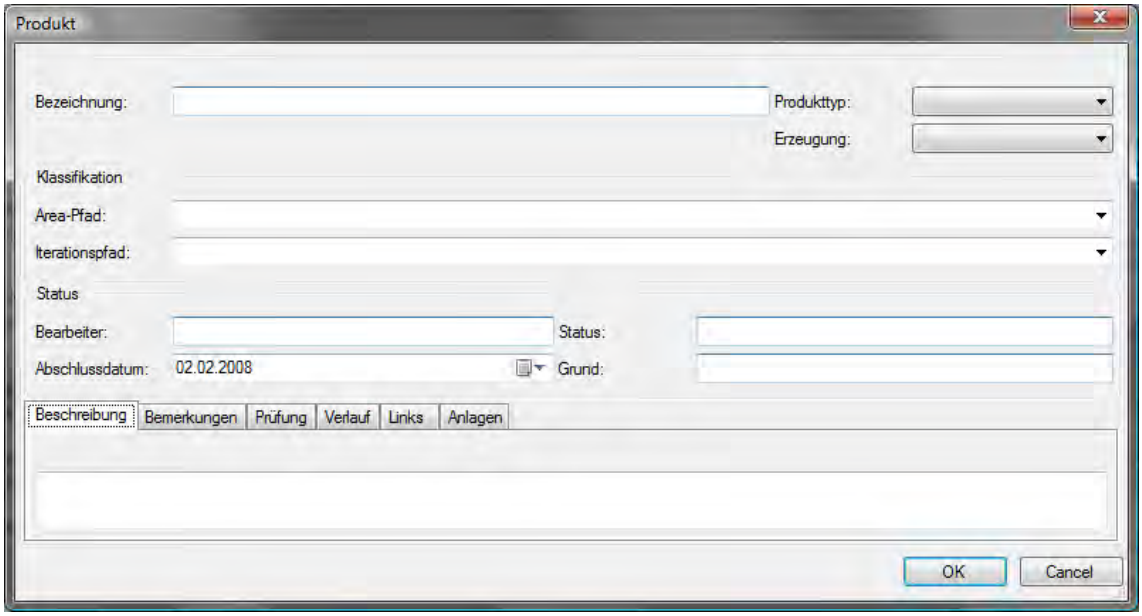


Abbildung B.2.: Formlayout für das Work Item Produkt

Abbildung B.2 zeigt das Formularlayout für den Work Item Typ *Produkt* (vgl. Kapitel 4.2.2). Im oberen Teil sind die Informationen für den *Produkttyp* und die *Produkterzeugung* zu finden. Der mittlere Bereich (*Klassifikation*) dient der Einplanung und Einordnung von Produkten in den Bereichs- und Iterationsgruppen des TFS. Darunter befindet sich der Statusbereich, in dem neben Bearbeiterinformationen und einem Plandatum der *Produktzustand* und der *Grund* für einen Zustandsübergang anzugeben sind. Die Logik hinter diesen Feldern wird durch den in Kapitel 4.2.2 beschriebenen Workflow bereitgestellt. Weiterhin können noch beschreibende Texte, Bemerkungen, Logs sowie Links und Dateinanlagen hinterlegt werden. Beschreibungstexte werden soweit beim Generierungsprozess verfügbar bereits automatisch mit erstellt. Produktexemplare in der SharePoint-Bibliothek oder Codes können durch Links referenziert werden. Dateianlagen können z. B. durch Prüfergebnisse, Screenshots etc. bereitgestellt werden.

Regel für Produkte. Wir betrachten nun die Regeln für den Work Item Typ *Produkt*. Wir wechseln dazu zur XML-Repräsentation des Work Item Typs, um die Regeln aus Anhang B.1 und deren Anwendung unmittelbar zu sehen. Die erste Regel bezieht sich auf die Eigenschaft `Titel`. Die Regel für das betreffende Feld ist `<REQUIRED />` und zeigt an, dass jede Instanz dieses Typs eine Bezeichnung haben muss:

```
<FIELD reportable="dimension" type="String" name="Titel" refname="System.Title">
        <HELPTEXT>Bezeichnung des Work Items.</HELPTEXT>
```

```
        <REQUIRED />
</FIELD>
```

Die folgende Regel regelt die Zuweisung eines Work Items an eine Benutzer. Die Regel `<VALIDUSER />` sorgt dafür, dass eine Zuweisung an einen gültigen Benutzer (des TFS-Systems) erfolgen muss:

```
<FIELD reportable="dimension" type="String" name="Zugewiesen␣an" refname="System.
    AssignedTo">
        <HELPTEXT>Der Verantwortliche für dieses Work Item.</HELPTEXT>
        <VALIDUSER />
</FIELD>
```

Die folgende Regel stellt fest: Wenn das Statusfeld `System.State` sich nicht geändert hat, dann ist das Feld `Microsoft.VSTS.Common.ClosedBy` als schreibgeschützt zu markieren. Die Regel `<WHENNOTCHANGED />` stellt Abhängigkeiten zwischen verschiedenen Feldern her. Hier betrifft das die Schließung einer Work Item Instanz. Hier ist die Aussage folgende: Annahme ist, dass das Work Item im Status *Abgeschlossen* (oder äquivalent) ist. Dann ist eine Person im Feld *Geschlossen von* eingetragen. Ändert sich der Zustand des Work Items nicht, so darf auch der Eintrag im Schließungsfeld nicht geändert werden:

```
<FIELD reportable="dimension" type="String" name="Geschlossen␣von" refname="Microsoft.
    VSTS.Common.ClosedBy">
        <HELPTEXT>Name desjenigen, der dieses Work Item geschlossen hat.</HELPTEXT>
        <VALIDUSER />
        <WHENNOTCHANGED field="System.State">
                <READONLY />
        </WHENNOTCHANGED>
</FIELD>
```

Dieselbe Regel wird auch für das Feld `Microsoft.VSTS.Common.ClosedDate` angewandt. Ist ein Work Item geschlossen, darf das Datum des Abschlusses nicht mehr geändert werden:

```
<FIELD reportable="dimension" type="DateTime" name="Schließungsdatum" refname="Microsoft
    .VSTS.Common.ClosedDate">
        <HELPTEXT>Datum, an dem dieses Work Item geschlossen wurde.</HELPTEXT>
        <WHENNOTCHANGED field="System.State">
                <READONLY />
        </WHENNOTCHANGED>
</FIELD>
```

Für die Produkterzeugung im V-Modell XT ist ein spezifischer Wertebereich für das Feld `VMXT.Erzeugung` vorzusehen. Anwendung findet hierbei die Regel `<ALLOWEDVALUES>...</ALLOWEDVALUES>`, die eine Menge von `<LISTITEM/>`-Elementen einschließt. `<ALLOWEDVALUES>` stellt für die Auswahlbox an der Benutzerschnittstelle die Listelemente zur Verfügung und legt weiterhin den gültigen Wertebereich für dieses Feld fest. Über die Regel `<DEFAULT/>` wird aus dieser Liste ein Standardwert angegeben, um eine gültige Belegung zu ermöglichen:

```
<FIELD type="String" name="Erzeugung␣(VMXT)" refname="VMXT.Erzeugung">
        <HELPTEXT>Die Art der Produkterzeugung</HELPTEXT>
        <ALLOWEDVALUES>
                <LISTITEM value="Initial" />
                <LISTITEM value="Abhängig" />
                <LISTITEM value="Extern" />
        </ALLOWEDVALUES>
        <DEFAULT from="value" value="Abhängig" />
</FIELD>
```

Neben den Regeln für die Felder eines Work Items, werden Regeln auch dann angewendet, wenn die Stati gesetzt werden. Regeln sind somit auch auf Zustände und Zustandsübergänge anzuwenden. Im Folgenden betrachten wir die Stati des Work Item Typs *Produkt*. Im Status *In Bearbeitung* ist das Feld `System.AssignedTo` verpflichtend anzugeben, was lediglich heißt, dass in diesem Zustand ein Bearbeiter zugewiesen werden muss. Dasselbe gilt für den Zustand *Vorgelegt*. Im Zustand *Fertig gestellt* greifen mehrere Regeln: Die Felder `Microsoft.VSTS.Common.ClosedDate` und `Microsoft.VSTS.Common.ClosedBy` werden in diesem Zustand vom Server automatisch mit Werten belegt (Regel `<SERVERDEFAULT/>`). Der Wert `from="clock"` legt dabei fest, dass das Abschlussdatum mit der aktuellen Serverzeit befüllt wird; der Wert `from="currentuser"` liefert die Kennung des aktuell angemeldeten Benutzers. Die Änderungsfähigkeit dieser beiden Felder wird dann mit den bereits weiter oben besprochenen Regeln festgelegt. Der Wert des Feldes

`System.AssignedTo` wird zurückgesetzt (`<EMPTY/>`), was zur Folge hat, dass bei der Wiedereröffnung eines geschlossenen Work Items eine Neuzuweisung an einen Bearbeiter erzwungen wird. Das Feld `VMXT.Assessment` ist als `<REQUIRED />` gekennzeichnet:

```
<STATE value="In␣Bearbeitung">
        <FIELDS>
                <FIELD refname="System.AssignedTo">
                        <REQUIRED />
                </FIELD>
        </FIELDS>
</STATE>
<STATE value="Vorgelegt">
        <FIELDS>
                <FIELD refname="System.AssignedTo">
                        <REQUIRED />
                </FIELD>
        </FIELDS>
</STATE>
<STATE value="Fertig␣gestellt">
        <FIELDS>
                <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
                        <SERVERDEFAULT from="clock" />
                </FIELD>
                <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
                        <SERVERDEFAULT from="currentuser" />
                </FIELD>
                <FIELD refname="System.AssignedTo">
                        <EMPTY />
                </FIELD>
                <FIELD refname="VMXT.Assessment">
                        <REQUIRED />
                </FIELD>
        </FIELDS>
</STATE>
```

Bei den Zustandsübergängen werden Gründe definiert (Feld `System.Reason`) aber auch Regeln, wie die Wertebelegung einzelner Felder geregelt wird. Beim Initialen Übergang von $\perp \Rightarrow$ *In Bearbeitung* wird das Feld `System.AssignedTo` mit einem Standardwert belegt: `<DEFAULT from="currentuser"/>`:

```
<TRANSITION from="" to="In␣Bearbeitung">
        <REASONS>
                <DEFAULTREASON value="Neu" />
        </REASONS>
        <FIELDS>
                <FIELD refname="System.AssignedTo">
                        <DEFAULT from="currentuser" />
                </FIELD>
        </FIELDS>
</TRANSITION>
```

Beim Übergang von *Fertig gestellt*⇒*In Bearbeitung*, also der Wiedereröffnung eines Produkts (aus den unten stehenden Gründen) werden im Wesentlichen die `<EMPTY/>`-Regeln angewendet. Die Felder `VMXT.Assessment`, `Microsoft.VSTS.Common.ClosedBy` und `Microsoft.VSTS.Common.-ClosedDate` werden hierbei gelöscht:

```
<TRANSITION from="Fertig␣gestellt" to="In␣Bearbeitung">
        <REASONS>
                <REASON value="Neubearbeitung␣(Qualität,␣Inhalt)" />
                <REASON value="Hinfällig" />
                <DEFAULTREASON value="Neubearbeitung␣(Konsistenz)" />
        </REASONS>
        <FIELDS>
                <FIELD refname="VMXT.Assessment">
                        <EMPTY />
                </FIELD>
                <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
                        <EMPTY />
                </FIELD>
                <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
                        <EMPTY />
                </FIELD>
        </FIELDS>
</TRANSITION>
```

B.2.2. Work Item: Aktivität

Für den Work Item Typ für Aktivitäten gilt dasselbe Konzept wie für Produkte. Auch Aktivitäten im Sinne des V-Modells modellieren wir als Repräsentanten. Die Formulare (Abbildung B.3) entsprechen sich auch im Aufbau; lediglich die nur Produkte relevanten Informationen (Typ und Erzeugung) finden wir hier nicht mehr. Wichtig bei Aktivitäten ist (vgl. Kapitel 2.2.3, dass Aktivitäten die Produkte, die sie erstellen, referenzieren. Die notwendigen Referenzinformationen sind in der Datenstruktur des Work Item Typs enthalen (jedoch nicht am Formular sichtbar). Hierfür verwenden wir die *Links*-Felder, die auf der Grundlage der Auswertung der Referenzen mit Verknüpfungen zwischen Aktivitäten und Produkten gefüllt werden. Die Logik hinter den Feldern für *Status* und *Grund* wird durch den Wokrflow in Kapitel 4.2.3 bereitgestellt.

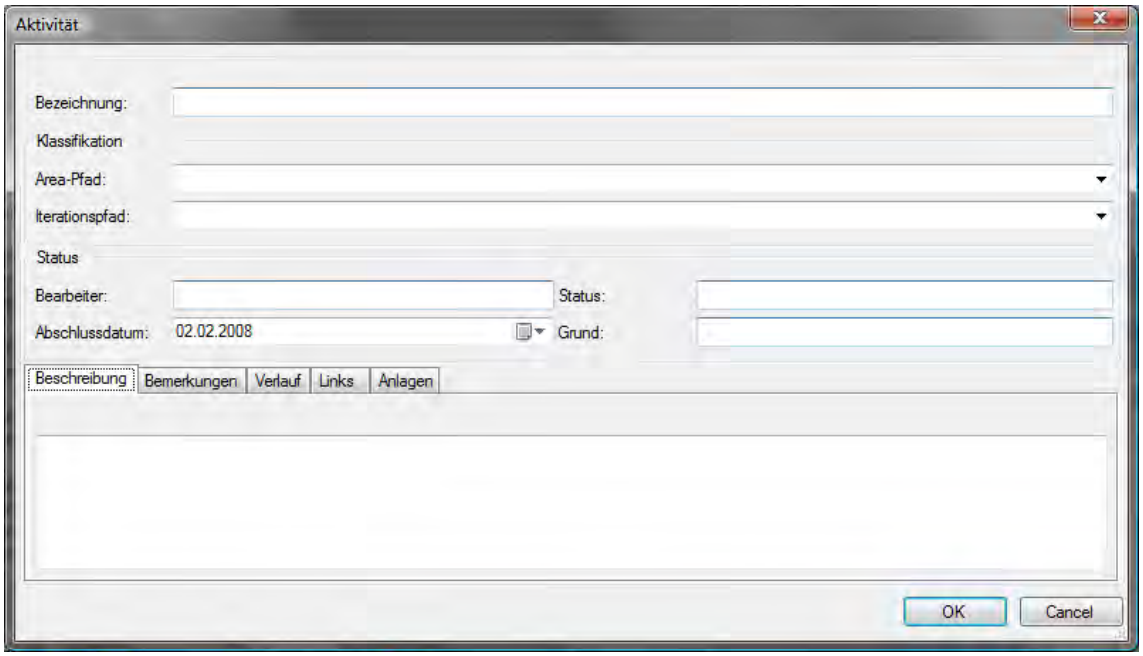


Abbildung B.3.: Formlayout für das Work Item Aktivität

Regeln für Aktivitäten. Auch für Aktivitäten definieren wir auf der Ebene von Work Items Regeln. An dieser Stelle wird bereits offensichtlich, dass wir mit einer sehr überschaubaren Menge von Regeln auskommen und diese sich immer wiederholen (sowohl Regeln als auch die referenzierten Elemente). Aktivitäten erfordern eine Bezeichnung (Feld `System.Titel` mit der Regel `<REQUIRED/>`). Ferner muss eine Aktivtät einem gültigen Benutzer des Systems zugewiesn werden (Feld `System.AssignedTo` mit der Regel `<VALIDUSER/>`).

```
<FIELD reportable="dimension" type="String" name="Titel" refname="System.Title">
        <HELPTEXT>Bezeichnung des Work Items.</HELPTEXT>
        <REQUIRED />
</FIELD>

<FIELD reportable="dimension" type="String" name="Zugewiesen an" refname="System.
    AssignedTo">
        <HELPTEXT>Der Verantwortliche für dieses Work Item.</HELPTEXT>
        <VALIDUSER />
</FIELD>
```

Analog zu Produkten, werden die Felder `Microsoft.VSTS.Common.ClosedBy` und `Microsoft.VSTS.Common.ClosedDate` schreibgeschützt, sofern eine Work Item Instanz abgeschlossen ist. Erläuterungen zu dieser Regel siehe oben:

```
<FIELD reportable="dimension" type="String" name="Geschlossen von" refname="Microsoft.
    VSTS.Common.ClosedBy">
  <HELPTEXT>Name desjenigen, der dieses Work Item geschlossen hat.</HELPTEXT>
  <VALIDUSER />
```

```
  <WHENNOTCHANGED field="System.State">
    <READONLY />
  </WHENNOTCHANGED>
</FIELD>
<FIELD reportable="dimension" type="DateTime" name="Schließungsdatum" refname="Microsoft
    .VSTS.Common.ClosedDate">
  <HELPTEXT>Datum, an dem dieses Work Item geschlossen wurde.</HELPTEXT>
  <WHENNOTCHANGED field="System.State">
    <READONLY />
  </WHENNOTCHANGED>
</FIELD>
```

Wir betrachten auch bei Aktivitäten einen Workflow, der verschiedene Stati und Übergänge zwischen diesen modelliert. Im Zustand *In Bearbeitung* muss ein Anwender zugewiesen werden (Feld `System.AssignedTo` mit der Regel `<REQUIRED/>`). Weiterhin werden im Zustand *Absgeschlossen* die Felder `Microsoft.VSTS.Common.ClosedDate` und `Microsoft.VSTS.Common.ClosedBy` mit Serverwerten (Regel `<SERVERDEFAULT/>`) befüllt und die Zuweisung eines Bearbeiters (Feld `System.AssignedTo`) mithilfe der Regel `<EMPTY/>` aufgehoben:

```
<STATE value="In␣Bearbeitung">
  <FIELDS>
    <FIELD refname="System.AssignedTo">
      <REQUIRED />
    </FIELD>
  </FIELDS>
</STATE>
<STATE value="Abgeschlossen">
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
      <SERVERDEFAULT from="clock" />
    </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
      <SERVERDEFAULT from="currentuser" />
    </FIELD>
    <FIELD refname="System.AssignedTo">
      <EMPTY />
    </FIELD>
  </FIELDS>
</STATE>
```

Beim Wiedereröffnen eines Work Items (Übergang *Abgeschlossen⇒Geplant*) werden die Felder `Microsoft.VSTS.Common.ClosedBy` und `Microsoft.VSTS.Common.ClosedDate` mithilfe der Regel `<EMPTY/>` geleert.

```
<TRANSITION from="Abgeschlossen" to="Geplant">
  <REASONS>
    <DEFAULTREASON value="Neubearbeitung␣erforderlich" />
  </REASONS>
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
      <EMPTY />
    </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
      <EMPTY />
    </FIELD>
  </FIELDS>
</TRANSITION>
```

B.2.3. Work Item: Entscheidungspunkt

Entscheidungspunkte dienen als Meilensteine, in denen Produkt und Aktivitäten zusammenlaufen (Befüllung der *Links*-Felder). Abbildung B.4 zeigt das Formular für Entscheidungspunkte, das wiederum denen für Produkte und Aktiviäten entspricht.

Entscheidungspunkte haben einen Typ. Dieser ist im oberen Teil des Formulars als Auswahlliste hinterlegt. Beim Anlegen eines Entscheidungspunkts muss dieser Typ angegeben werden. Die Logik hinter dem Formular (*Status*, *Grund*) entstammt dem Workflow aus Kapitel 4.2.4. Beachtet werden muss jedoch, dass dieser Workflow nur lokal gültig ist. Die Abhängigkeiten zwischen Entscheidungspunkten im Rahmen der Projektdurchführungsstrategien werden nicht geprüft.

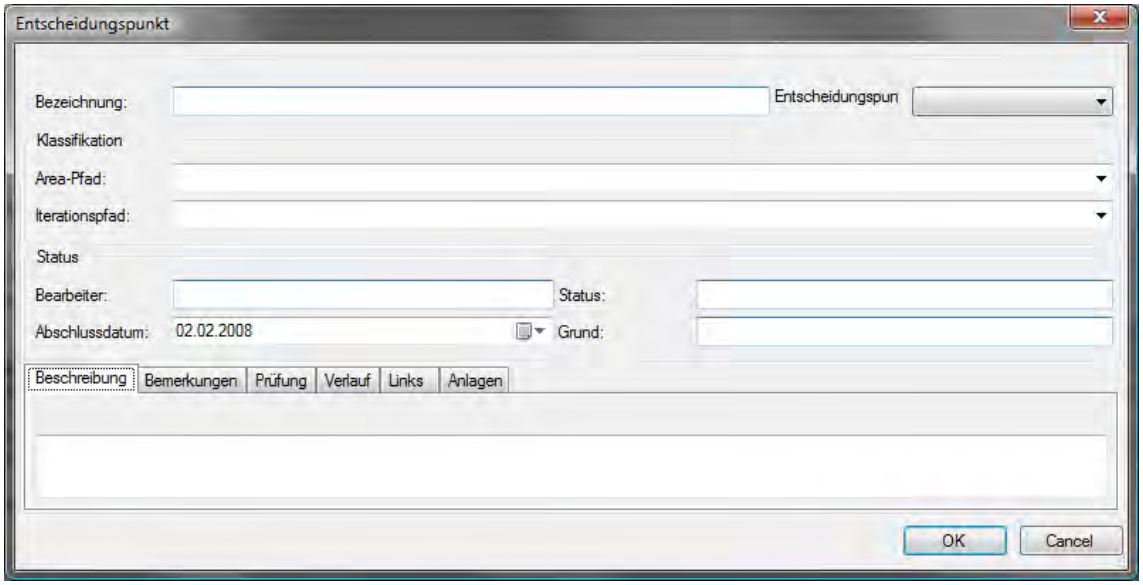


Abbildung B.4.: Formlayout für das Work Item Entscheidungspunkt

Regeln für Entscheidungspunkte. Entscheidungspunkte müssen eine Bezeichnung haben (Feld `System.Title` mit der Regel `<REQUIRED/>`) und sie müssen einem gültigen Systembenutzer zugewiesen werden (Feld `System.AssignedTo` mit der Regel `<VALIDUSER/>`):

```
<FIELD reportable="dimension" type="String" name="Titel" refname="System.Title">
  <REQUIRED /> </FIELD>

<FIELD reportable="dimension" type="String" name="Zugewiesen␣an" refname="System.
    AssignedTo">
  <VALIDUSER /> </FIELD>
```

Wie auch weiter oben, werden im Entscheidungspunkt die Felder `Microsoft.VSTS.Common.ClosedBy` und `Microsoft.VSTS.Common.ClosedDate` schreibgeschützt, wenn der Entscheidungspunkt im Abschlussstatus ist (siehe auch oben):

```
<FIELD reportable="dimension" type="String" name="Geschlossen␣von" refname="Microsoft.
    VSTS.Common.ClosedBy">
  <VALIDUSER />
  <WHENNOTCHANGED field="System.State"> <READONLY /> </WHENNOTCHANGED>
</FIELD>
<FIELD reportable="dimension" type="DateTime" name="Schließungsdatum" refname="Microsoft
    .VSTS.Common.ClosedDate">
  <WHENNOTCHANGED field="System.State"> <READONLY /> </WHENNOTCHANGED>
</FIELD>
```

Dem Work Item Typ *Entscheidungspunkt* ist auch ein so genannter *Entscheidungspunkttyp* zugeordnet (vgl. V-Modell-Projektpläne, in denen z. B. der Entscheidungspunkt *Iteration geplant* mehrfach vorkommt). Dieser wird im Feld `VMXT.Entscheidungspunkttyp` abgelegt und durch die Regel `<ALLOWEDVALUES>` mit unten stehender Liste eingeschränkt:

```
<FIELD type="String" name="Entscheidungspunkttyp␣(VMXT)" refname="VMXT.
    Entscheidungspunkttyp">
  <ALLOWEDVALUES>
    <LISTITEM value="Abnahme␣erfolgt" />
    <LISTITEM value="Anforderungen␣festgelegt" />
    <LISTITEM value="Angebot␣abgegeben" />
    <LISTITEM value="Feinentwurf␣abgeschlossen" />
    <LISTITEM value="Gesamtprojekt␣aufgeteilt" />
    <LISTITEM value="Gesamtprojektfortschritt␣überprüft" />
    <LISTITEM value="Iteration␣geplant" />
    <LISTITEM value="Lieferung␣durchgeführt" />
    <LISTITEM value="Projekt␣abgeschlossen" />
    <LISTITEM value="Projekt␣ausgeschrieben" />
    <LISTITEM value="Projekt␣beauftragt" />
    <LISTITEM value="Projekt␣definiert" />
```

```
    <LISTITEM value="Projektfortschritt␣überprüft" />
    <LISTITEM value="Projekt␣genehmigt" />
    <LISTITEM value="Systemelemente␣realisiert" />
    <LISTITEM value="System␣entworfen" />
    <LISTITEM value="System␣integriert" />
    <LISTITEM value="System␣spezifiziert" />
    <LISTITEM value="Nicht␣zugeordnet" />
  </ALLOWEDVALUES>
  <DEFAULT from="value" value="Nicht␣zugeordnet" />
</FIELD>
```

Der Default-Wert *Nicht zugeordnet* entspricht hierbei im Handling den *freien Meilensteinen*, wie sie im V-Modell XT Projektassistenten verwendet werden. Für die Stati des zugrunde liegenden Workflows sind folgende Regeln definiert: Im Status *Geplant* ist ein Bearbeiter zuzuweisen (Feld `System.AssignedTo` mit der Regel `<REQUIRED/>`). Im Status *Erreicht* (Abschlusstatus dieses Work Item Typs) sind die Felder `Microsoft.VSTS.Common.ClosedDate` und `Microsoft.VSTS.Common.ClosedBy` mit Serverwerten (Regel `<SERVERDEFAULT/>`) zu belegen, um den Abschluss anzuzeigen. Dabei wird das Feld `System.AssignedTo` mithilfe der Regel `<EMPTY>` geleert und es wird festgelegt, dass durch die Regel `<REQUIRED/>` das Feld `VMXT.Assessment` gefüllt werden muss:

```
<STATE value="Geplant">
  <FIELDS>
    <FIELD refname="System.AssignedTo">
      <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="Erreicht">
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
      <SERVERDEFAULT from="clock" /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
      <SERVERDEFAULT from="currentuser" /> </FIELD>
    <FIELD refname="System.AssignedTo"> <EMPTY /> </FIELD>
    <FIELD refname="VMXT.Assessment"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="Nicht␣erreicht">
  <FIELDS>
    <FIELD refname="System.AssignedTo"> <REQUIRED /> </FIELD>
    <FIELD refname="VMXT.Assessment"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
```

Bei den Transitionen ist lediglich die initiale Transition von $\perp \Rightarrow$*Geplant* mit einer Regel unterlegt. Wird ein Entscheidungspunkt geplant, so muss dem Feld `System.AssignedTo` mit der Regel `<DEFAULT from="currentuser"/>` ein Anwender zugeordnet werden. Initial ist der derjenige, der das Work Item anlegt:

```
<TRANSITION from="" to="Geplant">
  <REASONS> <DEFAULTREASON value="Neu" /> </REASONS>
  <FIELDS>
    <FIELD refname="System.AssignedTo"> <DEFAULT from="currentuser" /> </FIELD>
  </FIELDS>
</TRANSITION>
```

B.2.4. Work Item: Arbeitsauftrag

Der Arbeitsauftrag (Abbildung B.5) ersetzt als Work Item Typ in unserer Modellierung das V-Modell-Element *Teilaktivität*. Er orientiert sich am Aufbau (auch als Formular) an den gegebenen *Task* Work Items aus dem Microsoft Solutions Framework [Tur06].

Das Formular spiegelt die Modellierung aus Kapitel 4.2.5 wider. Im oberen Teil des Formulars findet sich neben dem obligatorischen Titel eine Auswahlgruppe für die *Disziplin*, in der der Arbeitsauftrag ausgeführt wird. Im Statusbereich gibt es Felder im Formular für *Status*, *Grund*, *Geblockt*, *Gewicht* und *Priorität*. Diese Felder sind mit definierten Wertebereichen im Work Item Typ hinterlegt. An verschiedenen Stellen im Workflow ist das Setzen dieser Felder erforderlich. Im unteren Bereich befindet sich ergänzend zu den Beschreibungs- (reines Textfeld) und Links-Boxen

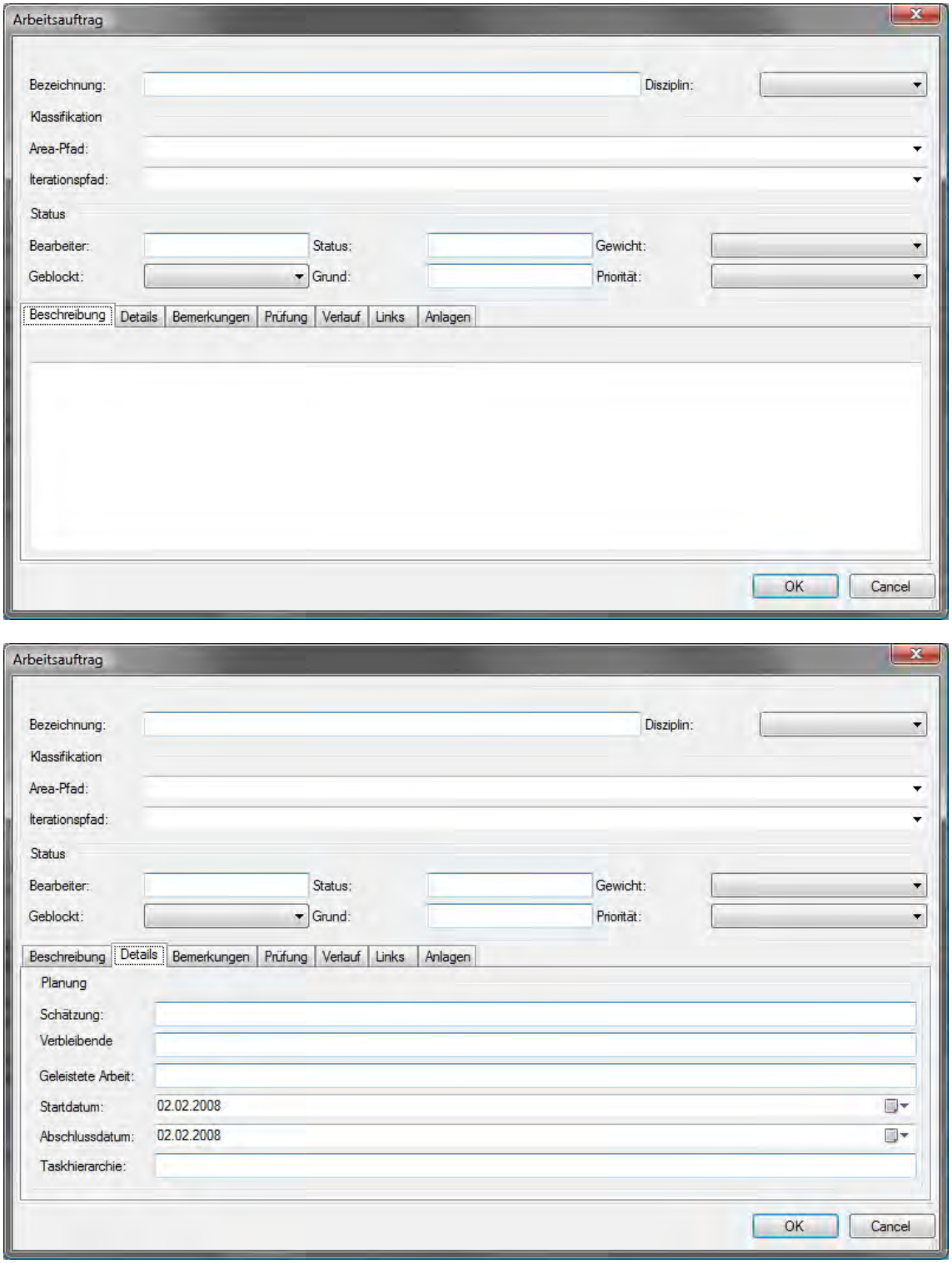


Abbildung B.5.: Formlayout für das Work Item Arbeitsauftrag

ein neuer *Details*-Bereich (Abbildung B.5). Die Felder, die hier zusammengefasst sind, dienen der Planung und Überwachung eines Arbeitsauftrags im Projektplan. Hier gibt es die Möglichkeit Schätz-/Planwerte zu hinterlegen und z. B. im Feld *Geleistete Arbeit* abzurechnen. Die hier relevanten Daten sind sowohl im TFS selbst als auch im externen Excel oder Project erfassbar.

Regeln für Arbeitsaufträge. Für Arbeitsaufträge gelten wie weiter oben bereits auch die Regeln `<REQUIRED/>` für das Feld `System.Title` und `<VALIDUSER/>` für das Feld `System.AssignedTo`:

```
<FIELD type="String" name="Titel" refname="System.Title">
  <REQUIRED /> </FIELD>

<FIELD type="String" name="Zugewiesen␣an" refname="System.AssignedTo">
  <VALIDUSER /> </FIELD>
```

Weiterhin sind im Abschlussstatus eines Work Items vom Typ *Arbeitsauftrag* auch die Felder `Microsoft.VSTS.Common.ClosedDate` und `Microsoft.VSTS.Common.ClosedBy` mithilfe der Regel `<WHENNOTCHANGED/>` mit einem Schreibschutz zu versehen. Ferner muss für das Feld `Microsoft.VSTS.Common.ClosedBy` durch die Regel `<VALIDUSER/>` ein gültige Benutzer des Systems benannt sein:

```
<FIELD reportable="dimension" type="DateTime" name="Schließungsdatum" refname="Microsoft
    .VSTS.Common.ClosedDate">
  <WHENNOTCHANGED field="System.State">
    <READONLY /> </WHENNOTCHANGED>
</FIELD>
<FIELD reportable="dimension" type="String" name="Geschlossen␣von" refname="Microsoft.
    VSTS.Common.ClosedBy">
  <VALIDUSER />
  <WHENNOTCHANGED field="System.State">
    <READONLY /> </WHENNOTCHANGED>
</FIELD>
```

Für einen Arbeitsauftrag muss (Regel `<REQUIRED/>`) eine Priorität festgelegt werden. Der Wertebereich des dafür verwendeten Feldes `Microsoft.VSTS.Common.Priority` wird durch die Regel `<ALLOWEDVALUES/>` festgelegt. Als Default-Wert fungiert der Wert *3*[1]:

```
<FIELD reportable="dimension" type="Integer" name="Priorität" refname="Microsoft.VSTS.
    Common.Priority">
  <REQUIRED />
  <ALLOWEDVALUES>
    <LISTITEM value="1" />
    <LISTITEM value="2" />
    <LISTITEM value="3" />
  </ALLOWEDVALUES>
  <DEFAULT from="value" value="3" />
</FIELD>
```

Ein Arbeitsauftrag kann sich stark auf ein Projekt auswirken. Das dafür vorgesehene Feld `Microsoft.VSTS.Common.Severity` muss (Regel `<REQUIRED/>`) mit einem Wert belegt werden, der die Schwere anzeigt. Der entsprechende Wertebereich wird wieder durch die Regel `<ALLOWEDVALUES/>` festgelegt, wobei als Default-Wert *Niedrig* festgelegt ist:

```
<FIELD type="String" name="Schweregrad" refname="Microsoft.VSTS.Common.Severity">
  <REQUIRED />
  <ALLOWEDVALUES>
    <LISTITEM value="Kritisch" />
    <LISTITEM value="Hoch" />
    <LISTITEM value="Mittel" />
    <LISTITEM value="Niedrig" />
  </ALLOWEDVALUES>
  <DEFAULT from="value" value="Niedrig" />
</FIELD>
```

Im Feld `VMXT.Discipline` wird festgehalten, zu welchem Aufgabengebiet, zu welcher Disziplin ein konkreter Arbeitsauftrag gehört. Mithilfe der Regel `<ALLOWEDVALUES/>` wird der dafür notwendige Wertebereich definiert:

1 1–3 ist eine abstrakte Skala ohne jede Vorinterpretation. Diese muss der Anwender übernehmen.

```
<FIELD reportable="dimension" type="String" name="Disziplin␣(VMXT)" refname="VMXT.
    Discipline">
  <ALLOWEDVALUES>
    <LISTITEM value="Entwicklung" />
    <LISTITEM value="Qualitätssicherung" />
    <LISTITEM value="Projektmanagement" />
    <LISTITEM value="Anforderungsfeststellung" />
    <LISTITEM value="Architektur,␣Entwurf,␣Spezifikation" />
    <LISTITEM value="Konfigurationsmanagement" />
    <LISTITEM value="Ausschreibungs-␣und␣Vertragswesen" />
    <LISTITEM value="Problem-␣und␣Änderungsmanagement" />
  </ALLOWEDVALUES>
</FIELD>
```

Das Feld `VMXT.Blocked` zeigt an, ob ein Arbeitsauftrag blockiert ist. Die Regel `<ALLOWEDVALUES/>` bildet den Wertebereich *Ja/Nein* ab, wobei *Nein* der Default-Wert ist. Das Feld muss (Regel `<REQUIRED/>`) mit einem Wert belegt werden:

```
<FIELD reportable="dimension" type="String" name="Blockiert␣(VMXT)" refname="VMXT.
    Blocked">
  <ALLOWEDVALUES>
    <LISTITEM value="Ja" />
    <LISTITEM value="Nein" />
  </ALLOWEDVALUES>
  <DEFAULT from="value" value="Nein" />
  <REQUIRED />
</FIELD>
```

Beim Arbeitsauftrag sind mit den Zuständen *Gelöst*, *Erledigt* und *In Bearbeitung* Regeln für die Wertebelegung von Feldern verbunden. Im Zustand *Gelöst* wird das Feld `VMXT.Assessment` durch die Regel `<EMPTY/>` wieder zurückgesetzt. Im Zustand *Erledigt* werden die Felder `Microsoft.VSTS.Common.ClosedBy` und `Microsoft.VSTS.Common.ClosedDate` durch die Regel `<SERVERDEFAULT/>` mit Serverwerten belegt. Das Feld `System.AssignedTo` wird durch die Regel `<EMPTY/>` zurückgesetzt und das Feld `VMXT.Assessment` wird durch die Regel `<REQUIRED/>` als verpflichtend auszufüllen markiert. Im Zustand *In Bearbeitung* ist das Feld `System.AssignedTo` durch die Regel `<REQUIRED/>` als verpflichtend gekennzeichnet:

```
<STATE value="Gelöst">
  <FIELDS>
    <FIELD refname="VMXT.Assessment"> <EMPTY /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="Erledigt">
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
      <SERVERDEFAULT from="currentuser" /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
      <SERVERDEFAULT from="clock" /> </FIELD>
    <FIELD refname="System.AssignedTo"> <EMPTY /> </FIELD>
    <FIELD refname="VMXT.Assessment"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="In␣Bearbeitung">
  <FIELDS>
    <FIELD refname="System.AssignedTo"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
```

Beim Zustandsübergang von *Gelöst⇒Bearbeitung* wird das Feld `VMXT.Assessment` durch die Regel `<REQUIRED/>` als verpflichtend auszufüllen markiert:

```
<TRANSITION from="Gelöst" to="In␣Bearbeitung">
  <REASONS>
    <DEFAULTREASON value="Nacharbeit␣erforderlich" />
  </REASONS>
  <FIELDS>
    <FIELD refname="VMXT.Assessment"> <REQUIRED /> </FIELD>
  </FIELDS>
</TRANSITION>
```

Bei Zustandsübergang von *Erledigt⇒Aktiv*, also bei der Wiedereröffnung eines Arbeitsauftrags, werden die Felder `Microsoft.VSTS.Common.ClosedDate`, `Microsoft.VSTS.Common.ClosedBy` und `VMXT.Assessment` durch die Regel `<EMPTY/>` zurückgesetzt.

```
<TRANSITION from="Erledigt" to="Aktiv">
  <REASONS>
    <DEFAULTREASON value="Reaktiviert" />
  </REASONS>
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate"> <EMPTY /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy"> <EMPTY /> </FIELD>
    <FIELD refname="VMXT.Assessment"> <EMPTY /> </FIELD>
  </FIELDS>
</TRANSITION>
```

B.2.5. Work Item: Risiko und Maßnahme

Das Risikomanagement ist nach dem Aufgabenmanagement der zweite komplexere Prozess, den wir in Form von Work Items modellieren.

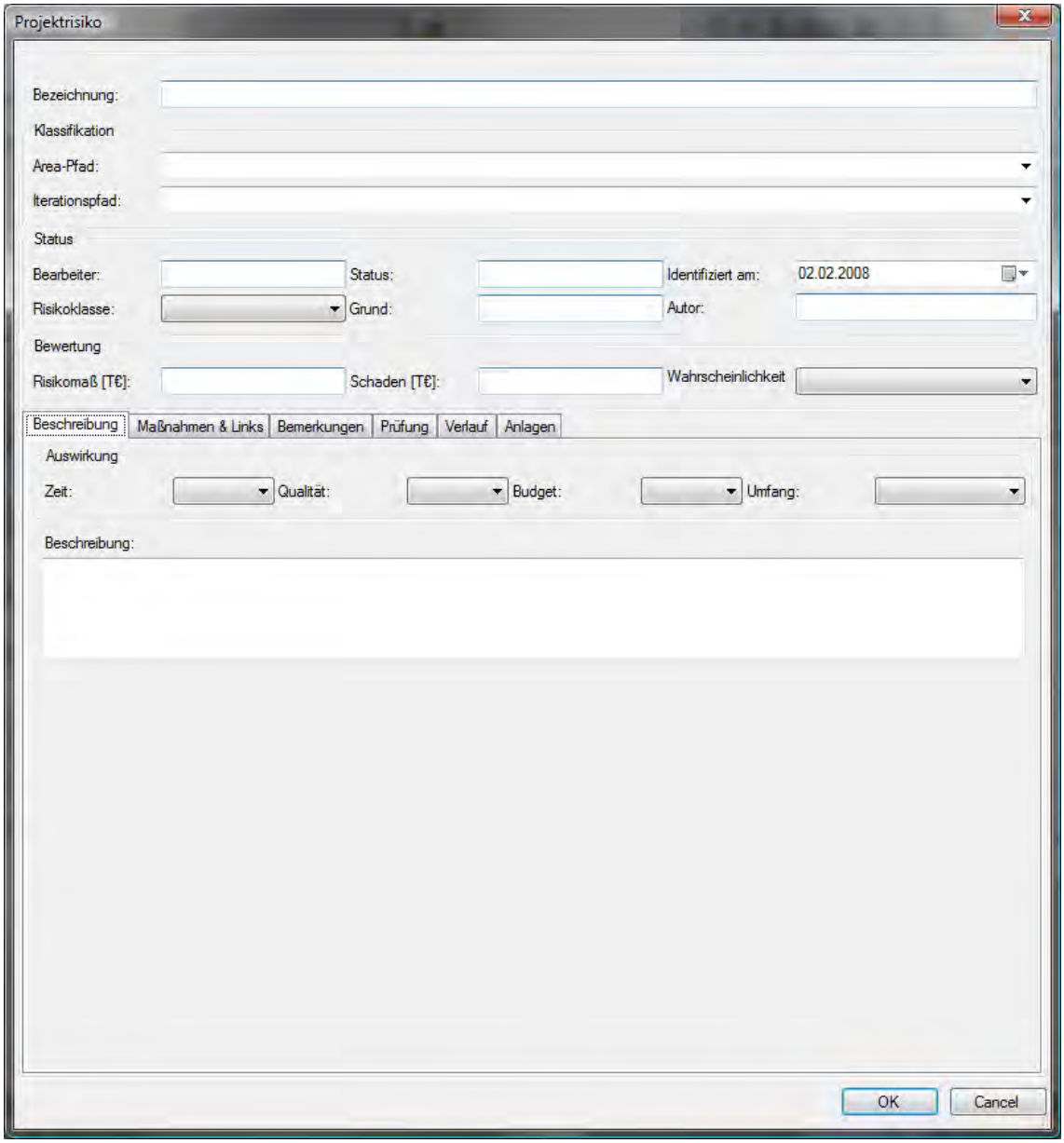


Abbildung B.6.: Formlayout für das Work Item Risiko

In Abbildung B.6 sind im oberen Teil des Formulars die Standarddaten für Bezeichnung und Klassifikation zu finden. Im darunter liegenden Statusbereich werden *Bearbeiter*, *Autor* (Melder),

Identifikationsdatum (*Identifiziert am)*, *Status* und *Grund* erfasst. Der Workflow, der die möglichen Übergänge definiert, ist in Kapitel 4.2.6 beschrieben. Im unteren Bereich des Beschreibungs-

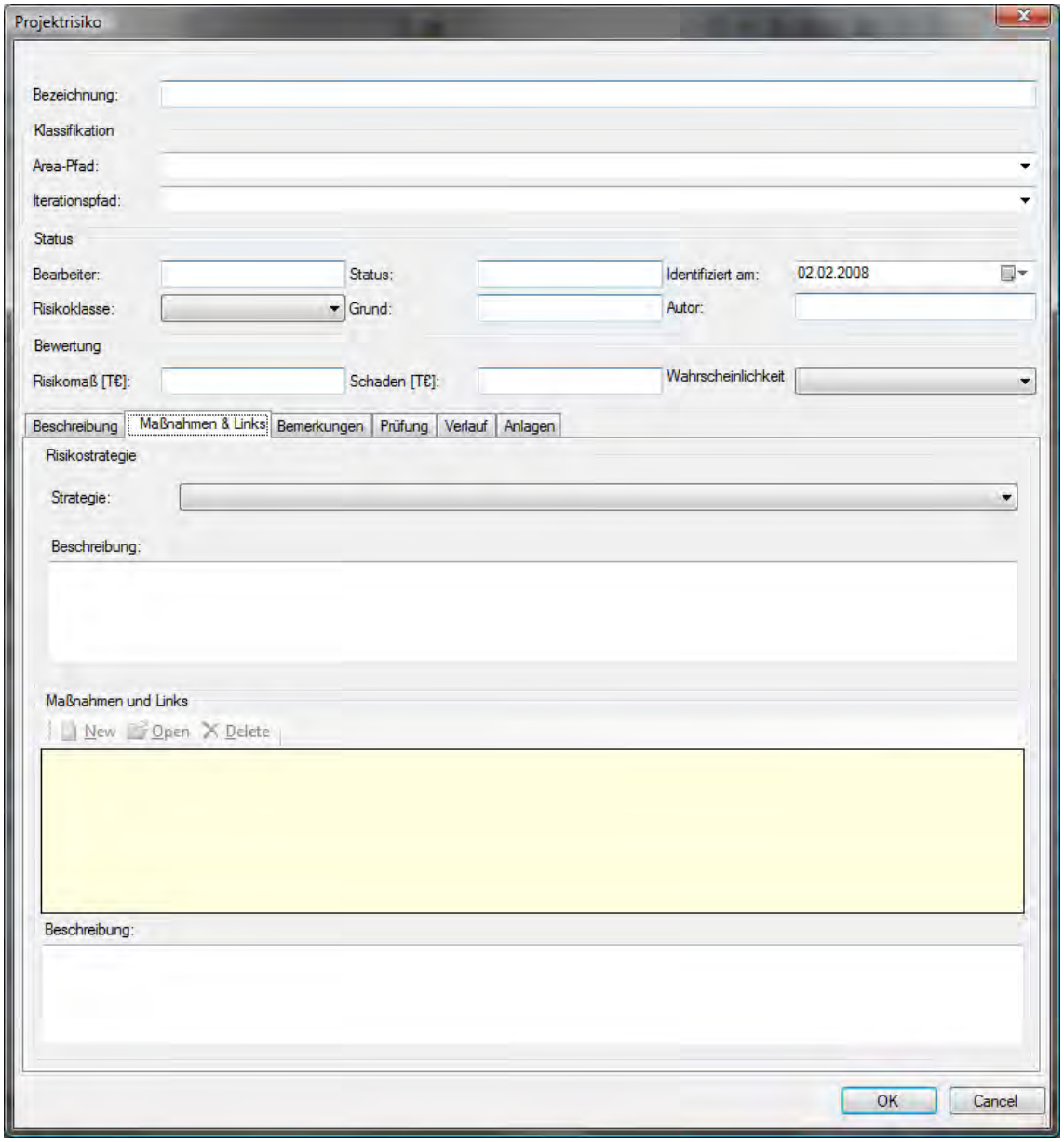


Abbildung B.7.: Formlayout für das Work Item Risiko

formulars können die Auswirkungen des identifizierten Risikos nach *Zeit*, *Qualität*, *Budget* und *Umfang* eingestuft werden. Zusätzlich befinden sich hier die üblichen Beschreibungs-, Log- und Anlagen-Felder. In Abbildung B.7 ist die Detaillierung des Reiters *Maßnahmen & Links* zu sehen. Hier kann dem Risiko eine *Strategie* zugewiesen und beschrieben werden. Darunter können beliebig viele *Maßnahmen* verlinkt und beschrieben werden.

Die Maßnahmen modellieren wir ebenfalls eigenen Work Item Typ. Abbildung B.8 zeigt das Formular für die Dateneingaben. Die Stati werden anhand der in Kapitel4.2.6 modellierten Übergänge angesprungen. Ansonsten definieren Maßnahmen lediglich noch einen *Auslöser*; entsprechen ansonsten aber im Wesentlichen den Formularlayouts für Produkte und Aktivitäten (siehe weiter oben).

Regeln für Risiken. Auch Risiken haben eine Bezeichnung (Feld `System.Title` mit der Regel `<REQUIRED/>`) und einen gültigen Systembenutzer (Feld `System.AssignedTo` mit der Regel `<VALIDUSER/>`) dem das Risiko zugewiesen ist.

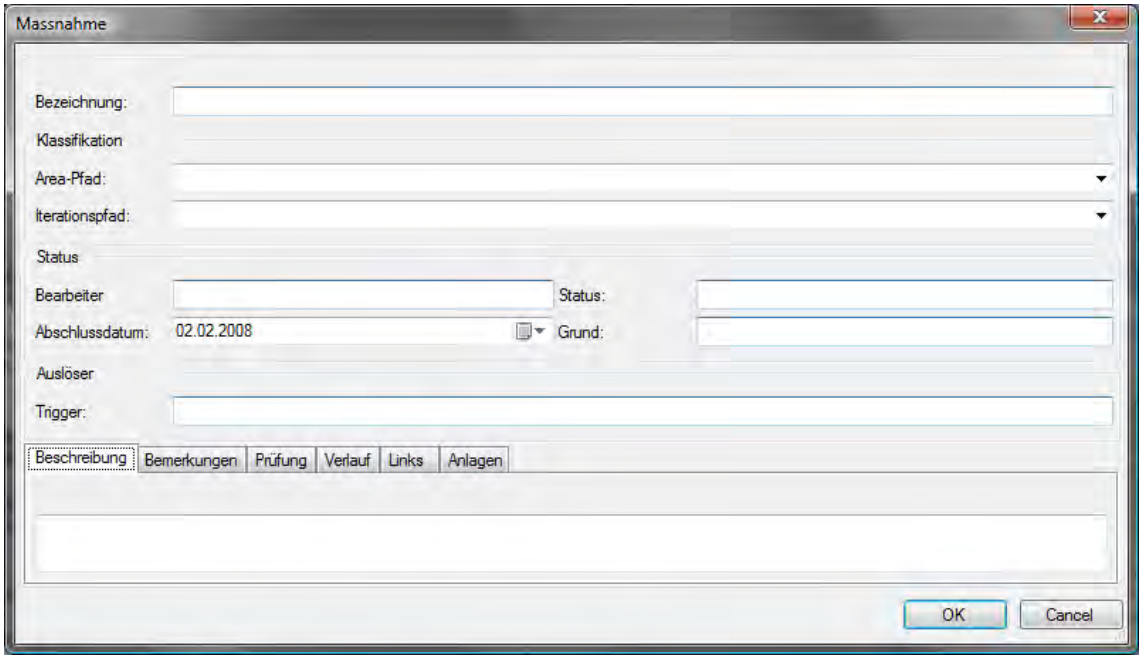


Abbildung B.8.: Formlayout für das Work Item Maßnahme

```
<FIELD reportable="dimension" type="String" name="Titel" refname="System.Title">
  <REQUIRED /> </FIELD>
<FIELD reportable="dimension" type="String" name="Zugewiesen an" refname="System.
    AssignedTo">
  <VALIDUSER /> </FIELD>
```

Analog zu den anderen Work Items, werden im Abschlussstatus die Felder `Microsoft.VSTS.Common.ClosedBy` und `Microsoft.VSTS.Common.ClosedDate` schreibgeschützt (Regeln `<WHENNOTCHANGED/>` und `<READONLY/>`). Ferner ist für das Feld `Microsoft.VSTS.Common.ClosedBy` ein gültiger Systembenutzer erforderlich (Regel `<VALIDUSER/>`).

```
<FIELD reportable="dimension" type="String" name="Geschlossen von" refname="Microsoft.
    VSTS.Common.ClosedBy">
  <VALIDUSER />
  <WHENNOTCHANGED field="System.State">
    <READONLY /> </WHENNOTCHANGED>
</FIELD>
<FIELD reportable="dimension" type="DateTime" name="Schließungsdatum" refname="Microsoft
    .VSTS.Common.ClosedDate">
  <WHENNOTCHANGED field="System.State">
    <READONLY /> </WHENNOTCHANGED>
</FIELD>
```

Das Feld `VMXT.Likelihood` gibt ein synthetisches Maß für die Eintrittswahrscheinlichkeit eines Risikos in % an. Um eine diskrete Skala zu erhalten, wird der Wertebereich dieses Felds mit der Regel `<ALLOWEDVALUES/>` eingeschränkt.

```
<FIELD type="Integer" name="Wahrscheinlichkeit (VMXT)" refname="VMXT.Likelihood">
  <HELPTEXT>Eintrittwahrscheinlichkeit des Risikos in %.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="90" />
    <LISTITEM value="70" />
    <LISTITEM value="50" />
    <LISTITEM value="30" />
    <LISTITEM value="10" />
  </ALLOWEDVALUES>
</FIELD>
```

Sofern Risiken identifiziert wurden, sind mögliche Auswirkungen auf Zeit (Feld `VMXT.ImpactTime`), Qualität (Feld `VMXT.ImpactQuality`), Budget (Feld `VMXT.ImpactBudget`) und Umfang (Feld `VMXT.ImpactScope`) abzuschätzen. Alle Felder sind über einem definierten Wertebereich mithilfe der Regel `<ALLOWEDVALUES/>` definiert.

```
<FIELD type="String" name="Auswirkung␣auf␣Zeit␣(VMXT)" refname="VMXT.ImpactTime">
  <HELPTEXT>Qualitative Auswirkung des Risikos auf die Zeit.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Kritisch" />
    <LISTITEM value="Hoch" />
    <LISTITEM value="Mittel" />
    <LISTITEM value="Gering" />
    <LISTITEM value="Keine" />
  </ALLOWEDVALUES>
</FIELD>

<FIELD type="String" name="Auswirkung␣auf␣Qualität␣(VMXT)" refname="VMXT.ImpactQuality">
  <HELPTEXT>Qualitative Auswirkung des Risikos auf die Qualität.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Kritisch" />
    <LISTITEM value="Hoch" />
    <LISTITEM value="Mittel" />
    <LISTITEM value="Gering" />
    <LISTITEM value="Keine" />
  </ALLOWEDVALUES>
</FIELD>

<FIELD type="String" name="Auswirkung␣auf␣Budget␣(VMXT)" refname="VMXT.ImpactBudget">
  <HELPTEXT>Qualitative Auswirkung des Risikos auf das Budget.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Kritisch" />
    <LISTITEM value="Hoch" />
    <LISTITEM value="Mittel" />
    <LISTITEM value="Gering" />
    <LISTITEM value="Keine" />
  </ALLOWEDVALUES>
</FIELD>

<FIELD type="String" name="Auswirkung␣auf␣Umfang␣(VMXT)" refname="VMXT.ImpactScope">
  <HELPTEXT>Qualitative Auswirkung des Risikos auf den Funktionsumfang.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Kritisch" />
    <LISTITEM value="Hoch" />
    <LISTITEM value="Mittel" />
    <LISTITEM value="Gering" />
    <LISTITEM value="Keine" />
  </ALLOWEDVALUES>
</FIELD>
```

Für ein identifiziertes Risiko kann eine Managementstrategie angegeben werden (Feld `VMXT.StrategyType`). Mithilfe der Regel `<ALLOWEDVALUES/>` wird der gültige Wertebereich festgelegt.

```
<FIELD type="String" name="Strategietyp␣(VMXT)" refname="VMXT.StrategyType">
  <HELPTEXT>Typ der Risikostrategie.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Vermeidung" />
    <LISTITEM value="Transfer" />
    <LISTITEM value="Akzeptanz" />
  </ALLOWEDVALUES>
</FIELD>
```

Ein identifiziertes Risiko kann in eine Risikoklasse (Feld `VMXT.RiskClass`) eingeordnet werden. Mithilfe der Regel `<ALLOWEDVALUES/>` wird der gültige Wertebereich festgelegt.

```
<FIELD type="String" name="Risikoklasse␣(VMXT)" refname="VMXT.RiskClass">
  <HELPTEXT>Qualitative Einstufung des Risikos anhand des Risikomasses.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Katastrophal" />
    <LISTITEM value="Kritisch" />
    <LISTITEM value="Unerwünscht" />
    <LISTITEM value="Tolerierbar" />
  </ALLOWEDVALUES>
</FIELD>
```

Für den Workflow sind in die Zuständen *In Bearbeitung*, *Gelöst* und *Erledigt* Regeln für Felder hinterlegt. Im Zustand *In Bearbeitung* ist das Feld `System.AssignedTo` durch die Regel `<REQUIRED/>` mit einem Wert zu belegen. Im Zustand *Gelöst* wird das Feld `VMXT.Assessment` durch die Regel `<EMPTY/>` zurückgesetzt. Im Zustand *Erledigt* (Abschlusszustand) werden die Felder `Microsoft.VSTS.Common.ClosedDate` und `Microsoft.VSTS.Common.ClosedBy` durch die Regel

<SERVERDEFAULT/> durch den TFS-Server mit Werten belegt. Das Feld VMXT.AssignedTo wird durch die Regel <EMPTY/> zurückgesetzt. Das Feld VMXT.Assessment wird durch die Regel <REQUIRED/> zwingend mit einem Wert belegt.

```
<STATE value="In␣Bearbeitung">
  <FIELDS>
    <FIELD refname="System.AssignedTo"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="Gelöst">
  <FIELDS>
    <FIELD refname="VMXT.Assessment"> <EMPTY /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="Erledigt">
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
      <SERVERDEFAULT from="clock" /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
      <SERVERDEFAULT from="currentuser" /> </FIELD>
    <FIELD refname="VMXT.Assessment"> <REQUIRED /> </FIELD>
    <FIELD refname="System.AssignedTo"> <EMPTY /> </FIELD>
  </FIELDS>
</STATE>
```

Im Zustandsübergang *Gelöst⇒In Bearbeitung* muss das Feld VMXT.Assessment mit einem Wert belegt werden. Die Regel <REQUIRED/> sorgt dafür. Bei der Wiedereröffnung eines Risikos im Zustandsübergang *Erledigt⇒Aktiv* müssen die beiden Felder Microsoft.VSTS.Common.ClosedBy und Microsoft.VSTS.Common.ClosedDate zurückgesetzt werden. Die Regel <EMPTY/> definiert das. Zusätzlich wird über dieselbe Regel das Feld VMXT.Assessment zurückgesetzt.

```
<TRANSITION from="Gelöst" to="In␣Bearbeitung">
  <REASONS>
    <DEFAULTREASON value="Nicht␣gelöst" />
  </REASONS>
  <FIELDS>
    <FIELD refname="VMXT.Assessment"> <REQUIRED /> </FIELD>
  </FIELDS>
</TRANSITION>
<TRANSITION from="Erledigt" to="Aktiv">
  <REASONS>
    <DEFAULTREASON value="Reaktiviert">
      <FIELDS>
        <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
          <EMPTY /> </FIELD>
        <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
          <EMPTY /> </FIELD>
      </FIELDS>
    </DEFAULTREASON>
  </REASONS>
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy"> <EMPTY /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate"> <EMPTY /> </FIELD>
    <FIELD refname="VMXT.Assessment"> <EMPTY /> </FIELD>
  </FIELDS>
</TRANSITION>
```

Regeln für Maßnahmen. Im Kontext von Risiken sind auch Maßnahmen definiert. Zusächst sind wieder die Standardregeln zu finden. Auch Maßnahmen haben eine Bezeichnung (Feld System.Title mit der Regel <REQUIRED/>) und einen gültigen Systembenutzer (Feld System.AssignedTo mit der Regel <VALIDUSER/>) dem die Maßnahmen zugewiesen ist.

```
<FIELD reportable="dimension" type="String" name="Titel" refname="System.Title">
  <REQUIRED /> </FIELD>
<FIELD reportable="dimension" type="String" name="Zugewiesen␣an" refname="System.
    AssignedTo">
  <VALIDUSER /> </FIELD>
```

Analog zu den anderen Work Items, werden im Abschlussstatus die Felder Microsoft.VSTS.Common.ClosedBy und Microsoft.VSTS.Common.ClosedDate schreibgeschützt (Regeln <WHENNOTCHANGED/> und <READONLY/>). Ferner ist für das Feld Microsoft.VSTS.Common.ClosedBy ein gültiger Systembenutzer erforderlich (Regel <VALIDUSER/>).

```
<FIELD reportable="dimension" type="DateTime" name="Schließungsdatum" refname="Microsoft
    .VSTS.Common.ClosedDate">
  <WHENNOTCHANGED field="System.State">
    <READONLY /> </WHENNOTCHANGED>
</FIELD>
<FIELD reportable="dimension" type="String" name="Geschlossen␣von" refname="Microsoft.
    VSTS.Common.ClosedBy">
  <VALIDUSER />
  <WHENNOTCHANGED field="System.State">
    <READONLY /> </WHENNOTCHANGED>
</FIELD>
```

In den Zuständen *In Bearbeitung*, *Erledigt* und *Gelöst* werden ebenfalls wieder Felder gesetzt. Im Zustand *In Bearbeitung* wird das Feld `System.AssignedTo` durch die Regel `<REQUIRED/>` als verpflichtend auszufüllen gekennzeichnet. Im Zustand *Erledigt* sind die Felder `Microsoft.VSTS.Common.ClosedDate` und `Microsoft.VSTS.Common.ClosedBy` durch die Regel `<SERVERDEFAULT/>` mit Werten vom Server zu befüllen. Das Feld `VMXT.Assessment` wird durch die Regel `<REQUIRED/>` als zwingend auszufüllen markiert und das Feld `System.AssignedTo` wird aufgrund der Regel `<EMPTY/>` in diesem Zustand zurückgesetzt.

```
<STATE value="In␣Bearbeitung">
  <FIELDS>
    <FIELD refname="System.AssignedTo"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="Erledigt">
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
      <SERVERDEFAULT from="clock" /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
      <SERVERDEFAULT from="currentuser" /> </FIELD>
    <FIELD refname="VMXT.Assessment"> <REQUIRED /> </FIELD>
    <FIELD refname="System.AssignedTo"> <EMPTY /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="Gelöst">
  <FIELDS>
    <FIELD refname="VMXT.Assessment"> <EMPTY /> </FIELD>
  </FIELDS>
</STATE>
```

Im Zustandsübergang *Gelöst⇒In Bearbeitung* ist das Feld `VMXT.Assessment` mit einem Wert zu belegen. Die Regel `<REQUIRED/>` setzt das durch. Im Übergang *Erledigt⇒Aktiv* werden die Felder `Microsoft.VSTS.Common.ClosedBy`, `Microsoft.VSTS.Common.ClosedDate` und `VMXT.Assessment` mithilfe der Regel `<EMPTY/>` zurückgesetzt.

```
<TRANSITION from="Gelöst" to="In␣Bearbeitung">
  <REASONS>
    <DEFAULTREASON value="Nacharbeit␣erforderlich" />
  </REASONS>
  <FIELDS>
    <FIELD refname="VMXT.Assessment"> <REQUIRED /> </FIELD>
  </FIELDS>
</TRANSITION>
<TRANSITION from="Erledigt" to="Aktiv">
  <REASONS>
    <DEFAULTREASON value="Reaktiviert" />
  </REASONS>
  <FIELDS>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy"> <EMPTY /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate"> <EMPTY /> </FIELD>
    <FIELD refname="VMXT.Assessment"> <EMPTY /> </FIELD>
  </FIELDS>
</TRANSITION>
```

B.2.6. Work Item: PÄM

Das Problem- und Änderungsmanagement (PÄM) modellieren wir als integrierten, einfachen Prozess (vgl. Kapitel 4.2.7). Obwohl wir uns um weitgehende Einfachheit bemüht haben, ist der

PÄM-Prozess sehr umfangreich ausgefallen, was sich auch in den Formularen für den Anwender zeigt.

Im oberen Teil des Formulars (Abbildung B.9) sind wieder die TFS-üblichen Felder für Bezeichnungen und Klassifikation in Bereiche/Iterationen zu finden. Im darunter liegenden Statusbereich sind wieder *Bearbeiter* und *Autor* sowie *Status* und *Grund* anzugeben. Der Workflow (gültige) Statusübergänge ist ebenfalls in Kapitel 4.2.7 beschrieben. Zusätzlich ist für ein PÄM eine *Kategorie* anzugeben und es ist vom Anwender eine Aussage zu treffen, ob das PÄM *Reproduzierbar* ist. Weiterhin gibt es einen Identifikationsbereich. Hier sind Felder vorgesehen, die eine Benennung des *Gegenstand*s und die Angabe einer *Version* erlauben. Mit diesen Informationen ist ein Problem- oder Änderungsgegenstand eindeutig identifizierbar.

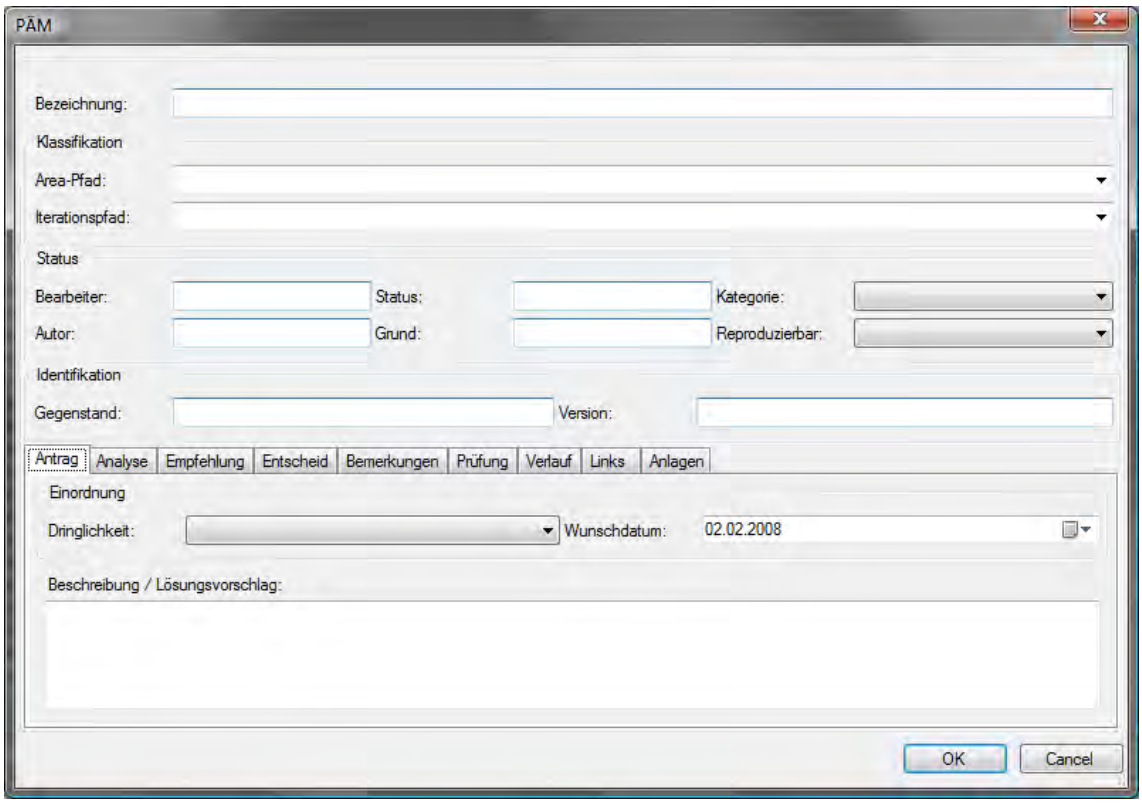


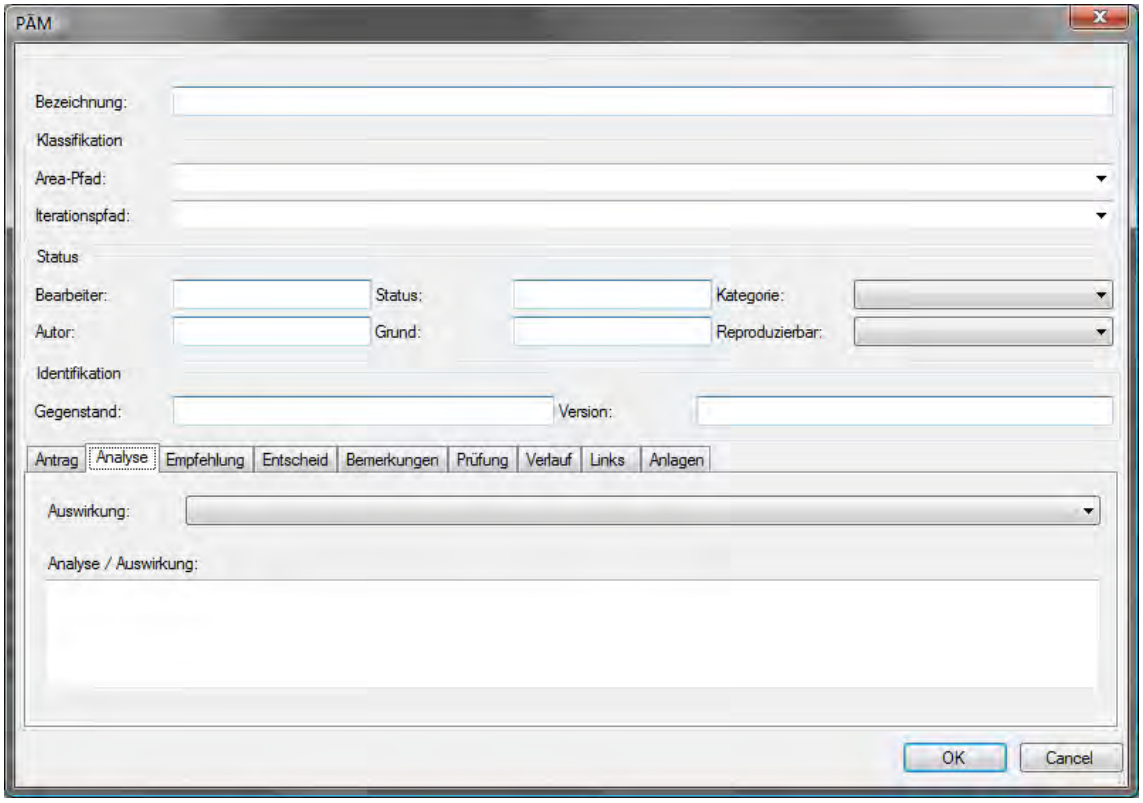
Abbildung B.9.: Formlayout für das Work Item PÄM 1

Der Detailbereich variiert dann wieder (Abbildungen B.9, B.10 und B.11). Zuerst betrachten wie den Reiter *Antrag*: Hier werden Daten erfasst, die für einen Änderungsantrag relevant sind. Es sind Felder für *Dringlichkeit*, *Wunschdatum* (für die Umsetzung) sowie eine/n *Beschreibung/Lösungsvorschlag* vorhanden. Die Reiter *Analyse* und *Empfehlung* sind in Abbildung B.10 zu sehen. Für die Analyse sind Felder vorhanden, die eine Klassifikation hinsichtlich einer *Auswirkung* (Liste) und eine entsprechende Beschreibung (*Analyse/Auswirkung*) erlauben. Für den Reiter Empfehlung sind analog Felder für eine *Empfehlung* (Liste) und eine Beschreibung mit Begründung (*Begründung/Lösungsvorschlag*).

Abbildung B.11 zeigt den Reiter *Entscheid*, der wie auch die üblichen Beschreibungsfelder als einfaches Textfeld hinterlegt ist. Weiterhin sind im Formular für diesen Work Item Typ die üblichen Eingabefelder für Bemerkungen, Logs sowie die Link-Felder für Anlagen verfügbar.

Regeln für PÄMs. Auch PÄMs haben eine Bezeichnung (Feld `System.Title` mit der Regel `<REQUIRED/>`) und einen gültigen Systembenutzer (Feld `System.AssignedTo` mit der Regel `<VALIDUSER/>`) dem das PÄMs zugewiesen ist.

```
<FIELD type="String" name="Titel" refname="System.Title">
  <REQUIRED /> </FIELD>
<FIELD type="String" name="Zugewiesen an" refname="System.AssignedTo">
  <VALIDUSER /> </FIELD>
```



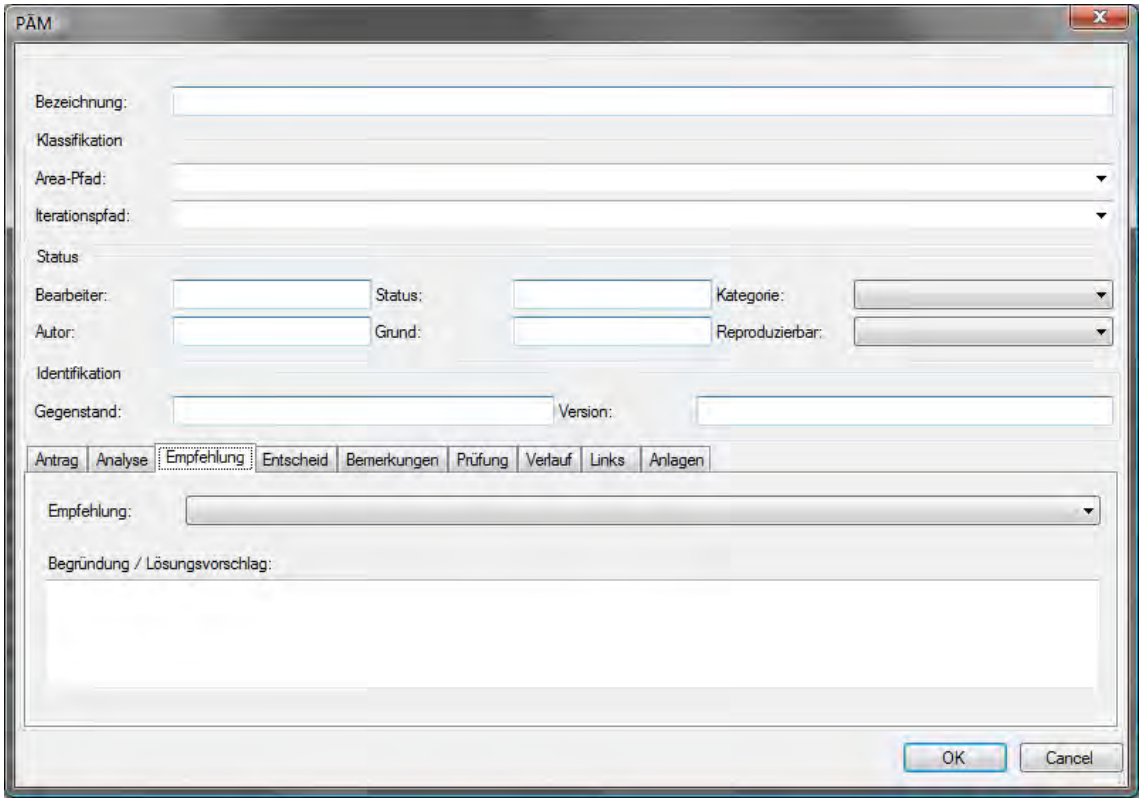


Abbildung B.10.: Formlayout für das Work Item PÄM 2

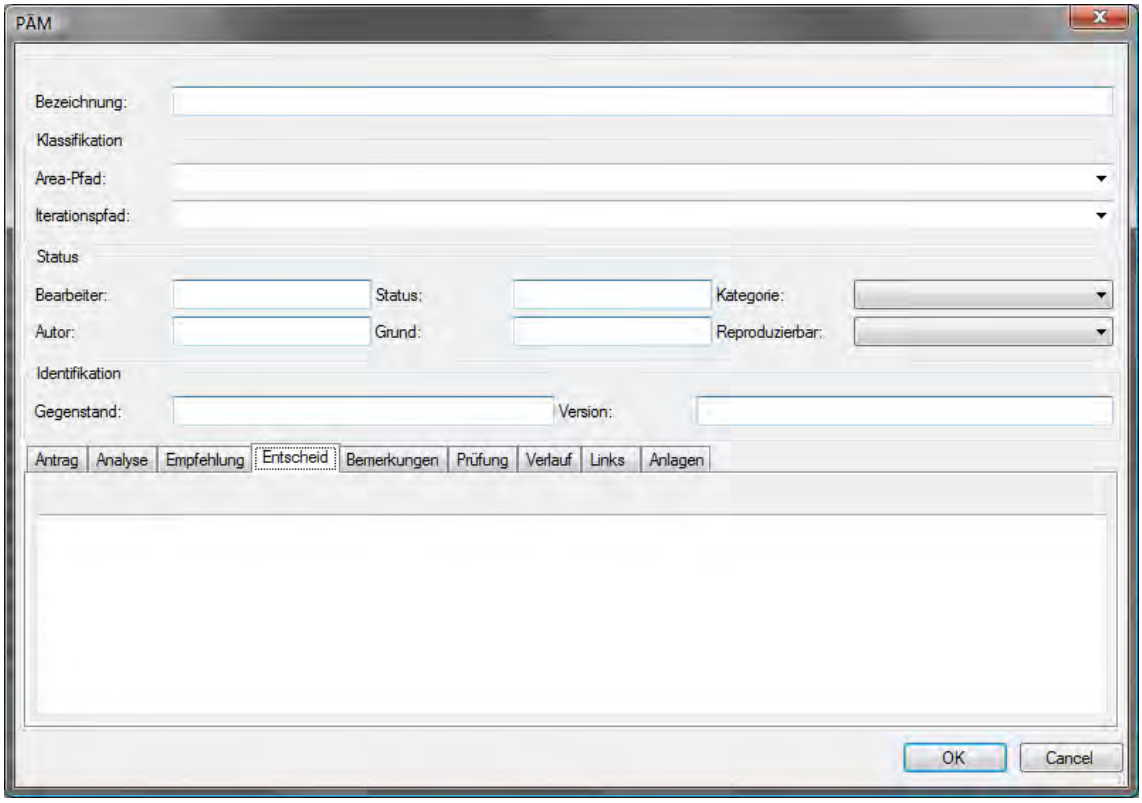


Abbildung B.11.: Formlayout für das Work Item PÄM 3

Analog zu den anderen Work Items, werden im Abschlussstatus die Felder `Microsoft.VSTS.Common.ClosedBy` und `Microsoft.VSTS.Common.ClosedDate` schreibgeschützt (Regeln `<WHENNOTCHANGED/>` und `<READONLY/>`). Ferner ist für das Feld `Microsoft.VSTS.Common.ClosedBy` ein gültiger Systembenutzer erforderlich (Regel `<VALIDUSER/>`).

```
<FIELD reportable="dimension" type="DateTime" name="Schließungsdatum" refname="Microsoft
    .VSTS.Common.ClosedDate">
  <WHENNOTCHANGED field="System.State">
    <READONLY /> </WHENNOTCHANGED>
</FIELD>
<FIELD reportable="dimension" type="String" name="Geschlossen␣von" refname="Microsoft.
    VSTS.Common.ClosedBy">
  <VALIDUSER />
  <WHENNOTCHANGED field="System.State">
    <READONLY /> </WHENNOTCHANGED>
</FIELD>
```

PÄMs haben i. d. R. Auswirkungen auf den Projektablauf. Im Feld `VMXT.ImpactOnProjectPromise` können diese erfasst werden. Über die Regel `<ALLOWEDVALUES/>` wird der zulässige Wertebereich festgelegt. Da dieses Feld mit einem Wert belegt werden muss (Regel `<REQUIRED/>`) ist durch die Regel `<DEFAULT/>` ein Standardwert definiert.

```
<FIELD type="String" name="Auswirkungen␣auf␣Projektziel␣(VMXT)" refname="VMXT.
    ImpactOnProjectPromise">
  <HELPTEXT>Auswirkung auf das Projekt/Tragweite.</HELPTEXT>
  <REQUIRED />
  <ALLOWEDVALUES>
    <LISTITEM value="Kritisch" />
    <LISTITEM value="Hoch" />
    <LISTITEM value="Mittel" />
    <LISTITEM value="Niedrig" />
  </ALLOWEDVALUES>
  <DEFAULT from="value" value="Niedrig" />
</FIELD>
```

Die Kategorie des PÄMs (Feld `VMXT.Kategorie`) weist ihn als Problemmeldung oder als Änderungsantrag aus. Dieser Wertebereich wird durch die Regel `<ALLOWEDVALUES/>` festgelegt. Das

Feld muss mit einem Wert belegt werden (Regel <REQUIRED/>). Vorgaben werden aber nicht gemacht.

```
<FIELD type="String" name="Kategorie␣(VMXT)" refname="VMXT.Kategorie">
  <HELPTEXT>Die Kategorie der PÄM</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Änderungsantrag" />
    <LISTITEM value="Problemmeldung" />
  </ALLOWEDVALUES>
  <REQUIRED />
</FIELD>
```

Das Feld VMXT.Reproduzierbar zeigt an, ob ein PÄM reproduzierbar ist. Die Regel <ALLOWEDVALUES/> legt wieder den gültigen Wertebereich fest. Es muss ein Wert angegeben werden (Regel <REQUIRED/>).

```
<FIELD type="String" name="Reproduzierbar␣(VMXT)" refname="VMXT.Reproduzierbar">
  <HELPTEXT>Flag, das anzeigt, ob die PÄM reproduzierbar auftritt.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Ja" />
    <LISTITEM value="Nein" />
    <LISTITEM value="Nicht␣relevant" />
  </ALLOWEDVALUES>
  <REQUIRED />
</FIELD>
```

Im Feld VMXT.Urgency wird die Dringlichkeit des PÄMs eingestuft. Die Regel <ALLOWEDVALUES/> legt wieder den gültigen Wertebereich fest.

```
<FIELD reportable="dimension" type="String" name="Dringlichkeit␣(VMXT)" refname="VMXT.
    Urgency">
  <HELPTEXT>Dringlichkeit des Work Items.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Kritisch" />
    <LISTITEM value="Sehr␣wichtig" />
    <LISTITEM value="Wichtig" />
    <LISTITEM value="Wünschenswert" />
  </ALLOWEDVALUES>
</FIELD>
```

Zu einem PÄM kann im Feld VMXT.Recommendation eine Entscheidungsempfehlung gegeben werden. Die Regel <ALLOWEDVALUES/> legt wieder den gültigen Wertebereich fest.

```
<FIELD type="String" name="Empfehlung␣(VMXT)" refname="VMXT.Recommendation">
  <HELPTEXT>Entscheidungsempfehlung für dieses Work Item.</HELPTEXT>
  <ALLOWEDVALUES>
    <LISTITEM value="Ablehnen" />
    <LISTITEM value="Annehmen" />
    <LISTITEM value="Zurückstellen" />
  </ALLOWEDVALUES>
</FIELD>
```

In den verschiedenen Zuständen eines PÄMs werden auch wieder verschiedene Felder mit Werten belegt. Im Zustand *In Bewertung* wird das Feld System.AssignedTo durch die Regel <REQUIRED/> als zwingend auszufüllen markiert. Im Zustand *In Entscheidung* wird das Feld System.AssignedTo durch die Regel <REQUIRED/> als zwingend auszufüllen markiert. Im Zustand *In Bearbeitung* werden die Felder System.AssignedTo und VMXT.Decision durch die Regel <REQUIRED/> als zwingend auszufüllen markiert. Der Zustand *Gelöst* setzt das Feld VMXT.Assessment durch die Regel <EMPTY/> zurück und markiert das Feld VMXT.Decision durch die Regel <REQUIRED/> als zwingend auszufüllen. Im Zustand *Erledigt* werden die Felder Microsoft.VSTS.Common.ClosedDate und Microsoft.VSTS.Common.ClosedBy wegen der Regel <SERVERDEFAULT/> mit Werten vom Server gefüllt. Das Feld VMXT.Assessment wird wegen der Regel <REQUIRED/> zum zwingenden Ausfüllen markiert, während das Feld System.AssignedTo wegen der Regel <EMPTY/> zurückgesetzt wird.

```
<STATE value="In␣Bewertung">
  <FIELDS>
    <FIELD refname="System.AssignedTo"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="In␣Entscheidung">
  <FIELDS>
```

```
    <FIELD refname="System.AssignedTo"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="In_Bearbeitung">
  <FIELDS>
    <FIELD refname="System.AssignedTo"> <REQUIRED /> </FIELD>
    <FIELD refname="VMXT.Decision"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="Gelöst">
  <FIELDS>
    <FIELD refname="VMXT.Assessment"> <EMPTY /> </FIELD>
    <FIELD refname="VMXT.Decision"> <REQUIRED /> </FIELD>
  </FIELDS>
</STATE>
<STATE value="Erledigt">
  <FIELDS>
    <FIELD refname="VMXT.Assessment"> <REQUIRED /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate">
        <SERVERDEFAULT from="clock" /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy">
      <SERVERDEFAULT from="currentuser" /> </FIELD>
    <FIELD refname="System.AssignedTo"> <EMPTY /> </FIELD>
  </FIELDS>
</STATE>
```

Im Zustandsübergang *Erledigt⇒Vorgeschlagen* werden bei der Wiedereröffnung des Work Items die Felder `VMXT.Assessment`, `Microsoft.VSTS.Common.ClosedDate` und `Microsoft.VSTS.-Common.ClosedBy` wegen der Regel `<EMPTY/>` zurückgesetzt.

```
<TRANSITION from="Erledigt" to="Vorgeschlagen">
  <REASONS>
    <DEFAULTREASON value="Wiedereröffnung" />
  </REASONS>
  <FIELDS>
    <FIELD refname="VMXT.Assessment"> <EMPTY /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedDate"> <EMPTY /> </FIELD>
    <FIELD refname="Microsoft.VSTS.Common.ClosedBy"> <EMPTY /> </FIELD>
  </FIELDS>
</TRANSITION>
```

B.2.7. Empfehlung für das Work Item Formular-Design

Es gibt von Microsoft *keine* verbindlichen Vorgaben für das Design von Work Items oder deren Formularen. Viele Ergebnisse dieses Teilprojekts sind zunächst im *trail and error*-Verfahren umgesetzt worden. Mit der Zeit hat sich jedoch (auch im Dialog mit Microsoft) konsolidiert, dass die Standardfelder für *Bezeichnung*, *Klassifikation*, *Bearbeiter* und *Status* immer vorhanden sein sollten. Layout-Vorgaben gibt es wiederum nicht. Die Layouts, die wir in diesem Abschnitt vorgestellt haben, orientieren sich an den Layouts der MSF Work Items.

Wir können nach Abschluss dieses Designs empfehlen, immer Bereiche für oben genannte Punkte in der oberen Hälfte des Formulars bereitzustellen (vgl. Abbildung B.2) im unteren Teil sollten dann zunächst die unmittelbar notwendigen Daten erfasst und angezeigt werden. Weitere Detailinformationen sollten auf weiteren Indexzungen (Reitern) untergebracht werden.

Weiterhin wichtig ist die Benennung der Work Item Felder, die eineindeutig sein müssen. Dabei wird zur Herstellung der Identität die Kombination der Felder `Name` und `RefName` verwendet. Dabei ist der `RefName` eine gewisse Konstante, wie wir in der Englischübersetzung im nächsten Abschnitt sehen werden.

B.3. Übersetzung für englische Systeme

Process Templates für den TFS sind nicht über Sprachen generisch, sondern immer an eine Sprache gebunden. besonders kritisch sind dabei die Work Items, die auf der Ebene der Datenstruktur lokalisiert werden müssen. In Konsequenz müssen dann auch Queries und ggf. Reports angepasst werden. In diesem Abschnitt geben wir für die Work Item Typen, die wir im Rahmen von CollabXT-TFS betrachten die Abbildung von Deutsch ⇔ Englisch an.

B.3.1. Anpassung der Work Items

Bei der Übersetzung (sprachlichen Anpassung) eines Process Templates, sind verschiedene Bereiche zu berücksichtigen. Zuerst sind hier die *Work Item Typen* zu nennen. Die Datenstrukturen müssen für die jeweiligen Sprachversionen angepasst werden. Besonders zu beachten ist die Benennung von Standardfeldern, wie z. B. dem Titel oder dem Verlauf. Diese Standardfelder müssen in der Kombination *Name* + *RefName* immer identisch bleiben. Konkret bedeutet dies für ein deutsches System heißt die Titel-Kombination immer *Titel, System.Title*; für ein englisches System heißt die Titelkombination immer *Title, System.Title*. Insbesondere der *RefName* darf hier nicht verändert werden. Ein Eingriff in diese Kombinationen kann dazu führen, dass Process Templates nicht mehr instanziieren.

In den Tabellen B.1 bis B.7 haben wir die Datenstrukturen der in Kapitel 4.2 entworfenen Work Item Typen englische Systeme übersetzt. Diese Übersetzung bezieht sich jedoch nur auf die Strukturen und hat noch keine Auswirkungen auf die Feldbezeichnungen in den Formularen (Anhang B.2). Diese ist von den Strukturen unabhängig.

B.3.2. Anpassung der Queries und Reports

Entsprechend der Anpassungen an den Work Item Typen sind auch die sich darauf beziehenden Queries und Reports anzupassen. Bei den Queries genügt hierbei eine einfache Aktualisierung sich ggf. geänderter bezeichner. Die Reports sind hingegen in weiten Bereichen sprachspezifisch und müssen daher neu erstellt werden.

B.3.3. Anpassung des Metatemplates

Für die Anpassung der Sprachversionen sind unter Umständen weiterhin Änderungen am Generatorcode oder am Metatemplate erforderlich. Zu den dafür notwendigen Informationen verweisen wir auf `http://www.codeplex.com/VModellXTTFS`.

Abbildungstabellen für die Work Item Typen

Name (DE)	Name (EN)	Reference Name
Titel	Title	System.Title
Zustand	State	System.State
Grund	Reason	System.Reason
Iterationspfad	Iteration Path	System.IterationPath
Zugewiesen an	Assigned To	System.AssignedTo
Beschreibung	Description	System.Description
Verlauf	History	System.History
Bereichspfad	Area Path	System.AreaPath
Geschlossen von	Closed By	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Closed Date	Microsoft.VSTS.Common.ClosedDate
Verbleibende Arbeit	Remaining Work	Microsoft.VSTS.Scheduling.RemainingWork
Startdatum	Start Date	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Finish Date	Microsoft.VSTS.Scheduling.FinishDate
Produkttyp (VMXT)	Product Type (VMXT)	VMXT.ProduktTyp
Erzeugung (VMXT)	Creation Style (VMXT)	VMXT.Erzeugung
vmxt_ref (VMXT)	vmxt_ref (VMXT)	VMXT.id
Bemerkungen (VMXT)	Log (VMXT)	VMXT.Log
Prüfung (VMXT)	Assessment (VMXT)	VMXT.Assessment

Tabelle B.1.: Datenstruktur (DE/EN) für Produkt-Work Items

Name (DE)	Name (EN)	Reference Name
Titel	Title	System.Title
Zustand	State	System.State
Grund	Reason	System.Reason
Iterationspfad	Iteration Path	System.IterationPath
Zugewiesen an	Assigned To	System.AssignedTo
Beschreibung	Description	System.Description
Verlauf	History	System.History
Bereichspfad	Area Path	System.AreaPath
Geschlossen von	Closed By	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Closed Date	Microsoft.VSTS.Common.ClosedDate
Verbleibende Arbeit	Remaining Work	Microsoft.VSTS.Scheduling.RemainingWork
Startdatum	Start Date	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Finish Date	Microsoft.VSTS.Scheduling.FinishDate
p_ref (VMXT)	p_ref (VMXT)	VMXT.PRef
ep_ref (VMXT)	ep_ref (VMXT)	VMXT.EPRef
vmxt_ref (VMXT)	vmxt_ref (VMXT)	VMXT.id
Bemerkungen (VMXT)	Log (VMXT)	VMXT.Log

Tabelle B.2.: Datenstruktur (DE/EN) für Aktivität-Work Items

Name (DE)	Name (EN)	Reference Name
Titel	Title	System.Title
Zustand	State	System.State
Grund	Reason	System.Reason
Iterationspfad	Iteration Path	System.IterationPath
Zugewiesen an	Assigned To	System.AssignedTo
Beschreibung	Description	System.Description
Verlauf	History	System.History
Bereichspfad	Area Path	System.AreaPath
Geschlossen von	Closed By	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Closed Date	Microsoft.VSTS.Common.ClosedDate
Verbleibende Arbeit	Remaining Work	Microsoft.VSTS.Scheduling. ↩ RemainingWork
Startdatum	Start Date	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Finish Date	Microsoft.VSTS.Scheduling.FinishDate
Entscheidungspunkttyp ↩ (VMXT)	Decision Gate ↩ Type (VMXT)	VMXT.Entscheidungspunkttyp
Prüfung (VMXT)	Assessment (VMXT)	VMXT.Assessment
vmxt_ref (VMXT)	vmxt_ref (VMXT)	VMXT.id
Bemerkungen (VMXT)	Log (VMXT)	VMXT.Log

Tabelle B.3.: Datenstruktur (DE/EN) für Entscheidungspunkt-Work Items

Name (DE)	Name (EN)	Reference Name
Titel	Title	System.Title
Zustand	State	System.State
Grund	Reason	System.Reason
Iterationspfad	Iteration Path	System.IterationPath
Zugewiesen an	Assigned To	System.AssignedTo
Beschreibung	Description	System.Description
Verlauf	History	System.History
Bereichspfad	Area Path	System.AreaPath
Geschlossen von	Closed By	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Closed Date	Microsoft.VSTS.Common.ClosedDate
Priorität	Priority	Microsoft.VSTS.Common.Priority
Schweregrad	Severity	Microsoft.VSTS.Common.Severity
Verbleibende Arbeit	Remaining Work	Microsoft.VSTS.Scheduling.RemainingWork
Abgeschlossene Arbeit	Completed Work	Microsoft.VSTS.Scheduling.CompletedWork
Startdatum	Start Date	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Finish Date	Microsoft.VSTS.Scheduling.FinishDate
Aufgabenhierarchie	Task Hierarchy	Microsoft.VSTS.Scheduling.TaskHierarchy
Disziplin (VMXT)	Discipline (VMXT)	VMXT.Disciplin
Schätzung (VMXT)	Estimate (VMXT)	VMXT.Estimate
Blockiert (VMXT)	Blocked (VMXT)	VMXT.Blocked
Prüfung (VMXT)	Assessment (VMXT)	VMXT.Assessment
Bemerkungen (VMXT)	Log (VMXT)	VMXT.Log

Tabelle B.4.: Datenstruktur (DE/EN) für Arbeitsauftrag-Work Items

Name (DE)	Name (EN)	Reference Name
Titel	Title	System.Title
Zustand	State	System.State
Grund	Reason	System.Reason
Iterationspfad	Iteration Path	System.IterationPath
Zugewiesen an	Assigned To	System.AssignedTo
Beschreibung	Description	System.Description
Verlauf	History	System.History
Bereichspfad	Area Path	System.AreaPath
Erstellungsdatum	Created Date	System.CreatedDate
Geschlossen von	Closed By	Microsoft.VSTS.Common. ↩ ClosedBy
Schließungsdatum	Closed Date	Microsoft.VSTS.Common. ↩ ClosedDate
Wahrscheinlichkeit (VMXT)	Likelihood (VMXT)	VMXT.Likelihood
Auswirkung auf Zeit (VMXT)	Impact Time (VMXT)	VMXT.ImpactTime
Auswirkung auf Qualität (VMXT)	Impact Quality (VMXT)	VMXT.ImpactQuality
Auswirkung auf Budget (VMXT)	Impact Budget (VMXT)	VMXT.ImpactBudget
Auswirkung auf Umfang (VMXT)	Impact Scope (VMXT)	VMXT.ImpactScope
Risikoschaden (VMXT)	Severity (VMXT)	VMXT.Severity
Strategietyp (VMXT)	Strategy Type (VMXT)	VMXT.StrategyType
Strategiebeschreibung ↩ (VMXT)	Strategy Description ↩ (VMXT)	VMXT.StrategyDescription
Maßnahmenbeschreibung ↩ (VMXT)	Measure Comment ↩ (VMXT)	VMXT.MeasureComment
Risikomaß (VMXT)	Risk Impact (VMXT)	VMXT.RiskImpact
Risikoklasse (VMXT)	Risk Class (VMXT)	VMXT.RiskClass
Prüfung (VMXT)	Assessment (VMXT)	VMXT.Assessment
Bemerkungen (VMXT)	Log (VMXT)	VMXT.Log
Autor (VMXT)	Author (VMXT)	VMXT.Autor

Tabelle B.5.: Datenstruktur (DE/EN) für Risiko-Work Items

Name (DE)	Name (EN)	Reference Name
Titel	Title	System.Title
Zustand	State	System.State
Grund	Reason	System.Reason
Iterationspfad	Iteration Path	System.IterationPath
Zugewiesen an	Assigned To	System.AssignedTo
Beschreibung	Description	System.Description
Verlauf	History	System.History
Bereichspfad	Area Path	System.AreaPath
Geschlossen von	Closed By	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Closed Date	Microsoft.VSTS.Common.ClosedDate
Abschlussdatum	Finish Date	Microsoft.VSTS.Scheduling.FinishDate
Prüfung (VMXT)	Assessment (VMXT)	VMXT.Assessment
Bemerkungen (VMXT)	Log (VMXT)	VMXT.Log
Trigger (VMXT)	Trigger (VMXT)	VMXT.Trigger

Tabelle B.6.: Datenstruktur (DE/EN) für Maßnahme-Work Items

Name (DE)	Name (EN)	Reference Name
Titel	Title	System.Title
Zustand	State	System.State
Grund	Reason	System.Reason
Iterationspfad	Iteration Path	System.IterationPath
Zugewiesen an	Assigned To	System.AssignedTo
Beschreibung	Description	System.Description
Verlauf	History	System.History
Bereichspfad	Area Path	System.AreaPath
Geschlossen von	Closed By	Microsoft.VSTS.Common.ClosedBy
Schließungsdatum	Closed Date	Microsoft.VSTS.Common.ClosedDate
Verbleibende Arbeit	Remaining Work	Microsoft.VSTS.Scheduling. ↩ RemainingWork
Startdatum	Start Date	Microsoft.VSTS.Scheduling.StartDate
Abschlussdatum	Finish Date	Microsoft.VSTS.Scheduling.FinishDate
Prüfung (VMXT)	Assessment (VMXT)	VMXT.Assessment
Bemerkungen (VMXT)	Log (VMXT)	VMXT.Log
Empfehlung (VMXT)	Recommendation (VMXT)	VMXT.Recommendation
Entscheid (VMXT)	Decision (VMXT)	VMXT.Decision
Analyse (VMXT)	Analysis (VMXT)	VMXT.Analysis
Auswirkungen auf ↩ Projektziel (VMXT)	Impact On Project ↩ Promise (VMXT)	VMXT.ImpactOnProjectPromise
Tatsächliche Lösung ↩ (VMXT)	Corrective Action ↩ Actual Solution (VMXT)	VMXT.CorrectiveActionActualSolution
Gewünschter ↩ Fertigstellungszeitpunkt (VMXT)	Target Date ↩ (VMXT)	VMXT.TargetDate
Kategorie (VMXT)	Category (VMXT)	VMXT.Kategorie
Reproduzierbar (VMXT)	Repeatable (VMXT)	VMXT.Reproduzierbar
Autor (VMXT)	Author (VMXT)	VMXT.Autor
Dringlichkeit (VMXT)	Urgency (VMXT)	VMXT.Urgency
Gegenstand (VMXT)	Item (VMXT)	VMXT.Item
Version (VMXT)	Item Version (VMXT)	VMXT.ItemVersion

Tabelle B.7.: Datenstruktur (DE/EN) für PÄM-Work Items

Literaturverzeichnis

[AEH+07] ARMBRUST, O., J. EBELL, U. HAMMERSCHALL, J. MÜNCH und D. THOMA: *Prozesseinführung und -reifung in der Praxis: Erfoolgsfaktoren und Erfahrungen*. In: *Proceedings des 14. Workshop der Fachgruppe WI-VM der Gesellschaft für Informatik e.V. (GI)*, Nummer ISBN: 978-3-8322-6111-5, Seiten 3–15. Shaker Verlag, apr 2007.

[bAB05] AHMED BOULILA, N. BEN: *A Framework for Distributed Collaborative Software Design Meeting*. Doktorarbeit, Technische Universität München, 2005.

[BEJ06] BUSCHERMÖHLE, R., H. ECKHOFF und B. JOSKO: *Success – Erfolgs- und Misserfolgsfaktoren bei der Durchführung von Hard- und Softwareentwicklungsprojekten in Deutschland*. Nummer ISBN 978-3-8142-2035-2. BIS-Verlag der Carl von Ossietzky Universität Oldenburg, 2006.

[Ber06] BERGNER, K.: *Plangenerierung im V-Modell XT 1.2*. In: HOCHBERGER, CHRISTIAN und RÜDIGER LISKOWSKY (Herausgeber): *Informatik 2006 - Beiträge der 36. Jahrestagung der Gesellschaft für Informatik e.V.(GI)*, Band P-94 der Reihe *Lecture Notes in Informatics*. Gesellschaft für Informatik, oct 2006. available at http://www.gi-ev.de/LNI.

[BULL07] BADI ULMER, P., A. LORENZ und G. LORENZ: *Kennzahl-getriebenes Controlling zur Optimierung der Softwareentwicklung und -pflege – Ein Praxisbericht*. In: KOSCHKE, RAINER, OTTHEIN HERZOG, KARL-HEINZ RÖDIGER und MARC RONTHALER (Herausgeber): *Informatik 2007 - Beiträge der 37. Jahrestagung der Gesellschaft für Informatik e.V.(GI),*, Band P-110 der Reihe *Lecture Notes in Informatics*. Gesellschaft für Informatik, 2007.

[Fri06] FRIEDRICH, J.: *Technische und semantische Transformation von Vorgehensmodellen*. Diplomarbeit, Technische Universität München, 2006.

[Gna05] GNATZ, M.: *Vom Vorgehensmodell zum Projektplan*. Doktorarbeit, Technische Universität München, 2005.

[Gna06] GNATZ, M.: *V-Modell XT: Meta-Modell und Konsistenzbedingungen*. Online, jan 2006. Version 1.1.

[GP06] GUCKENHEIMER, S. und J. J. PEREZ: *Software Enginnering with Microsoft Visual Studio Team System*. Addison Wesley, 2006.

[GS04] GREENFIELD, J. und K. SHORT: *Software Factories*. Nummer ISBN: 978-0471202844. Wiley & Sons, 2004.

[HKST07] HAMMERSCHALL, U., M. KUHRMANN, M. SIHLING und T. TERNITÉ: *Strategischer Vorteil - Das V-Modell XT an Unternehmen anpassen (Teil 2)*. iX - Magazin für professionelle Informationstechnik, (05/07):142–145, apr 2007. available at http://www.heise.de.

[JRH+03] JECKLE, M., C. RUPP, J. HAHN, B. ZENGLER und S. QUEINS: *UML 2 glasklar*. Nummer ISBN: 978-3446225756. Hanser, Erste Auflage, 2003.

[KHTS07] KUHRMANN, M., U. HAMMERSCHALL, T. TERNITÉ und M. SIHLING: *Individueller Standard - Das V-Modell XT an Unternehmen anpassen (Teil 1)*. iX - Magazin für professionelle Informationstechnik, (04/07):134–138, mar 2007.

[KK07] KALUS, G. und M. KUHRMANN: *CollabXT – Ein Ansatz zur automatischen Erzeugung von Kollaborationsportalen aus dem V-Modell XT*. In: *Proceedings des 14. Workshop der Fachgruppe WI-VM der Gesellschaft für Informatik e.V. (GI)*, Nummer ISBN: 978-3-8322-6111-5, Seiten 29–40. Shaker Verlag, apr 2007.

[KMR06] KUHRMANN, M., J. MÜNCH und A. RAUSCH: *Metamodellbasierte Integration von Projekt Controlling Mechanismen in das V-Modell XT - Positionspapier*. In: HOCHBERGER, CHRISTIAN und RÜDIGER LISKOWSKY (Herausgeber): *Informatik 2006 - Beiträge der 36. Jahrestagung der Gesellschaft für Informatik e.V.(GI)*, Band P-94 der Reihe *Lecture Notes in Informatics*, Seiten 103–109. Gesellschaft für Informatik, oct 2006. available at http://www.gi-ev.de/LNI.

[Kne06] KNEUPER, R.: *CMMI*. Nummer ISBN: 978-3898643733. dpunkt.verlag, 2006.

[KT06] KUHRMANN, M. und T. TERNITÉ: *Implementing the Microsoft Solutions Framework for Agile Sw-Development as Concrete Development-Method in the V-Modell XT*. International Transactions on Systems Science and Applications (ITSSA), Special Issue Sections in ENASE 06, 1(ISSN 1751-147X):119 – 126, sep 2006.

[Kuh06] KUHRMANN, M.: *Projektspezifische Anpassungen nach dem Tailoring des V-Modell XT durchführen*. In: BISKUP, HUBERT und RALF KNEUPER (Herausgeber): *13. Workshop der Fachgruppe WI-VM der Gesellschaft für Informatik e.V. (GI) zum Thema: Nutzen und Nutzung von Vorgehensmodellen*, Seiten 27–42. Shaker Verlag, mar 2006.

[Kuh07] KUHRMANN, M.: *Prozessintegration und -anpassung*. Forschungsbericht TUM-I0712, Technische Universität München, 2007.

[Mic07] MICROSOFT CORPORATION (Herausgeber): *Team Development with Visual Studio Team Foundation Server*. Nummer ISBN-13: 978-0735625716. Microsoft Press, 2007.

[NK05] NIEBUHR, D. und M. KUHRMANN: *Projekt-Tüv*. iX - Magazin für professionelle Informationstechnik, 1(1/2006):126–129, dec 2005.

[OMG06a] OMG: *Meta Object Facility (MOF) Core Specification –Version 2.0*. Meta Object Facility (MOF) Core Specification, Object Management Group, 2006.

[OMG06b] OMG: *Object Constraint Language (OCL) – Version 2.0*. OMG Available Specification, Object Managment Group, 2006.

[OMG07a] OMG: *Unified Modeling Language (UML): Infrastructure – Version 2.1.1*. Technischer Bericht, Object Management Group, 2007.

[OMG07b] OMG: *Unified Modeling Language (UML): Superstructure – Version 2.1.1*. Technischer Bericht, Object Management Group, 2007.

[Tur06] TURNER, M.: *Microsoft Solutions Framework Essentials*. Nummer ISBN: 0-7356-2353-8. Microsoft Press, 2006.