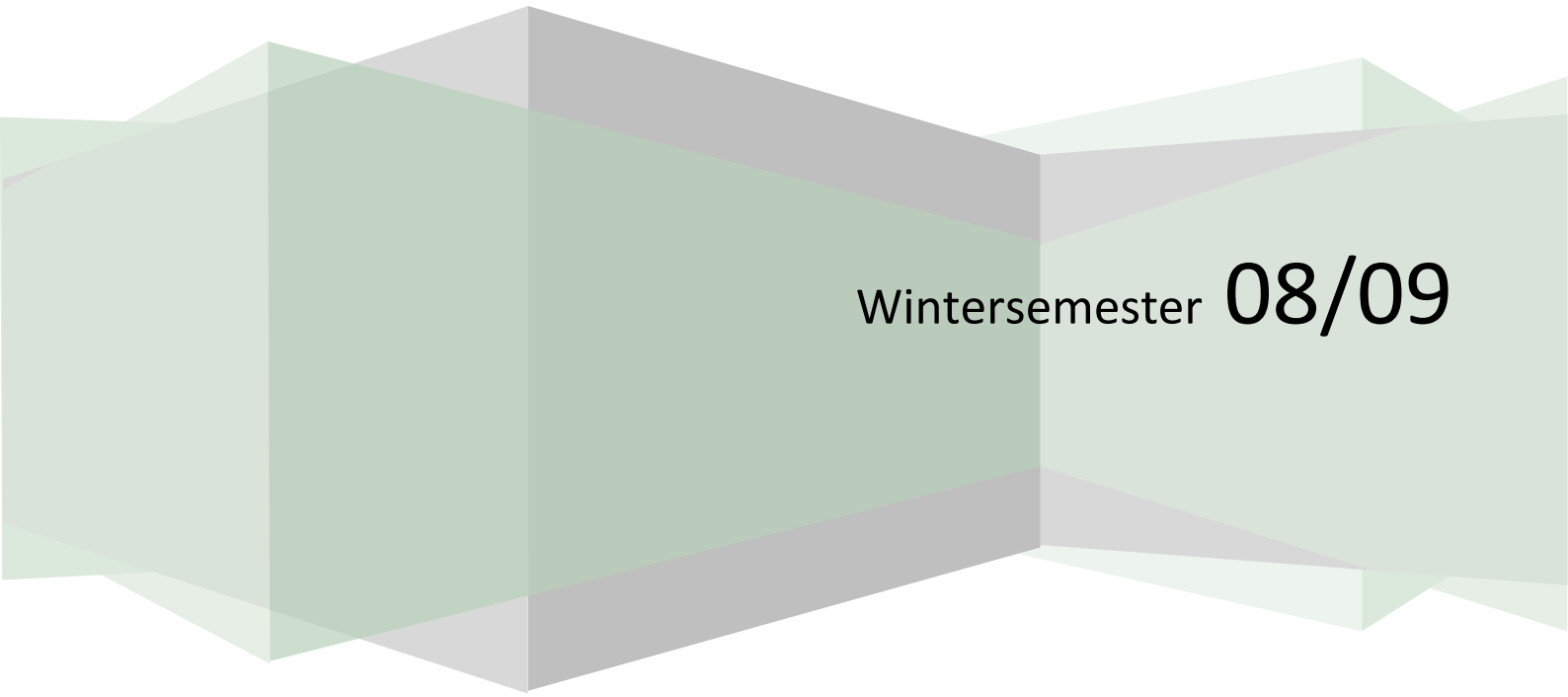
Hochschule Karlsruhe Technik & Wirtschaft

Software-Entwicklungsprozesse mit Team Foundation Server 2008

Studienarbeit

Manuel Naujoks (www.neokc.de)



Wintersemester 08/09

Zusammenfassung

Im Rahmen dieser Studienarbeit wird der Team Foundation Server evaluiert. Als serverseitige Komponente in Visual Studio Team System unterstützt Team Foundation Server Teams bei der Entwicklung von Software nach definierten Prozessen. In Zusammenarbeit mit Visual Studio ist er ein entscheidender Faktor in der systematischen Entwicklung von .NET Anwendungen.
In dieser Ausarbeitung werden zunächst die wichtigsten Funktionen der Anwendung vorgestellt, bevor die unterstützten Prozessmodelle untersucht werden. Mit einem kleinen Szenario wird die Unterstützung für Agile Software Development, CMMI und einer Scrum-Variante näher betrachtet. Mithilfe der Entwicklung einer Beispielanwendung werden die Prozessmodelle analysiert und dessen Stärken und Schwächen aufgezeigt. Dabei wird ein Schwerpunkt auf die verwendeten Work Items gelegt und gezeigt wie sie die Dynamik eines Prozesses steuern können. Zum Schluss werden Möglichkeiten beschrieben, wie eventuelle Schwächen vielleicht auszugleichen sind. Es wird untersucht, in wie fern sich der Team Foundation Server anpassen lässt, um ihn beispielsweise in einem bereits bewährten Umfeld zu etablieren.

Inhaltsverzeichnis

1 Einführung

Die Entwicklung von Software findet heutzutage immer häufiger nach strukturierten Prozessen statt. Mithilfe von Prozessen kann eine effektive Zusammenarbeit in Teams erreicht, sowie die Entwicklung formalisiert werden. Für die unterschiedlichsten Bereiche und Anforderungen existieren spezielle Prozessmodelle, die Entwicklern und Projektmanagern helfen können, die Kontrolle über Projekte nicht zu verlieren, sondern diese erfolgreich abzuschließen. Doch soviel zur Theorie.
In der Praxis gibt es oft Schwierigkeiten Software nach einem definierten Prozess zu entwickeln. Zum Beispiel fällt es manchen Teammitgliedern schwer, den Prozess zu leben und sich an spezielle Vorgehensweisen zu halten. Häufig spielt sich der Prozess auch nur in den Köpfen einiger versierter Mitarbeiter ab und wird im besten Fall mit Hilfe von Microsoft Excel und Microsoft Project koordiniert. Die verwendeten Tools bestehen oft nur aus IDE und Versioning Control. Dabei ist die Unterstützung durch Software essentiell für die Umsetzung eines Entwicklungsprozesses. Zum Beispiel muss eine Arbeitsverfolgung benutzt werden, um Aufgaben zu verteilen und zukünftige Schritte planen zu können. Erledigte Aufgaben müssen systematisch ausgewertet werden können und anhand einer Projektübersicht grafisch dargestellt werden. Es muss jederzeit ersichtlich sein, wie das Projekt läuft und wie es sich entwickelt. Nur dann können Projektmanager und Verantwortliche rechtzeitig reagieren und das Scheitern verhindern. Natürlich spielen IDE und Source Control eine sehr wichtige Rolle, allerdings müssen diese Entwicklungswerkzeuge mit den prozesssteuernden Tools zusammenarbeiten können. Zum Beispiel kann ein Projektmanager nachvollziehen wollen, welche Check-Ins mit einer von ihm erstellten Aufgabe zu tun haben. Außerdem ist es wichtig, dass eingecheckter Code und dessen Änderung in die Auswertung des Projektzustandes mit einbezogen werden können.
Software, die hilft einen Prozess zu leben ist also von hoher Bedeutung. Das Team muss zusammenarbeiten können, ohne nur noch an den Prozess zu denken. Genau an dieser Stelle kommt es darauf an, eine Softwarelösung zu verwenden, die sämtliche Bedürfnisse der unterschiedlichsten Beteiligten an dem Softwareprojekt adressiert und ihnen hilft, ihre Position im Prozess zu meistern. Bei einer solchen Lösung sollte ein vollständig integriertes System in Frage kommen. Visual Studio Team System (VSTS) von Microsoft entspricht einer solchen Endlösung für Teams, die Softwareentwicklung mit Prozessen steuern wollen. Aus diesem Grund wird VSTS im Rahmen dieser Studienarbeit genauer betrachtet.

2 Visual Studio Team System

Mit VSTS bietet Microsoft eine integrierte Umgebung für den kompletten Softwarelebenszyklus. Dabei besteht diese Lösung aus mehreren verschiedenen Komponenten. Den vermutlich wichtigsten Teil stellt die Visual Studio Team Suite dar, die in Unterabschnitt 2.1 Visual Studio 2008 näher betrachtet wird. Auf der Server Seite ist der Team Foundation Server zu betrachten. Unterabschnitt 2.2 Team Foundation Server 2008 beschäftigt sich dabei ausführlich mit dessen Architektur. Zusätzlich existiert in Team System eine Komponente für Microsoft Office. Projektmanager können daher mit Microsoft Excel oder Microsoft Project ihre Arbeit verrichten. Mit einer Oberfläche für das Web können alle Projektmitarbeiter eine Übersicht über sämtliche Teamarbeiten erhalten. Dafür brauchen sie keine IDE, sondern nur einen Browser. Die Verbindung all dieser Komponenten erfolgt über sogenannte Process Templates, die in Abschnitt 5 Prozessvorlagen genauer betrachtet werden. Mit den Prozessen, die damit abgebildet werden können, lassen sich sämtliche Funktionen, wie zum

Beispiel Version Control oder automatisierte Builds nutzen. Die untere Übersichtsgrafik von Microsoft zeigt alle Komponenten in ihrem strukturellen Zusammenhang.

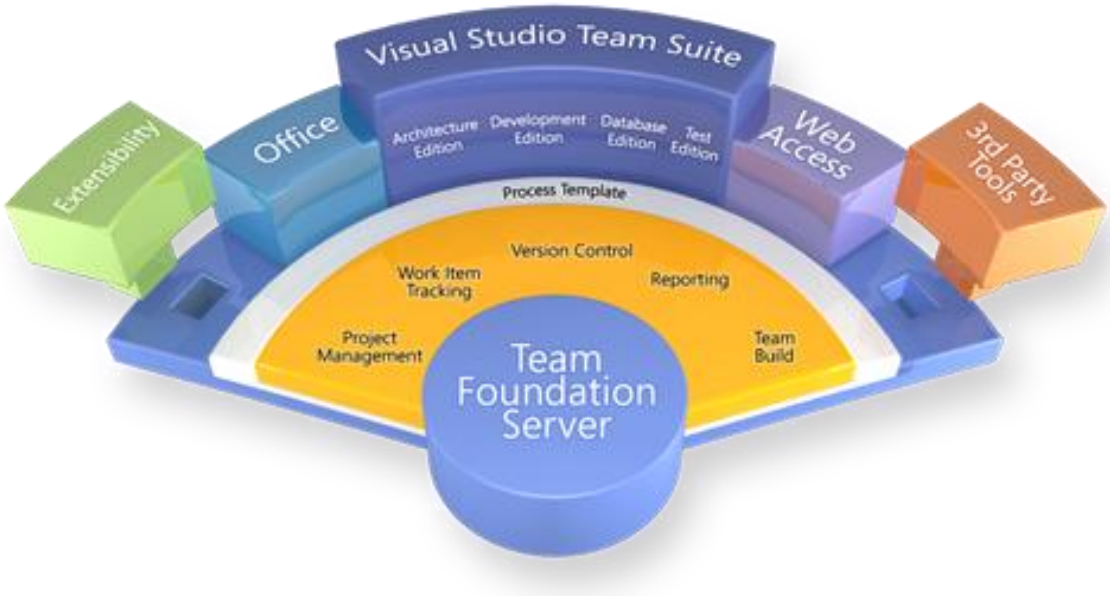


Abbildung 1 Visual Studio Team System Übersichtsgrafik[1]

Wie aus der Grafik entnommen werden kann, ist der Visual Studio Team System erweiterbar. Laut Microsoft [Mic0713] gibt es ein SDK, womit eigene Anwendungen in der VSTS Umgebung entwickelt werden können. Im Rahmen dieser Studienarbeit kann aufgrund des begrenzten Umfangs nicht näher auf dieses SDK eingegangen werden. Wie man es aber erwarten würde gibt's es auch eine Reihe an Produkten von Dritt-Anbietern, die Tools für den Team Foundation Server entwickelt haben. Das bekannteste wird wohl das Eclipse Plugin von Teamprise [Tea08] sein. Mit diesem Plugin kann nun auch die Eclipse Umgebung und damit Java integriert werden. Normalerweise beschränkt sich Team System ausschließlich auf die Visual Studio Entwicklungsumgebung und unterstützt aus diesem Grund nur .NET Sprachen sowie C++ für Windows Betriebssysteme.

2.1 Visual Studio 2008

Aktuell ist Microsofts integrierte Entwicklungsumgebung in der Version 9 (2008) vorhanden. Insgesamt gibt es fünf verscheiden Visual Studio Produkte, die auch auf der entsprechenden Microsoft Website [Mic08] vorgestellt werden. Die Architecture Edition ist laut Microsoft für Bereichsleiter und Software Architekten gedacht, da sie Werkzeuge für die Planung und den Entwurf von Softwaresystemen beinhaltet. Die Development Edition bietet sämtliche Funktionalität für den eigentlichen Softwareentwickler. Dazu gehören unter anderem statische Codeanalysen um eine hohe Softwarequalität zu gewährleisten. Die dritte Version ist die Test Edition. In dieser Version sind Werkzeuge für das Testen von Software und Webanwendungen enthalten. Neu in der Reihe der Visual Studio Produkte ist die Database Edition. Diese Version bietet Tools, die die Datenbank Entwicklung unterstützen und vereinfachen sollen. Das fünfte und damit letzte Visual Studio ist die Team Suite. Sie beinhaltet die komplette Funktionalität der anderen vier Produkte und stellt damit die wirklich integrierte Anwendung dar, die den Lebenszyklus eines Softwareprojekts durchgehend unterstützen kann. Damit repräsentiert Visual Studio die Seite der Clients in der Team System

[1] Informationen zu Visual Studio Team System im Microsoft Developer Network über wichtige Aspekte des Team Foundation Server in der Version 2008: http://msdn.microsoft.com/de-de/vsts2008/products/bb964615.aspx.

Umgebung. Die Teammitglieder arbeiten mit einer dieser fünf Versionen an den Softwareprojekten. Damit diese Arbeit auch koordiniert erfolgen kann, ist auch die serverseitige Anwendung erforderlich.

2.2 Team Foundation Server 2008

Der Team Foundation Server stellt die Kernkomponente in Visual Studio Team System, sowie die Serverseite dar. Damit ein Visual Studio Client eine Verbindung zu ihm herstellen kann, wird der Team Explorer benutzt. Diese integrierte Komponente erlaubt einem Anwender sämtliche Projekte, zu denen er einen berechtigten Zugriff hat, einzusehen und an ihnen zu arbeiten. Im Folgenden werden die Funktionen des Team Foundation Servers vorgestellt, die im Rahmen der Evaluation als besonders wichtig angesehen werden.

2.2.1 Team Projekte

Um Team Projekte verwalten zu können, wird eine neue Projektstruktur verwendet, die unabhängig von Projekten und Solutions in Visual Studio ist. Dabei werden Projekten Teammitglieder zugeordnet um ihnen Rollen und Berechtigungen zuteilen zu können. Diese Mitglieder müssen anhand von lokalen oder Active Directory Benutzen im Team Foundation Server registriert werden. Zusätzlich lassen sich pro Team Projekt Bereiche und Iterationen festlegen, die dazu dienen die Arbeit grob zu strukturieren. Selbstverständlich können dabei die Zugriffsberechtigungen der Benutzer sehr granular eingestellt werden, um nur bestimmten Personen die Arbeit an bestimmten Teilen zu erlauben. In jedem Team Projekt muss ein Prozessmodell verwendet werden, welches bei der Erstellung ausgewählt werden kann. Ein solches Prozessmodell beschreibt den Entwicklungsprozess, der in diesem Projekt verwendet werden soll. Wie dies genau aussieht wird in Abschnitt 5 Prozessvorlagen genauer beschrieben. Anhand der Prozessvorlage wird ein SharePoint Portal eingerichtet, dass den Zustand des Team Projekts präsentiert und als zentrale Anlaufstelle für die Mitglieder desselben dient. Dort können sämtliche Tätigkeiten geschehen, die nicht im Rahmen der konkreten Entwicklung in Visual Studio erfolgen. Zum Beispiel existieren ein Forum zur Diskussion, ein Kalender, sowie eine zentrale Dateiablage für projektrelevante Dokumente. Neben dem SharePoint Portal wird auch eine sogenannte Process Guidance erstellt. Dabei handelt es sich um eine Website, die den verwendeten Prozess innerhalb eines Projekts erklärt. In Abschnitt 5 Prozessvorlagen wird auf diese Hilfe näher eingegangen. Das SharePoint Portal ist das Projektportal.

2.2.2 Work Item Tracking

Die wohl wichtigste Funktion im Team Foundation Server ist das Work Item Tracking. Work Items stellen Arbeitseinheiten dar und enthalten sämtliche Information zu den zu erledigenden Tätigkeiten. Alles, zu dem Informationen im Rahmen einer Entwicklung abgelegt werden kann, kann also ein Work Item sein. Dabei ist es egal, ob es sich um eine funktionale, nichtfunktionale oder logische Komponente handelt. Diese Arbeitseinheiten stellen die Dynamik in einem Prozess dar und beinhalten Zustandsinformationen anhand dessen sich der Status des Team Projekts verändern kann. Ein Bug wäre ein gutes Beispiel für ein Work Item. Er wird mithilfe von relevanten Informationen erstellt und befindet sich dann in einem definierten Zustand. Ein Zustandswechsel kann erfolgen, wenn der Bug zum Beispiel geprüft wurde. Anschließend kann er behoben werden, was den Zustand erneut verändern würde. Je nachdem in welchem Zustand er sich befindet, kann der Fortschritt des Team Projekts gemessen und verwendete Prozess gelebt werden. Work Items sind Bereichen und Iterationen zugeordnet und können auch einzelnen Personen zugeteilt werden. Jede Person kann so an einer Menge von Arbeitseinheiten arbeiten und so Abläufe koordinieren. Um ein Work Item zu erstellen wird ein Work Item Template benötigt, das definiert, von welchem Typ ein Work Item ist.

Also ob es sich um einen Bug, eine Anforderung oder etwas anderes handelt. Ein Work Item ist also eine Instanz eines Work Item Templates, auch Work Item Typ genannt.

2.2.3 Source Control

Zu einem Team Projekt gehört in der Regel auch Quellcode der verwaltet werden muss. Dafür wird eine eigene Versionsverwaltung benutzt, die komplett in Team Foundation Server integriert ist. Jedes erstellte Projekt enthält hier einen eigenen Bereich, indem beliebig viele Visual Studio Solutions und Projekte in einer freiwählbaren Struktur hinterleget werden können. Dabei werden alle bekannten Funktionen einer Code-Verwaltung geboten, wie zum Beispiel Check-In/Out, Vergleich unterschiedlicher Versionen und Auflisten aller Versionen einer Komponente. Zusätzlich können Richtlinien definiert werden, die ein Check-In-Vorgang erfüllen muss. Zum Beispiel kann eine bestimmte Codequalität nach einer statischen Codeanalyse oder ein erfolgreicher Testlauf vorausgesetzt werden. Außerdem kann festgelegt werden, dass nur Code eingecheckt wird, der einem Work Item zugeordnet ist. Auf diese Weise kann nachverfolgt werden, welche Check-Ins zum Beispiel an einem Bug-Fix beteiligt waren oder welche Änderung am Code eine Anforderung realisiert hat. Dabei kann ein Check-In auch den Zustand eines Work Items entsprechend ändern, sodass ein Bug entweder endgültig gefixt werden oder nur dazu beitragen kann. Bei jedem Check-In kann zudem eine Menge an zusätzlicher Information angegeben werden, wie zum Beispiel eine Beschreibung oder die Namen der Code Reviewer. Eine weitere Funktion der Quellcode-Verwaltung ist das sogenannte Shelving. Code, der noch nicht fertig ist und aus diesem Grund noch nicht eingecheckt werden kann und darf, kann in Shelves abgelegt werden. Dieser Shelve wird damit auch serverseitig gesichert und kann jederzeit wieder abgeholt und sogar an andere Mitarbeiter weitergegeben werden.

2.2.4 Continuous Integration

Neben der Code-Verwaltung besitzt der Team Foundation Server auch ein integriertes Buildsystem. So lassen sich Buildvorgänge einrichten, die in jeder Nacht eine Menge an definierten Visual Studio Solutions kompilieren und die erzeugten Assemblies in einem freigegebenen Verzeichnis ablegen. Solche Buildvorgänge können auch nach jedem erfolgreichen Check-In erfolgen, sodass die Integrität der kompletten Anwendung jederzeit geprüft werden kann. Diese Prüfung lässt sich zum Einen mit einer statischen Codeanalyse auf der Serverseite durchführen. Zum anderen können Unit-Tests definiert werden, die den Code aus der Codeverwaltung vor dem Kompilieren testen. Ein fehlgeschlagener Build, also wo ein Test nicht erfolgreich ausgeführt werden konnte, könnte in diesem Zusammenhang sofort den fehlerhaften Code identifizieren und den entsprechenden Entwickler, Tester oder Vorsitzenden automatisch informieren.

2.2.5 Reporting

Normalerweise würde die Entwicklung einer Anwendung mit Visual Studio in einem Team ohne Team Foundation Server nicht sehr transparent erfolgen. Das bedeutet, dass es für einen Vorgesetzten oder Teamleiter schwierig sein würde, den aktuellen Stand des Projekts in Erfahrung zu bringen, ohne zeitaufwendige Interviews mit seinen Entwicklern führen zu müssen. Im Team Foundation Server wird dieses Problem mithilfe einer integrierten Reporting-Funktionalität vorgebeugt. Jedes Prozessmodell beinhaltet eine Reihe vordefinierter Berichte, die über das SharePoint Projektportal oder direkt über Visual Studio aufgerufen werden können. So existieren unter anderem Berichte über Builds sowie über den Fortschritt des gesamten Projekts. Dieser wird anhand von Check-Ins, Bugs, erfolgreichen oder fehlgeschlagenen Tests definiert. Die Datenaufbereitung erfolgt automatisch und stündlich über einen OLAP Cube und verwendet SQL Server Reporting Services. Dies ermöglicht das

einfache Erstellen von neuen und angepassten Berichten, um spezielle Informationen besser verfolgen zu können. Der OLAP Cube kann sogar über Excel abgefragt werden, was die Projekttransparenz sicherstellt. Interessant ist, dass beim Reporting im Team Foundation Server Informationen sämtliche Features zusammenkommen. So werden die Bereiche und Iterationen eines Projekts mit den zugehörigen Work Items, ihren Codeänderungen aus der Quellcode-Verwaltung und den erfolgreichen Builds mit ihren Unit-Tests in Verbindung gebracht, um einen möglichst realistischen Projektfortschritt zu messen.

3 Projektumgebung

Um den Team Foundation Server zu evaluieren wurde eine Demonstrationsumgebung eingerichtet. Auf dem Server ist Windows Server 2008 mit SQL Server 2008 installiert. Außerdem wurde die Workgroup Edition des über MSDNAA verfügbaren Team Foundation Server 2008 mit SP1 installiert. Zusätzlich bestehen auf dem Server unteranderem drei Benutzerkonten zur Administration. Der Benutzer TFSSETUP wurde angelegt um den Team Foundation Server zu installieren und zu warten. Als zu registrierenden Servicebenutzer wurde bei der Installation TFSSERVICE verwendet. Dann besteht noch das Konto TFSBUILD, welches den Build Agenten ausführt und somit Anwendungen der Team Projekte insoliert und gesichert kompilieren kann. Das untere Schaubild zeigt die Benutzer der Server und der Client Seite.

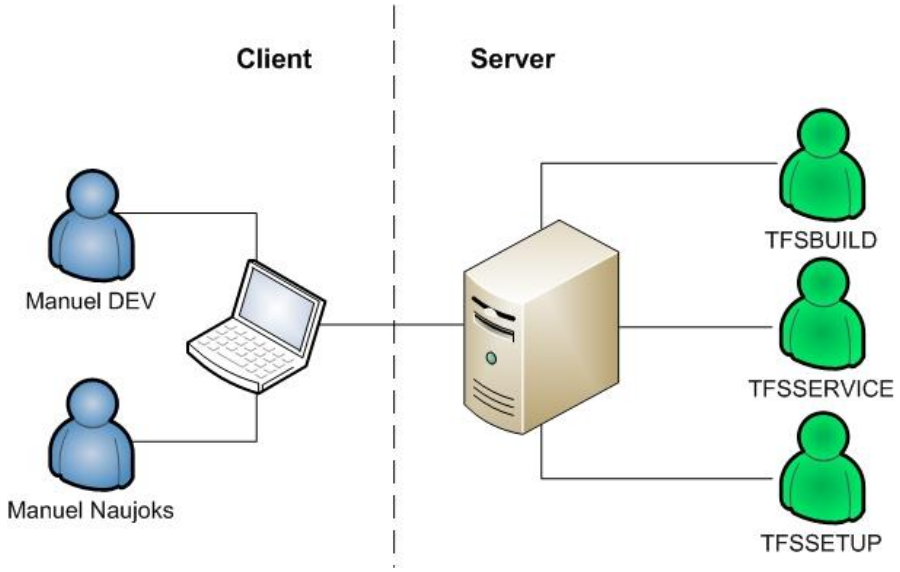


Abbildung 2 Projektumgebung des Team Foundation Server

Auf der Client Seite läuft ein Windows Vista SP1 mit Visual Studio Team Suite 2008 SP1, also der allesumfassenden Team Edition, und integriertem Team Explorer. Der Client verwendet zwei Benutzerkonten, um den Team Server zu verwenden. Zusammen existieren also fünf Benutzer, die im Team Foundation Server lizensiert werden müssen. Mehr als fünf Benutzer würden in der Workgroup Edition auch nicht unterstützt werden. Es ist noch anzumerken, dass auf dem Server zusätzlich ein IIS 7 Webserver installiert ist, um die SharePoint Portale zu benutzen. Aus diesem Grund ist auch Windows SharePoint Services 3 installiert.

Die Berechtigungen sind so eingestellt, dass TFSSETUP der Administrator des Team Foundation Servers ist. Die Benutzer auf dem Client erhalten lediglich die Rechte der Teammitglieder. Das

bedeutet, nur TFSSETUP kann ein neues Teamprojekt anlegen und Einstellungen an der Versionsverwaltung oder dem Build System vornehmen. Weiterhin repräsentiert der Benutzer „Manuel DEV“ den Entwickler im Team. „Manuel Naujoks“ soll in der Umgebung eine Kombination aus Entwickler und Teamleiter sein. Nachdem die Umgebung vorgestellt wurde, kann die Evaluation des Team Foundation Server beginnen.

4 Szenario

Um die Softwareentwicklung mit dem Team Foundation Server evaluieren zu können, wird in dieser Studienarbeit ein kleines Szenario verwendet. Es soll eine einfache Software Anwendung entwickelt werden, die es einem Benutzer ermöglicht eine Zahl zu würfeln. Dabei soll es sich wirklich um ein sehr einfaches und triviales Programm handeln, da der Schwerpunkt auf der strukturierten Entwicklung mithilfe der einzelnen Prozessmodelle im Team Foundation Server liegt. Das untere Bild zeigt die fertige Anwendung, wie sie schnell und ohne Struktur zusammenprogrammiert wurde.

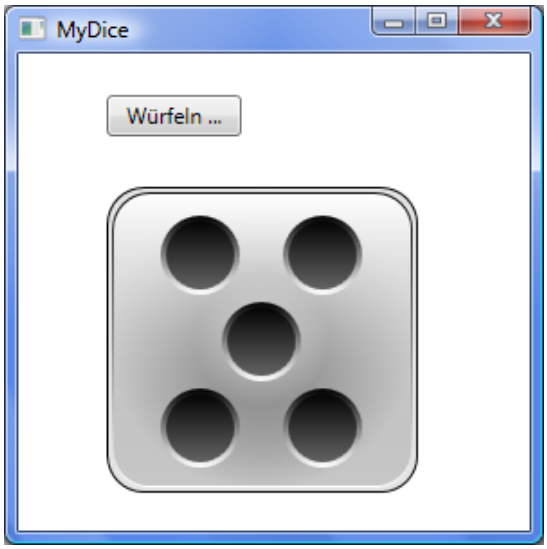


Abbildung 3 Beispielanwendung MyDice

Die Funktionalität der Anwendung ist denkbar einfach. Klickt der Benutzer auf den Button „Würfeln…“, soll der grafische Würfel das Würfelergebnis darstellen. Dabei handelt es sich um Zahlen zwischen eins und sechs.
Im Rahmen des Szenarios soll gezeigt werden, wie das Beispielprogramm unter Verwendung der einzelnen Prozessmodelle des Team Foundation Server entwickelt werden kann. Dabei soll die Funktionalität und auch die Benutzeroberfläche von dem Würfel beachtet werden. Bei der systematischen Durchführung der Entwicklung sollen die Möglichkeiten des Team Foundation Server demonstriert und evaluiert werden. Dabei wird für jeden zu betrachtenden Prozess ein eigenes Teamprojekt mit der entsprechenden Prozessvorlage erstellt. Es sollen die Unterschiede der Prozessmodelle sowie dessen Schwerpunkte und Eignungen untersucht werden. Das Ziel ist es nicht, in jedem dieser Teamprojekte die Beispielanwendung bis zum Ende zu entwickeln, sondern einen Überblick über Funktionalitäten und Prinzipien zu verschaffen. Abschließend soll eine bewertende Schlussfolgerung zu dem jeweiligen Prozessmodell und dem Team Foundation Server gezogen werden.

5 Prozessvorlagen

Um ein Software Projekt von dem Team Foundation Server unterstützen zu lassen, muss zunächst ein Teamprojekt angelegt werden. Beim Erstellen eines solchen muss eine Prozessvorlage ausgewählt werden, die die Vorgehensweise und den Prozess definieren, nachdem entwickelt werden soll. In diesem Abschnitt wird das Prinzip der Vorlagen vorgestellt und die Entwicklungsmodelle, die der Team Foundation Server von Haus aus unterstützt analysiert. Zur Evaluation werden drei Teamprojekte erstellt, die auf unterschiedlichen Prozessvorlagen basieren: AgileTest, CMMITest und ScrumTest. Um die Projekte erstellen zu können, muss der Benutzer TFSSETUP verwendet werden, da die anderen Benutzer standardmäßig nicht die nötige Berechtigung besitzen. Da sich diese Studienarbeit auf die Entwicklungsunterstützung bezieht, werden administrative Aspekte weitgehend außer acht gelassen. Alle diese drei Projekte haben das Ziel die Beispielanwendung MyDice zu entwickeln. Nach den Analysen werden zudem Schlussfolgerungen getroffen, die die Vor- und Nachteile der behandelten Modelle zusammenfassen.

5.1 MSF for Agile Software Development – v4.2

Die erste Prozessvorlage, die im Team Foundation Server standardmäßig schon vorinstalliert ist, ist „MSF for Agile Software Development – v4.2“. Nachdem das Projekt mit dem Namen „AgileTest“ angelegt wurde, wurden automatisch einige Work Items erstellt, die das Einrichten des neuen Teamprojekts erleichtern sollen. Im unteren Screenshot kann man den Team Explorer, also die zentrale Team Foundation Server Anbindung in Visual Studio, erkennen. Dort können alle Informationen zum aktuellen Projekt abgerufen und Arbeiten an demselben vorgenommen werden. Eine Liste aller existierender Work Items kann über die Team Query „All Work Items“ aufgerufen werden.

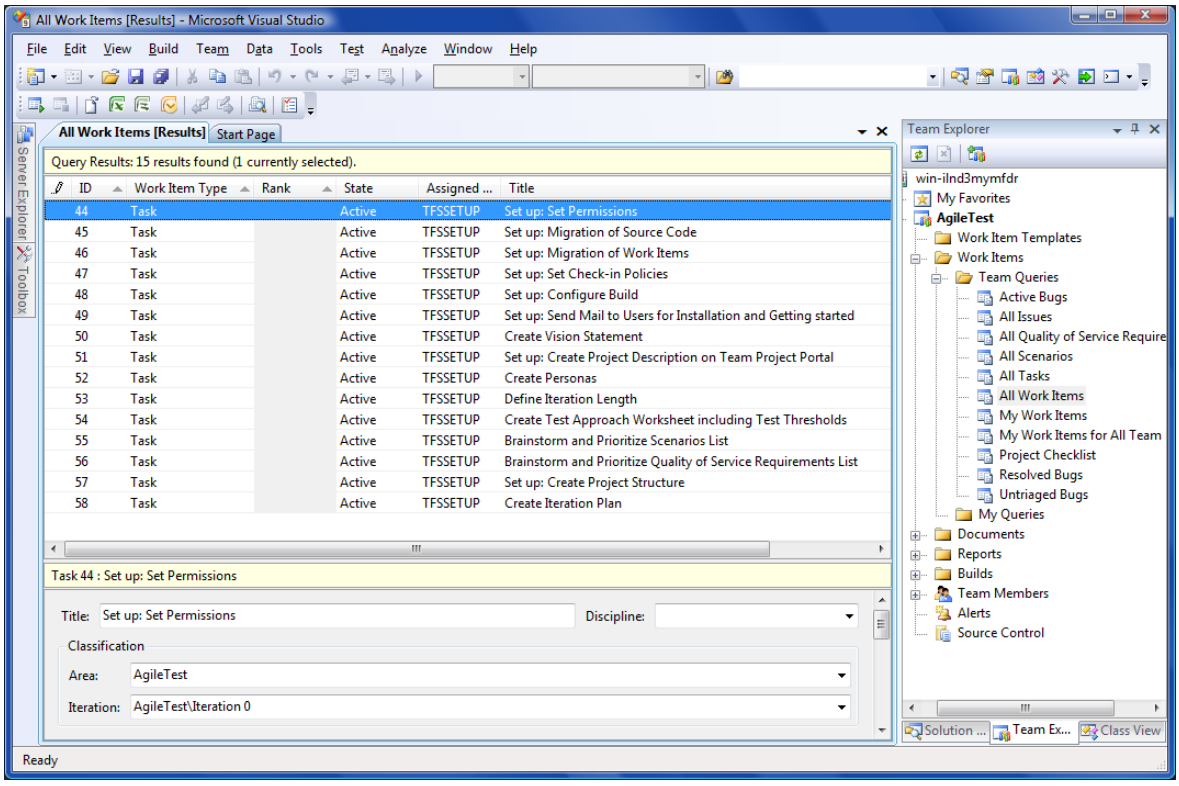


Abbildung 4 Liste aller Work Items im Teamprojekt AgileTest

Wird ein Work Item ausgewählt, können dessen Eigenschaften und Attribute eingesehen und bearbeitet werden. In diesem Abschnitt wird die Entwicklung exemplarisch vorgenommen und dabei die Möglichkeiten der MSF Agile Prozessvorlage gezeigt.

5.1.1 Strukturierung des Projekts

Wenn man das Work Item mit der ID 57 aus der Abbildung betrachtet, so handelt es sich um einen Task. Was ein Task ist und welche Typen es in dieser Vorlage noch gibt, wird in Unter-Unterabschnitt 5.1.2 Verfügbare Work Item Typen behandelt. Im Rahmen dieses Tasks soll die Struktur des Projekts geplant werden. Im Team Foundation Server können dafür prozessunabhängige und sogenannte Areas definiert werden. Sämtliche Work Items müssen einer Area zugeordnet werden. Im Falle der Beispielanwendung wird das Projekt in drei Areas unterteilt, der eigentlichen Logik, dessen Tests und der Benutzeroberfläche.

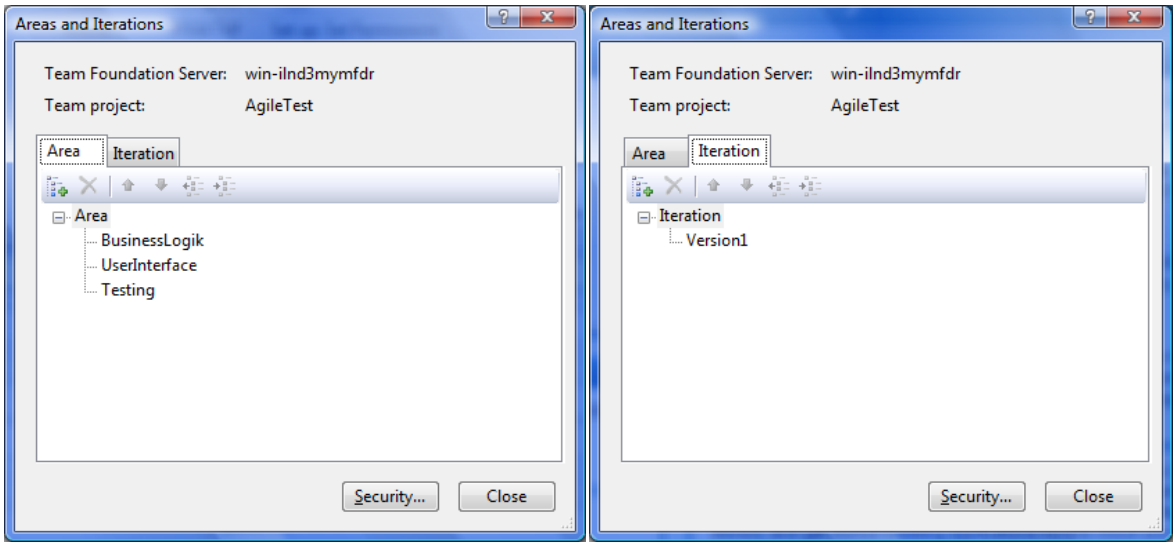


Abbildung 5 Areas und Iterations in AgileTest

Nachdem das Projekt strukturiert wurde, kann das Work Item 57 als abgeschlossen markiert werden. Dieser Vorgang wird im weiteren Verlauf noch genauer behandelt. Das Work Item 58 ist ebenfalls ein Task und betrifft die Planung einer Iteration. Eine Iteration kann als die Entwicklung eines Meilensteins angesehen werden. Für MyDice wird nur ein einziger Meilenstein geplant, da es sich ja nur um eine kleine Anwendung handelt. Die ersten beiden Work Items können also bereits geschlossen werden. Eine umfangreichere Planung der Areas und Iterations kann mit entsprechenden Dokumenten des Teamprojekts im Teamportal erfolgen.

57	Task	Closed	TFSSETUP	Set up: Create Project Structure
58	Task	Closed	TFSSETUP	Create Iteration Plan

Abbildung 6 Tasks für Areas und Iterations abgeschlossen

In der Praxis können Iterations und Areas sehr komplex und auch nachträglich verändert oder erstellt werden. Selbstverständlich lassen sich beide Mechanismen auch beliebig verschachteln. Zur Planung des Teamprojekts gehören auch das Sammeln von Anforderungen und das Hinzufügen von Arbeitspaketen. Beides kann optimal über Work Items vorgenommen werden.

5.1.2 Verfügbare Work Item Typen

In diesem Unter-Unterabschnitt werden die Work Item Typen vorgestellt, die die Prozessvorlage MSF Agile „installiert“. Zunächst ist zu erwähnen, dass jedes Teamprojekt eine Process Guidance [Mic0714] besitzt. In diesem Dokument werden alle wichtigen Bestandteile des eingesetzten Prozessmodells erklärt. Zusätzlich werden Arbeitsabläufe beschrieben, wie sie im Rahmen eines richtigen Teams sein könnten. Dabei handelt es sich also um eine Hilfestellung für sämtliche Teammitglieder, die den Prozess „leben“ und ihre Arbeitsabläufe optimieren und strukturieren möchten.

Abbildung 7 Process Guidance für AgileTest

Diesem Dokument können die Informationen über die verschiedenen Work Item Typen entnommen werden. Bevor die Work Items für die Beispielanwendung erstellt werden können, sollten die verschiedenen Typen erst einmal vorgestellt werden. Sämtliche Zustandsdiagramme der Work Items wurden direkt aus den entsprechenden Seiten der Process Guidance entnommen.

5.1.2.1 Risk

Laut Process Guidance soll für jede Information die einen negativen Einfluss auf das Projekt hat, ein Risk angelegt werden. Dabei kann das Risiko technischer oder organisatorischer Natur sein. Wenn es beispielsweise über den Team Explorer erstellt wird, ist der Zustand automatisch Active. Zusätzlich sollte eine erklärende Beschreibung, sowie eine geschätzte Auswirkung auf das Projekt angegeben werden. Wenn ein Risiko durch Änderung abgeschwächt (Resolved), sehr unwahrscheinlich ist (Inactive), ein anderes Projekt betrifft (Transferred), einfach akzeptiert wird (Accepted) oder das Problem umgangen wurde (Avoided), dann kann das Work Item durch Angabe eines dieser Gründe in den Zustand Closed übergehen. Sollte sich eine dieser Gründe als nicht wahrheitsgemäß herausstellen, kann das Risiko wieder aktiviert werden (Reactivated).

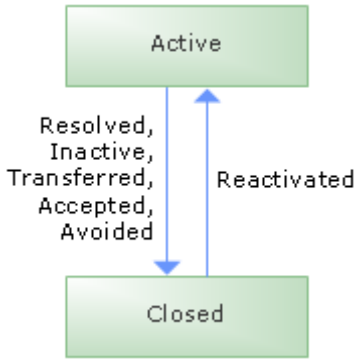


Abbildung 8 Zustände von Risk in AgileTest

5.1.2.2 Quality of Service Requirement

Als Quality of Service Anforderung kann alles in Frage kommen, was ein System charakterisiert. Laut Process Guidance kann es sich dabei beispielsweise um Geschwindigkeit, Verfügbarkeit, Benutzbarkeit und Wartbarkeit handeln. Die Qualitätsanforderung wird dabei im Zustand Active angelegt und geht zum Beispiel in Resolved über, wenn sie implementiert wurde (Completed). Für die Implementierung liegt nahe, dass sich die Zerlegung in Tasks eignet, um konkrete Tätigkeiten mit dieser Anforderung zu verbinden. Andererseits kann sie auch durch anderweitige Änderungen am System blockiert und damit zurückgestellt werden (Deferred). Sollte sie überhaupt nicht umsetzbar sein, kann sie auch entfernt werden (Removed). Das gleiche gilt auch, wenn das Work Item in den Zustand Closed übergehen soll. Im positiven Fall wird die Anforderung getestet und bei bestehen geschlossen (Completed). Sollte ein solcher Test fehlschlagen, muss die Implementierung erneut vorgenommen (Test Failed) und der Zustand wieder auch Active gesetzt werden.

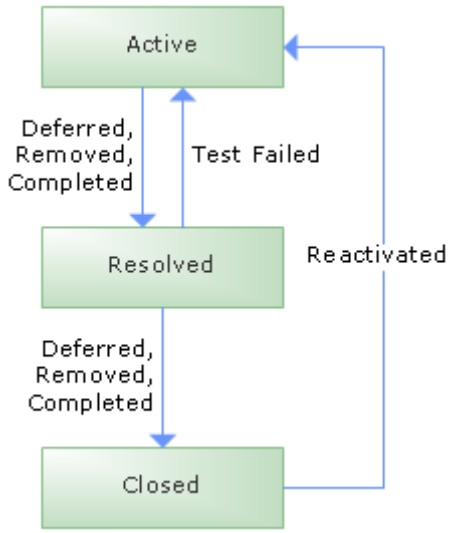


Abbildung 9 Zustände von QoS Requirement in AgileTest

5.1.2.3 Scenario

Ein Use Case wird über ein Szenario abgebildet. Es handelt sich dabei um eine Interaktion eines Benutzers oder eines anderen Systems mit einem System. Laut Process Guidance kann eine solche Interaktion auf mehrere verschiedene Wege erfolgen und dabei ein erwünschtes Ziel erreichen oder nicht. Nachdem ein solcher Use Case angelegt wurde, befindet er sich im Zustand Active und kann bearbeitet werden. Genau wie im QoS Requirement sollte ein Szenario von Task Work Items repräsentiert werden. Durch diese Zerlegung kann ein komplexer Vorgang mithilfe übersichtlicher Arbeitspakete mehreren Personen zugeordnet werden. Sollte die Implementierung zu

unübersichtlich werden, beziehungsweise das Szenario zu umfangreich, so kann es in beliebig viele kleinere Szenarios unterteilt werden (Split). Dadurch geht das Master-Szenario in den Zustand Resolved über. Ein Szenario kann zurückgestellt (Deferred), entfernt (Removed) und abgeschlossen (Completed) werden um ebenfalls in den Zustand Resolved überzugehen. Dort kann es dann getestet werden. Natürlich müssen sämtliche Testfälle auch implementiert werden, wenn es sich um Unit-Tests handelt. Sollte der Test fehlschlagen geht der Use Case wieder in den Zustand Active zurück (Test Failed). Beim Bestehen kann er geschlossen werden (Completed). Das Szenario kann auch geschlossen werden, wenn es entfernt (Removed), aufgeteilt (Split) oder zurückgestellt (Deferred) wird.

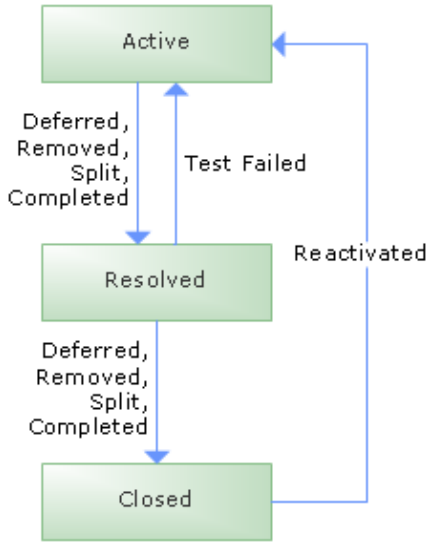


Abbildung 10 Zustände von Scenario in AgileTest

5.1.2.4 Bug

Ein Problem im System wird durch einen Bug repräsentiert, welcher in einer Softwareanwendung eine Fehlfunktion darstellt. In der Regel treten Bugs beim Testen auf und müssen dokumentiert werden, damit sich ein Entwickler mit dem Problem beschäftigen kann. Der Team Foundation Server erstellt automatisch einen Bug, wenn ein Build fehlschlägt.

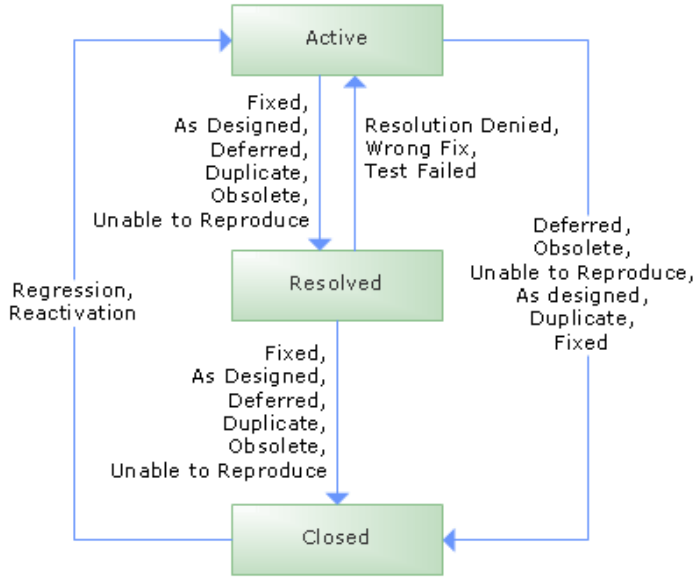


Abbildung 11 Zustände von Bug in AgileTest

Wenn das Problem lokalisiert werden konnte kann es behoben werden (Fixed) um in den Zustand Resolved zu wechseln. Sollte es sich um ein Problem handeln, dass nicht reproduziert werden konnte (Unable to Reproduce), nicht mehr aktuell ist (Obsolete), bereits vermerkt ist (Duplicate) oder entsprechend desgined wurde (As Designed), kann der Bug auch in den Zustand Resolved oder direkt in Closed wechseln. Befindet sich das Work Item im Zustand Resolved und ein Test schlägt erneut fehl (Test Failed), dann wird es erneut auf Active gesetzt. Das gleiche gilt auch, wenn die Behebung nicht korrekt ist (Wrong Fix) oder nicht akzeptiert werden kann (Resolution Denied). Laut Process Guidance besteht die Möglichkeit einen Bug über einen nicht näher betrachteten Regressions Test wieder Active zu setzten.

5.1.2.5 Task

Der letzte Work Item Typ der MSF Agile Prozessvorlage stellt eine einfache Aufgabe dar. Ein Task kann dabei in verschiedenen Kontexten genutzt werden. Zum Beispiel rollenabhängig. Ein Projektmanager kann Tasks als unabhängige Arbeitspakete verteilen, wobei Entwickler oder Tester eigene Tasks erstellen, um die zu erledigende Arbeit besser zu strukturieren. Zum Beispiel können QoS Requirements oder Scenarios mittels Tasks granularer verwaltet werden. Man kann also sagen, dass ein Task immer dann erstellt werden soll, wenn es Arbeit gibt, die in Zusammenhang mit dem System des Teamprojekts steht. Ein Task kann als Closed betrachtet werden, wenn die damit zusammenhängende Arbeit abgeschlossen wurde (Completed), die Aufgabe zurückgestellt werden musste (Deferred) oder sie nicht mehr relevant für das Projekt ist (Obsolete). Eine Aufgabe kann zusätzlich abgeschlossen werden, wenn die Funktionalität, auf die sie sich bezieht, aus dem System entfernt wurde (Cut).

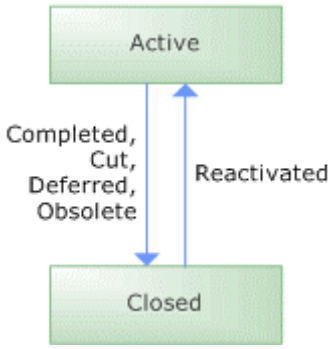


Abbildung 12 Zustände von Task in AgileTest

5.1.3 Praktischer Entwicklungsprozess

Nachdem die fünf verschiedenen Work Item Typen vorgestellt wurden, können sie nun benutzt werden, um die Beispielanwendung zu entwickeln. An dieser Stelle beginnt die eigentliche Entwicklung von MyDice. Dazu erfolgt zunächst eine Planungsphase, beziehungsweise eine Analyse oder ein Design. MSF Agile stellt im Teamprojekt eine Reihe von Vorlagen für Dokumente zur Verfügung, die benutzt werden können um beispielsweise eine Anforderungsspezifikation in Textform zu formalisieren. Zusammen mit einer umfangreichen Planung von Meilensteinen und Bereichen, die für dieses Beispiel in Unter-Unterabschnitt 5.1.1 Strukturierung des Projekts beschränkt vorgenommen wurde, kann ein Projekt sehr detailiert und formal geplant werden. Da es sich bei MSF Agile um ein agiles Prozessmodell handelt, werden Dokumente in diesem Zusammenhang nicht näher betrachtet. Stattdessen werden Anforderungen direkt im Team Foundation Server als Work Items angelegt. Ein Design der zu entwickelnden Anwendung wird in

dieser Studienarbeit nicht beschrieben. Für MyDice ist zunächst ein Use Case vorgesehen. Dieser beschreibt den Vorgang des Würfelns und wird als Scenario im Teamprojekt angelegt. Zu Scenarios können Dokumente verlinkt werden, sodass beispielsweise UML-Diagramme oder Spezifikationen hinterlegt werden können. Ein weiteres Work Item im Testprojekt ist eine Quality of Service Anforderung. Und zwar soll der Würfel bei drei würfen mindestens einmal eine sechs anzeigen. Desweiteren werden noch fünf konkrete Aufgaben angelegt, die direkt als Arbeitspakete für die Entwicklung zu verstehen sind. Dabei geht es um das Anlegen einer Struktur für Visual Studio, das Schreiben von Testfällen und Code, sowie das Gestalten der Benutzeroberfläche der Anwendung.

Project: **AgileTest** Server: **win-ilnd3mymfdr** Query: **[None]**

ID	Rank	Work Item Type	State	Title	Area Path	Iteration Path	Description
59	1	Scenario	Active	Würfeln	\	\Version1	Der Benutzer würfelt eine Zahl zwischen 1 und 6.
60	1	QoS Requirement	Active	Jeder dritte Wurf 6	\BusinessLogik	\Version1	Wenn der Benutzer dreimal würfelt muss der Würfel mindestens einmal 6 anzeigen.
61	1	Task	Active	GUI designen	\UserInterface	\Version1	Oberfläche mit Würfelanzeige und Button designen.
62	1	Task	Active	Würfelvorgang implementieren	\BusinessLogik	\Version1	Ergebnisbestimmung des Würfelvorgangs.
63	1	Task	Active	3D Würfel	\UserInterface	\Version1	Der Würfel soll sich drehen in 3D.
64	1	Task	Active	Tests schreiben	\Testing	\Version1	Testfälle der Programm Logik schreiben.
65	1	Task	Active	VS Struktur anlegen	\	\	Visual Studio Solution und Struktur als Basis für die Entwicklung anlegen.

Tabelle 1 Konkrete Work Items in AgileTest

In der oberen Tabelle werden die Work Items beschrieben, die für die Entwicklung der Beispielanwendung verwendet werden sollen. Ein Projektmanager könnte diese über den Team Explorer oder über die Office Integration mit Excel erstellen und dann den Teammitgliedern zur Bearbeitung zuweisen. Hier ist zu erwähnen, dass sämtliche Work Items der Iteration Version1 zugewiesen sind. Zum Beispiel könnten Szenarien und Anforderungen einem Meilenstein „Designspezifikation“ zugeordnet werden, um eine bessere Strukturierung im Prozess zu erreichen.

5.1.3.1 Erstellen der Code Grundlage

Um mit der Entwicklung zu beginnen, muss zunächst ein Gerüst, also mindestens eine Solution in Visual Studio erstellt werden, die dann der Versionsverwaltung hinzugefügt werden kann. Jedes Teamprojekt besitzt einen eigenen Bereich, in dem Visual Studio Projekte und Quellcode hinterlegt und versioniert werden kann, was dank Team Explorer direkt in Visual Studio möglich ist. Für die Beispielanwendung wurde eine neue Visual Studio Solution mit einem C# WPF-Projekt als Gerüst erstellt. Über einen Rechtsklick auf das Projekt im Solution Explorer von Visual Studio kann es dann der Versionsverwaltung hinzugefügt werden. Wie in der unteren Abbildung werden sämtliche Dateien, die zu dem Projekt gehören und jetzt eingecheckt wurden, mit einem blauen Schloss dargestellt. Es existiert jetzt eine verwaltete Version im Team Foundation Server sowie eine lokale

Version auf der Festplatte des Benutzers, der eingecheckt hat. Diese lokale Kopie wird auch als Workspace bezeichnet. Sollten jetzt Änderungen an der lokalen Version vorgenommen werden, wird dies bemerkt und der Team Foundation Server listet die geänderten Dateien in einem weiteren Fenster als Pending Changes auf. Sollte die Datei zwischenzeitlich von einem anderen Benutzer verändert worden sein und daher lokal nicht mehr auf dem aktuellen Stand ist, wird dies in der Versionsverwaltung angezeigt.

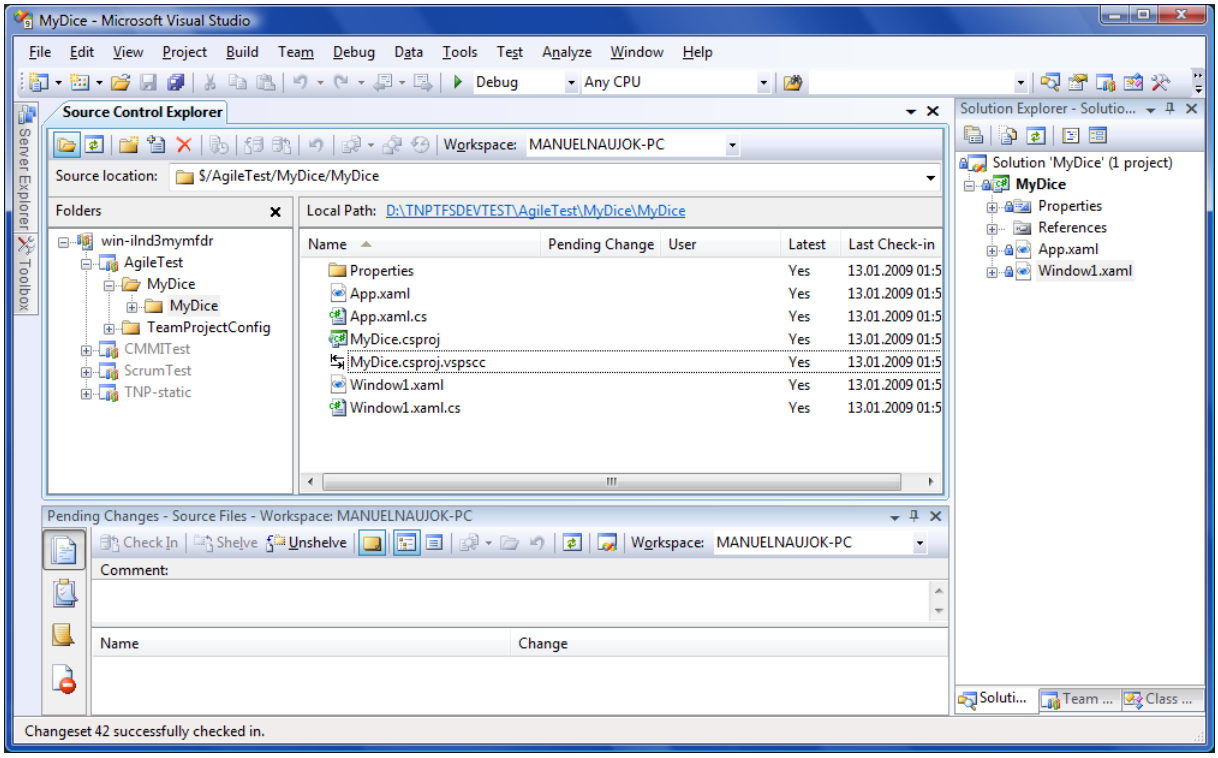


Abbildung 13 Versionsverwaltung in AgileTest

Selbstverständlich bietet die Versionsverwaltung vom Team Foundation Server alles, was Entwickler brauchen. So werden zum Beispiel Branches unterstützt, sodass an mehreren Versionen gleichzeitig gearbeitet werden kann. Zusätzlich lassen sich spezielle Regeln festlegen, sodass nicht jeder irgendwas einchecken kann. Diese sogenannten Check-In Policies werden beim Einrichten des Buildsystems vorgestellt. Die Projektstruktur ist jetzt erst mal sicher archiviert, sodass der entsprechende Task als abgeschlossen markiert werden kann.

65	Task	1	Closed	Manuel DEV	VS Struktur anlegen

Abbildung 14 Visual Studio Solution archiviert

Als nächstes würde ein Projektmanager seine Entwickler und Tester anhalten Testfälle für die Anwendung zu entwickeln, sodass später automatisiert getestet werden kann.

5.1.3.2 Schreiben der Testfälle

Dafür muss sich das Team zunächst Gedanken machen, was alles getestet werden soll. Erwartungsgemäß gibt es auch dafür Dokumentvorlagen um diese Testfälle formal zu beschreiben. An dieser Stelle wird angenommen, dass sämtliche Testfälle bekannt sind, sodass die Entwickler die

erste Architektur und die Tester zumindest die Testfälle schreiben können. Als Architektur wird im Rahmen der Beispielanwendung lediglich die Klasse Dice betrachtet. Wie im unteren Klassendiagramm erkennbar ist, besitzt diese Klasse neben einem Konstruktor zwei Methoden. Da es in dieser Studienarbeit darum geht den Team Foundation Server vorzustellen, wird auf Details des Designs der Beispielanwendung nicht näher eingegangen. Um das Gerüst als Basis allen Teammitgliedern zur Verfügung zustellen, muss es eingecheckt werden.

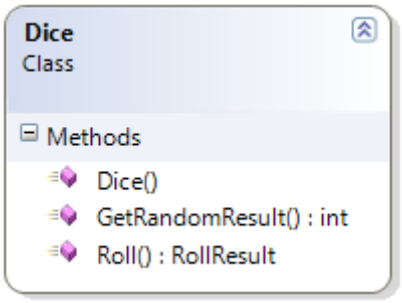


Abbildung 15 Klassendiagram zu Klasse Dice in AgileTest

Anschließend können die Entwickler und Tester die aktuellsten Quellen abfragen und Testfälle erzeugen. In Visual Studio können Unit Tests zu einer Klasse einfach über einen Rechtsklick in der IDE auf den Klassennamen erzeugt werden. Dadurch wird ein neues Projekt in der Solution angelegt, welches die Unit Tests für die Methoden der Originalklasse bereitstellt. Nachdem diese vervollständigt wurden, sollten sie zu einer Testliste zusammengefasst werden. Visual Studio stellt dafür einen Testlisten Editor zur Verfügung.

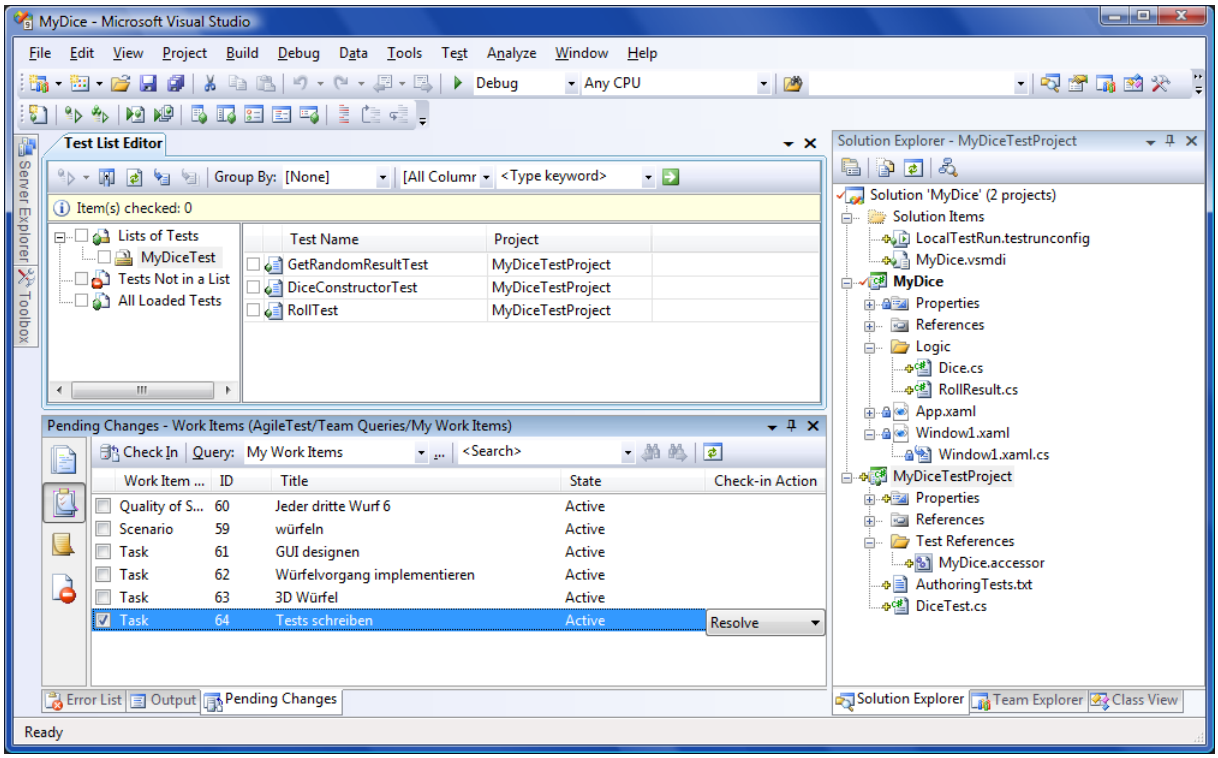


Abbildung 16 Erstellung einer Testliste in AgileTest

Wie im oberen Bild gezeigt, müssen die erstellten Testfälle und die Testliste auch eingecheckt werden, damit sie von alle Teammitgliedern verwendet werden können. Dabei wird in der Abbildung auch schön gezeigt, wie die geänderten Dateien eingecheckt werden. Zum Beispiel können Check-Ins mit Work Items assoziiert werden. Im Beispiel kann das Work Item „Tests schreiben“ dadurch abgeschlossen, also auf „Resolve“ gesetzt werden.

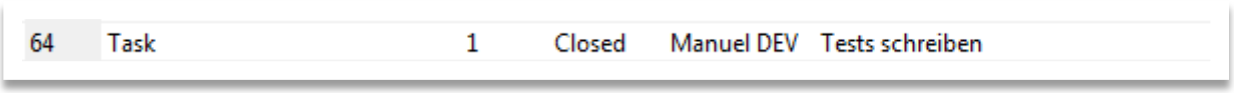


Abbildung 17 Testfälle geschrieben

Nachdem nun die erste Architektur und die Tests der Beispielanwendung von den Teammitgliedern erstellt und archiviert wurden, kann das Buildsystem eingerichtet werden.

5.1.3.3 Einrichten des Buildsystems

Da es sich hierbei um eine administrative Tätigkeit handelt, muss dies ein Benutzer tun, der die nötigen Rechte hat. Während die bisherigen Aktivitäten im Testprojekt vom Benutzer Manuel DEV durchgeführt werden konnten, muss jetzt der Benutzer TFSSETUP verwendet werden. Dafür wird Visual Studio im entsprechenden Benutzerkontext gestartet und über den Punkt Build im Projekt AgileTest im Team Explorer eine neue Build Definition erstellt. Dabei muss zunächst das zu kompilierende Projekt aus der Versionsverwaltung ausgewählt werden. Anschließend muss ein MSBuild Projekt erstellt werden, dass übrigens ebenfalls archiviert wird. Glücklicherweise führt der Team Foundation Server den Benutzer dabei durch einen selbsterklärenden Assistenten. Bei den Optionen wird festgelegt, dass bei jedem Build die erstellte Testliste ausgeführt werden soll. Zusätzlich soll auch die statische Codeanalyse durchgeführt werden. Dadurch wird erreicht, dass zu jedem Build eine Aussage der Qualität gemacht werden kann.

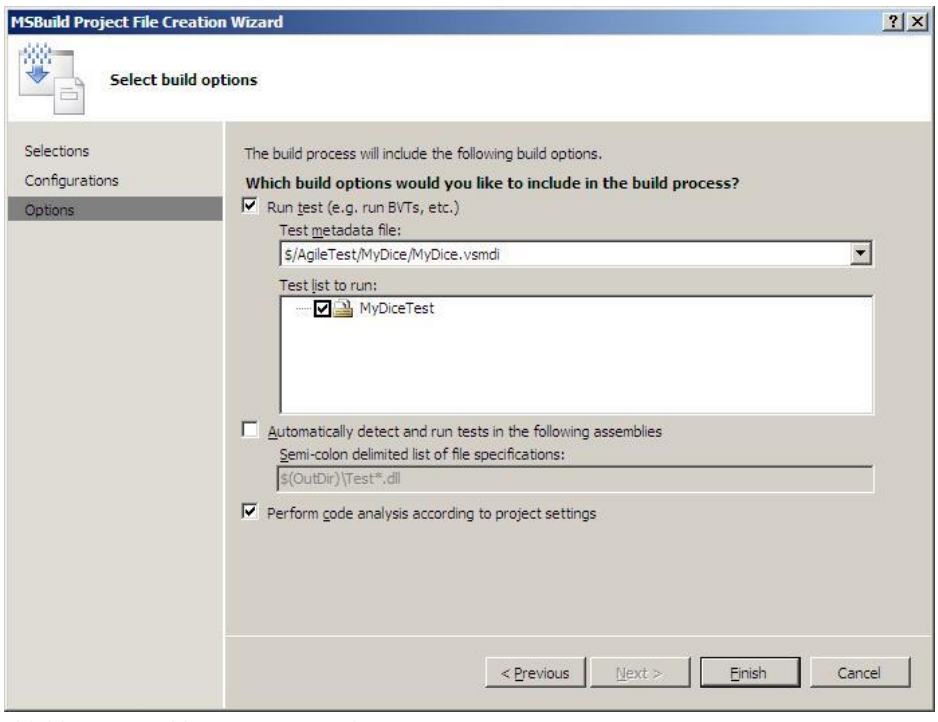


Abbildung 18 Build-Optionen in AgileTest

In der oberen Abbildung ist erkennbar, dass die Testliste direkt aus der Quellcodeverwaltung geladen wird. Würden mehrere Testlisten existieren, könnten diese hier ausgewählt werden. Nachdem definiert wurde „was“ und „wie“ kompiliert werden soll, muss noch eingestellt werden, „wann“ ein Build erstellt werden soll. Der bereits erwähnte Assistent bietet dafür mehrere Optionen an und wird in der unteren Abbildung dargestellt. Für das Testprojekt wird ein Build immer dann gestartet, wenn ein Check-In erfolgt ist. Dadurch wird eine Codeänderung automatisch in Bezug auf Korrektheit der gesamten Anwendung hin überprüft.

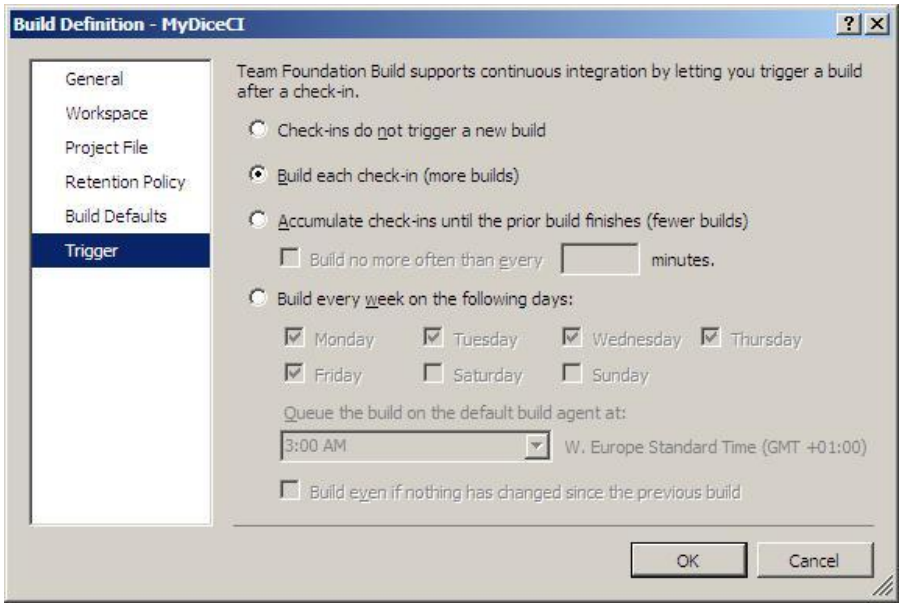


Abbildung 19 Build-Ereignisse in AgileTest

Damit ist das Buildsystem im Testprojekt eingerichtet.

5.1.3.4 Einrichten der Check-In Policies

Sollte jetzt ein Check-In erfolgen, wird automatisch die aktualisierte Version im Team Foundation Server kompiliert. Da ein Buildprozess einiges an Ressourcen beansprucht, sollten schlechte Codeänderung nicht erlaubt werden. Eine Möglichkeit das zu kontrollieren besteht in dem Einrichten sogenannter Check-In Policies. Der Benutzer TFSSETUP verfügt über die nötige Berechtigung diese Einstellung vorzunehmen. Wie im unteren Bild gezeigt, bietet der Team Foundation Server vier Regeln: Builds, Code Analysis, Testing Policy und Work Items. Dank letzterem kann vorgeschrieben werden, dass Check-Ins grundsätzlich mit einem Work Item assoziiert werden müssen. Dadurch kann sichergestellt werden, dass nicht irgendwas eingecheckt wird, sondern nur Relevantes. Das gleiche gilt für das lokale Ausführen einer statischen Codeanalyse oder einer Testliste. Codeänderungen werden also nur dann archiviert, wenn eine gewisse Codequalität messbar ist. Eine letzte Policy unterbindet Check-Ins, wenn der letzte Build fehlgeschlagen ist. Dadurch kann aufgrund des letzten gültigen commits der Schuldige ermittelt und die archivierte Version zuerst repariert werden, bevor weitere Änderungen vorgenommen werden können.

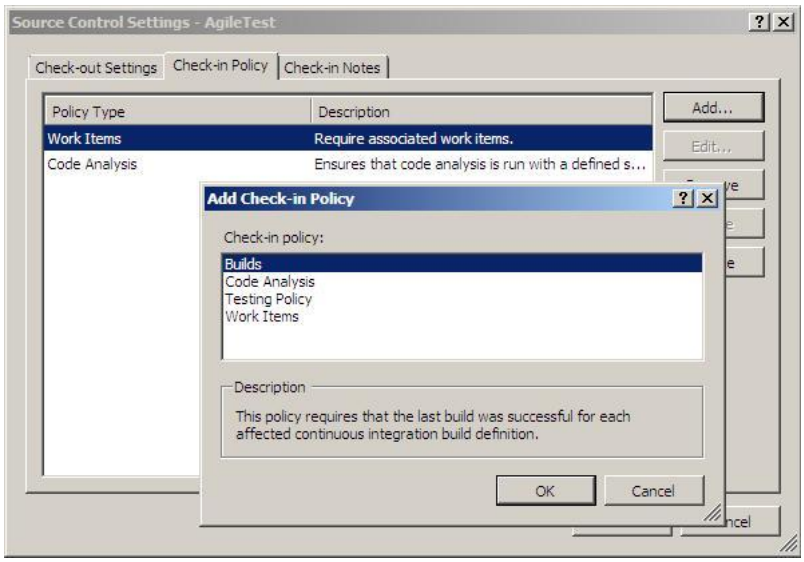


Abbildung 20 Check-In Policies in AgileTest

In dem Testprojekt zur Entwicklung von MyDice auf agile Art und Weise werden lediglich die Codeanalyse und Verbindung zu Work Items vorausgesetzt und als Regel aktiviert. Dadurch können die Tasks 47 und 48 abgeschlossen werden.

47	Task	Closed	TFSSETUP	Set up: Set Check-in Policies
48	Task	Closed	TFSSETUP	Set up: Configure Build

Abbildung 21 Buildsystem und Check-In Policies eingerichtet

Anschließend werden noch die Work Items betrachtet, die bei der Erstellung des Testprojekts automatisch angelegt wurden. In dieser Studienarbeit kann aufgrund des Umfangs leider nicht auf alle Möglichkeiten des Projektmanagements mit dem Team Foundation Server eingegangen werden. Daher werden vorzugsweise die Aspekte vorgestellt, die für den Entwickler interessant sind. Sämtliche Tasks, die aus diesem Grund nicht näher bearbeitet werden, werden als obsolet betrachtet und geschlossen.

44	Task	Closed	TFSSETUP	Set up: Set Permissions
45	Task	Closed	TFSSETUP	Set up: Migration of Source Code
46	Task	Closed	TFSSETUP	Set up: Migration of Work Items
49	Task	Closed	TFSSETUP	Set up: Send Mail to Users for Installation and Getting started
50	Task	Closed	TFSSETUP	Create Vision Statement
51	Task	Closed	TFSSETUP	Set up: Create Project Description on Team Project Portal
52	Task	Closed	TFSSETUP	Create Personas
53	Task	Closed	TFSSETUP	Define Iteration Length
54	Task	Closed	TFSSETUP	Create Test Approach Worksheet including Test Thresholds
55	Task	Closed	TFSSETUP	Brainstorm and Prioritize Scenarios List
56	Task	Closed	TFSSETUP	Brainstorm and Prioritize Quality of Service Requirements List

Abbildung 22 Obsolete Tasks in AgileTest

Dadurch ist sämtliche Vorbereitung getroffen und die einzigen Work Items, die noch aktiv sind, betreffen die Entwicklung der Beispielanwendung. Ein Auszug aus Tabelle 1 aus Unter-Unterabschnitt 5.1.3 Praktischer Entwicklungsprozess ist hier abgebildet.

ID	Work Item Typ	Zustand	Name	Zugewiesen
59	Scenario	Active	Würfeln	Manuel DEV
60	QoS Requirement	Active	Jeder dritte Wurf 6	Manuel Naujoks
61	Task	Active	GUI designen	Manuel DEV
62	Task	Active	Würfelvorgang implementieren	Manuel DEV
63	Task	Active	3D Würfel	Manuel Naujoks

Tabelle 2 Aktive Work Items in AgileTest, die noch bearbeitet werden müssen

Ursprünglich waren diese fünf Work Items dem Benutzer Manuel DEV zugeordnet, da sie von diesem angelegt worden sind. Um die weitere Entwicklung etwas aufzuteilen, wurde Item 60 und 63 dem Benutzer Manuel Naujoks zugewiesen.

5.1.3.5 Integration von Codeänderungen

In Unterabschnitt 5.1 MSF for Agile Software Development – v4.2 wurde eine Team Query zum Anzeigen aller Work Items verwendet. Eine weitere Abfrage listet nur auf, was dem angemeldeten Benutzer auch wirklich zugewiesen wurde. Zum Beispiel würde Manuel DEV sich eine Übersicht über alle seine offenen Aufgaben verschaffen und diese dann abarbeiten. So würde er sehen, dass er zunächst den Use Case des Würfelvorgangs implementieren soll. Dafür muss zunächst der aktuelle Code ausgecheckt werden um sicherzustellen, dass die neusten Quellen verwendet werden. Ein Doppelklick auf die Solution in der Versionsverwaltung lädt das aktuelle Projekt, öffnet es und sperrt automatisch die Dateien, die verändert werden.

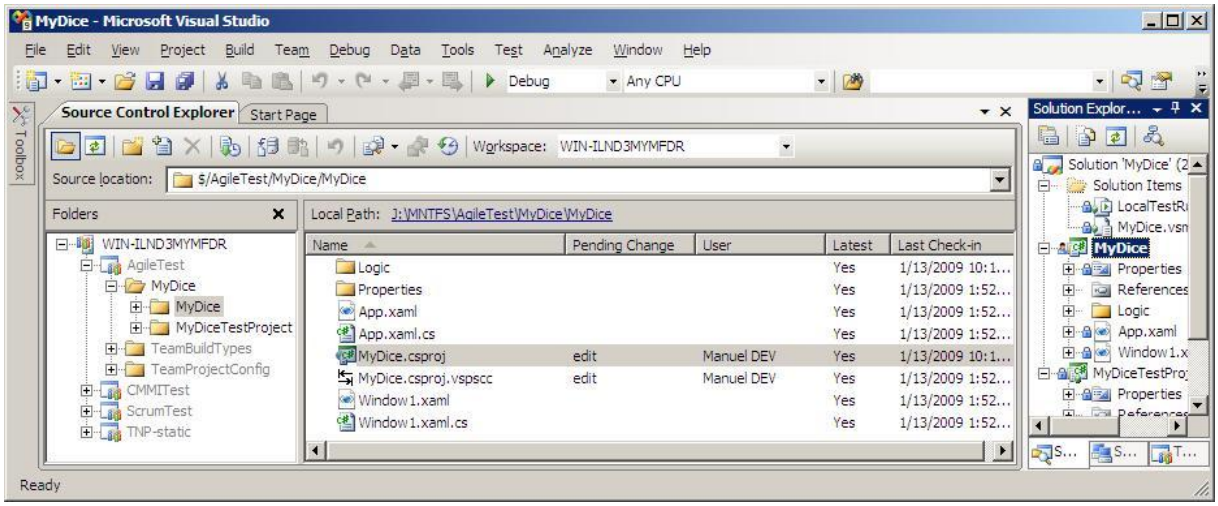


Abbildung 23 Visual Studio sperrt geänderte Dateien automatisch in der Versionskontrolle

Zur gleichen Zeit erkennt auch der Benutzer Manuel Naujoks seine Aufgaben und versucht ebenfalls die aktuellsten Quellen auszuchecken. Dabei sieht er, dass Manuel DEV bestimmte Dateien im Projekt bereits zur Verarbeitung gesperrt hat. Nachdem die Quellen ausgecheckt wurden, erscheint bei manchen Dateien anstelle des blauen Schlosses ein Avatar, der signalisiert, dass die Datei von

einem anderen Benutzer bereits gelockt wurde. Jedenfalls versucht Benutzer Manuel Naujoks das Work Item 63 zu bearbeiten und checkt am Abend sein Ergebnis ein. Dabei markiert er den Task nicht als abgeschlossen, sondern assoziiert ihn nur mit seinem Check-In. Das wird auch in der History des Work Items angezeigt, in der alle Änderungen protokolliert werden. Die untere Abbildung zeigt, dass das Changeset 47, also die eingecheckte Codeänderung eine Auswirkung auf die Aufgabe hat.

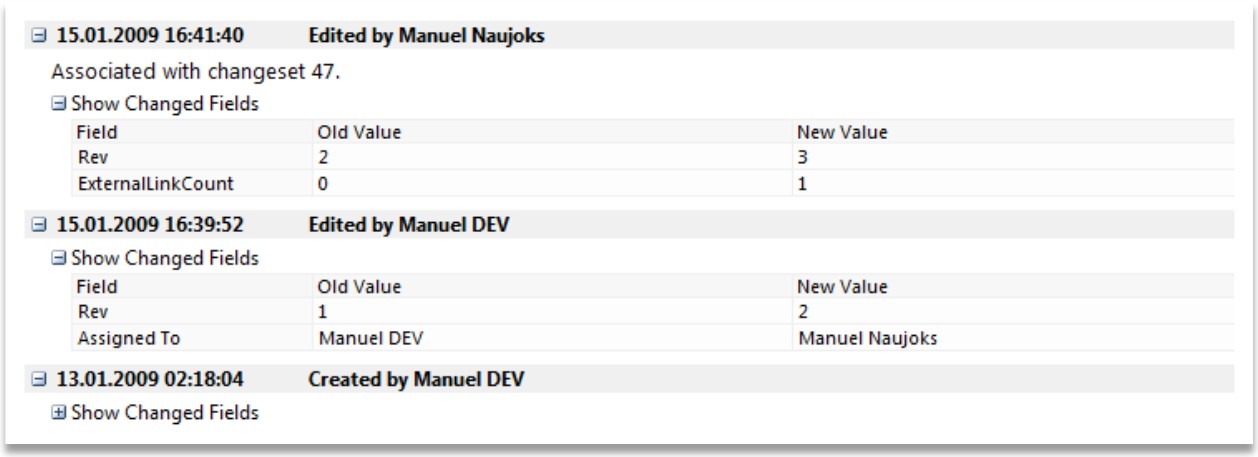


Abbildung 24 History von Task 63 "3D Würfel" in AgileTest

Es wird angezeigt, dass das Feld ExternalLinkCount durch den Check-In verändert wurde. Ein Blick in das Linkverzeichnis des Work Items zeigt, dass das entsprechende Changeset direkt verbunden wurde und mit einem Doppelklick jederzeit geöffnet werden kann. An dieser Stelle sei angemerkt, dass IDs von Work Items und Changesets über sämtliche Teamprojekte im Team Foundation Server eindeutig sind. Über Links können auch andere Work Items und Dokumente verlinkt werden.

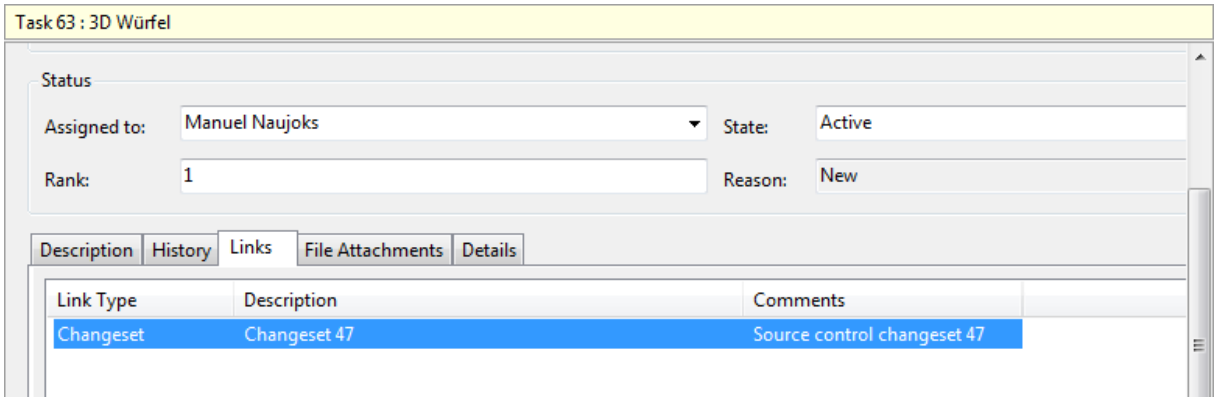


Abbildung 25 Linkverzeichnis von Task 63 "3D Würfel" in AgileTest

Nachdem Manuel Naujoks eingecheckt hat, möchte auch der Benutzer Manuel DEV seine Änderung am Code auch archivieren. Neben der eigentlichen Logik musste zusätzlich ein Fehler in den Testfällen korrigiert, sowie zwei neue Unit Tests geschrieben und der bereits erwähnten Testliste hinzugefügt werden. Dementsprechend wird der entsprechende Check-In mit dem Use Case „würfeln“, der Aufgabe zur Implementierung und der bereits geschlossenen Aufgabe zur Entwicklung der Tests assoziiert. Task 62 wird sogar als abgeschlossen betrachtet und daher mit „Resolve“ committed, was in der unteren Abbildung gezeigt wird.

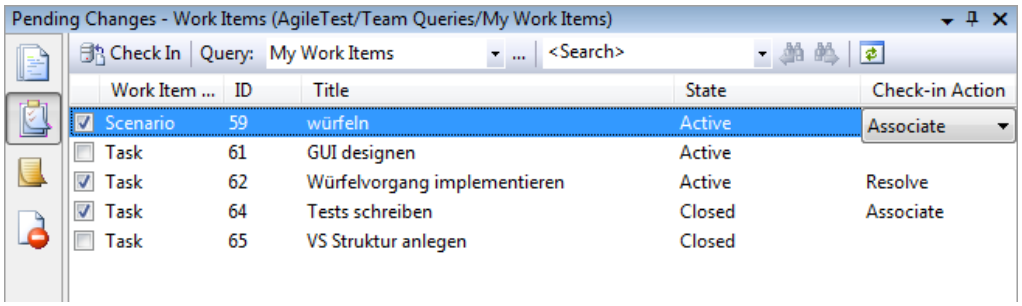


Abbildung 26 Check-In des Codes zur Logik des Würfels

Obwohl Task 64 bereits geschlossen ist, können Check-Ins vorgenommen werden, die ihm zugeordnet sind. Dadurch wird das Work Item allerdings nicht wieder aktiviert. Anschließend führt der Team Foundation Server automatisch ein Integration Build durch und testet die Codeänderung. Über den Team Explorer können sämtliche Builds des Testprojekts eingesehen werden. Mit einem Doppelklick auf den aktuellsten Build wird das Protokoll angezeigt. Außerdem wird angezeigt, dass nur drei von fünf Tests erfolgreich waren. Wie die untere Abbildung zeigt, sind immer noch zwei fehlgeschlagen, sodass die Implementierung des Würfels noch einmal überarbeitet werden muss.

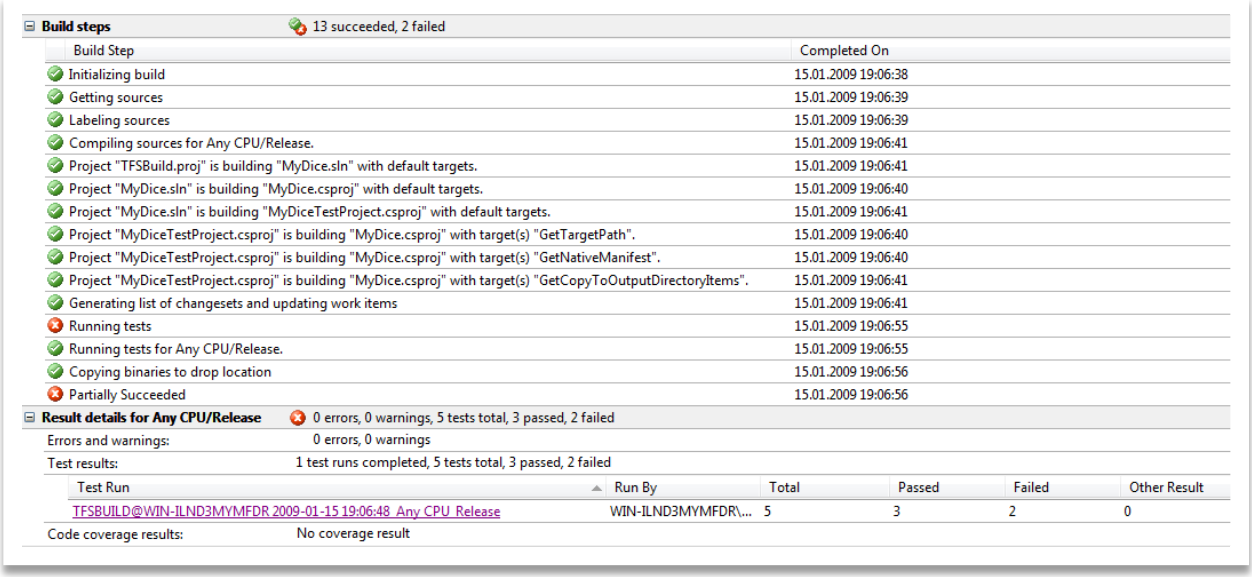


Abbildung 27 Protokoll eines Integration Builds in AgileTest

Zum Beispiel könnte jetzt ein Bug angelegt werden, der mit dem Testprotokoll aus dem Buildprotokoll verbunden wird. Anschließend kann dieser Bug einem Tester oder Entwickler zugewiesen werden, sodass er bearbeitet und behoben werden kann. Das Prinzip ist dabei genau das gleiche, wie bei der Bearbeitung einer normalen Aufgabe. Die Entwicklung einer Anwendung kann strukturiert mithilfe von Work Items erfolgen. Diese werden vom Projektmanager und den Entwicklern entweder manuell bearbeitet oder werden automatisch von Team Foundation Server Komponenten aktualisiert. Zum Beispiel können so Codeänderungen oder Builds zugeordnet werden. Einem Continuous Integration Build werden alle Work Items zugeordnet und umgekehrt, die seit dem letzten Build abgeschlossen wurden. Dadurch kann jederzeit ermittelt werden, ab welcher Version ein spezielles Feature oder ein Bug-Fix Einzug in die Anwendung erhalten hat. Durch diese Möglichkeiten bietet der Team Foundation Server eine sehr hohe Transparenz für Entwickler.

Doch wie sieht diese Durchsichtigkeit aus der Richtung des Managements aus?

5.1.4 Management

Normalerweise arbeitet ein Team an einem Projekt und lässt sich daher nur ungern stören um den Stand der Entwicklung mit Vorgesetzten und Managern zu besprechen. Aus diesem Grund sind die Daten der Projekte, die mit dem Team Foundation Server unterstützt werden nicht nur für Entwickler verfügbar. Über eine bereits erwähnte Projektwebsite in Form eines SharePoint Portals, können sämtliche Informationen zu einem Teamprojekt eingesehen und bedingt verwaltet werden. Die Seite wird automatisch durch das Anlegen eines Teamprojekts erstellt und hängt von der Vorlage des ausgewählten Prozessmodells ab. Das Portal des AgileTest Projekts kann über einen Browser von jedem Teammitglied aufgerufen werden. Wie im unteren Bild gezeigt, bietet es über den linken Navigationsbereich unter anderem Verweise zu wichtigen Dokumenten, wie zum Beispiel der Process Guidance, an. Zusätzlich stehen ein Kalender und ein Diskussionsforum bereit, über das Informationen zu dem gemeinsamen Projekt gesammelt werden können. Am wichtigsten für Benutzer aus dem Management werden aber wohl die dort einsehbaren Berichte sein. Über SQL Server Reporting Services können sämtliche Daten aus einem OLAP Cube des Team Foundation Servers angezeigt werden.

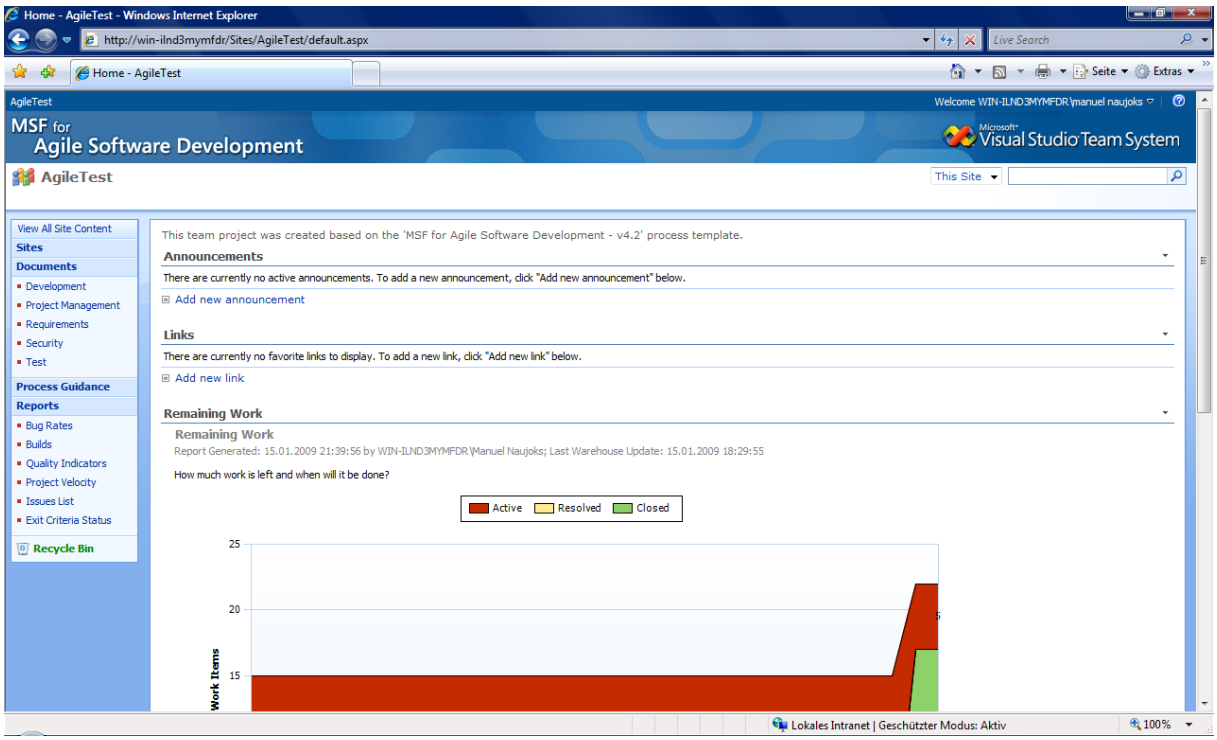


Abbildung 28 Projektportal von AgileTest

Da die Reports den SQL Server 2008 benutzen, können natürlich jederzeit eigene Reports definiert werden. Dennoch bietet MSF Agile einige interessante Berichte, die Informationen zur Vorgehensweise und Arbeit des Teams grafisch darstellen können. Zum Beispiel möchte der Entwickler und Teamleiter Manuel Naujoks über die Geschwindigkeit der MyDice Entwicklung informiert werden ohne mit seinen Teammitgliedern zu reden. Dazu muss er lediglich auf das Projektportal des Projekts AgileTest gehen und sich den Project Velocity Report ansehen. Dieser Bericht stellt zu einer Zeitachse die Menge an Transaktionen im Projekt dar, die zum lösen (Resolved) oder schließen (Closed) eines Work Items geführt haben. Dabei unterscheidet der Report zwischen

den unterschiedlichen Typen, sodass ein realistischer Eindruck gewonnen werden kann. Die untere Abbildung zeigt den Report im Browser.

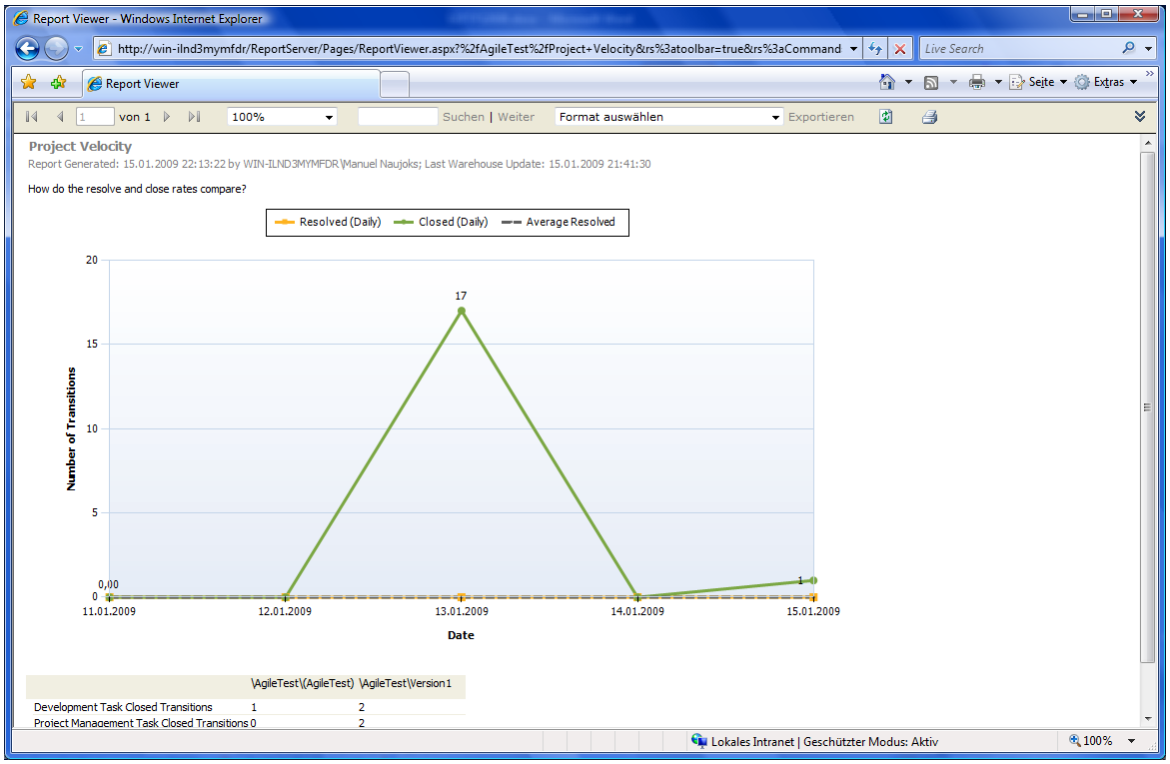


Abbildung 29 Project Velocity von AgileTest

Ein weiterer aussagekräftiger Bericht ist der Ramaining Work Report, der unten dargestellt ist.

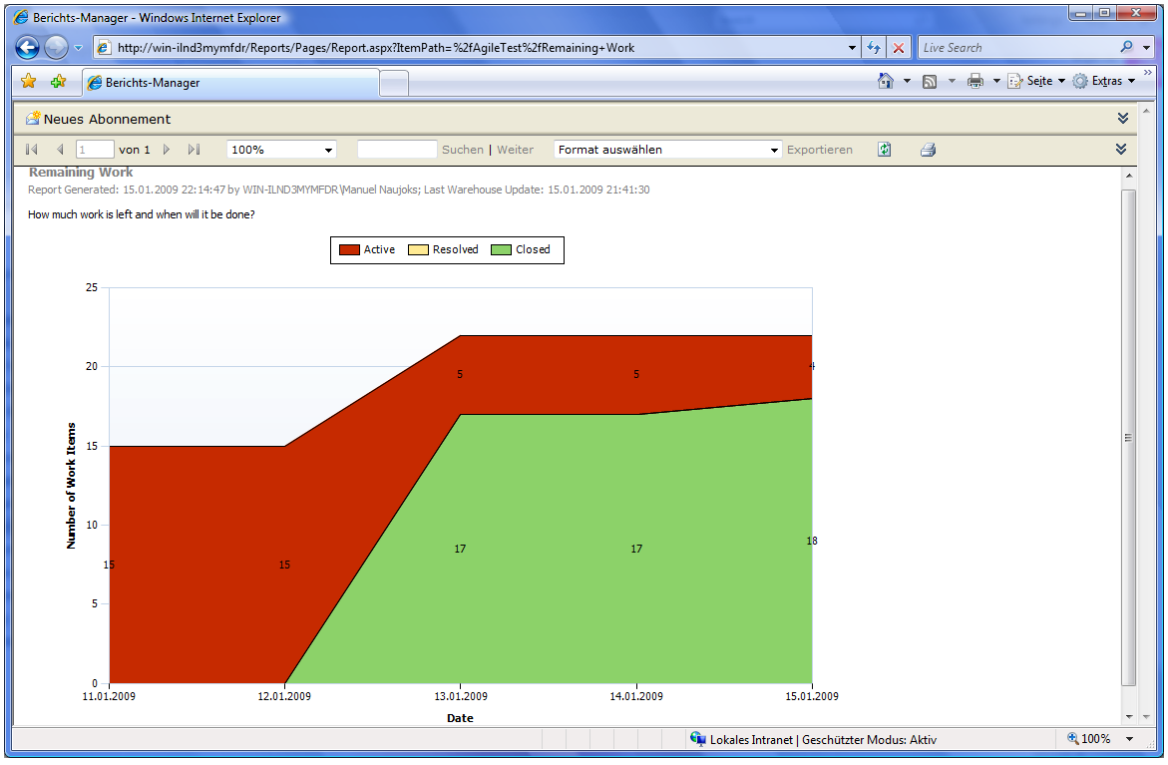


Abbildung 30 Remaining Work in AgileTest

In dieser Grafik wird die Anzahl der Work Items im Projekt in aktiv, bereits gelöst und geschlossen aufgeteilt. Zusätzlich wird über einen Zeitstrahl die Veränderung der Aufteilung angezeigt, sodass sich der Teamleiter jederzeit ein Bild darüber machen kann, wie viel Arbeit noch in dem Projekt zu tun ist. Nachdem der Managementbenutzer den Überblick über das Projekt gewonnen hat, möchte er vielleicht wissen, ob bereits etwas ausgeliefert werden kann. Dies kann der Quality Indicator Report anzeigen. Hier werden zu jedem Build die Anzahl der Testfälle angezeigt, die bestanden oder nicht bestanden wurden. Daraus kann eine qualitative Einschätzung des entstehenden Programms vorgenommen werden. Die untere Abbildung zeigt dass sich die Beispielanwendung MyDice mit den Check-In Vorgängen kontinuierlich verbessert hat, da immer mehr Testfälle erfolgreich abgeschlossen werden.

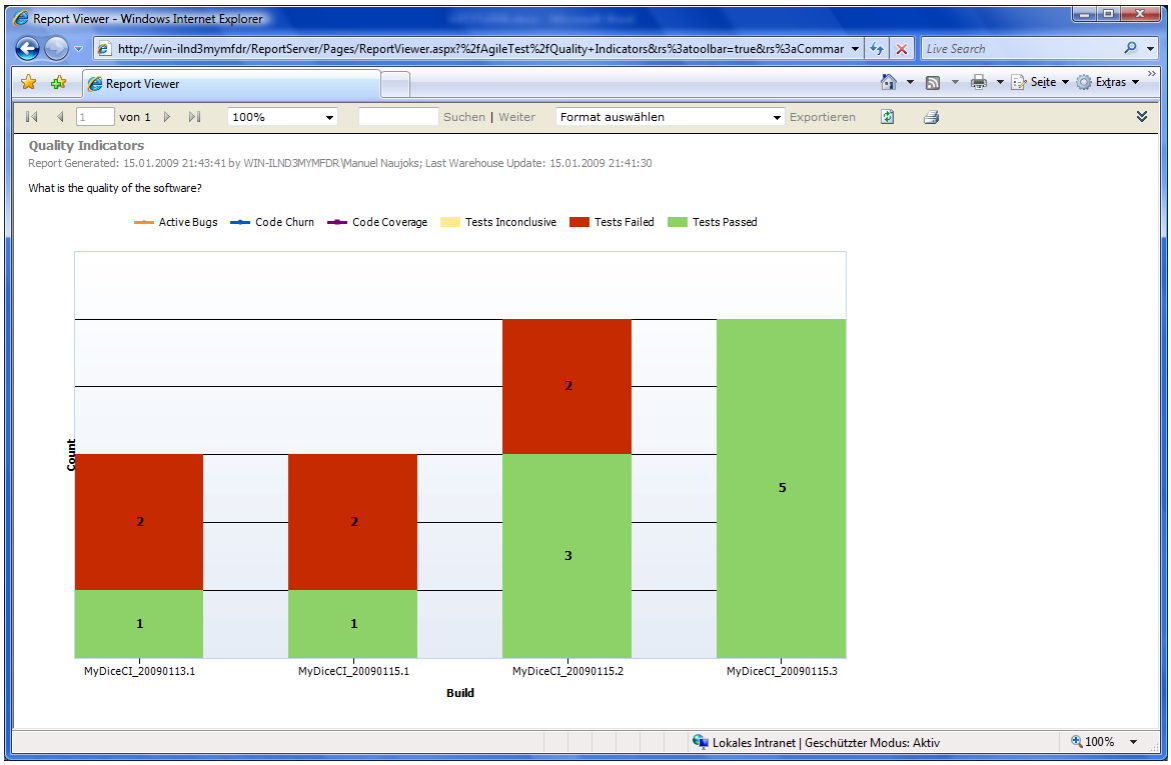


Abbildung 31 Quality Indicators in AgileTest

Dadurch würde ein Teamleiter in Erfahrung bringen können, dass der letzte Build vielleicht schon an den Kunden ausgeliefert werden könnte, da sämtliche Features erfolgreich getestet werden konnten. Neben diesen interessanten Berichten gibt es noch weitere, die hier nicht näher betrachtet werden. Je umfangreicher die Projekte werden, desto ausführlicher werden auch die Reports. Sollten diese Informationen nicht ausreichen oder der Teamleiter möchte genaueres über bestimmte Work Items wissen oder diese sogar steuern, müsste er eigentlich den Team Explorer verwenden. Da Manager in der Regel keine Visual Studio Version installiert haben sondern nur mit Microsoft Office umgehen können, bietet Team Foundation Server eine Integration in Microsoft Excel und Microsoft Project. Zunächst kann in Excel eine Team Query genutzt werden, um eine Mappe mit einer Liste sämtlicher Work Items zu füllen. Diese können dann verändert und Teammitgliedern zugewiesen werden, wie im unteren Screenshot gezeigt. Anschließend können die Änderungen über den Button Publish gespeichert werden. Natürlich können die Daten auch in Excel weiterverarbeitet werden um beispielsweise ein schnelles Diagramm zu erzeugen.

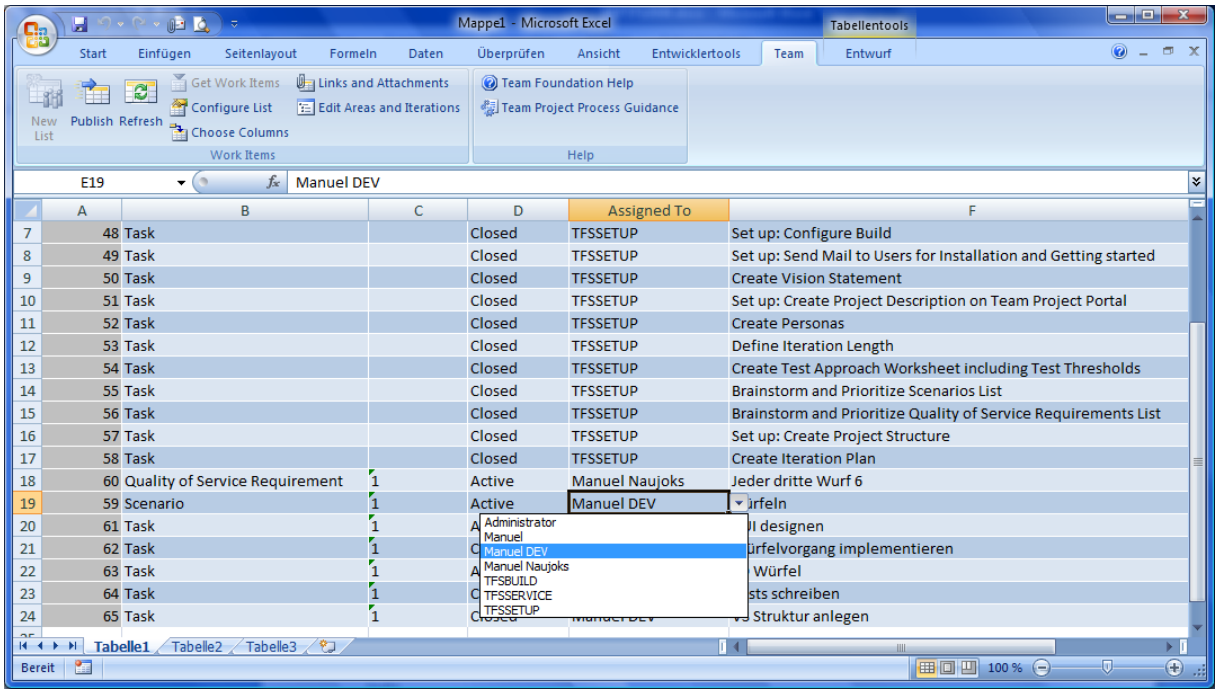


Abbildung 32 Work Items von AgileTest in MS Excel

Für eine zeitliche Strukturierung von Projekten, verwenden Teamleiter häufig Project. Dank dessen Anbindung zum Team Foundation Server können genau wie in Excel Team Queries geladen werden um Work Items bearbeiten zu können. Diese lassen sich dann wie gewohnt als Arbeitspakete zeitlich und topologisch anordnen. So kann wie in der unteren Abbildung gezeigt, der Use Case „würfeln“ aus den verschiedenen Tasks bestehen.

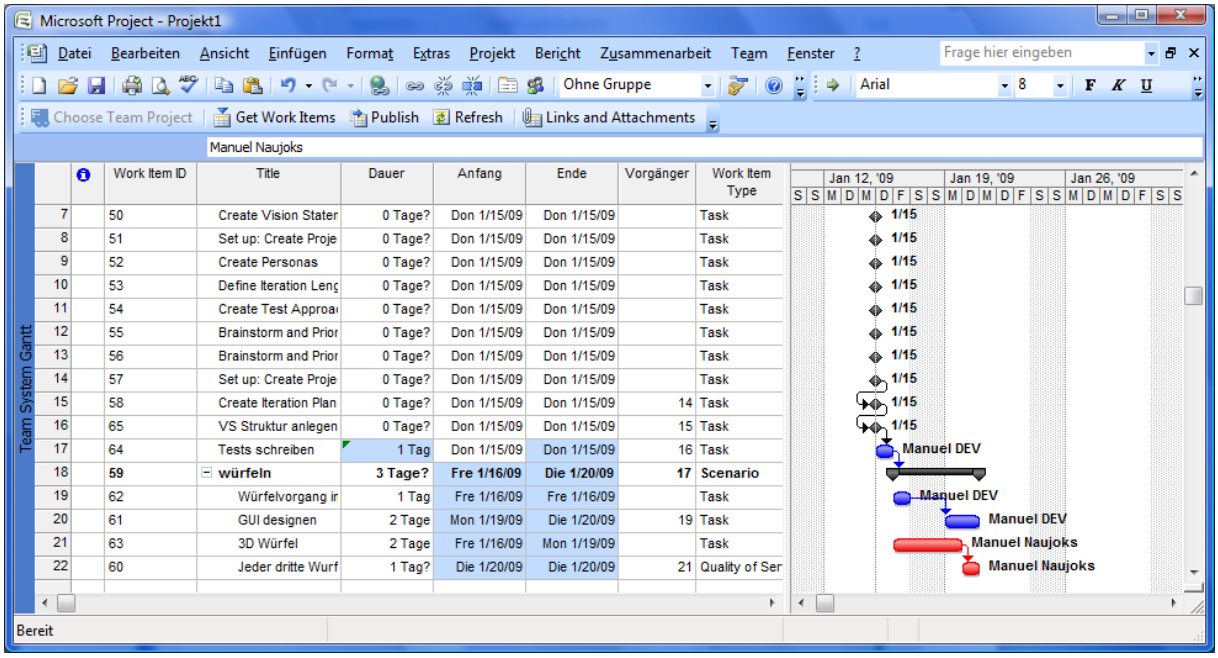


Abbildung 33 Work Items von AgileTest in MS Projekt

Nach einer zeitlichen Neusortierung können Änderungen der Work Items gespeichert werden. Die entstehenden Microsoft Office Dokumente können ebenfalls archiviert werden, indem sie im

Projektportal von AgileTest unter der Kategorie Shared Documents hochgeladen werden. Zusätzlich zur integrierten Anbindung an den Team Foundation Server existiert noch eine weitere Möglichkeit der Informationsbeschaffung mit Excel. So kann der Cube direkt als externe Datenquelle angegeben und dessen Daten für das Erzeugen einer Pivot-Tabelle genutzt werden. Dadurch können sämtliche Informationen beliebig miteinander kombiniert werden um sehr umfangreiche Statistiken zu erstellen. Das untere Bild zeigt als eine solche Statistik wie viele Work Items jeder Benutzer in welchen Zuständen zugeteilt bekommen hat. Die Datei kann dabei ebenfalls unter Shared Documents gespeichert und somit mehreren Personen zugänglich gemacht werden. Selbst Mitarbeiter die ansonsten keine Berechtigung haben irgendwelche Daten aus dem TFS einzusehen, könnten die Pivot-Tabelle als statische Tabelle verwenden.

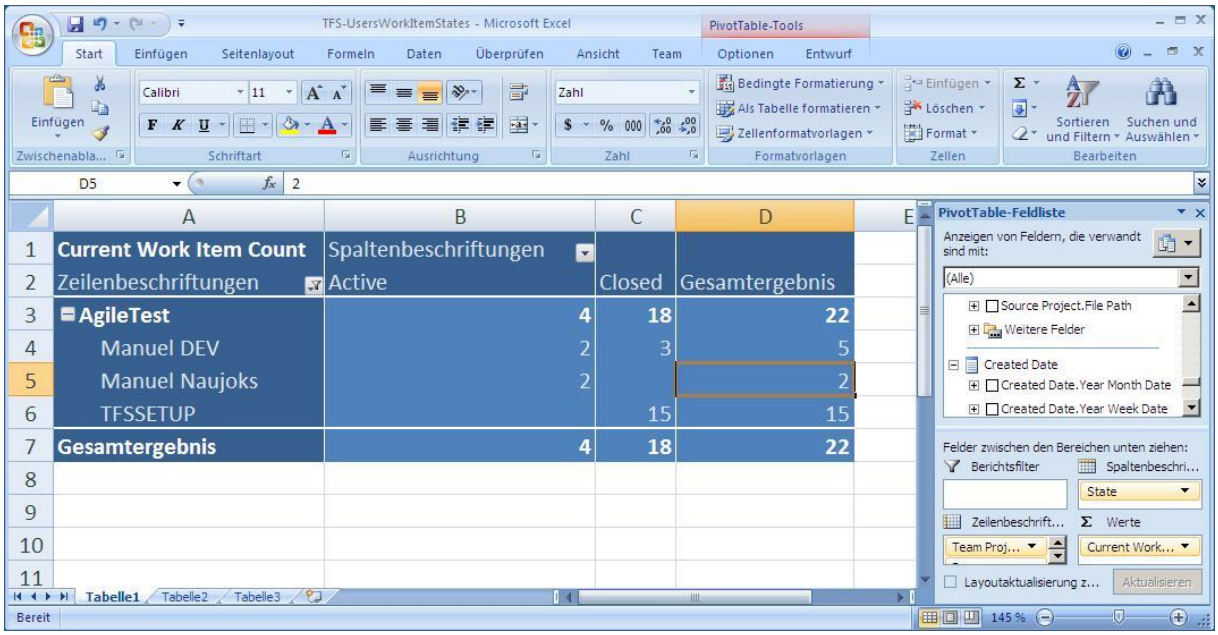


Abbildung 34 Excel Statistik der Work Items in AgileTest

Auf diese Weise kann sich der Teamleiter oder Manager die Informationen beschaffen und verwenden, die er benötigt. Natürlich kann es sein, dass viele Informationen nicht jedem zur Verfügung stehen sollen, was durch die Vergabe von entsprechenden Berechtigungen in der zuständigen Gruppe im SQL Server realisiert werden kann.

5.1.5 Schlussfolgerung

In diesem Unterabschnitt wurde das Prozessmodell MSF for Agile Software Development vorgestellt. Dabei wurden die wichtigsten Funktionen des Team Foundation Server anhand der Entwicklung der Beispielanwendung MyDice demonstriert. Es wurden Work Item verwendet, um die Arbeit zu strukturieren und Teammitgliedern zuzuweisen. Dadurch kann ein agiles Vorgehen erreicht werden. Besonderheiten des verwendeten Prozessmodells liegen vor allem in den verschiedenen Typen der Work Items. Das Buildsystem und die Versionsverwaltung können mit diesen Work Items arbeiten, sodass sich jederzeit nachverfolgen lässt wann eine Funktionalität kompiliert wurde, welche Test sie bestanden hat und wie sie eingecheckt wurde. Dadurch kann eine transparente Entwicklung gewährleistet werden, die selbst von Teamleitern und Managern beobachtet werden kann. Dafür wurde gezeigt, wie mit Berichten der Zustand eines Projekts ermittelt werden kann, ohne mit den zuständigen Entwicklern zu reden. Auch wenn die Beispielanwendung aufgrund des beschränkten

Umfangs dieser Ausarbeitung nicht vollständig entwickelt werden konnte, wurde das Prinzip der Prozesssteuerung mit Work Items ausführlich aufgezeigt. Leider existieren auch einige Einschränkungen. Zum Beispiel ist es nicht möglich Work Items zu schachteln um sie hierarchisch aufzubauen. Die einzige Möglichkeit die Entwicklungsaufgaben dem Use Case „würfeln“ zuzuordnen, besteht in dem Setzen von Links zu den jeweiligen Tasks. Ein umfangreicherer Editor zur besseren Unterstützung der Top-Down Entwicklung wäre daher noch wünschenswert. Desweiteren können keine Abhängigkeiten zwischen Work Items erstellt werden. Sämtliche Aufgaben und Anforderungen sind daher als parallelisierbar zu betrachten. Die einzige Möglichkeit zur zeitlichen Strukturierung besteht in dem Hinzufügen von externer Semantik über Microsoft Office Project. An dieser Stelle soll der Team Foundation Server Microsoft Office natürlich nicht ersetzen, aber einfache Abhängigkeiten sollten in Form eines Vorgängerfeldes den Work Items hinzugefügt werden können. Dann wurde demonstriert, dass Mitglieder des Teams in AgileTest fast nach Belieben Work Items editieren und damit deren Zustand verändern können. Auch wenn der Team Foundation Server an dieser Stelle ein sehr mächtiges Hilfsmittel im Prozess ist, ersetzt er keine Teambesprechungen und sonstige Koordination. Ein letzter Punkt, der nachteilig aufgefallen ist, besteht darin, dass Work Items immer nur einer Person zugeordnet werden können. Zwar kann diese Zuordnung nachträglich geändert werden, allerdings wäre es praktisch ein Scenario mehreren Entwicklern zuordnen und allen anderen Benutzern die Arbeit an dem selbigen verweigern zu können. Zwar besitzt der Team Foundation Server ein sehr umfangreiches Rechtemanagement allerdings sollte eine solche Einstellung auch direkt im Editor der Work Items vorgenommen werden können. Zusammenfassend ist zu sagen, das MSF Agile ein sehr unkompliziertes Prozessmodell ist, dass die Entwickler in der Standardausprägung wenig einschränkt und sämtliche Vorgehensweisen nur minimal, nämlich in den Workflows der Work Items definiert. Trotzdem kann die voll Funktionsvielfalt des Team Foundation Servers in einem MSF Agile Projekt genutzt werden.

5.2 MSF for CMMI Process Improvement – v4.2

Die zweite Prozessvorlage, die im Team Foundation Server standardmäßig schon vorinstalliert ist, ist „MSF for CMMI Process Improvement – v4.2“. Nachdem ein neues Teamprojekt mit diesem Template erstellt wurde, wurden ähnlich wie bei MSF for Agile mehrere Work Items erstellt, die den Begin des Projekts unterstützen sollen. Da im vorangegangenen Unterabschnitt bereits sämtliche Funktionalitäten des Team Foundation Servers 2008 vorgestellt wurden, werden im Folgenden lediglich die Besonderheiten des CMMI-Modells beschrieben. CMMI ist die Abkürzung von Capability Maturity Model Integration und dient der systematischen und formalen Abwicklung von Prozessen und Vorgängen, sowie dessen Verbesserung und Weiterentwicklung. Im Gegensatz zu dem bereits erwähnten agilen Prozessmodell MSF for Agile besitzt die CMMI Version im Team Foundation Server ein wesentlich umfangreicheres Sortiment an Dokumenten. Dadurch kann eine gewisse Formalität und Bürokratie erreicht werden, die für die Entwicklung von professioneller Software vielleicht erforderlich ist. Dabei gliedert sie sich laut der CMMI Process Guidance in fünf sogenannte Tracks [Mic0715] die umfangreich beschrieben werden. Ein Projekt durchläuft diese Tracks, die auch als Phasen gesehen werden können. Es beginnt mit der Envision-Phase, also der Projektdefinition. Anschließend kommt die Plan-Phase, in der das Projekt hauptsächlich geplant wird und in die Build-Phase überleitet. Schließlich folgt die Stabilize-Phase und die Deploy-Phase. Dabei sei angemerkt, dass diese fünf Tracks sich wie in der unteren Grafik dargestellt, überlappen und so die erforderliche Dynamik erzeugen. Zwischen diesen Überlappungen finden die Iterationen statt, die zum Abschluss der jeweiligen Tracks führen können, wenn ein entsprechendes Kriterium erreicht wird. Die Planung der Tracks beziehungsweise der Iterationen erfolgt dabei hauptsächlich durch die Dokumente des

Teamprojekts. Ansonsten unterstützen entsprechende Word und Excel Vorlagen auch die Aktivitäten innerhalb der beschrieben Phasen. So können beispielsweise entsprechende Spezifikationen erstellt, Testpläne und Änderungen formalisiert sowie Risiken und Verbesserungsvorschläge dokumentiert werden.

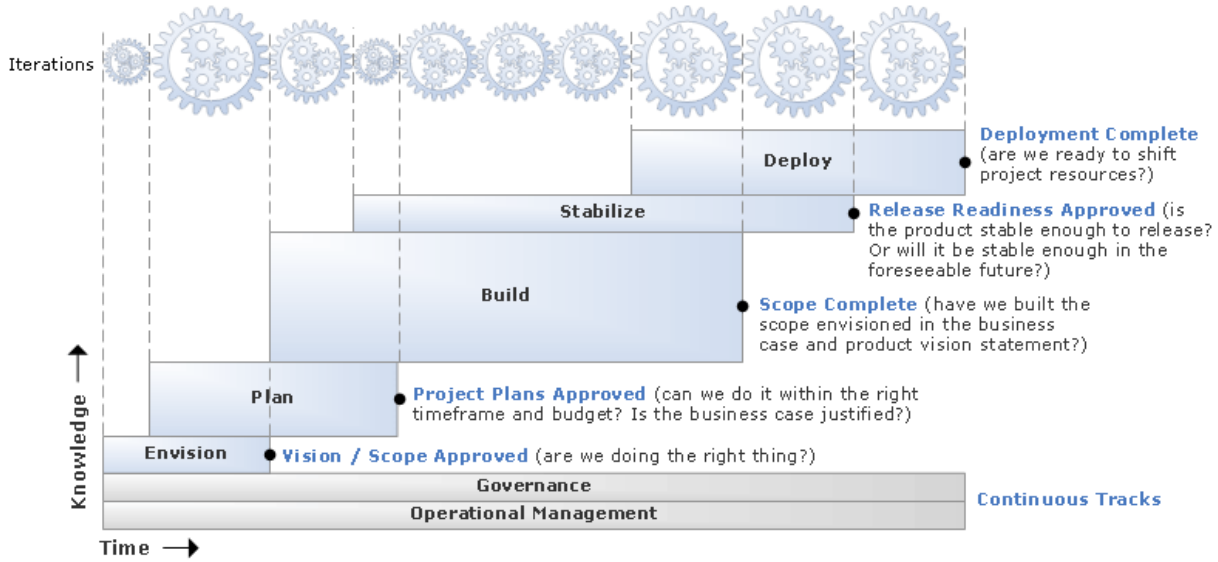


Abbildung 35 Tracks in MSF for CMMI Process Improvement

Natürlich ist eine solche umfangreiche Planung und Formalität bei kleinen oder agilen Projekten nicht sinnvoll. In Bezug auf die Beispielanwendung MyDice, die im Teamprojekt „CMMITest“ entwickelt werden kann, ist Bürokratie eher nicht erforderlich. Aus diesem Grund eignet sich MSF for CMMI eher für größere Projekte, bei denen eine gewisse Professionalität von vornerein vorausgesetzt wird. Das CMMI-Prozessmodell dient dabei nicht nur dem Zweck ein Projekt nach einem definierten Prozess abzuschließen, sondern adressiert auch, wie dieser Prozess optimiert und verbessert werden kann. Dabei wird kontinuierliche und schrittweise Verbesserung unterschieden. Eine genauere Übersicht und Definition kann übrigens aus dem entsprechenden Dokument auf der Website des Software Engineering Institute [Sof06] entnommen werden. In diesem Dokument wird unteranderem eine Einstufung nach Fähigkeit und Reife eines Prozesses im Rahmen von CMMI vorgestellt. Dabei existieren Level von 1 bis 5 beziehungsweise 0 bis 5. Level 0 bedeutet, dass ein Prozess nicht in der Lage ist, ein Projektziel zu erreichen. Level 1 bedeutet, dass ein Prozess in der Lage ist, ein Projekt durchzuführen. Die restlichen Levels geben an, welche Prinzipien und Vorgänge existieren oder genutzt werden um einen Prozess als fähig und reif zu bewerten. Im diesem Zusammenhang wird der Begriff der Prozessgebiete, sogenannten Process Areas als Disziplinen und Prinzipien eines Prozesses definiert. Abhängig davon welche dieser Process Areas in einem Prozess adressiert werden, wird ein Level zwischen 2 und 5 erreicht. Der Team Foundation Server dient diesbezüglich als Plattform für einen CMMI-Prozess mit stufenweiser Prozessverbesserung, wobei ein Reifegrad von Level 3 nahezu mühelos erreicht werden kann. Level 3 bedeutet, dass der Prozess unterstützt wird (Level 2) und dass er klar definiert ist. Ein Prozess ist laut dem Software Engineering Institute [Sof06] dann definiert, wenn er Zweck, Eingaben, Eingangskriterien, Aktivitäten, Rollen, Maßnahmen, Nachweismöglichkeiten, Ausgaben und Ausgangskriterien definiert und besitzt. Die Möglichkeiten der Benutzerverwaltung, des Reporting, des Projektmanagement und des Work Item Tracking stellen dafür die Grundlage dar. Höhere Level können erreicht werden, indem die Prozessverbesserung ermöglicht und definiert wird. Die Process Guidance des MSF for CMMI Templates zeigt, welche

Process Areas für welche Reifegrade relevant sind. Dabei deckt sich die Menge der Process Areas inhaltlich mit denen aus der Beschreibung des Software Engineering Institutes. Weiterhin gibt die Prozesshilfe im Team Foundation Server [Mic0715] zu jedem dieser Gebiete eine Hilfestellung und beschreibt, wie Arbeitsabläufe und Aktivitäten in Work Items und Dokumenten des Teamprojekts abgebildet werden können. Mithilfe dieser Beschreibungen können Process Areas verstanden und nach und nach eingeführt werden, sodass ein Prozess „reifen“ kann und somit ein höheres Level erreicht. Das Level eines CMMI-Prozesses ist als Klassifizierung der Qualität einer Projektdurchführung anerkannt. In der unteren Tabelle werden die Reifegrade zusammen mit den ihnen zugeordneten Prozessgebieten dargestellt. Level 1 setzt diese Gebiete nicht voraus.

Level 1	Level 2	Level 3	Level 4	Level 5
Performed/Initial	**Managed**	**Defined**	**Quantitatively Managed**	**Optimizing**
	Project Planning	Organizational Process Focus	Organizational Process Performance	Organizational Innovation and Deployment
	Project Monitoring and Control	Organizational Process Definition	Quantitative Project Management	Causal Analysis and Resolution
	Supplier Agreement Management	Organizational Training		
	Requirements Management	Integrated Project Management for IPPD		
	Configuration Management	Risk Management		
	Process and Product Quality Assurance	Integrated Teaming		
	Measurement and Analysis	Integrated Supplier Management		
		Requirements Development		
		Technical Solution		
		Product Integration		
		Verification		
		Validation		
		Organizational Environment for Integration		
		Decision Analysis and Resolution		

Tabelle 3 Reifegrade in MSF for CMMI Process Improvement

Für ein kleines Projekt, wie die Beispielanwendung MyDice ist es eigentlich nicht erforderlich einen gereiften Prozess zu nutzen. Trotzdem kann auch hier die Entwicklung nach CMMI-Level 3 stattfinden, da beispielsweise Requirement Management und Validation sogar schon in MSF for Agile verwendet wurden. In CMMITest müssten lediglich die restlichen Prozessgebiete irgendwie realisiert werden. Es liegt dabei aber nahe, dass beispielsweise Risikomanagement oder Entscheidungsanalyse und Entschließung eher für Projekte geeignet sind, die einen größeren Umfang haben und sich über

einen gewissen Zeitraum erstrecken. Kurze Projekte wie MyDice sind nach kurzer Zeit abgeschlossen und damit nicht mehr in der Lage den zugrundliegenden Prozess nach CMMI-Kriterien zu verbessern. Aufgrund des beschränkten Umfangs dieser Ausarbeitung, können die einzelnen Process Areas nicht näher betrachtet werden. Die CMMI Process Guidance stellt aber auch noch allgemeine Vorgänge vor, sogenannte Generic Practices (GP), die dazu dienen die Generic Goals zu erreichen. Jeder Reifegrad verfügt über ein solches Ziel, nämlich ihn zu erreichen. Damit sind diese GPs den Reifegraden, also den Levels zugeordnet, wobei lediglich Vorgänge zu Level 2 und 3 verfügbar sind, diese aber ausführlich beschrieben werden. CMMI kennt auch noch spezielle Vorgänge (Specific Practices), die eine Verbesserung abhängig von Process Areas beschreiben, allerdings in der Process Guidance für das MSF for CMMI Process Improvement Template nicht erwähnt werden. Auch hier können die allgemeinen Vorgänge nur aufgelistet und aus bereits erwähntem Grund nicht näher betrachtet werden. Es sei aber darauf hingewiesen, dass für jeden dieser Vorgänge eine umfangreiche Beschreibung existiert, die zeigt, wie eine Umsetzung im Team Foundation Server erfolgen kann. Die untere Tabelle zeigt, welche Vorgänge für welchen Reifegrad beziehungsweise welches Generic Goal relevant sind.

Level 2 (Generic Goal: Manage)	Level 3 (Generic Goal: Define)
GP 2.1 Establish an Organizational Policy - CO 1	GP 3.1 Establish a Defined Process - AB 1
GP 2.2 Plan the Process - AB 2	GP 3.2 Collect Improvement Information - DI 4
GP 2.3 Provide Resources - AB 3	
GP 2.4 Assign Responsibility - AB 4	
GP 2.5 Train People - AB 5	
GP 2.6 Manage Configurations - DI 1	
GP 2.7 Identify and Involve Relevant Stakeholders - DI 2	
GP 2.8 Monitor and Control the Process - DI 3	
GP 2.9 Objectively Evaluate Adherence - VE 1	
GP 2.10 Review Status with Higher Level Management - VE 2	

Tabelle 4 Generic Practises in MSF for CMMI Process Improvement

Um Aufgaben einer Aktivität im Rahmen einer Iteration einer Person im Team zuweisen zu können, werden nach wie vor Work Items verwendet. Die CMMI Process Improvement Vorlage bringt dafür neue Work Item Typen mit. Im Gegensatz zu MSF Agile existiert der Typ Requirement anstelle von Quality of Service Requirement. Ansonsten gibt es auch wieder Bug, Risk, Task sowie Change Request, Issue und Review. Dabei ist es interessant, dass die Initialzustände verändert wurden. So wird ein Work Item nicht mehr als aktiv angelegt, sondern muss zuvor genehmigt werden. Erst nach einer Genehmigung geht die Arbeitseinheit in den Zustand aktiv über und kann bearbeitet werden. Dadurch kann erreicht werden, dass nur bestimmte Aufgaben bearbeitet werden können, die von einer berechtigten Person zuvor freigegeben werden. Dadurch können Entwickler von einer Flut an Arbeiten geschützt, sowie sichergestellt werden, dass Reports auch wirklich nur Work Items auswerten, an denen wirklich gearbeitet wird. Im Rahmen von CMMITest und MyDice sind Risk, Change Request, Issue und Review eher nicht relevant, da es sich um ein kleines Projekt mit wenig Verbesserungspotential handelt. Die MyDice Entwicklung würde auch nach CMMI zunächst nur Work Items wie Bug, Task und Requirement verwenden, wie es schon in AgileTest der Fall gewesen ist.

5.2.1 Schlussfolgerung

Das Prozessmodell MSF for CMMI Process Improvement stellt die Grundlage für eine formalisierte Entwicklung dar und kann dank der Möglichkeiten des Team Foundation Servers auch ohne viel Aufwand effektiv genutzt werden. So helfen umfangreichere Zustandsübergänge den Projektmanagern sämtliche Work Items besser zu verwalten und so tatsächliche Arbeiten formal zu koordinieren. Diese Formalität kann auch durch das Sortiment an Dokumentvorlagen weiter erhöht werden. Auch wenn in diesem Abschnitt die Möglichkeiten der Prozessvorlage nur erwähnt werden konnten, bietet die CMMI Process Guidance [Mic0715] doch umfangreiche Informationen zu Arbeitsabläufen nach der durch das Software Engineering Institute [Sof06] veröffentlichten offiziellen CMMI-Spezifikation. Dabei kann ohne viel Aufwand ein Prozess mit dem Reifegrad 3 entstehen, der ein systematisches und umfangreich dokumentiertes Projekt und dessen Durchführung ermöglicht und effektiv unterstützt. Natürlich kann ein solcher Prozess noch stufenweise verbessert werden um selbst in kleinen Teams höhere Level zu erreichen, ohne den Überblick über das Projekt zu verlieren. Dabei ist CMMI aber auch mit wesentlich mehr Arbeit neben der reinen Entwicklungsarbeit verbunden. Aus diesem Grund eignet sich die Prozessvorlage nicht für kleine oder kurze Projekte, in denen die Entwicklung selbst schon sehr kurz ist. Größere Projekte, die über einen längeren Zeitraum entwickelt werden und mehrere Releases enthalten, können von den Möglichkeiten von CMMI Process Improvement Gebrauch machen und sich somit stufenweise verbessern um eine hohe Qualität sicherzustellen.

5.3 Light Weight Scrum – v2.1

An dieser Stelle wäre die Prozessvorlage Light Weight Scrum – v2.1 vorgestellt worden. Leider ist der Umfang dieser Studienarbeit begrenzt, sodass dieses Modell nicht mehr betrachtet werden kann. Im Gegensatz zu MSF for Agile und MSF for CMMI, ist Light Weight Scrum nicht im Team Foundation Server vorinstalliert, sondern muss nachträglich hinzugefügt werden. Wie das nachträgliche Installieren von Prozessvorlagen funktioniert, wird in Abschnitt 6 Anpassbarkeit beschrieben. Dabei handelt es sich bei dieser Scrum-Variante um ein Codeplex-Projekt und ist damit Open Source. Weitere Informationen können auf der Website des Projekts [scrum08] nachgelesen werden.

5.4 Fazit

Prozessvorlagen stellen die zentrale Komponente im Team Foundation Server dar. Sie werden benutzt um Teamprojekten eine Struktur zu geben und definieren dabei grundlegende Entwicklungsprozesse. Dabei kann ein Teamprojekt nur nach einer Prozessvorlage erstellt werden, was sich auch im späteren Projektverlauf nicht mehr ändern lässt. Alles, was zu einem Teamprojekt gehört, wird durch die entsprechende Vorlage erstellt. Dadurch können unterschiedliche Ansprüche abhängig vom Projektumfang befriedigt werden. Zum Beispiel kann ein kleines Projekt mit wesentlich weniger Work Items und Dokumenten zurechtkommen, als ein Projekt, das über einen langen Zeitraum existieren soll und sogar wächst. Obwohl die Prozessmodelle und deren Templates unterschiedlich sind, werden die Grundprinzipien des Team Foundation Servers immer genutzt. Damit unterscheiden sich Vorlagen lediglich in der Art der Work Items, der Reports, der Dokumentenvielfalt und der Process Guidance. Diese Prinzipien wurden ausgiebig vorgestellt und an dem Beispielprojekt MyDice demonstriert. Team Foundation Server 2008 bringt in der Standardinstallation zwei Modelle mit, nämlich MSF for Agile Software Development und MSF for CMMI Process Improvement, beides in Version 4.2. Ersteres bietet die Möglichkeit kleinere und kürzere Projekte nach einem agilen Prozess zu entwickeln, während letzteres für größere und langfristige Projekte besser geeignet ist. Ein größeres Projekt mit einer agilen Prozessvorlage zu

entwickeln könnte dazu führen, dass die Übersicht verloren geht, da weniger systematisiert und formalisiert wird. In der Regel ist eine hohe Bürokratie für große Projekte erforderlich und genau das wird durch CMMI ja ermöglicht. Beide Vorlagen wurden in dieser Ausarbeitung vorgestellt und Vor- und Nachteile entsprechend erörtert. Dadurch kann Team Foundation Server sofort genutzt werden um Projekte zu unterstützen, ohne vorher noch Vorlagen installieren zu müssen. Dies kann natürlich auch gemacht werden, ist aber nicht unbedingt erforderlich. Eine dieser externen Vorlagen ist das Light Weight Scrum Template, dass aus einem Codeplex-Projekt entstanden ist. Es existieren viele solcher externen Templates, die an Unternehmen oder Probleme angepasste Prozesse abbilden. Team Foundation Server kann so auf eine einfache Weise genutzt werden, um unterschiedlichsten Anforderungen gerecht zu werden. In Bezug auf die Beispielanwendung MyDice ist zu sagen, dass es sich dabei um ein sehr kleines Projekt gehandelt hat. Natürlich kann man auch ein kleines Projekt nach CMMI entwickelt, allerdings wäre das nicht effizient, da wesentlich mehr Prozessmanagement als Entwicklung betrieben werden müsste. In der Praxis gilt es einen Mittelweg zwischen Management eines Projekts und dem eigentlichen Entwicklungsaufwand zu finden. Die Wahl eines angemessenen Prozessmodells steht also noch vor dem Beginn des eigentlichen Projekts an. Das Konzept der Templates ist daher eines der wichtigsten Features des Team Foundation Servers, da es den zu verwendenden Entwicklungsprozess steuert und Teammitglieder entsprechend unterstützen kann.

6 Anpassbarkeit

Nachdem die Prozessmodelle vorgestellt wurden, die der Team Foundation Server enthält und versucht wurde mit diesen ein Projekt zu realisieren, könnte man meinen, dass die Standardunterstützung nicht ausreicht. Man könnte seinen eigenen Prozess nutzen wollen. Ein eigener Prozess könnte sich zum Beispiel in einer Organisation bereits bewährt haben, sodass eine Umstellung auf Agile, CMMI oder Scrum nicht vorgenommen werden soll. Auch diese Anforderung erfüllt der Team Foundation Server indem er Prozessvorlagen als Basis verwendet. Diese können bei Bedarf angepasst werden, oder als Basis für eine neue Vorlage verwendet werden. Wenn ein neues Teamprojekt erstellt werden soll, muss der zugrundeliegende Prozess ausgewählt werden. Um verfügbare Modelle verwalten zu können, gibt es den Process Template Manager. Dort können neue Vorlagen installiert oder bestehende gelöscht oder lokal gespeichert werden. Um eine neue Vorlage zu erstellen speichert man eine bestehende Vorlage lokal und ändert diese so ab, dass sie den eigenen Bedürfnissen entspricht. Anschließend installiert man die geänderte Vorlage wieder. Im Folgenden soll eine neue Prozessvorlage erstellt werden, dessen Prozess Bugs nach Exceptions behandelt. Das bedeutet, dass Bugs nur erstellt werden, wenn ein zugehöriger Fehlerzustand in der Software eingetreten ist. Als Basis wird MSF for Agile Software Development verwendet. Geändert werden soll nicht viel, es sollen lediglich die Möglichkeiten einer Anpassung demonstriert werden. Die erwähnte Abhängigkeit von Exceptions soll erreicht werden, indem bei der Erstellung der Typ der Ausnahme sowie der StackTrace angegeben werden muss. Das untere Bild zeigt den Prozessvorlagenmanager mit administrativen Rechten. Über den Button Download im Process Teamplate Manager kann das Template MSF for Agile gespeichert werden. Am Speicherort entsteht ein Verzeichnis mit mehreren Unterverzeichnissen, SQL Reports, Office Dokumenten und XML-Dateien. Natürlich kann das XML direkt manipuliert werden, allerdings bietet Visual Studio 2008 einen Process Template Editor.

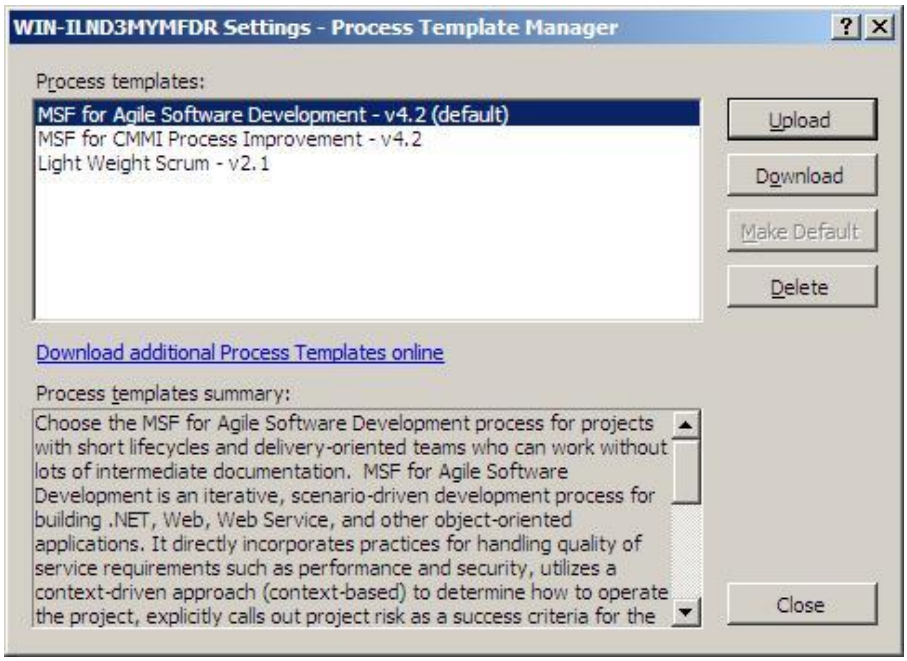


Abbildung 36 Process Template Manager zum Verwalten der Prozessvorlagen

Die XML Datei, die direkt im erstellten Verzeichnis liegt, kann in Visual Studio geöffnet werden, um den Editor zu starten. Dabei ist zu beachten, dass der Editor nur existiert, wenn die Power Tools für Team Foundation Server [Mic081] installiert wurden.

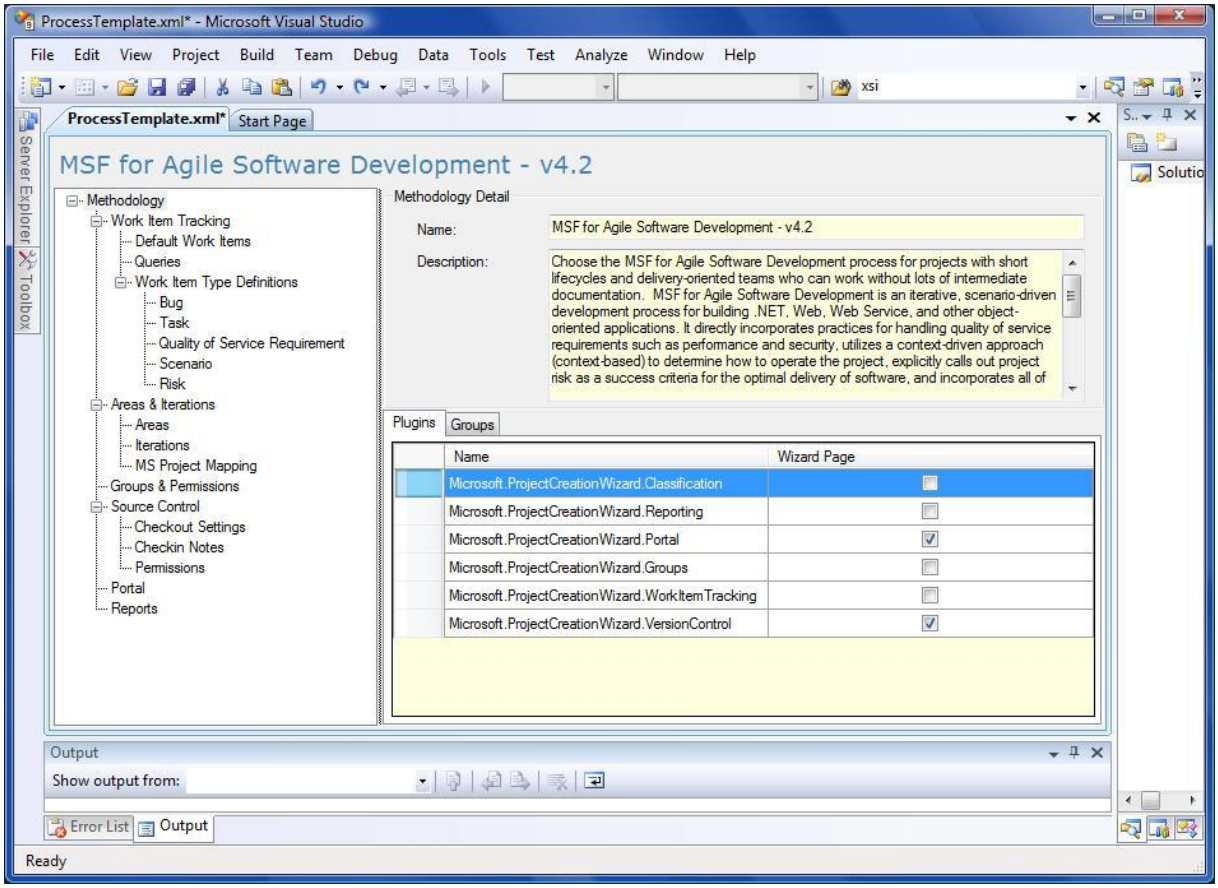


Abbildung 37 Process Template Editor in Visual Studio 2008

Im oberen Screenshot sieht man, dass eine Prozessvorlage aus sechs Komponenten besteht, die in einer Baumansicht dargestellt werden. In der Wurzel dieses Baums, der „Methodology“, lassen sich der Name der Vorlagen, die Beschreibung, sowie sogenannte Plugins festlegen. Plugins sind Schritte, die bei der Installation durchgeführt werden um das Teamprojekt einzurichten. Für die neue Vorlage ist dies aber nicht relevant. Als Knoten existieren Work Item Tracking, Areas & Iterations, Groups & Permissions, Source Control, Portal und Reports. Das Gebiet, das für die Anpassung relevant ist, ist natürlich Work Item Tracking. In dessen Unterknoten Work Item Type Definitions sind alle verfügbaren Typen von Arbeitseinheiten aufgelistet. Ein Doppelklick auf Bug öffnet die Datei Bug.wit in Visual Studio. Auch für diese Work Item Type-Dateien (*.wit), existiert ein Editor. An dieser Stelle sei angemerkt, dass aus einem bestehenden Teamprojekt die WIT-Dateien exportiert, geändert und wieder importiert werden können. Das untere Bild zeigt den Bug und seinen Workflow nach Unter-Unter-Unterabschnitt 5.1.2.4 Bug im Work Item Type Editor.

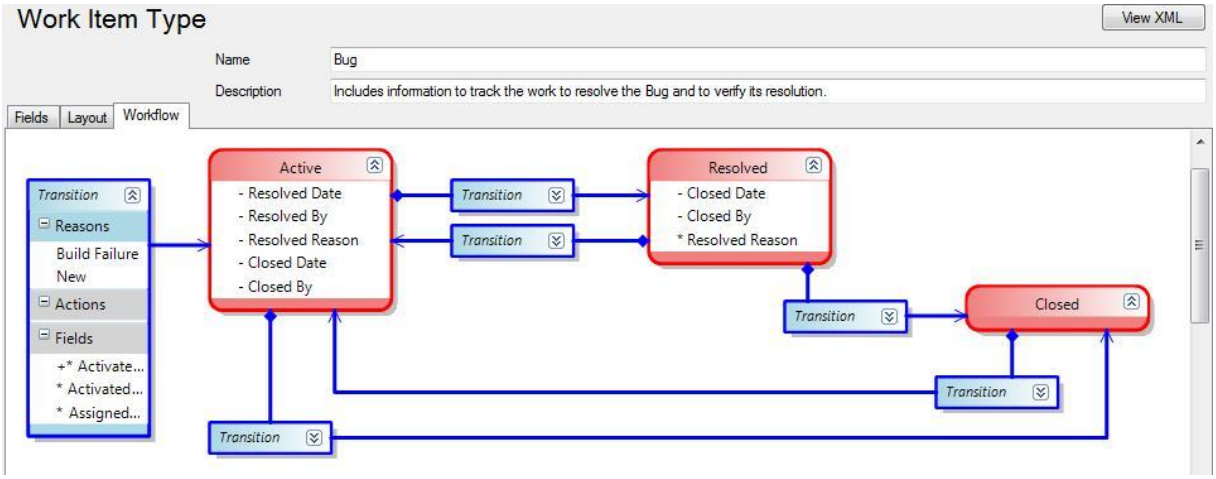


Abbildung 38 Bug mit Workflow im Work Item Type Editor

Neben Name und Beschreibung des Work Items, können Felder, Workflow sowie das Layout des Bugfensters angepasst werden. Zunächst werden also zwei neue Felder erstellt, die die Eingabe des Typs der Exception und dessen StackTrace ermöglichen.

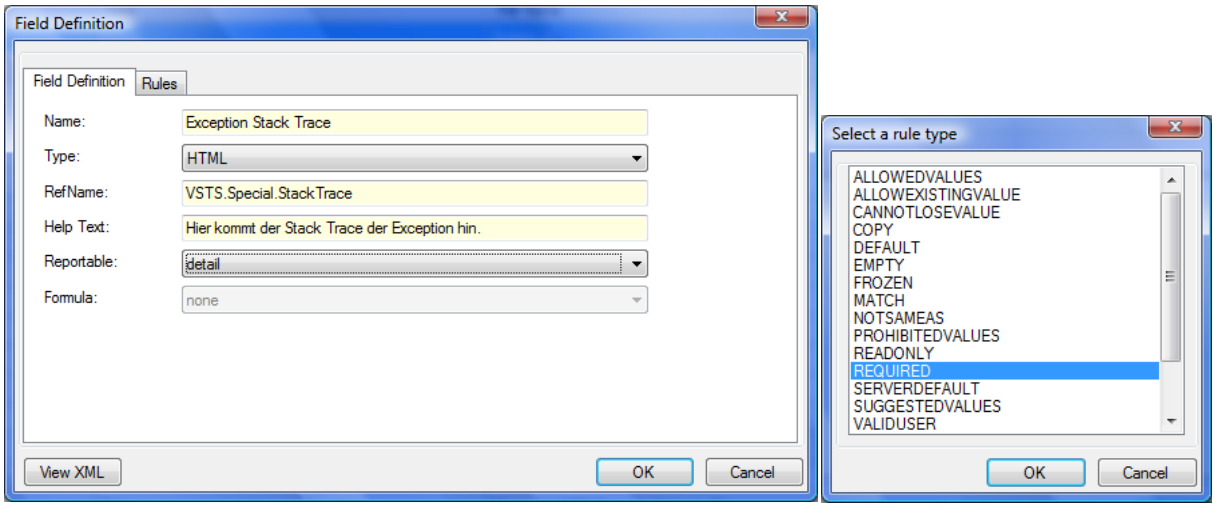


Abbildung 39 Hinzufügen des Feldes "Exception Stack Trace"

Neben den Labeln für die Oberfläche müssen auch Typen und eindeutige Referenznamen angegeben werden. Anschließend muss noch festgelegt werden, dass es sich um Pflichtfelder handelt, die in jedem Fall ausgefüllt werden müssen. Nachdem die Felder wie im oberen Bild dargestellt angelegt wurden, müssen sie auf der Oberfläche angezeigt werden. Dazu kann das Layout des Bugfensters angepasst werden. Wo die zwei neuen Felder tatsächlich erscheinen, ist an dieser Stelle nicht wichtig. Wichtig ist, dass hier die gesamte Oberfläche verändert werden kann. Dabei können auch eigene Controls implementiert und dargestellt werden. Über den eindeutigen Referenznamen können die Bestandteile des Fensters mit den Feldern verbunden werden. Um Änderungen auf visuelle Plausibilität überprüfen zu können, kann die veränderte Oberfläche direkt in einer Vorschauansicht angezeigt werden. Das untere Bild zeigt den Oberflächeneditor.

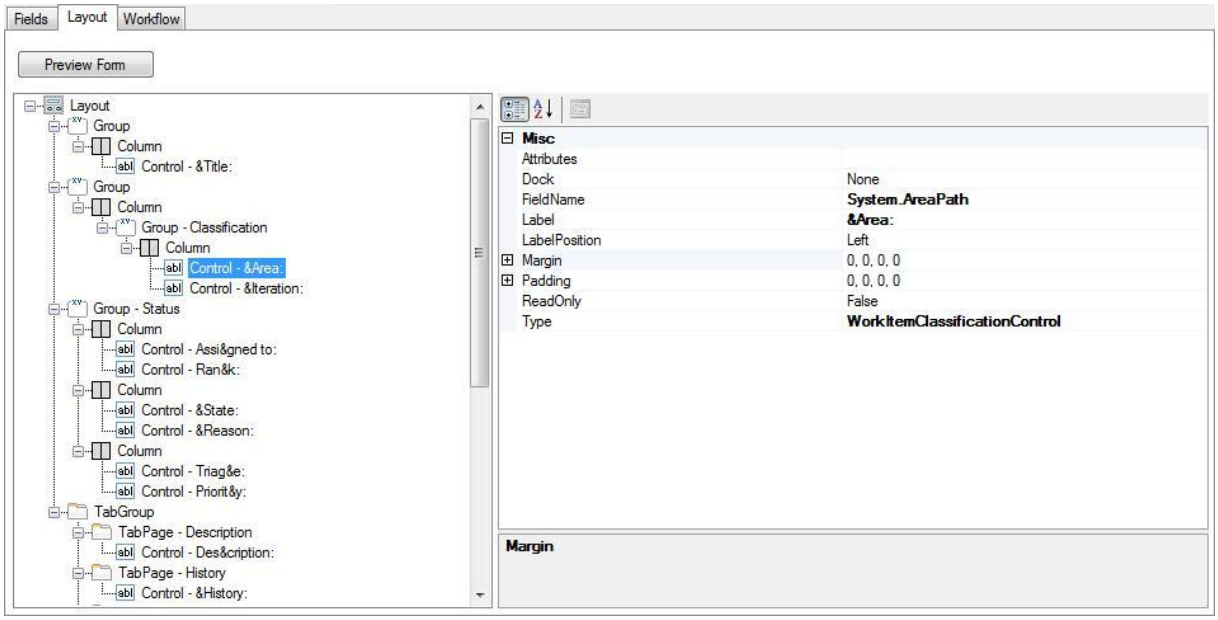


Abbildung 40 Oberflächeneditor im Work Item Type Editor

Nachdem die Felder mit Controls auf der Oberfläche verbunden wurden, kann die Datei Bug.wit sowie Änderungen an der Prozessvorlage gespeichert und mit Hilfe des Process Template Managers über den Button Upload installiert werden. Beschreibung und Name der Vorlage sollten angepasst werden, damit die Vorlagen unterschieden werden können. Anschließend steht der neue Prozess bei der Erstellung eines Teamprojekts zur Verfügung. Wenn ein Teamprojekt erstellt wird, wird die Struktur des ausgewählten Prozessmodells kopiert, sodass Änderungen an MSF for Agile zum Beispiel keine Auswirkungen auf bereits bestehende Projekte mit diesem Template haben.

6.1 Fazit

Wenn ein Unternehmen den Team Foundation Server einführen will, aber die komplette Entwicklung nicht auf eins der mitgelieferten Prozessmodelle umstellen will, kann es ein eigenes Template erstellen, dass den momentanen Anforderungen gerecht wird. Dazu wurde demonstriert, wie Änderungen auf Basis einer bestehenden Vorlage vorgenommen werden können. So kann die Firma eigene Dokumente und Office-Vorlagen spezifizieren, eigene SQL Server Reports definieren und das Work Item Tracking entsprechend an eigene Bedürfnisse anpassen. Eigene Abläufe und Abhängigkeiten bei der Arbeitsabwicklung können im Workflow der Work Item Typen abgebildet werden. Zusätzlich kann die SharePoint Seite als Projektportal angepasst sowie der verwendete Prozess in Form einer eigenen Process Guidance dokumentiert werden. All dies kann in einem

Process Template zusammengefasst für sämtliche Projekte verwendet werden. Dabei ist die Erstellung einer Vorlage aus einer vorhandenen sehr einfach vorzunehmen, wenn es sich nur um wenige Änderungen handelt. Sollten die Änderungen komplexer Natur sein, besteht die Gefahr schnell den Überblick zu verlieren. So sind beispielsweise Work Item Typen Basis für viele Referenzen. Sämtliche Felder werden in SQL Server Reports, Work Item Querys und an der Oberfläche verwendet, was die Manipulation bestehender Felder erschwert. Abhängigkeiten sind somit auch schwer zu erkennen und lassen sich mit den mitgelieferten Editoren nur bedingt verfolgen. Zusätzlich bedürfen Änderungen die weitere Systeme integrieren einer aufwendigen Entwicklung. Mithilfe von Plugins oder eigenen Controls auf Oberflächen von Work Items lassen sich externe Anwendungen einbinden, allerdings ist dessen Entwicklung nicht mit den Bordmitteln des Team Foundation Server zu erledigen. Da Process Templates in mehreren Dateien im XML-Format vorliegen, ist es prinzipiell machbar nicht über die Editoren von Visual Studio sondern direkt Änderungen vorzunehmen. Eine ausführliche Dokumentation des Formats konnte leider nicht gefunden werden. Dennoch ist Anpassung der Prozessmodelle in Team Foundation Server möglich, wobei keine direkten Einschränkungen festgestellt werden konnten. Sogar die Integration von Microsoft Office Project, die in Unter-Unterabschnitt 5.1.4 Management demonstriert wurde, lässt sich im Prozessvorlageneditor steuern.

7 Zusammenfassung

In dieser Ausarbeitung wurde der Team Foundation Server 2008 evaluiert. Zusammen mit Visual Studio 2008 Clients ist Team System die ideale Entwicklungsumgebung für Entwicklungsteams. Warum das so ist wurde in diesem Dokument demonstriert und erörtert. In einer fiktiven Projektumgebung wurde ein Beispielprojekt durchgeführt um Funktionalität und Prinzipien vorzustellen und zu zeigen, wie diese in einem echten Projekt zur Unterstützung von Teams genutzt werden können. Dabei wurde die Entwicklung der Anwendung MyDice durch Team Foundation Server unterstützt. Es wurde gezeigt, wie ein Teamprojekt geplant werden kann und wie Aufgaben mithilfe von Work Items Teammitgliedern zugewiesen werden können. Mit Visual Studio wurden sämtliche Entwicklungsaufgaben vorgenommen und anschließend in das zentrale Code-Verwaltungssystem eingecheckt. Durch Richtlinien konnte festgelegt werden, dass nur Code mit hoher Qualität akzeptiert wird. Solche Eincheckvorgänge haben Integration Builds und eine Reihe von definierten Testläufen ausgelöst. Dadurch konnten Fehler rechtzeitig erkannt und behoben werden. In Form von Reports standen die Resultate allen Projektmitgliedern über Visual Studio und dem zugehörigen Projektportal und Managern zusätzlich über Microsoft Office Excel und Project zur Verfügung.

Das innovative an Team Foundation Server ist dabei aber nicht, dass solche Vorgänge unterstützt werden, sondern wie sie unterstützt werden. Es lassen sich Vorgänge nach verschiedenen Prozessmodellen steuern, die sogar selbst definiert werden können. Dafür werden Prozessvorlagen verwendet, die einen speziellen Entwicklungsprozess vorschreiben können. Dadurch wird es Teammitgliedern erleichtert sich an einen Prozess zu halten und ihn sogar aktiv zu leben. Standardmäßig sind zwei Vorlagen installiert, nämlich MSF for Agile Software Development und MSF for CMMI Process Improvement in Version 4.2, es können aber beliebig viele nachträglich installiert werden. Für jedes Teamprojekt muss zu Beginn ein Modell ausgewählt werden, dass unter anderem angibt, welche Work Items erstellt werden können und wie der Arbeitsfluss ist, der mit ihnen in Verbindung steht. Zusätzlich stehen Dokumentenvorlagen zur Verfügung, die Planung, Spezifizierung

und Dokumentation ermöglichen. Über Work Item Tracking und Reporting kann zudem der aktuelle Status des Projekts ermittelt sowie Verifikation der Spezifizierung vorgenommen werden. Je nach Prozess existiert eine unterschiedliche Vielfalt und Formalität. So eignet sich die CMMI-Vorlage für die formale Projektabwicklung und einer stufenorientierten Prozessverbesserung, während die Agile-Vorlage ohne viel Dokumentation auskommt und eher auf Kundenorientiertheit und schnelle Realisierung setzt. Die Betrachtung hat gezeigt, dass CMMI eher für längere und größere Projekt verwendet werden sollte und agile Prozesse für kurzfristigere und kleinere Projekte. Wenn diese Vorlagen nicht dem entsprechen, wie etwas gemacht werden soll, können eigene Vorlagen nach eigenen Bedürfnissen erstellt werden. Außerdem existiert bereits eine Vielzahl an Process Templates im Internet, die sofort installiert und verwendet werden können. Team Foundation Server unterstützt nicht nur Softwareentwicklung sondern eignet sich dank der Prinzipien des Work Item Tracking, Reporting und Projektmanagement auch für viele andere Arten von Projekten, wie zum Beispiel reinen Dokumentierungsprojekten. Zu jedem Teamprojekt existiert ein SharePoint Portal, das als zentrale Ablage aller projektrelevanten Informationen genutzt werden kann. So können organisatorische Informationen beispielsweise im Kalender festgehalten werden und Diskussionsforen genutzt werden. Das wichtigste an diesem Projektprotal ist aber die Dokumentenverwaltung. Alles, was nicht im Team Foundation Server spezifiziert und geplant wird, kann über zusätzliche Dokumente vorgenommen werden, die natürlich auch mit Work Items verlinkt werden können. Damit spielen Dokumente in der Abwicklung eines Projekts zusammen mit den Work Items wohl die wichtigste Rolle. Team Foundation Server soll an dieser Stelle bestehende Projektmanagement-Ansätze mit beispielsweise Microsofts Office-Anwendungen nicht ersetzten, sondern lediglich eine übergeordnete Plattform für dieselben darstellen. Excel und Project sind häufig genutzt um Vorgänge und Projekte zu planen und erhalten daher sogar eine direkte Anbindung an die Daten des Team Foundation Servers. Office Project ist wie gezeigt in der Lage, Work Items in einer zeitlichen Reihenfolge sowie einer Hierarchie zu ordnen. Leider werden viele dieser Informationen nicht direkt in den Work Items sondern nur im Project-Dokument gespeichert. Zum Beispiel wäre es sehr praktisch Work Items direkt im Team Foundation Server verschachteln und Abhängigkeiten definieren zu können. Dadurch könnte ein Projekt besser strukturiert werden, was eine effektivere Top-Down Vorgehensweise ermöglichen würde. Außerdem lassen sich Work Items in den vorinstallierten Vorlagen nur einer Person zuweisen. Um diese Funktionalitäten nachzurüsten müsste der Nutzer die Prozessvorlagen selber ändern, indem er weitere Datenfelder für mehrere Personen, Vorgänger, Nachfolger und übergeordnetes Work Item mit dem Work Item Editor hinzufügt. Diese Felder müssen aber auch in den SQL Server Reports berücksichtigt werden, was schnell zu einer komplexen Änderung ausarten kann. Es wäre daher wünschenswert solche Anpassungen in der nächsten Version des Team Foundation Server nativ umzusetzen.

Die Berichte der Teamprojekte bieten die Möglichkeit viele Kenngrößen eines Projekts grafisch darzustellen und Managern so einen aktuellen Statusbericht zu liefern. Allerdings sind diese Reports nur bedingt flexibel und können von Projektmanagern nicht einfach angepasst und verändert werden. Office Excel bietet dafür aber die Möglichkeit auf die Datenquelle des Team Foundation Server als OLAP Cube zuzugreifen. Dadurch kann jedes Teammitglied seine eigenen Berichte in Form von Excels Pivot-Tabellen erstellen.

Abschließend ist nichts Negatives über den Team Foundation Server 2008 zu sagen. Lediglich wünschenswert wäre die Auslieferung mehrerer Prozessvorlagen, wie zum Beispiel ein Template für den Rational Unified Process (RUP) und Scrum. Zwar kann eine Vorlage für Scrum auf Codeplex gefunden werden, allerdings muss diese erst installiert werden. Leider konnte sie in dieser Studienarbeit nicht betrachtet werden. Für den RUP gibt es noch keine kostenfreie Vorlage.

8 Anhang

8.1 Literaturverzeichnis

fabcamara, mazocar, steven_borg, wschaub, CareBear. 2008. VSTS Scrum Process Template. *VSTS Scrum Process Template.* [Online] 26. Juni 2008. [Zitat vom: 10. Dezember 2008.] http://www.codeplex.com/VSTSScrum.

Microsoft. 2007. MSF for Agile Software Development. *MSF for Agile Software Development 4.2.* [Online] 2007. [Zitat vom: 1. Januar 2009.] Team Foundation Server.

—. 2007. MSF for CMMI Process Improvement. *MSF for CMMI Process Improvement 4.2.* [Online] 2007. [Zitat vom: 16. Februar 2009.] Team Foundation Server.

—. 2008. Team Foundation Server Power Tools. *Team Foundation Server Power Tools.* [Online] Oktober 2008. [Zitat vom: 24. Februar 2009.] http://msdn.microsoft.com/en-us/teamsystem/bb980963.aspx.

—. 2007. Team Foundation Server SDK. *Team Foundation Server SDK.* [Online] November 2007. [Zitat vom: 8. Dezember 2008.] http://msdn.microsoft.com/en-us/library/bb130146.aspx.

—. 2008. Visual Studio Team System 2008 – Produktinformationen. *Visual Studio Team System 2008 – Produktinformationen.* [Online] 2008. [Zitat vom: 9. Dezember 2008.] http://msdn.microsoft.com/de-de/vsts2008/products/bb933734.aspx.

Software Engineering Institute. 2006. CMMI for Development. *CMMI for Development, Version 1.2.* [Online] August 2006. [Zitat vom: 16. Februar 2009.] http://www.sei.cmu.edu/pub/documents/06.reports/pdf/06tr008.pdf.

Teamprise. 2008. Teamprise Plugin for Eclipse. *Teamprise Plugin for Eclipse.* [Online] 2008. [Zitat vom: 8. Dezember 2008.] http://www.teamprise.com/products/plugin/.

8.2 Abbildungsverzeichnis