HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

ScrumTable

TFS Scrum Meeting Unterstützung auf dem MS-Surface interaktiven Tisch

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Abstract

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

1 Abstract

Abteilung	Informatik
Namen der Studierenden	Michael Gfeller Silvan Gehrig Patrick Boos
Studienjahr	FS 2010
Titel der Studienarbeit	ScrumTable: TFS Scrum Meeting Unterstützung auf dem MS-Surface interaktiven Tisch
Examinatorin / Examinator	Prof. Dr. Markus Stolze
Themengebiet	Software
Projektpartner	HSR Institut für Software

Scrum als agiler Entwicklungsprozess gewinnt immer mehr an Bedeutung. Ein zentrales Element in Scrum bildet das Task-Board in Form eines A0-Posters. Auf diesem tragen die Teams ihre Tasks ein, welche häufig auch in einem Projektmanagement System erfasst sind. Die Folge daraus sind redundante Daten, und daher die Gefahr von Inkonsistenzen.

ScrumTable löst dieses Problem, indem das Task-Board auf dem interaktiven Microsoft Surface Table simuliert wird und die Informationen direkt im Projektmanagement System (z.B. Microsoft Team Foundation Server) gehalten werden. ScrumTable ermöglicht also das Durchführen von Scrum Meetings auf eine intuitive Art, welche dem physisches Erlebnis sehr nahe kommt.

Die Synchronisation mit der zentralen Datenquelle ist modular gestaltet um auch verschiedenste Datenquellen zu unterstützen.

Die Usability von ScrumTable wurde mittels User Centered Design nach Garrett und Goodwin optimiert.

Die Bachelorarbeit wird in Zusammenarbeit mit Herrn Professor Dr. Markus Stolze und mit dem HSR Institut für Software durchgeführt.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

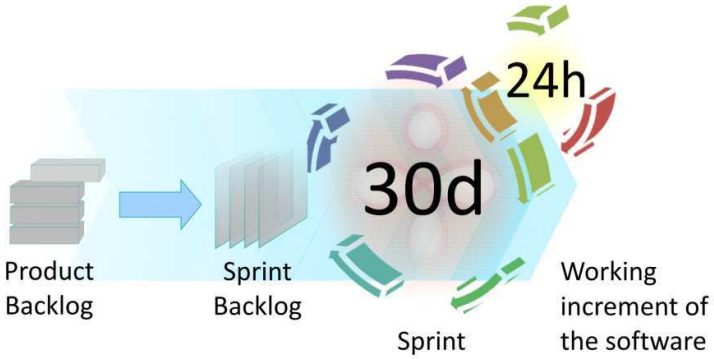
Management Summary

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

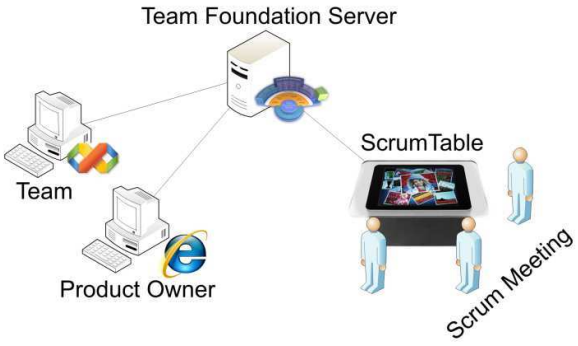
2 Ausgangslage

Scrum ist eine moderne Methode zur agilen Entwicklung von Software. Immer mehr Teams nutzen diese Methode. Ein zentrales Element von Scrum ist das Task-Board. Dies ist ein speziell aufgeteiltes A0-Poster, welches zur täglichen Planung und Fortschrittsüberprüfung des Projektes genutzt wird. Aufgrund der Popularität von Scrum unterstützen auch neuere Systeme für Software-Projektmanagement (z.B. Microsoft Team Foundation Server) die Bewältigung von Scrum Projekten. Da die Teams aber weiter auf ihren A0-Postern arbeiten, ergibt sich häufig eine Doppelspurigkeit. Informationen müssen von Hand zwischen Task-Board und Projektmanagement System übertragen werden. Dies ist aufwändig und kann zu Fehlern führen. ScrumTable löst dieses Problem, indem das Task-Board auf einem interaktiven Tisch simuliert wird und die Informationen direkt im Projektmanagement System gehalten werden.



3 Vorgehen/Technologien:

Schon im Vorfeld der Bachelorarbeit stand fest, dass der Microsoft Surface Tisch für diese Arbeit genutzt werden sollte. Daher mussten wir uns in die für uns neuen Design- und Entwicklungskonzepte für Multi-Touch Software einarbeiten. Beim Analysieren der Anforderungen war es uns wichtig, dass der Benutzer im Mittelpunkt steht: Die Einfachheit und das Erlebnis des Task-Boards sollten bei der Benutzung von ScrumTable nicht verloren gehen. Durch fundierte Interviews erhielten wir Einblick, wie Scrum in realen Projekten eingesetzt wird und was bei der Unterstützung der Scrum Prozesse zu beachten ist. Während der Entwicklung führten wir wiederholt Usability-Tests durch, um die Benutzbarkeit von ScrumTable zu gewährleisten und zu verbessern.



Eingesetzte Technologien

- C#, WPF
- Microsoft Surface
- Microsoft Team Foundation Server

4 Resultat

ScrumTable ermöglicht das Durchführen von Scrum Meetings auf eine intuitive Art, welche dem physischen Erlebnis sehr nahe kommt. Die Synchronisation mit der zentralen Datenquelle ist modular gestaltet um auch verschiedenste Datenquellen zu unterstützen.

Aktuell sind dies:

- Microsoft Team Foundation Server
- Lokale Datenbestände

5 Ausblick

Während der Arbeit sind viele Wünsche aufgekommen, welche im Umfang dieser Arbeit nicht betrachtet werden konnten. Darunter fallen: Kollaboration, Desktop Gadgets und Anzeigen vom Task-Board im Stand-By-Modus. Diese können in einer nächsten Phase implementiert werden.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Inhaltsverzeichnis

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

Inhaltsverzeichnis

5.3 Umsetzungskonzept [TBU]

5.4 Resultate, Bewertung und Ausblick [TBR]

6 Teil II - SW-Projektdokumentation

6.1 Vision [PDV]

Kann in Kapitel 5.1.2 (Teil I: Technischer Bericht – Einführung – Vision) gefunden werden.

6.2 Anforderungsspezifikation [PDAS]

6.3 User Centered Design [PDU]

6.4 Interviews [PDI]

6.5 Analyse [PDA]

6.6 Design [PDD]

6.7 Paper Prototype [PDPP]

6.8 Implementation (Entwicklung) und Test [PDIT]

6.9 Code Reviews [PDCR]

6.10 Usability Tests [PDUT]

6.11 Resultate und Weiterentwicklung [PDR]

6.12 Projektmanagement [PDP]

6.13 Softwaredokumentation [PDS]

7 Anhang

- Glossar
- Wikipedia Scrum PDF
- Vereinbarung über Urheber- und Nutzungsrechte
- Erklärung über eigenständige Arbeit

Aufgabenstellung Bachelorarbeit
Silvan Gehrig, Michael Gfeller, Patrick Boos

ScrumTable for Team Foundation Server

1. Betreuer

Betreuer HSR:

Prof. Dr. Markus Stolze, Institut für Software mstolze@hsr.ch

2. Ausgangslage

Der TFS 2010, welcher momentan als Release Candidate 1 vorliegt, bietet die Möglichkeit Work Items für Scrum zu erfassen und zu verwalten. Die Abläufe von Scrum sind innerhalb des TFS verteilt und teilweise schwer nachzuvollziehen.

Der MS Surface Table bietet eine multitouch-fähige Oberfläche. Mit dem dafür erhältlichen SDK kann man multitouch-fähige Programme entwickeln, welche sich auf diesem Surface Table nutzen lassen.

3. Ziele und Aufgabenstellung

Ziel dieses Projektes ist es eine Anwendung auf dem MS Surface zu entwickeln welche den Scrum Prozess und dessen Abläufe interaktiv unterstützt. Diese Anwendung soll vor allem für die verschiedenen Scrum Meetings verwendet werden können.

Zwischen dem TFS und dem MS Surface soll eine starke Kopplung bestehen. Änderungen sollen jeweils auch auf der anderen Plattform sichtbar werden.

4. Zur Durchführung

Die HSR stellt den Studenten den Surface Tisch zur Verfügung. Dieser Tisch ist allerdings nicht zur exklusiven Verfügung der Studenten, sondern muss bei Bedarf mit anderen Bachelor und Master-Studenten geteilt werden.

Mit den HSR-Betreuern finden in der Regel wöchentliche Besprechungen statt. Neben Herrn Stolze kann auf Herrn Kevin Gaunt (Assistent IFS) für inhaltliche Surface Fragen. Zusätzliche Besprechungen sind nach Bedarf zu veranlassen.

Alle Besprechungen sind von den Studenten mit einer Traktandenliste vorzubereiten und Beschlüsse in einem Protokoll zu dokumentieren, das den Betreuern per E-Mail zugestellt wird.

Für die Durchführung der Arbeit ist ein Projektplan zu erstellen. Dabei ist auf einen kontinuierlichen und sichtbaren Arbeitsfortschritt zu achten. Arbeitszeiten sind zu dokumentieren. Entscheidungen zum Externen Design sind so früh wie möglich mittels Usability Test zu validieren.

Reviews der Resultate der Usability Test sind einzuplanen und Anforderungsdokumentation und Architekturdokumentation sollten im Laufe des Projektes mittels Milestone abgeschlossen werden. Zu den abgegebenen Arbeitsresultaten wird ein vorläufiges Feedback abgegeben. Eine definitive Beurteilung erfolgt auf Grund der am Abgabetermin abgelieferten Dokumentation.

5. Dokumentation

Über diese Arbeit ist eine Dokumentation gemäss den Richtlinien der Abteilung Informatik zu verfassen. Die zu erstellenden Dokumente sind im Projektplan festzuhalten. Alle Dokumente sind nachzuführen, d.h. sie sollten den Stand der Arbeit bei der Abgabe in konsistenter Form dokumentieren. Die Dokumentation ist vollständig auf CD/DVD in 5 Exemplaren abzugeben. Daneben ist ein Papierausdruck S/W in zweifacher Ausführung zu erstellen. Für die Projektdokumentation wird eine (nicht bindende) Vorlage und Kriterien abgegeben. Zudem ist eine kurze Projektresultatdokumentation im Wiki von Prof. Stolze zu erstellen. Weiterhin erwünscht ist die Erstellung eines kurzen Videos.

6. Termine

Siehe auch Terminplan auf https://www.hsr.ch/Termine-Diplom-Bachelor-und.5142.0.html?&L=0

Montag, den 22. Februar 2010	Beginn der Bachelorarbeit, Ausgabe der Aufgabenstellung durch die Betreuer
Mai 2010	Fotoshooting
11.06.2010	Abgabe Kurzbeschreibung an das Abteilungssekretariat mit folgendem Formular gemäss https://www.hsr.ch/Allgemeine-Infos-Diplom-Bach.4418.0.html?&L=0
Freitag, den 18. 06 2010, 12:00	Abgabe des Berichtes an den Betreuer bis 12.00 Uhr. Fertigstellung des A0-Posters bis 12.00 Uhr. Abgabe der Posters im Studiengangsekretariat 6.113.
21.06. - 31.08.2010	Mündliche BA-Prüfung
25.06.2010	HSR-Forum, Vorträge und Präsentation der Bachelor- und Diplomarbeiten, 13 bis 18 Uhr

Allfällige weitere Termine sind am Sekretariat der Abteilung Informatik zu erfragen und sollten entsprechend in einem Sitzungsprotokoll dokumentiert werden.

7. Beurteilung

Eine erfolgreiche Bachelorarbeit zählt 12 ECTS-Punkte pro Studierenden. Für 1 ECTS Punkt ist eine Arbeitsleistung von ca. 25 bis 30 Stunden budgetiert. Siehe auch http://unterricht.hsr.ch/staticWeb/allModules/10939_M_BAI.html für die Modulbeschreibung der Bachelorarbeiten.

Für die Beurteilung sind die HSR-Betreuer verantwortlich unter Einbezug des Feedbacks des Auftraggebers.

Gesichtspunkt	Gewicht
1. Organisation, Durchführung	1/6
2. Berichte (Abstract, Mgmt Summary, techn. u. persönliche Berichte) sowie Gliederung, Darstellung und Sprache der gesamten Dokumentation.	1/6
3. Inhalt*)	3/6
4. Mündliche Prüfung zur Bachelorarbeit	1/6

*) Die Unterteilung und Gewichtung von 3. Inhalt wird im Laufe dieser Arbeit festgelegt.

Im Übrigen gelten die Bestimmungen der Abt. Informatik zur Durchführung von Studienarbeiten.

Rapperswil, den 22. September 2010

Prof. Dr. Markus Stolze
Institut für Software
Hochschule für Technik Rapperswil

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Einleitung

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument beschreibt die Vision und Ziele der Bachelorarbeit

1.2 Gültigkeitsbereich

Die Gültigkeit dieses Dokumentes erstreckt sich über die gesamte Projektdauer hinweg.

1.3 Übersicht

Dieses Dokument ist der Einstiegpunkt für die Bachelorarbeit.

2 Vision

Scrum ist eine moderne Methode zur agilen Entwicklung von Software. Immer mehr Teams nutzen diese Methode. Ein zentrales Element von Scrum ist das Task-Board. Dies ist ein speziell aufgeteiltes A0-Poster, welches zur täglichen Planung und Fortschrittsüberprüfung des Projektes genutzt wird. Aufgrund der Popularität von Scrum unterstützen auch neuere Systeme für Software-Projektmanagement (z.B. Microsoft Team Foundation Server) die Bewältigung von Scrum Projekten. Da die Teams aber weiter auf ihren A0-Postern arbeiten, ergibt sich häufig eine Doppelspurigkeit. Informationen müssen von Hand zwischen Task-Board und Projektmanagement System übertragen werden. Dies ist aufwändig und kann zu Fehlern führen.
Es ist also eine Software wünschenswert, welche dieses Problem löst, indem das Task-Board auf einem interaktiven Tisch simuliert wird und die Informationen direkt im Projektmanagement System gehalten werden.

3 Ziele von ScumTable

Das Ziel dieser Arbeit ist eine Softwareapplikation, welche ein einfaches & effzientes Scrum Projektmanagement mit dem Microsoft Surface Tisch ermöglicht:

Die wichtigsten Ziele dabei sind:

- Intuitive und natürliche Benutzung von Gesten (physisches Erlebnis)
  - Zuweisen von Elementen mit ziehen
  - Öffnen/Mutieren von Elementen
- Unterstützung der Scrum-Meetings
  - Projekt Planung
  - Sprint Planung
  - Daily Scrum
- Typische Scrum-Elemente
  - Durchführen von Scrum Poker
  - Anzeigen des Burn Down Chart des aktuellen Sprints
  - Task-Board
  - Story-Board
- Mutieren/Erfassen von User Stories
- Mutieren/Erfassen von Tasks
- Anbinden des Team Foundation Servers als Datenquelle

4 Rahmenbedingungen

Für die Bachelorarbeit sind folgende Technologien vorgeben:

- Team Foundation Server als Datenquelle
- Programmiersprachen: C#, .Net, WPF, Surface Toolkit 1.0
- Microsoft Surface Tisch

4.1 Vorgehen

Das Vorgehen soll dem Leser eine grobe Übersicht auf das Projektvorgehen vermitteln. Die diversen Punkte werden im Teil 2 SW-Projektdokumentation im Kapitel Projektmanagement detaillierter analysiert und beschrieben.

4.1.1 Organisation

Im Projekt sind zwei Parteien beteiligt: Bachelor-Team (Studenten), Lehrbeauftragte (Professor / Experte)

4.1.2 Planung & Entwicklung

Das Projekt wird in drei Major-Meilensteine und zwei Minor-Meilensteine gegliedert. Das Vorgehen im Projekt folgt dem agilen Prinzip.

4.1.3 Meilensteine

Die Meilensteine sind dazu da, das Projekt in granulare und überwachbare Einheiten zu gliedern.

1. Der erste Meilenstein dient dazu, die Anforderungen zu spezifizieren und den Paper-Prototype zu erarbeiten.
    a. Dieser Minor Meilenstein beinhaltet die Risikoabklärung mit dem Microsoft Surface Tisch.
2. Bis zum zweiten Meilenstein sind die Interviews zum Validieren der Spezifikation abgeschlossen. Die Erkenntnisse sind aufbereitet für eine Zwischenpräsentation.
    a. Dieser Minor Meilenstein beinhaltet die Entwicklung des ersten Prototypens für Usability Tests.
3. Der dritte Meilenstein beinhaltet alle geplanten Funktionen und die finale Dokumentation.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Stand der Technik

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quellenverzeichnis

Agilo for Scrum. (2010). Abgerufen am 10. März 2010 von http://www.agile42.com/cms/pages/agilo/

IceScrum. (2009). Abgerufen am 10. März 2010 von http://www.icescrum.org/index.php/product/

MSF for Agile Software Development, Version 5.0. (2010). Abgerufen am 11. Juni 2010 von http://msdn.microsoft.com/de-de/library/dd380647.aspx

Nori Scrum Meeting Table. (17. November 2009). Abgerufen am 10. März 2010 von http://www.youtube.com/watch?v=xsLszO-KdDg

Planning Poker (Cards). (2010). Abgerufen am 10. März 2010 von http://www.crisp.se/planningpoker

Planning Poker (Web). (2010). Abgerufen am 10. März 2010 von http://www.planningpoker.com/

Scrum Dashboard. (2010). Abgerufen am 8. Juni 2010 von http://scrumdashboard.codeplex.com/

Scrum for Team Systems. (2010). Abgerufen am 8. Juni 2010 von http://www.scrumforteamsystem.com

Scrum Planning Poker. (2010). Abgerufen am 10. März 2010 von http://scrumpoker.codeplex.com/

Scrum Sprint Monitor. (2010). Abgerufen am 8. Juni 2010 von http://scrumsprintmonitor.codeplex.com/

ScrumDesk. (16. März 2010). Abgerufen am 2010. März 2010 von http://www.scrumdesk.com

ScrumWorks. (2010). *ScrumWorks.* Abgerufen am 10. März 2010 von http://danube.com/scrumworks/basic/features

Surface Scrum Poker. (2009). Abgerufen am 10. März 2010 von http://www.blackspike.com/site/wpf/surface-scrum-poker

Urban Turtle. (2010). Abgerufen am 8. Juni 2010 von http://www.urbanturtle.com/

1 Einführung

1.1 Zweck

Das Dokument zeigt Produkte oder Arbeiten, welche im Bereich Scrum bereits erstellt wurden. Besonders beachtet werden dabei Produkte oder Arbeiten, welche versuchen Scrum interaktiver zu machen oder welche auf dem Microsoft Surface funktionstüchtig sind.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Wie unterstützt Team Foundation Server Scrum. Im zweiten Teil der Vergleich zwischen schon existenten Scrum Unterstützungstools.

2 Was ist Scrum

Scrum ist eine agile Vorgehensweise für die bearbeiten von Software Projekten. Die Beschreibung dieses Vorgehen kann unter folgenden Quellen nachgeschlagen werden:

Kurzzusammenfassung:

> http://de.wikipedia.org/wiki/Scrum
> Dieses Dokument ist im Anhang hinterlegt.

Deutsch:

> http://www.scrum.org/storage/scrumguides/Scrum%20Guide%20-%20DE.pdf#view=fit

Englisch:

> http://www.scrum.org/storage/scrumguides/Scrum%20Guide.pdf#view=fit

3 Scrum in Team Foundation Server

Microsoft hat mit der neusten Version von Team Foundation Server 2010 erkannt, dass Scrum schon längst kein Hype mehr ist, sondern bereits eine bereite produktive Anwendung findet. Microsoft unterstützt deshalb in der aktuellsten Version Scrum. Die Unterstützung wird durch das MSF for Agile Software Development, Version 5.0[1] Template geboten.

In diesem Kapitel wird aufgezeigt, wie der Team Foundation Server, mithilfe von MSF Agile v5.0, Scrum unterstützt.

3.1 Scrum Elemente

Die Elemente (User Story, Tasks, …) von Scrum sind im Team Foundation Server in Form vom Work Items gespeichert.

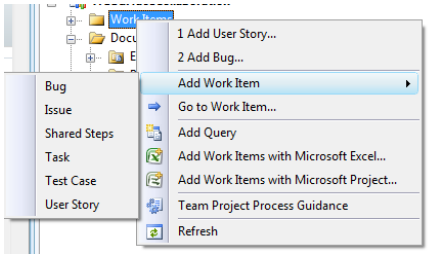


Abbildung 1 Team Foundation Server: Scrum Elemente

Das Arbeiten mit den Work Items erweist sich als leicht verständlich aber ist auch etwas umständlich. Als Beispiel, das Erstellen eines Tasks und das Verlinken zu einer User-Story. Dafür wird benötigt:

- 6 Maus-Klicks
- 3 modale Fenster

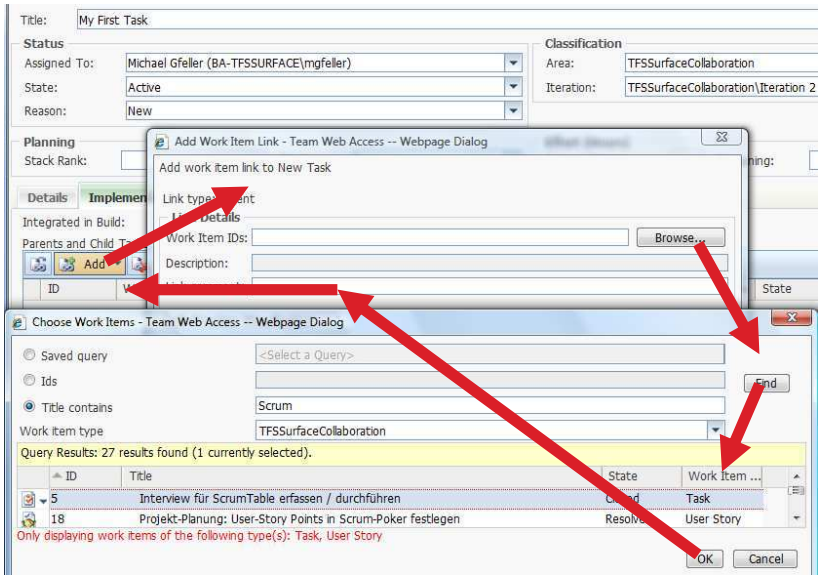


Abbildung 2 Task mit User-Story verknüpfen

[1] (MSF for Agile Software Development, Version 5.0, 2010)

3.2 Scrum-Tools von Team Foundation Server

Der Team Foundation Server stellt Hilfsmittel zur Verfügung für die Datenerfassung. Dies sind meist spezielle Excel-Dateien. In diesen werden Daten aufbereitet und gesammelt. Als Beispiel der Sprint Backlog, in diesem können Start/Ende und die Kapazität, inklusive Ferien und Abwesenheiten, eingetragen werden. Daraus werden dann die exakten Reporte erstellt.

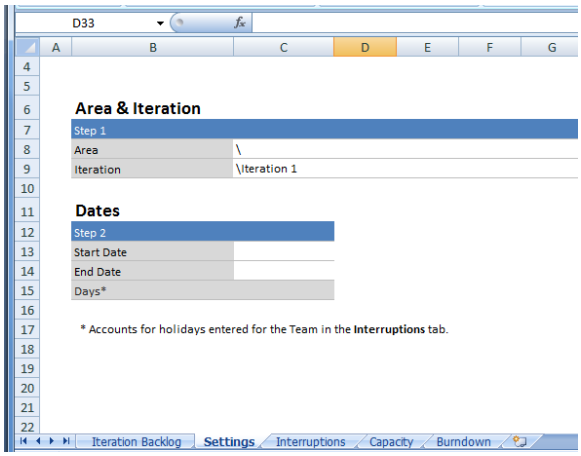


Abbildung 3 Excel: Sprint Backlog

Weitere Hilfsmittel bieten spezifische Queries. Diese dienen zur Übersicht über die eingegebenen Daten. Das Ergebnis der Abfrage kann in ein Excel-Sheet exportiert werden und dort einfach bearbeitet werden. Die Queries werden in einer speziellen, SQL ähnlichen und hierarchischen Abfragesprache (WIQL - Work Item Query Language) gespeichert.

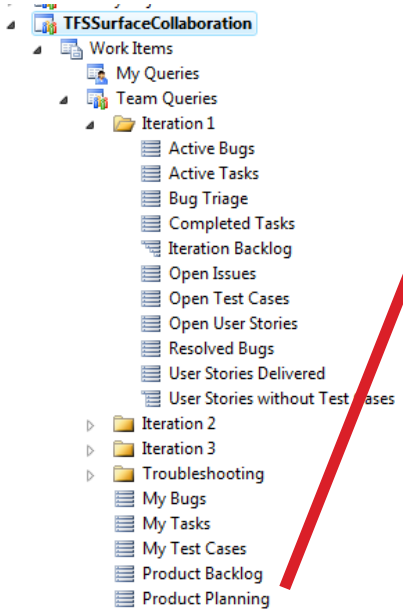


Abbildung 5 Scrum Queries

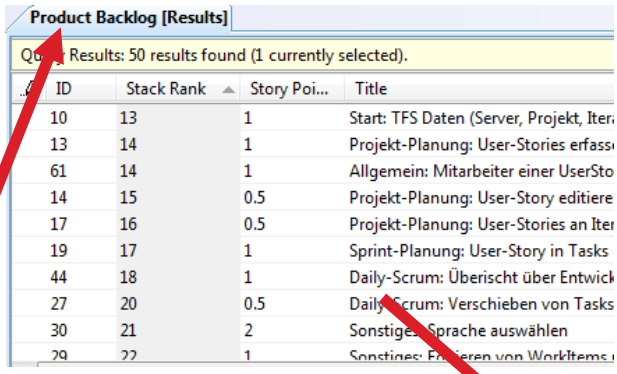


Abbildung 4 Scrum Querie: Produkt Backlog

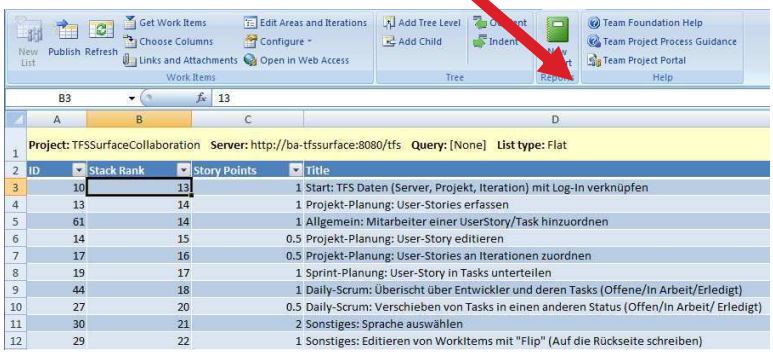


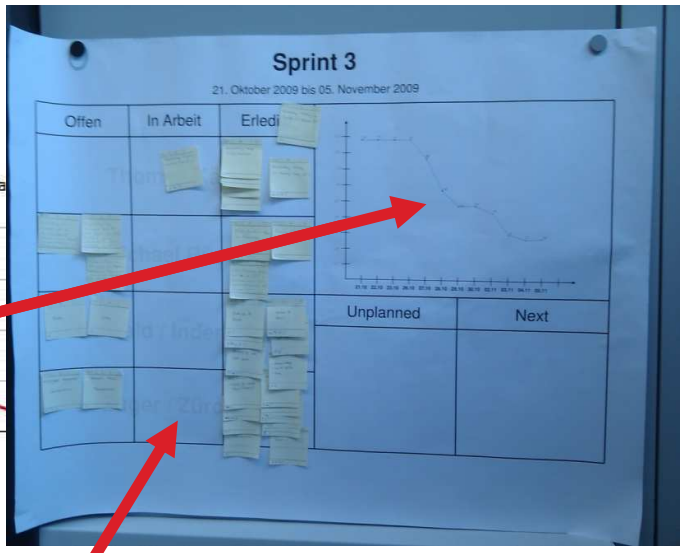
Abbildung 6 Scrum Querie im Excel

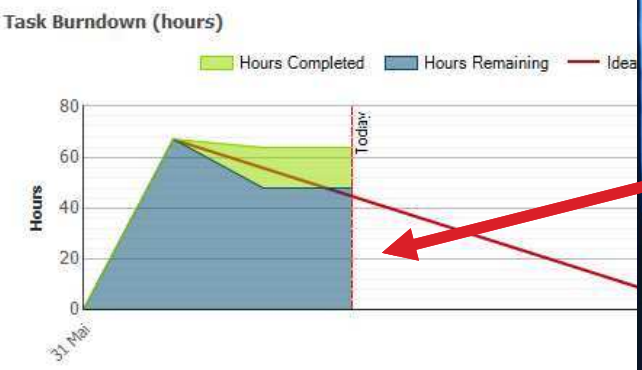
3.3 Scrum „Boards“

Ein Kern-Element von Scrum ist das physische Board, wie z.B. das Task-Board oder User Story-Board. Dieses dient dem Scrum Master als Übersicht und zeigt den Vorschritt des Teams auf. In Meetings wird auch sehr gerne auf diese Boards zurückgegriffen, da diese einem sehr schnell zeigen, wo das Projekt steht und wo Probleme vorhanden sind. Im Team Foundation Server sind diese Boards in diesem Sinn nicht vorhanden. Es wird jedoch das Burndown und Burnrate Chart angezeigt. Dies ist jedoch kein Ersatz für das ganze Board.

Als Beispiel dient der Vergleich vom Team Foundation Server Team-Portal zu einem realen Task-Board.

Wie klar ersichtlich ist, sieht beim TFS das Task-Board nicht so intuitiv aus wie bei physischen.





All Tasks of actual Iteration

New Work Item | Refresh | Column Options

Query Results: 33 results found (33 top level, 0 linked items, 1 currently selected).

ID	Title	Assigne...	Remainin...	Complet...
335	Unit-Tests SZ	Stefan Zü...	0	3
336	Iterationsplanung SZ	Stefan Zü...	0	3
339	Issue bezüglich Menu-Animation auf RobotController fixen	Stefan Zü...	0	2
340	Architektur RemoteController überarbeiten SZ	Stefan Zü...	0	4
343	Architektur des ViewDesign Blend Projekts überarbeiten (mehrere UserC...	Stefan Zü...	0	9
321	Persistierung/Laden der Alarmkonfiguration	Stefan Zü...	0	4.5
322	Implementierung behandeln von Alarme	Stefan Zü...	0	8
325	UI Implementation Alarm anzeigen auf RemoteController	Stefan Zü...	0	3
329	Meeting bei Zühlke SZ	Stefan Zü...	0	5

3.4 Fazit

Die Datenerfassung und das Analysieren der Daten wird vom Team Foundation Server durch unzählige Tools unterstützt. Bekannte Scrum-Ansichten fehlen jedoch im Team Foundation Server. Wesentliche Elemente von Scrum bilden gerade das Erlebnis, das Diskutieren der Mitarbeiter und das gemeinsame Verändern der Daten. Diese Funktionalitäten werden vom Team Foundation Server nicht geboten. Daraus folgt, dass ein Task-Board parallel geführt werden muss und die manuelle Synchronisation in Kauf genommen werden muss.

4 Bestehende Lösungsansätze

4.1 IceScrum

IceScrum[2] ist eine J2EE Applikation um nach Scrum zu arbeiten. Der Zugriff geschieht über den Web Browser. Sie stellt virtuelle Anschlagbretter mit Post-Its für Sprint Backlog, Product Backlog und weitere Items zur Verfügung. Es bietet alles an, was zu Scrum gehört.

Die Daten werden in einer Datenbank gespeichert. Es besteht die Möglichkeit Eclipse per Mylyn Web Connector an IceScrum anzubinden und zu synchronisieren.

IceScrum ist Open Source.

4.2 Agilo for Scrum

Agilo™ for Scrum[3] ist ein einfaches, web-basiertes und zielgerichtetes Werkzeug zur Unterstützung des Scrum Frameworks. Es ist flexibel an alle unternehmens- und teamspezifischen agilen Arbeitsweisen anpassbar.

Agilo for Scrum lässt sich mit Trac, SVN und Eclipse (Mylyn) integrieren.

Agilo for Scrum ist gratis und Open Source. Jedoch gibt es Agilo Pro, welches eine kostenpflichtige Erweiterung darstellt.

[2] (IceScrum, 2009)
[3] (Agilo for Scrum, 2010)

4.3 ScrumWorks basic und pro

ScrumWorks[4] besteht aus drei Komponenten. Einem Server, Desktop Client und Web Client. Ebenfalls wird eine API angeboten, über welche andere Applikationen auf die Daten von ScrumWorks zugreifen können.

orks basic Logo

Kollaboration wird nur von ScrumWorks pro angeboten. ScrumWorks pro integriert Bugzilla, JIRA und Tasktop als Eclipse Plugin.

orks pro Logo

4.4 Planning Poker (Cards)

Planning Poker[5], oder auch Scrum Poker genannt, ist eine Methode, wie man das Bestimmen von Story Points für eine User Story lebendiger gestaltet. Dabei erhält jeder Sitzungsteilnehmer einen Stapel von Karten. In einem solchen Stapel befinden sich jeweils folgende Karten:

ing Poker (Cards)

- Zahlenkarten (½, 1, 2, 3, 4, 8, 13, 20, 40, 100)
- 0 – Karte (Wird gespielt, wenn man denkt die Story bereits implementiert ist oder nur wenige Minuten benötigt)
- ? – Karte (Wird gespielt, wenn man keine Ahnung hat, was man Schätzen soll)
- Kaffee-Karte (Wird gespielt, wenn man der Meinung ist man brauche eine Pause)

Der Scrum Master legt dann eine zu bewertende User Story vor. Jeder der Teilnehmer legt die Karte ab, die seiner Schätzung entspricht. Wenn alle eine Karte abgelegt haben, werden diese aufgedeckt. Nun kann diskutiert werden oder einfach der Durchschnitt genommen werden. Planning Poker dient hauptsächlich dazu, dass die Diskussion angeregt wird und allfällige Fragen besprochen werden können.

Es gibt mehrere Firmen, die solche Decks an Karten verkaufen.

4.5 Planning Poker (Web)

Hierbei handelt es sich um eine Online-Version von Planning Poker[6], welche kostenlos von Mountain Goat Software angeboten wird.

.3: Planning Poker

Eine Person loggt sich auf der Seite als Moderator ein und lädt die anderen Teilnehmer ein. Der Moderator gibt dann eine User Story ein, welche bewertet werden soll. Alle Spieler geben ihre Wertung ein. Nachdem dies alle getan haben, werden die Werte angezeigt.

Ein Nachteil hierbei ist, dass jede beteiligte Person an einem Computer sitzen muss, welche sich nicht unbedingt im gleichen Raum befinden.

[4] (ScrumWorks, 2010)
[5] (Planning Poker (Cards), 2010)
[6] (Planning Poker (Web), 2010)

Dann müsste man eine Telefonkonferenz aufrechterhalten, damit man trotzdem miteinander diskutieren kann.

4.6 Surface Scrum Poker

Surface Scrum Poker[7] ist eine Microsoft Surface Applikation welche von Stuart Harris und Felix Corke entwickelt wurde. Die User Stories werden hier direkt aus Visual Studio geholt und auch dahin zurück gespeichert.

Verwendet werden Planning Poker Karten, auf welche ein Tag gedruckt ist. Diese Karte wird dann von jedem Teilnehmer auf den Microsoft Surface Tisch gelegt und die Applikation erkennt anschliessend, welche Karte gelegt wurde. Wenn alle Teilnehmer ihre Karte abgelegt haben, kann man auf „Reveal all“ klicken, wodurch die Werte der abgelegten Karten angezeigt wird und ein Durchschnitt berechnet wird.

Abbildung 14: Aussehen der Surface Scrum Poker Applikation

4.7 Scrum Planning Poker

Scrum Planning Poker[8] ist eine Desktop Applikation unter der Microsoft Public License, mit welcher Planning Poker durchgeführt werden kann. Dabei werden Items vom Microsoft Team Foundation Server gelesen und auch dahin zurück gespeichert.

[7] (Surface Scrum Poker, 2009)
[8] (Scrum Planning Poker, 2010)

Die Applikation besteht aus einer Project Owner- und Developer-Ansicht. Über die Project Owner-Ansicht wird die Verbindung zum Microsoft Team Foundation Server angegeben und das zu beurteilende Item ausgewählt. Über die Developer-Ansicht sieht man das ausgewählte Item und wählt die Karte aus.

4.8 Nori Scrum Meeting Table

Bei Nori[9] handelt es sich um eine Microsoft Surface Applikation, welche auf der ITS 2009 Conference vorgeführt wurde. Die Applikation wurde Henning Voss und Georg Schneider der Fachhochschule Trier entwickelt. Da es sich bei der Quelle nur um einen Youtube-Video handelt und darüber nicht viel Informationen verfügbar sind, kann nicht genau gesagt werden, was die Applikation alles kann.

Dem Video kann entnommen werden, dass die Applikation folgende Funktionalitäten umfasst:

- Anforderungen erfassen/bearbeiten
- Anforderungen mit Prioritäten versehen
- Teams erstellen
- Den Teammitgliedern Rollen zuweisen

Am Ende des Videos wird erwähnt, dass das Programm den kompletten Scrum Prozess, alle Meeting Typen von Scrum und Ticketing unterstützt. Ebenfalls prüft das Programm, ob das Team die Regeln des Entwicklungsprozesses einhalten.

Abbildung 15: Nori auf Microsoft Surface

[9] (Nori Scrum Meeting Table, 2009)

4.9 ScrumDesk

ScrumDesk[10] ist eine Desktop Applikation, welche den gesamten Scrum Prozess unterstützt. Die Applikation ist für bis zu 5 Benutzer kostenlos. Für mehrere Benutzer ist sie kostenpflichtig.

Über den ScrumDesk Hub lässt sich der Microsoft Team Foundation Server 2008 mit ScrumDesk abgleichen. In Entwicklung ist auch eine Verbindung zu Mantis.

Features, welche auf der WebSeite angegeben sind und nicht genauer erläutert werden:

- Projekte
- Teams
- Stories board, story cards
- Sprints und Releases planen
- Planning Poker®
- Retrospektive
- Fortschritt überwachen
- Kollaboration
- Integration
- Dokumente mit PDF support

4.10 Scrum for Team Systems

Scrum for Team Systems[11] ist ein Prozess Template für den Team Foundation Server 2010. Es stellt die Items für Scrum zur Verfügung. Es ist stärker auf Scrum ausgerichtet als das MSF for Agile v5.0 von Microsoft.

Nebst den Standard Scrum-Komponenten bietet Scrum for Team Systems:

- Kompatibilität mit Microsoft Test Manager
- Eine Vielzahl von Reports
- Komplett modellierter Lebenszyklus für Projektverfolgung und -planung
- Support für verschiedenste Team set ups

4.11 Scrum Dashboard

Scrum Dashboard[12] ist ein Web Front-end für Scrum for Team Systems auf dem Team Foundation Server 2010. Das Hauptziel ist es, das Whiteboard zu ersetzen und die tägliche Arbeit mit Scrum Artefakten in Team Foundation Server zu vereinfachen.

Funktionalität:

- Ajax-basiertes Web Interface mit Support für Internet Explorer und Firefox
- Alle Features fokussieren auf den Sprint für die tägliche Arbeit in dem Sprint
- Erstellen, hinzufügen und editieren von Product Backlogs

[10] (ScrumDesk, 2010)
[11] (Scrum for Team Systems, 2010)
[12] (Scrum Dashboard, 2010)

- Erstellen und editieren Sprint Backlog Items (Task, Impediment, Bug)
- Drag and Drop von Sprint Backlog Items zwischen Status
- Inline editieren von "Verbleibende Arbeit" während dem benutzen
- Farbliche Sprint Backlog Items um den Fortschritt und die ungeplante Arbeit visuell darzustellen
- Importiere Bugs von „Maintenance Tasks" (selbst von anderen TFS Projekten)
- Statistik und Sprint Burndown Chart einfach Sichtbar für das Team
- RSS Feed zur Anzeige von Veränderungen im Product Backlog
- Automatische Anzeige von neuen Projekten, Sprints und Team vom TFS ohne Konfiguration

4.12 Urban Turtle

Urban Turtle[13] ist ein intuitives Agile Project Management Tool (plug-in) für Visual Studio Team System. Es wurde entwickelt um die Software Entwicklungszyklen zu vereinfachen.

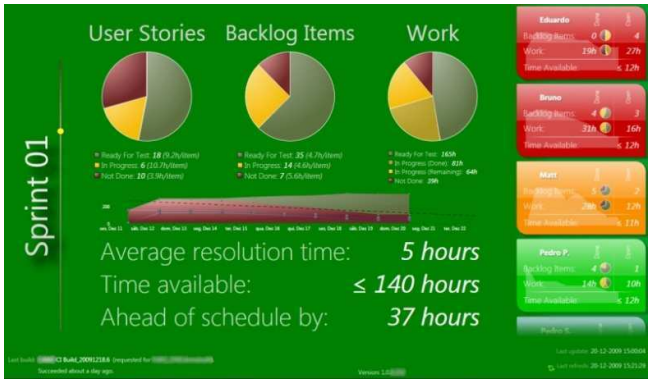
Urban Turtle arbeitet mit den Items des Team Foundation Server zusammen. Dabei wird das Standard MSF Agile 5.0 Template unterstützt. Urban Turtle lässt den Benutzer über den Browser folgende Tätigkeiten einfacher ausüben:

Turtle Logo

- Planen
  - Release und Sprint Backlog per Drag and Drop
  - Priorisieren der Arbeit
- Arbeiten
  - Gruppierte Ansicht von hierarchischen Items
  - Task Status durch Drag and Drop verändern
- Verfolgen
  - Live Iterations Statistiken
  - Ansicht von Team Fortschritt für Stand-Up Meetings

4.13 Scrum Sprint Monitor

Scrum Sprint Monitor[14] ist ein Bildschirmschoner, welcher zur Anwendung auf einem öffentlichen LCD zur stetigen Anzeige des Sprintstatus gedacht ist. Es kann jedoch auch auf dem eigenen Desktop laufen gelassen werden.



Als Datenquelle dient der Team Foundation Server mit dem Scrum for Team Systems Template (siehe oben).

[13] (Urban Turtle, 2010)
[14] (Scrum Sprint Monitor, 2010)

5 Gegenüberstellung

	IceScrum	Agilo	ScrumWorks basic	ScrumWorks pro	Planning Poker (Cards)	Planning Poker (Web)	Surface Scrum Poker	Scrum Planning Poker	Nori	ScrumDesk	Scrum for Team Systems	Urban Turtle	ScrumTable
Desktop Applikation	-	-	Ja	Ja	-	-	-	Ja	-	Ja	-	-	**-**
Per Browser bedienbar	Ja	Ja	Ja	Ja	-	Ja	-	-	-	-	Ja	Ja	**-**
Auf Microsoft Surface	-	-	-	-	-	-	Ja	-	Ja	-	-	-	**Ja**
Kompletten Scrum Prozess	Ja	Ja	Ja	Ja	-	-	-	-	Ja	Ja	Ja	Ja	**Ja**
Für Sitzungen Nutzbar	-	-	-	-	Ja	Nur bedingt	Ja	Nur bedingt	Ja	-	-	-	**Ja**
Scrum Poker	-	-	-	-	Ja	Ja	Ja	Ja	?	Ja	-	-	**Ja**
Eigene Datenbank	Ja	Ja	Ja	Ja	-	-	-	-	?	Ja	-	-	**-**
Collaboration	Mylyn	Trac, SVN, Eclipse (Mylyn)	-	Bugzilla, JIRA, Eclipse (via Tasktop)	-	-	Visual Studio	Team Foundation Server	?	Team Foundation Server	Team Foundation Server	Team Foundation Server	**Team Foundation Server**
Programmiersprache	J2EE	Python	?	?	-	?	C#	C#	C#	?	Template für TFS	ASP.NET	**C#**
Open Source	Ja	Ja	-	-		-	-	-	?	-	?	-	**-**
Kosten	-	-	-	> $289 US	9.73 CHF	-	?	-	?	< 5 Users kostenlos	-	$450+	**?**

Tabelle 1: Gegenüberstellung der Produkte

6 Fazit

Es gibt bereits viele brauchbare Applikationen, die den Scrum-Prozess unterstützen. Davon sind einige per Desktop Applikation, Web Browser oder beidem zugänglich. Wieder andere sind nur auf dem Microsoft Surface benutzbar. Eine Anbindung an den Microsoft Team Foundation Server gibt es jedoch nur bei Desktop Applikationen oder Plugins für Visual Studio Team System. Sitzungen werden nur teilweise unterstützt.

Die meisten Produkte, welche mit den Items von Team Foundation Server arbeiten, benutzen das Scrum for Team Systems Template. Ein paar wenige nutzen das MSF Agile 5.0 Template, welches standardmässig im Team Foundation Server 2010 enthalten ist. ScrumTable baut auf das MSF Agile 5.0 Template auf. Dadurch muss nicht zusätzlich ein Template installiert werden.

ScrumTable versucht durch die einfache Bedienung und durch die Nutzbarkeit in Meetings zu punkten. Die meisten anderen Applikationen sind Desktop oder Browser Applikationen, welche zwar den Umgang erleichtern und das Task-Board im Browser oder auf dem Desktop abbilden. Das Erlebnis des Task-Boards kann dadurch jedoch nicht ersetzt werden. Auch können diese Applikationen nur schlecht in Sitzungen benutzt werden, da man dazu jeweils einen Beamer braucht oder alle vor einem Bildschirm sitzen und das Verschieben der Items per Maus vornehmen müssen. Nori kommt dem Erlebnis des Task-Boards näher, da es auf dem Surface Tisch arbeitet und durch Touch bedient wird. Auch unterstützt Nori den ganzen Scrum Prozess. Wie schön und angenehm das Erlebnis dabei ist, konnte jedoch Anhand der wenigen Informationen nicht herausgefunden werden. Unklar ist ebenfalls, ob bei Nori eine Anbindung an den Microsoft Team Foundation Server oder andere Datenquellen bietet. Es wird jedoch vermutet, dass dies nicht der Fall ist. Dies ist ein wichtiger Punkt, durch den ScrumTable an mehr Bedeutung gewinnt.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Umsetzungskonzept

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einführung

1.1 Zweck

In diesem Dokument wird die konzeptionelle Umsetzung beschrieben.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Dieses Dokument beinhaltet das Umsetzungskonzept.

2 Einfache Bedienung und dennoch sicher

Ein Ziel von ScrumTable ist das Verwenden vom Team Foundation Server als Datenquelle

Aus Benutzerperspektive ist der konzeptionelle Ablauf der Applikation wie folgt umgesetzt:

1. Eine nicht registrierte Karte auf den Tisch legen.
2. Diese Karte wird aktiviert, indem die Funktion „Activate Card“ angewählt wird.
3. Den Team Foundation Server als Datenquelle auswählen.
4. Die Login Daten für den Team Foundation Server eingeben (Name, Password).
5. Die Eingabe mit einem Pin sichern.
6. Als Abschluss das Projekt und die Iteration auswählen.

Abbildung 1 Team Foundation Projekt laden

3 Herausforderung: Physisches Erlebnis

Das physische Board soll nachgebildet werden und das physische Erlebnis soll dabei nicht verloren gehen. Ebenfalls soll die Nutzung in Gruppen, das heisst gleichzeitig an demselben Tisch, möglich sein.

Damit das physische Erlebnis nicht verloren geht, wird sehr stark von Drag and Drop gebraucht gemacht. Dabei können alle Scrum-Elemente von Status zu Status verschoben, oder per „fallen lassen“ auf ein anderes Element, diesem zugewiesen werden.

Ebenfalls werden Tags eingesetzt, welche z.B. auf der Login Karte sowie auf den Scrum Poker Karten aufgedruckt werden, so dass diese auf den Tisch gelegt und erkannt werden können.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Resultate, Bewertung und Ausblick

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

Inhaltsverzeichnis

Tabellenverzeichnis

1 Einführung

1.1 Zweck

In diesem Dokument wird auf den Projektverlauf zurück geblickt. Ebenfalls wird ein Blick in die Zukunft geworfen und aufgelistet, welche Weiterentwicklungen möglich wären.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Als erstes wird aufgezeigt, welche Ziele erreicht wurden und wo es Probleme gab bzw. welche Funktionalitäten nicht umgesetzt werden konnten. Zusätzlich werden Weiterentwicklungsmöglichkeiten erläutert. Abschliessend folgen die persönlichen Berichte und der Dank.

2 Zielerreichung

Die Ziele von ScrumTable können in Teil I Technischer Bericht im Kapitel Ziele von ScrumTable nachgelesen werden.

Gesetzte Ziele	Erreicht
• Intuitive und natürliche Benutzung von Gesten (Physisches Erlebnis)	Ja
o Zuweisen von Elementen mit ziehen	Ja
o Öffnen/Mutieren von Elementen	Ja
• Unterstützung der Scrum-Meetings	Ja
o Projekt Planung	Ja
o Sprint Planung	Ja
o Daily Scrum	Ja
• Typische Scrum-Elemente	Ja
o Durchführen von Scrum Poker	Ja
o Anzeigen des Burn Down Chart des aktuellen Sprints	Ja
o Task-Board	Ja
o Story-Board	Ja
• Mutieren/Erfassen von User Stories	Ja
• Mutieren/Erfassen von Tasks	Ja
• Anbinden des Team Foundation Servers als Datenquelle	Ja

Tabelle 1 Erreichte Ziele von ScrumTable

2.1 Zielanalyse

2.1.1 Intuitive und natürliche Benutzung von Gesten (Physisches Erlebnis)

Da das Erlebnis des Task- oder Story-Boards eines der wichtigsten Elemente von Scrum ist, musste dies auch auf ScrumTable erhalten bleiben. Dabei war der limitierte Platz auf dem Microsoft Surface Tisch das grösste Problem. Aus diesem Grund kann nicht so viel Information auf ein Mal dargestellt werden wie auf einem A0 Task Board. Es wurde jedoch versucht, dies so gut wie möglich mit verschiedenen Ansichten nachzubilden. So bekommt man jeweils diejenigen Informationen zu sehen, welche man für die Aufgabe benötigt.
Nach Feedbacks von einigen Entwicklern, die gerade bezüglich dem Task-Board kritisch an ScrumTable herantraten, können wir sagen, dass das physisches Erlebnis und die intuitive und natürliche Benutzung erreicht werden konnte. Die Entwickler, welche ScrumTable ausprobiert haben, meinten, dass sie sich vorstellen könnten, mit ScrumTable das Task Board zu ersetzen.

Das Zuweisen von Elementen per Ziehen funktioniert sehr gut und ist an fast allen Stellen möglich. Das beinhaltet:

- Verschieben von Status zu Status
- Verschieben von Product Backlog zu Sprint (und auch wieder zurück)
- Priorisieren der User Stories durch Verschieben
- Zuweisen von Elementen
  - Entwickler an Task zuweisen (umgekehrt ebenfalls möglich)
  - Entwickler an User Story zuweisen (umgekehrt ebenfalls möglich)
  - Task einer anderen User Story zuweisen
  - Priorität einem Task zuweisen

2.1.2 Unterstützung der Scrum-Meetings

Es wurden alle Sitzungen in der Applikation eingebunden. Der Fokus liegt jedoch auf der Projekt Planung, Sprint Planung und dem Daily Scrum. Review und Retrospektive sind zwar vorhanden, jedoch ohne Mutationsmöglichkeiten. Lediglich eine kurze Anleitung zum Vorgehen der Retrospective sowie ein Burn Down- und Burn Rate Chart sind vorhanden.

2.1.3 Typische Scrum-Elemente

Vorhanden sind:

- Scrum Poker
- Burn Down Chart
- Burn Rate Chart
- Task-Board
- Story-Board

2.1.4 Mutieren / Erfassen von User Stories

Während der Projekt- und auch in der Sprint Planung lassen sich neue User Stories erstellen. Diese können überall, wo sie sichtbar sind, durch einen Doppelklick mit zwei Fingern zum Editieren geöffnet werden.

2.1.5 Mutieren / Erfassen von Tasks

Während der Sprint Planung können neue Tasks erstellt und einer User Story zugewiesen werden. Überall wo Tasks sichtbar sind, können diese durch einen Doppelklick mit zwei Fingern editiert werden.

2.1.6 Anbinden des Team Foundation Servers als Datenquelle

Die Anbindung des Team Foundation Servers besteht und wurde auch bereits im Rahmen des Projektes ScrumTable getestet. Sobald ScrumTable mit dem Team Foundation Server nutzbar war, wurde ScrumTable für die Sprint Planung sowie den Daily Scrum eingesetzt. Dabei tauchten keine grösseren Probleme auf. Einziger Nachteil ist die Ladezeit, welche ein paar Sekunden dauern kann.

2.2 Bestehende Unschönheiten / Probleme

2.2.1 Grösse und Positionierung

Die Oberfläche auf dem Microsoft Surface Tisch ist leider etwas klein und erreicht dadurch nicht die Grösse eines A0 Task Boards. Deswegen konnte das Task Board nicht 100%ig so nachgestellt werden, wie es sich die meisten Entwickler gewohnt sind. Jedoch wurde es auf die Grösse der Oberfläche des Surface angepasst und ist dadurch ein guter Ersatz. Jedoch kann ein Überblick, wie es ein A0 Task-Board bietet, nicht vollständig geboten werden.

Ebenfalls ist der Microsoft Surface Tisch vom Bau her etwas unpraktisch. So ist er etwas tief und man muss am Tisch sitzen. Gerade für Daily Scrum wo man eigentlich stehen möchte, ist dies etwas unpraktisch. Ebenfalls wäre es schön, wenn das Ganze eher als grosser LCD an die Wand hängen könnte, was aber mit dem Microsoft Surface nicht möglich ist.

2.2.2 Tastatur

Die Tastatur, welche im Microsoft Surface eingeblendet und per Touch bedienbar ist, gestaltet sich etwas schwierig in deren Benutzung. Für kurze Eingaben ist diese On-Screen Tastatur praktisch, aber für länger Sätze oder sogar Texte ist diese ungeeignet. Abhilfe schafft hierbei eine separate Bluetooth Tastatur. Somit können längere Eingaben darüber tätigt werden.

2.2.3 Memory Leaks

Obwohl während der Entwicklung auf die Abräumung nicht mehr verwendeter Resourcen geschaut wurde, sind immer noch merkbare paar Memory Leaks vorhanden. Leider wurde dies erst gegen Ende der Arbeit bemerkt und konnte deswegen nicht mehr komplett behoben werden.

3 Abgrenzung

Dieses Kapitel zeigt auf, was wir nicht selbst gemacht haben, sondern von dritter Seite übernommen haben. Dabei wurde darauf geachtet, dass wir gegen keine Lizenzen verstossen und die Erlaubnis haben, diese Quellen zu nutzen.

3.1 Icons

http://www.icon-king.com/projects/nuvola/ (GNU LGPL 2.1)
http://www.openclipart.org/ (Public Domain)

3.2 Weitere Quellen

Log4NET
http://logging.apache.org/log4net/

Microsoft Unity Application Block 1.1
http://msdn.microsoft.com/en-us/library/ff649080.aspx

Localization
http://blogs.microsoft.co.il/blogs/tomershamam/archive/2007/10/30/wpf-localization-on-the-fly-language-selection.aspx

PieChart
http://www.codeproject.com/KB/WPF/PieChartDataBinding.aspx

WrapPanelAnimated (StackPanelAnimated)
http://www.switchonthecode.com/tutorials/wpf-tutorial-creating-a-custom-panel-control

WPFAnimatedControl
http://www.codeproject.com/KB/WPF/wpfgif.aspx

ParseGif
http://www.codeproject.com/KB/WPF/wpfgif.aspx

ListBoxExtensions (ScrollIntoViewCentered)
ttp://blogs.msdn.com/delay/archive/2009/03/30/don-t-let-the-gotchas-getcha-adding-the-scrollintoviewcentered-method-to-wpf-s-listbox.aspx

Relay Command
Eine Tool-Klasse welche im MS-Projekt Modul (2009) in der HSR verwendet wurde.

Multithreading-Komponenten
Die Komponenten wurden vom SE2-Projekt von Michael Gfeller, Silvan Gehrig und Andreas Zollinger übernommen.

ViewModelChangeDispatcher
http://pradeepmahdevu.blogspot.com/2009/04/running-animations-in-mvvm.html

Prozess Model Elemente
http://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf

4 Ausblick: Weiterentwicklung

ScrumTable hat bereits das Potential, produktiv eingesetzt zu werden. Jedoch gibt es einige Weiterentwicklungen, die realisiert werden sollten, um dem Benutzer ein besseres Erlebnis zu bieten. Dazu zählen Verbesserungen und Erweiterungen an der Surface Applikation selbst, sowie weitere Tools, welche auf dem Desktop eingesetzt werden können. Diese Desktop-Tools helfen den Entwicklern, die für sie wichtigen Informationen gleich auf ihrem Desktop sehen zu können, ohne jeweils zum Surface Tisch gehen zu müssen.

Folgend werden die möglichen Weiterentwicklungen aufgelistet und in ihrer Wichtigkeit (1=unwichtig, 10=sehr wichtig) bewertet und sortiert.

Bereich	Beschreib	Wichtig-keit	Aufwand
Surface	Memory Leaks beheben	10	Mittel
Surface	Performanz verbessern	10	Gering
Desktop Gadget	Ein Windows Gadget zur Anzeige der Tasks des Entwicklers (eigene).	9	Gering
Desktop	Ansicht des Story-, Task-Boards oder Burn Down Chart auf dem Desktop. Als Gadget oder/und als eigene Applikation möglich.	9	Mittel
Surface	Anzeigen des Task-/Story-Boards im Stand-By Modus	8	Gering
Surface	Löschen von Items und Löschen von der Verknüpfung Task/Story zu Entwickler.	8	Mittel
Surface	Skizze per Zeichnen zu User Story oder Task hinzufügen.	8	Mittel
Surface	Skizze per Bluetooth (von Kamera) zu User Story oder Task hinzufügen.	8	Mittel
Surface	Automatisches Auslesen der IDs der Sprints (Iteration) aus dem SQL MDX Server.	8	Mittel
Surface	Area-/Iterationen-Hierarchie (Bsp.: Construction -> Sprint 3, Sprint 4, ...)	8	Hoch
Surface	Sprint Planning: Kapazitäten, Ferien und Abwesenheiten erfassen und editieren	7	Hoch
Surface	Verschlüsselte Verbindung (HTTP**S**: TLS / SSL) zum Team Foundation Server	7	Mittel
Surface	Weitere Zielsysteme neben TFS. Denkbar sind JIRA, Trac, usw.	7	Hoch
Surface	Undo und Redo	7	Sehr Hoch
Surface	Project Planning: Sprints hinzufügen	6	Gering
Surface	Kollaboration mehrerer Surface Tische, die ScrumTable verwenden. Dadurch wäre eine Sitzung am selben Projekt mit einem entfernten Team mit zusätzlicher Videokonferenz möglich.	6	Hoch
Surface	Speichern, welche Sprache bei einer LoginKarte ausgewählt wurde und beim hinlegen der Karte automatisch auf diese umschalten.	6	Gering
Surface	Änderungen vom Team Foundation Server als Ereignisse entgegennehmen und behandeln.	5	Mittel
Surface	Bugs sollen ebenfalls sichtbar und Entwicklern zugewiesen werden können. Diese tauchen dann auch im Daily Scrum auf.	5	Hoch
Surface	Erfassen von speziellen „Sprint-Learnings“-Notizen. Diese würden dann auch im Daily Scrum oder beim Sprint Planning angezeigt werden, damit sie immer präsent sind.	5	Mittel

Desktop	Ausdrucken von Scrum Poker Karten für ScrumTable.	5	Gering
Surface	Ein Installer soll das Mapping von speziellen TFS Schema auf ScrumTable Felder konfigurierbar machen.	4	Mittel
Surface	Möglichkeit, eine User Story in zwei kleinere aufzuteilen.	4	Gering
Surface	DailyScrum: Entfernen von nicht anwesenden Entwicklern aus der Liste (rausziehen).	3	Gering
Surface	Release Backlog	3	Hoch

Eine genauere Beschreibung der einzelnen Weiterentwicklungen kann im Teil II unter 6 – Resultate und Weiterentwicklungen gefunden werden.

5 Persönliche Berichte

5.1 Michael Gfeller

Mit Vorfreude auf eine interessante Arbeit mit dem Microsoft Surface Tisch startete das neue Semester. In der ersten Woche war das Meeting mit dem externen Partner der ETH Zürich. In diesem Meeting stellte sich aber heraus, dass die ETH ihre eigenen Ideen hatten und keinen Platz für externe Entscheide hatten. Da die Arbeit auch noch geschützt wäre, haben wir entschieden, die Zusammenarbeit zu beenden. Der Rest der Woche war geprägt mit dem Auffinden von weiteren Projekt-Ideen, welche den Surface Tisch optimal gebrauchen und ein gewisser Grad von einem Business-Wert hat. Deshalb war unsere Wunschidee von einem Spiel von Anfang an zum Scheitern verurteilt. Parallel dazu haben wir die Surface SDK studiert und die Funktionalität getestet, was uns bei der späteren Risiko-Planung half.

Der grosse Meilenstein war die Idee von einem Mitstudenten, welcher mit Scrum schon die Semesterarbeit durchgeführt hat. Die Grundidee war das Task-Board zu visualisieren. Da dies nicht ausreichend war für eine Bachelor Arbeit, mussten wir die Idee noch erweitern. Silvan Gehrig hatte dann die Idee den Team Foundation Server als Datenquelle zu benutzen und den Ablauf zu unterstützen. Der Vorteil ist: Wer sich ein TFS leisten kann, kann sich auch den teuren Microsoft Surface Tisch leisten, der 10000 Euro aufwärts kostet.

Die Idee kam bei allen gut an und darauf folgte die übliche Projekt Planung und Arbeitsteilung. Das spezielle für mich war bei dieser Arbeit die Fokussierung auf den Usability Teil. Der grösste Teil hat Silvan Gehrig davon beigetragen. Für mich interessant war das Interview mit zwei Scrum Masters bei einer Firma, die Scrum produktiv einsetzt. Die Erkenntnisse daraus haben mir stark geholfen, da meine Kenntnisse über Scrum nur auf theoretischem Wissen basierten. Scrum ist nicht gleich Scrum war die wichtigste Erkenntnis für uns und ich war „froh“ das wir uns an den Team Foundation Server halten können, welcher einen marginalen Standard vorgibt.

Am Anfang der Entwicklungsphase haben wir alle Features als User-Storys erfasst. Unsere Idee war es, Scrum von Beginn an einzusetzen. Die Erfassung von Task erwies sich aber als Umständlicher und da wir die Zeiterfassung in einem Excel File vorgenommen haben, wurde die Datenhaltung im TFS etwas vernachlässigt.

Mein persönliches Highlight war ein Studenten-Tag bei Namics. Alle Teilnehmer (6 an der Zahl) mussten ein Projekt vorstellen. Patrick Boos und ich haben unsere Bachelorarbeit gewählt. Das Feedback war super und sie könnten sich diese Applikation auch im produktiven Einsatz vorstellen; wenn der Tisch nur nicht so teuer wäre.

Abschliessend, die Arbeit war eine interessante Erfahrung für mich. Multi-Touch Geräte werden wahrscheinlich in der Zukunft interessanter, da immer mehr Touch-Bildschirme erscheinen. Es ist wohl nur eine Frage der Zeit, bis auch diese Multi-Touch unterstützen.

5.2 Silvan Gehrig

Die Bachelorarbeit versprach schon im Vorfeld Gutes: Für das ETH ValueLab sollte eine Multi-Touch Applikation um Remote-Collaboration erweitert werden. Diese Arbeit bot aus technischer Sicht sehr interessante Herausforderungen. So entschieden wir (Gruppe Gfeller, Boos, Gehrig) uns für eine Zustimmung zur Arbeit. Doch kurze Zeit später, zu Beginn der Bachelorarbeit, stellte sich unmissverständlich heraus: Das ETH ValueLab verfolgte andere Ziele als die Zusammenarbeit mit uns; dies trotz intensivster Vorabklärungen. Die Idee platzte in der Luft – unsere Bachelorarbeit stand unweigerlich dort, wo diese sich schon drei Monate zuvor befand: Nämlich ganz am Anfang und ohne klares Ziel. Unsere Erkenntnis: Die Arbeit sollte einfach auf dem Microsoft Surface Table durchgeführt werden.

Guter Rat war teuer – und die Zeit für die Bachelorarbeit lief... Wohin sollte unsere Bachelorarbeit gehen? Was sollten die Ziele der Bachelorarbeit sein? Sollten wir ein Spiel ohne fundierte Kenntnisse im Game-Design Bereich eine Anwendung entwickeln? Die Diskussionen und Verhandlungen mit unserem Bachelorarbeit-Dozenten Herr Stolze dauerten Stunden. Doch lag die Lösung so nah vor unseren Augen und wir sahen sie nicht! Der teure gute Rat, das Fundament, die innovative Idee, plötzlich war sie da – eingebracht von den Studienkollegen Andermatt und Trecco und verfeinert mit eigenen Zutaten. Die Arbeit trägt heute den Namen ScrumTable und soll die Scrum Meetings auf dem Microsoft Surface Table unterstützen.

Mit wiedergefundenem Elan und neuer Motivation stürzten wir uns auf die Arbeit. Parallel zur Bachelorarbeit fand auch das Modul User Interface 2 statt, welches von Prof. Dr. Markus Stolze an der HSR Rapperswil unterrichtet wird. In den ersten Lektionen lernten wir, wie nach User Centered Design vorzugehen ist. Da sich der Fokus unserer Bachelorarbeit ScrumTable sehr stark auf das User Interface ausrichtet, entschieden wir uns am User Centered Design Prozess zu orientieren. So durchwanderten wir die 5 Schichten nach Garrett und erarbeiteten die geforderten Artefakte nach Goodwin. Die Entwicklungsmethode von Scrum versuchten wir von Beginn weg nach Scrum zu gestalten. Leider aber liessen unsere Scrum Kenntnisse gerade bei Bachelorarbeitsbeginn zu wünschen übrig. So fanden die Daily Meetings erst im Laufe der Arbeit regelmässig statt.

Mich ermunterten vor allem die vielen positiven Feedbacks zu unserem Vorhaben. Wir stiessen bei einigen externen Firmen auf offene Ohren. So verfolgte beispielsweise ein Mitarbeiter der Zühlke AG das Ziel, bei den neuen Zühlke Scrum Poker-Karten die von uns verwendeten Markierungen (Tags) mit auf die Poker-Karten zu drucken. Leider konnte das Vorhaben schlussendlich nicht durchgeführt werden, da die neuen Karten kurz zuvor bereits gedruckt wurden. Trotzdem konnte unsere Arbeit von verschiedenen dozierenden Personen an der HSR Rapperswil bereits im frühen Stadium demonstriert werden. Mir zeigte dies das echte Potential von Scrum an der eigenen Bachelorarbeit auf: Scrum ermöglicht, dass man die Software bereits nach der Hälfte der Entwicklungszeit präsentieren kann.

Die Bachelorarbeit gestaltete sich für mich gerade im Bereich der Software Methodik Scrum und User Centered Design als sehr lehrreich. Das Vorgehen nach Garrett und Goodwin erweiterte meinen Horizont im Bereich Requirements Engineering und der Entwicklung von grafischen Benutzerschnittstellen. So denke ich, einiges vom Gelernten ins Berufsleben mitnehmen zu können.

5.3 Patrick Boos

Eine Arbeit auf dem Microsoft Surface Tisch mit Multi-touch hat uns drei sehr interessiert, wodurch wir mit Herr Stolze in Kontakt waren um darauf eine Arbeit schreiben zu können. Dabei bot sich ein Projekt in Zusammenarbeit mit der ETH an. Es klang anfangs auch sehr interessant, bis sich dann in der ersten Woche in der Kick-Off Sitzung mit der ETH herausstellte, dass eine Zusammenarbeit praktisch unmöglich ist. Dies aus dem Grunde, da die ETH ihren eigenen Zeitplan hatte und wir nur gerade am Anfang ein paar Abklärungsarbeiten ohne wirkliches Endprodukt hätten tätigen können. Dies lag natürlich nicht wirklich in unserem Interesse und wir waren etwas enttäuscht. So beschlossen wir gemeinsam mit Herrn Stolze ein neues Projekt zu finden, welches wir auf dem Surface Tisch umsetzen können.

Nach einer verlorenen Woche haben wir dann, dank der Idee eines Mitstudenten, endlich ein interessantes Projekt gefunden, für das sich alle begeistern konnten. Dadurch entstand die Idee ScrumTable und wurde dann noch weiter verfeinert.

Zu Beginn der Arbeit haben wir einige Abklärungen getätigt um zu sehen, wie wir das Projekt auf dem Surface Tisch umsetzen können und was uns dieser Tisch überhaupt bietet. Dabei sahen wir, dass wir auf Windows Vista arbeiten mussten. So benötigten wir doch etwas Zeit um unsere Computer und den Team Foundation Server einzurichten, damit wir eine Umgebung hatten, in der wir arbeiten konnten.

Anschliessend haben wir die Anforderungen aufgenommen. Dazu haben wir auch einige Interviews mit Personen gemacht, welche auf Scrum arbeiteten oder immer noch darauf arbeiten. Ganz interessant war dabei eine Firma zu besuchen, die Software entwickelt und dabei Scrum verwendet. Gemeinsam mit Michael Gfeller ging ich da hin und wir hatten ein interessantes Interview mit zwei Scrum Mastern. Wir erkannten sehr schnell, dass es schwierig werden wird, das Erlebnis des Task-Boards auf dem Tisch nachzubilden. Doch haben wir diese Herausforderung gerne angenommen. Ich habe hierbei erkannt, wie wichtig diese Interviews wirklich sind. Denn daraus konnten wir sehr viele Informationen für unser Projekt sammeln. Gerne hätten wir weitere Interviews durchgeführt, aber hatten leider nicht die Zeit dafür.

Wir haben zwar schon während den Interviews mit dem Entwickeln von ScrumTable begonnen, konnten aber einiges, was wir schon begonnen hatten, bestätigen und anderes verbessern oder verändern. So reifte ScrumTable heran und wurde um Funktionen erweitert. Die Applikation konnte sich dann auch schon bald sehen lassen. So kamen dank der Werbung von Herr Stolze immer mehr Leute vorbei um ScrumTable einmal live zu erleben. Dabei kam auch ein Entwickler von Zühlke vorbei, der anfangs etwas skeptisch war, da ihm das Task-Board sehr wichtig ist und es gross sein sollte. Wir waren natürlich alle erfreut, als er uns mitteilte, dass er sich vorstellen könnte, mit ScrumTable zu arbeiten und dies eine gelungene Umsetzung sei.

Ebenfalls durften Michael Gfeller und ich gemeinsam an dem Students Day von Namics in St.Gallen unser Projekt für 20 Minuten vorstellen. Das Feedback war sehr positiv und auch der Youtube Video, den wir ins Internet stellten kriegte bereits viele Hits.

Wir wollten von Anfang an gemeinsam nach Scrum vorgehen und haben auch schön die User Stories auf dem Team Foundation Server erfasst. Wir wollten jedoch kein Task-Board aufhängen, da wir

dann das Problem der Synchronisation gehabt hätten. Und auf dem TFS bemerkten wir, dass es mühsam ist, Tasks mit User Stories zu verlinken. So sahen wir gerade den Nutzen von ScrumTable. Als ScrumTable dann mit dem TFS nutzbar war, konnten wir dann auch wirklich nach Scrum arbeiten und sahen, wie hilfreich ScrumTable dabei ist. Ebenfalls konnten wir so noch ein paar weitere Probleme entdecken und beheben.

Ich bin sehr froh, dass wir ScrumTable als Bachelorarbeit machen konnten. Ich habe dabei sehr viel gelernt und bin sicher vieles davon später im Berufsleben anwenden zu können.

6 Dank

Wir möchten uns besonders bedanken:

- bei Metromec für die Interviews. Diese waren für uns eine sehr grosse Hilfe fürs Verständnis von Scrum. Auch konnten wir durch die Erkenntnisse aus den Interviews viele positive Verbesserungen in ScrumTable einfliessen lassen.
- bei Meinrad Andermatt und Mischa Trecco für die grundlegende Idee von ScrumTable. Mit ihrer innovativen Eingebung haben sie den Grundstein für eine solide und erfolgreiche Bachelorarbeit gelegt.
- bei Andreas Zollinger für die Mitarbeit an ScrumTable im Zusammenhang mit dem User Interface 2 Projekt. Dank seiner Hilfe konnten wir das User Centered Design effizienter durchführen.
- bei Daniel Tobler von der Firma Zühlke für seine Inputs, sein Feedback und sein Interesse an ScrumTable.
- bei Markus Flückiger von der Firma Zühlke für seine Inputs und sein Bemühen, dass wir an einem Scrum Meeting bei Zühlke teilhaben können.
- bei Michael Klenk vom Institut für Software (IFS) der HSR Rapperswil fürs Interview, den Scrum Artefakten sowie sein Feedback zur Usability. Auf seinen hilfreichen Feedbacks bauten wir die ScrumTable Applikation auf. Auch konnten wir dank seinen Inputs die Usability massgebend steigern.
- bei Kevin Gaunt vom Institut für Software (IFS) der HSR Rapperswil fürs Code Review sowie die nützlichen Usability-Ergänzungen.
- bei Prof. Dr. Markus Stolze vom Institut für Software (IFS) der HSR Rapperswil für seine Unterstützung, welche massgebend fürs Gelingen der Bachelorarbeit war. Vor allem machte sich Herr Stolze für die Arbeit im Bereich Publicity und User Centered Design stark. Hierfür möchten wir uns speziell bedanken. Herr Stolze ermöglichte die Durchführung eines derartigen Projekts am Institut für Software (IFS) der HSR Rapperswil.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Anforderungs-spezifikationen

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quellenverzeichnis

Cockburn, A. (5. März 2010). *Alistair Cockburn*. Abgerufen am 5. März 2010 von Alistair Cockburn: http://alistair.cockburn.us/Structuring+use+cases+with+goals

log4net. (10. März 2010). Abgerufen am 10. März 2010 von http://logging.apache.org/log4net/

Microsoft Application Block. (12. März 2010). *Microsoft Application Block*. Abgerufen am 12. März 2010 von Microsoft Application Block: http://msdn.microsoft.com/en-us/library/cc468366.aspx

Microsoft Team Foundation Server. (9. März 2010). *Team Foundation Server 2010.* Abgerufen am 9. März 2010 von Team Foundation Server 2010: http://msdn.microsoft.com/en-us/vstudio/dd582936.aspx

MSDN User Experience Guidelines for Surface. (5. März 2010). *User Experience Guidelines for Surface SDK Controls*. Abgerufen am 5. März 2010 von User Experience Guidelines for Surface SDK Controls: http://msdn.microsoft.com/en-us/library/ee804943.aspx

Persona Service Verwaltungs AG & Co. KG. (17. März 2010). *Der Mensch im Mittelpunkt*. Abgerufen am 17. März 2010 von Der Mensch im Mittelpunkt: http://www.persona.de/typo3temp/GB/404bc155b9.jpg

Scrum Alliance. (5. März 2010). *Scrum Alliance*. Abgerufen am 5. März 2010 von Scrum Alliance: http://www.scrumalliance.org/pages/scrum_student_resources

Shamam, T. (6. Februar 2009). *WPF Localization - On-the-fly Language Selection.* Abgerufen am 11. März 2010 von http://blogs.microsoft.co.il/blogs/tomershamam/archive/2007/10/30/wpf-localization-on-the-fly-language-selection.aspx

Stolze, M. (17. März 2010). *UInt2 - FS 2010*. Abgerufen am 17. März 2010 von UInt2 - FS 2010: \\vf3\skripte\Informatik\Fachbereich\User_Interfaces_2\UInt2\Vorlesungsfolien\

www.wikipedia.ch IEC_9126. (5. März 2010). Abgerufen am 5. März 2010 von wikipedia: http://de.wikipedia.org/wiki/IEC_9126

www.wikipedia.ch Two-Factor Authentication. (5. März 2010). Abgerufen am 5. März 2010 von www.wikipedia.ch Two-Factor Authentication: http://en.wikipedia.org/wiki/Two-factor_authentication

1 Einführung

Die Anforderungsanalyse wird zu Beginn des Entwicklungsprozesses durchgeführt. Sie soll Auskunft über die Anforderungen seitens der Benutzer und seitens der Funktionalitäten geben.

1.1 Zweck

Ziel dieses Dokument ist, die Anforderungen auf SEA-LEVEL[1] (falls nicht explizit anders angegeben) zu analysieren und die Ergebnisse aufzeigen. Dieses Dokument richtet sich an eingeschulte Requirements Engineers, Requirements Engineering Begriffe werden nicht separat erläutert.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Die Anforderungsanalyse beschreibt die wesentlichen Punkte der funktionalen sowie nicht funktionalen Requirements an die Applikation. Weiteres werden Prototypen sowie Szenarien und Personas dokumentiert, welche mit Benutzern durchgeführt wurden. Diese Informationen zusammen bilden die Basis für die Qualitätssicherung und die Tests.

[1] (Cockburn, 2010)

2 Business Use Cases

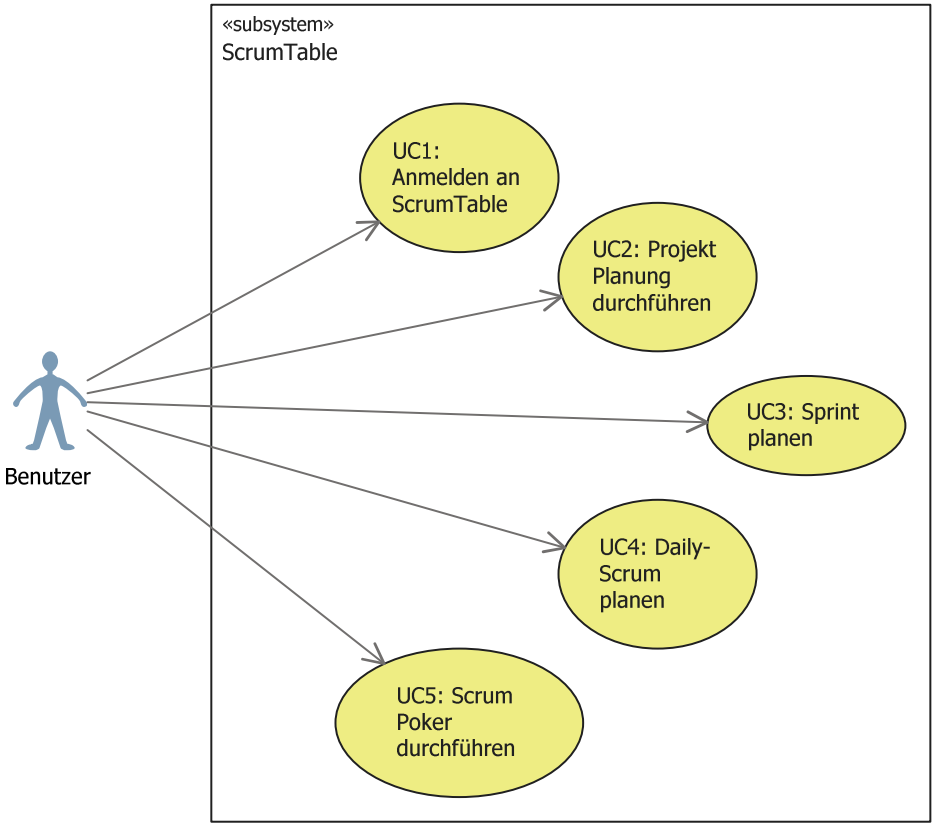


Abbildung 1 Business Use Cases aus Sicht des Benutzers

Das Use Case Diagramm beschreibt die Anwendungsfälle auf dem BUSINESS-LEVEL. Die genaueren SEA-LEVEL Anforderungen können der Use Case Beschreibung sowie den Szenarien entnommen werden.

ID	Beschreibung
1	Der Benutzer meldet sich an. Dies umfasst: - Anmelden an die ScrumTable-Applikation - Anmelden am Zielsystem (Microsoft Team Foundation Server 2010[2], kurz TFS) - Anmeldeinformationen werden in einer Konfiguration gespeichert
2	Der Benutzer plant das Projekt. Dies umfasst: - User Stories planen o Erfassen/mutieren o Priorisieren o Umfang schätzen - Sprint planen o Dauer bestimmen o User Stories zuweisen
3	Der Benutzer plant die Sprints. Dies umfasst: - User Stories für aktuellen Sprint auswählen - User Stories in Tasks aufteilen o Zeit schätzen o Verantwortlichkeit zuweisen - Anzeige der Kapazitäten o Team-Kapazität anzeigen o Personen-Kapazitäten anzeigen
4	Der Benutzer plant den Daily-Scrum. Dies umfasst: - Burn Down - und Burn Rate Chart anzeigen - Jeder Entwickler berichtet über seinen Stand
5	Scrum Poker durchführen um User Stories zu schätzen

[2] (Microsoft Team Foundation Server, 2010)

3 Personas

3.1 Daniel Devello


[3]

Daniel Devello

23 Jahre

Bachelor of Science, Computer Technology HSR

Ledig, keine Kinder

Seit 2 Jahren als Software Test Entwickler tätig

Sehr gute Kenntnisse im SW Bereich

Ist mit Scrum gut vertraut

Daniel arbeitet in einem sechs köpfigen Team direkt unter seinem Teamchef, dem Scrum Master. In diesem Team ist er verantwortlich für die Erstellung, Prüfung und Wartung von Business Applikationen.

Gearbeitet wird dabei in einem grossen Raum, welcher sich Daniel mit weiteren Arbeitskollegen teilt. Weitere Teammitarbeiter sind in anderen Räumen, welche sich nicht imselben Gebäude befinden, tätig. Daniel's Arbeitsstation umfasst einen Platz für einen Rechner mit zwei Bildschirmen sowie einen Platz für Papierarbeiten. Jedes Teammitglied besitzt ein eigenes stationäres Telefon. Besprechungen im Team werden in einem ebenfalls anliegenden Besprechungsraum abgehandelt.

Daniel entwickelt alle Artefakte nach den Vorgaben der HSR und besitzt meist einen aufgeräumten Arbeitsplatz. Seine technische Arbeit erledigt er korrekt und mit grossem Elan.
Scrum kennt Daniel erst seit diese Technik vor einiger Zeit in diesem Team vom Dozenten für eine neues Projekt eingeführt wurde. Die Grundlegenden Fakten zum Thema Scrum hat er schnell verstanden und kann sich selber dabei auch schon einbringen.

Das grösste Optmierungspotential sieht Daniel einerseits im Reduzieren des Papierkriegs, anderseits in der Zentralisierung der einzelnen dezentralen Scrum Artefakten:
„Während einer Besprechung muss der Scrum Master immer Protokoll führen, sobald etwas am Whiteboard geändert wird. Anschliessend muss er die Änderungen und Erweiterungen ins zentrale Wiki übertragen, damit es später für alle Projektteilnehmer dokumentiert ist. Ich vermisse die Sicht auf den Laptop des Scrum Masters, da dieser häufig am anderen Enden des Tisches sitzt. Aus meiner Perspektive geht das Vier-Augenprinzip verloren.“

[3] (Persona Service Verwaltungs AG & Co. KG, 2010)

3.2 Marco Maestro


[4]

Marco Maestro

32 Jahre

Dipl. Ing. FH

Familienvater von 3 Kindern (12, 15, 19)

Seit 4 Jahren als Scrum Master tätig

Guten Kenntnisse im Managen vom Projekten

Absolvierte mehrere Weiterbildungen im Bereich Agile Software-Entwicklung

Beherrscht Scrum

Marco leitet seit mehreren Jahren Projekte. Er koordiniert seine Mitarbeiter mit grosser Vorliebe und delegiert, erfasst und priorisiert Aufgaben für seine Teammitglieder. Die ständige Kontrolle des Projektes und des Projektfortschrittes ist eine seiner grossen Aufgaben.

Das Scrum Meeting findet jeden Tag am Morgen um 9:00 Uhr, im anliegenden Besprechungsraum statt. Dabei werden die diversen User Stories und Tasks besprochen, und zusammen mit dem gesamten Team genauer analysiert. Marco schaut sehr genau darauf, dass das Meeting nicht länger als 15 Minuten dauert.

Falls erforderlich, werden die Prioritäten und Risiken der Tasks und User Stories während eines Meetings angepasst. Dafür wird jedes Teammitglied zum Status seiner Arbeit befragt.

Marco verfügt über einen Laptop, auf welchem er die Tasks und User Stories erfasst hat und diese nun mutiert und ergänzt. Zusätzlich hat er am eigenen Arbeitsplatz eine Dockingstation, mit welcher er auf das Netzwerk zugreifen kann, um die Daten aus den Meetings auf die zentrale Dokumentenablage hochzuladen.

Für Marco ist der reibungslose Projektablauf sehr wichtig:
„Für mich ist ein Sprint in Scrum dann erfolgreich, wenn wir beim Review sagen können: ‚Jawohl, alles erledigt, alles Funktioniert, zur Zufriedenheit des Kunden‘.“

Durch die Weiterbildung zum Scrum Master im Fachbereich Agile Software Development hat sich Marco viel Scrum-Wissen erarbeitet. Dieses setzt er auch tatkräftig in die Praxis um. Neuerungen geht er nicht aus dem Weg und versucht jeweils die positiven Aspekte einer neuen Technologie gleich auszukosten.

Marco ist sehr zufrieden über die Art, wie er dank Scrum mit seinem Team die Meetings durchführen kann:
„Wichtig beim Daily Scrum Meeting ist, dass man am Story-Board ein physisches Erlebnis hat. Ob man die User Stories nun auf einem Multi-Touch-Tisch umher schiebt und mit einer Datenbank abgleicht ist einerlei. Wichtig ist einfach, dass es den Daily Scrum gibt, dass man zusammen sitzt, und dass man sieht, wie der Andere den Zettel mit der User Story bewegt.“

Marco sieht bei seinem täglichen Ablauf mit Scrum vor allem Optimierungsbedarf bei der Planung der Abarbeitungsreihenfolge der Tasks. Dazu führt er eine separate Tabelle, welche er nach jedem Meeting mit dem Story-Board abgleichen muss. Wie schnell doch etwas vergessen geht und dann wieder zu Missverständnissen und unnötigen Diskussionen führt!

[4] (Stolze, 2010)

4 Negativ Personas

ScrumTable Version 1 enthält nur Personas für die Entwicklung von kundenspezifischer Software. Die Personas für die Standard-Software werden bewusst weggelassen und gelten als negativ Personas, da das vorgesehene Zielsystem (Team Foundation Server) die entsprechenden Funktionalitäten für die Standard-Software Scrum Prozesse von Haus aus nicht vollständig unterstützt.

5 Ist-Scenarios

5.1 Projekt Planung: User Stories erfassen und Priorisieren

Der Product Owner berichtet über seine Anforderungen und seine Ziele, was er mit der Applikation erreichen möchte. Die Teammitglieder hören aufmerksam zu. Marco sowie auch Daniel hören mit und bringen immer wieder Ideen ein, wie eine User Story der Anforderung entsprechend erfasst werden soll.

So erwähnt der Product Owner die Anforderung, dass der Endbenutzer sich am Kundenportal, welches realisiert werden sollte, anmelden können sollte. Der Product Owner meint, dass dies eine „must"-Anforderung ist. Daniel ergreift sofort das Wort und meldet die Beschreibung der User Story: „As a User I'd like to log in in order to get the advanced portal features.". Marco editiert unterdessen diese User Story in dem Excel Sheet. Zu dieser User Story sollen nun auch die erforderlichen Aufwände erfasst werden. Dazu schlägt Daniel in diesem Fall das Scrum Poker vor, da sich die Entwickler untereinander nicht einig über den entsprechenden Aufwand sind.

5.2 Projekt Planung: Sprints planen

Nachdem die User Stories erfasst, priorisiert und durch Scrum Poker geschätzt wurden, schlägt Marco vor, dass man vorläufig alle User Stories auf Sprints verteilt. Dabei wird auf die Priorisierung der User Stories geachtet. Die Wichtigeren sollen so früh wie möglich gemacht werden. Ebenfalls schauen sie auf den geschätzten Aufwand der User Stories, um die Sprints etwa gleichmässig aufzuteilen.

Marco bemerkt beim Eintragen im Excel Sheet, dass der Aufwand aller User Stories im zweiten Sprint etwas zu hoch ist. Deshalb wird eine User Story von Sprint 2 nach Sprint 3 verschoben.

5.3 Sprint Planung: User Stories für nächsten Sprint auswählen

Marco, Daniel und die übrigen Entwickler sitzen in einem Project Planing Meeting. Der letzte Sprint wurde gerade abgeschlossen und nun wird der Product Backlog neu betrachtet.

Marco nimmt das Excel Sheet hervor und sieht darin, welche User Stories bereits erledigt wurden. Er zeigt diese allen. Dazu verwendet er einen Beamer, damit jeder die Excel Liste sehen kann. Nun werden die am höchsten Priorisierten User Stories genommen und in dem nächsten Sprint hinzugefügt. Daniel meint, dass anstelle der grossen User Story, die nächste genommen werden soll. Diese wäre etwas kleiner. Die grössere sei in der Zeit nicht umsetzbar, aber die kleinere kann gemacht werden. Die anderen Entwickler stimmen dem zu. Anschliessend werden nun die User Stories in Tasks unterteilt.

5.4 Sprint Planung: User Stories in Tasks aufteilen

Das Team hat bereits die User Stories dem nächsten Sprint zugeteilt. Marco wählt nun die erste User Story aus dem Excel Sheet aus und das Team bespricht die Tasks, welche zur User Story gehören. Für diese Tasks wird auch der Zeitaufwand in Stunden geschätzt. Daniel bringt ebenfalls sein Wissen ein und verpflichtet sich auch einzelne Tasks zu übernehmen. Marco trägt die Tasks unterdessen im Excel Sheet ein. So wird eine User Story nach der anderen in Tasks unterteilt, welche dann auch den Entwicklern zugewiesen werden.

Daniel erwähnt während dem Meeting noch, dass er die eine Woche nicht anwesend ist, da er in der Woche Ferien habe. Dies wurde doch total vergessen und es werden einige seiner Tasks anderen Entwicklern zugeordnet um ihn zu entlasten. Zum Glück wurde dieser Einwand am Anfang eingebracht, wodurch nicht so viel geändert und um geplant werden musste.

Marco bemerkt im Excel, dass die bereits geplante Anzahl Stunden mehr ist, als das Team im nächsten Sprint schaffen kann. Das Team entschliesst sich eine grössere User Story, welche nicht so wichtig ist, in den Product Backlog zurück zu legen und eine etwas kleinere zu nehmen. Diese wird wiederum in Tasks unterteilt. Marco sieht nun, dass die totalen Stunden etwas niedriger sind und dies als Team in der Zeit erreicht werden kann.

5.5 Daily Scrum durchführen

Die 15 minütigen Daily Scrum Meeting leitet Marco mit grossem Elan. Marco sitzt um 9:00 Uhr im Besprechungszimmer. Die Team-Mitglieder sind bereits im Raum anwesend, so dass Marco das Meeting starten kann. Marco berichtet kurz über seine erreichten Ziele des letzten Arbeitstages und was er an diesem Tag machen möchte. Er meint, dass er gut vorankomme und momentan auch nichts hat, was ihn hindert vorwärts zu kommen. Anschliessend geht er in die Runde und weist das erste Team-Mitglied an, seine erreichten Lösungen, arbeitsbedingten Ziele und Probleme darzulegen. So geht Marco durch die Runde. Jedes Team-Mitglied hat anschliessend über die wichtigsten Ereignisse des letzten Arbeitstages berichtet und weiss genau, was er am heutigen Tag tun wird.

Falls bei Team-Mitgliedern grössere Probleme anstehen, unterbricht Marco die Diskussion und beruft eine Sitzung nach dem Daily Scrum ein. Somit erreicht Marco, dass das Meeting maximal 15 Minuten dauert.

Anschliessend trägt Marco das Burn Down Chart nach, um die Errungenschaften klar sichtbar darzustellen. Falls die abzuarbeitenden Tätigkeiten immer mehr zunehmen, lädt Marco seine Mitarbeiter nach dem Scrum Meeting zu einer Krisensitzung ein. Ein Burn Rate Chart wird keines geführt, da der Aufwand für den Burn Down Chart genug gross ist und alles nötige darauf bereits ersichtlich ist.

5.6 Scrum Poker durchführen

Marco und Daniel befinden sich im Projekt Planungsmeeting. Die User Stories wurden bereits erfasst und priorisiert. Jetzt möchte das Team gemeinsam schätzen, wie Umfangreich die einzelnen User Stories in etwa sind. Marco möchte damit auch bezwecken, dass eine Diskussion angeregt wird und allfällige Fragen oder Unklarheiten bewältigt werden können. Dazu möchte er Scrum Poker anwenden, damit die Meinungen nicht durch einen vorherigen Sprecher beeinflusst werden. Er hat dazu für jeden Entwickler ein Set von Scrum Poker Karten mitgebracht. Er gibt Daniel und den anderen Entwicklern je ein Set von Karten und behält für sich auch ein Set.

Marco nimmt die erste User Story mit dem Namen „Statistik drucken“ aus dem Product Backlog und liest die Beschreibung vor. Er fordert die Entwickler auf zu überlegen, wie gross der Aufwand für die User Story sei. Ein Story Point bedeute in diesem Projekt in etwa der Komplexität von einem Tag Arbeit. Daniel hat bereits einige Erfahrung darin, wie man programmgesteuert druckt. Er schätzt, dass dies etwa ein Aufwand von einem Arbeitstag sein wird. Deshalb wählt er die Karte mit der 1 und legt sie umgekehrt auf den Tisch. Marco hat selbst etwas weniger Ahnung von Drucken, aber meint trotzdem, dass es wohl etwa 2 Tage dauern würde. Und deshalb legt er die Karte mit der 2 verdeckt auf den Tisch. Die anderen Entwickler legen ebenfalls ihre Karten auf den Tisch. Dabei legt ein Entwickler eine 0.5. Wieder ein Anderer eine „?“-Karte, da er keine Ahnung über das Drucken hat und nicht weiss, was er von der User Story halten soll. Weitere Karten sind 3 und 1 die abgelegt wurden.

Marco fragt den Entwickler mit der 0.5, wieso er meint, dass es nur so wenig Aufwand sei. Dieser erwähnt, dass in einem anderen Projekt einmal Code geschrieben wurde, der genau dies erledigt. Die anderen Entwickler erinnern sich nun auch und stimmen zu, dass dies dadurch schneller als gedacht implementiert werden kann. Aus diesem Grund wird diese User Story mit 0.5 Story Points versehen. Marco trägt dies im Excel nach.

So wird auch mit den anderen User Stories verfahren bis alle User Stories abgearbeitet wurden und bewertet wurden. Teilweise tauchten Fragen auf, was denn genau mit der User Story gemeint ist. Diese wurden dann etwas besser erklärt oder in wenigen Fällen sogar in zwei User Stories aufgeteilt, da bemerkt wurde, dass sich darin eigentlich zwei Anwendungsfälle befinden.

6 Soll-Scenarios

6.1 Anmelden + Konfigurieren

Marco möchte mit seinem Team und dem Product Owner die Projekt Planung durchführen. Mit seinem Team trifft er sich im Sitzungszimmer, in welchem auch der Microsoft Surface Tisch mit ScrumTable steht. Er startet den Tisch und dann die Applikation ScrumTable. Er meldet sich mit seiner Login-Karte mit einem Pin ein. Somit ist er nun auch schon bereit die Projekt Planung zu starten. Denn die Verbindung zum Team Foundation Server und das Projekt wurden bereits aus den Einstellungen entnommen, welche der Karte hinterlegt ist.

6.2 Projekt Planung: User Stories erfassen und Priorisieren

Marco hat sich mit seinem Team, zu welchem auch Daniel gehört, um Tisch mit ScrumTable versammelt. Auch der Product Owner ist anwesend. Sie möchten gemeinsam die Projekt Planung durchführen. Er hat sich bereits auf ScrumTable mit seiner Login-Karte angemeldet.

Der Product Owner hat die User Stories bereits auf dem Team Foundation Server erfasst. Dazu hat er das Excel Sheet verwendet, welches auf dem SharePoint Server unter „Shared Documents“ liegt und vom Team Foundation Server angelegt wurde. Dabei hat er nur den Namen und den Stack Rank gesetzt. Die übrigen Felder hat er nicht beachtet. Er zeigt nun am Microsoft Surface Tisch, welche User Stories ihm am wichtigsten sind und welche nicht so wichtig sind. Dazu unterteilt er sie in „must“, „should“ und „could“ User Stories. Durch verschieben auf dem Tisch weist er sie den jeweiligen zu. Ebenfalls innerhalb eines solchen Bereiches kann er sie in der Reihenfolge priorisieren.

Daniel und die anderen Entwickler geben zu verschiedensten Zeitpunkten ihre Meinung dazu ab oder stellen Fragen.

6.3 Projekt Planung: Sprints planen

Nachdem der Product Owner gemeinsam mit dem Scrum Master und den Entwicklern die User Stories priorisiert hat, möchte Marco eine etwaige Verteilung der User Stories auf die Sprints vornehmen. Er wechselt zur der entsprechenden Ansicht und nimmt die wichtigsten User Stories zuerst und plaziert diese im ersten Sprint. Weitere User Stories legt er dann in den zweiten Sprint und so weiter. Dabei beachtet er die gesamte Anzahl von Story Points, welche sich in einem Sprint befinden und kann somit ziemlich genau schätzen, was möglich sein sollte.

6.4 Sprint Planung: User Stories für nächsten Sprint auswählen

Marco, Daniel und die übrigen Entwickler sitzen im Meeting für die Sprint Planung. Der letzte Sprint wurde gerade abgeschlossen und nun wird der Product Backlog neu betrachtet. In diesem Projekt wurden die User Stories in der Projekt Planung noch nicht an die Iterationen (Sprints) verteilt. Im Raum sind der Scrum Master, der Product Owner und die anderen Entwickler anwesend.

Der Scrum Master hat sich in ScrumTable mit seiner Login-Karte angemeldet und in die Sprint Planung gewechselt. Alle User Stories, welche sich noch im Product Backlog befinden und keinem Sprint zugeordnet sind werden angezeigt. Ebenfalls solche, welche bereits diesem Sprint zugeteilt worden sind. Das Team beschliesst zusammen, welche User Stories im nächsten Sprint abgearbeitet werden sollten.

6.5 Sprint Planung: User Stories in Tasks aufteilen

Marco, Daniel und die anderen Entwickler haben bereits ausgewählt, welche User Stories sie in diesem Sprint erledigen wollen. Nun möchten Sie die User Stories gemeinsam in Tasks unterteilen. Für jeden Task schätzen sie die ungefähre Arbeitszeit in Stunden und weisen ihn einem Entwickler zu. Dabei sieht das ganze Team auch jeweils die Auslastung der einzelnen Entwickler.

Dank der Ansicht der Auslastung merkt der Scrum Master, dass einer der Entwickler zu fest ausgelastet wird und Daniel im Moment nur sehr wenig Arbeit für den kommenden Sprint zugewiesen bekommen hat. Sie beschliessen gemeinsam, dass Daniel einen Auftrag des Kollegen übernimmt.

Während dem Erstellen der Tasks der letzten User Story wird bemerkt, dass die User Story doch etwas grösser ist als angenommen. Deswegen gibt es zu viel Arbeit für das ganze Team. So viel Arbeit kann das Team in diesem Sprint nicht bewältigen. Gemeinsam wird beschlossen, diese User Story lieber im nächsten Sprint zu machen. Aus diesem Grund wird die User Story, mit ihren Tasks, in den Product Backlog zurück verschoben. Anstelle dieser User Story wird eine kleinere genommen und diese wieder in Tasks unterteilt. Jetzt ist es nicht zu viel Arbeit für das Team. Das Team meint nun, dass man dies alles in diesem Sprint erledigen kann.

Durch das stetige Beachten der Auslastung der einzelnen Entwickler sind nun alle Mitarbeiter für den folgenden Sprint ideal ausgelastet. Nachdem die Sitzung geschlossen wurde, weiss Daniel was er im nächsten Sprint zu erwarten hat.

6.6 Daily Scrum: Burndown und Burnrate Chart ansehen

Zu Beginn des Daily Scrum am Morgen um 9 Uhr möchte Marco ganz kurz den Burn Down Chart ansehen und auswerten. Diesen zeigt er im Daily Scrum in ScrumTable an. Es freut ihn, dass dieser automatisch berechnet wurde und er ihn nicht jeden Tag neu von Hand zeichnen muss. Beim Betrachten des Burn Down Chart sieht er, dass sie bisher gut vorankommen mit dem Projekt. Dies macht ihn sowie die anderen Entwickler glücklich und motiviert sie weiterhin ihr bestes zu geben um das gemeinsam gesteckte Ziel zu erreichen.

Der Burn Rate Chart der angezeigt wird zeigt Marco und dem Team ebenfalls, dass sie auf richtigem Kurs sind.

6.7 Daily Scrum: Jeder Entwickler berichtet über seinen Stand

Marco möchte im Daily Scrum Meeting jedem Entwickler die Möglichkeit geben, kurz über den Stand seiner Arbeit zu berichten. Dafür hat er bereits ScrumTable gestartet und in die Ansicht für den Daily Scrum gewechselt. Hier sieht man die verschiedenen Entwickler in einer zufälligen Reihenfolge. Dies bringt Abwechslung in das tägliche Meeting. Daniel sieht, dass er heute wieder ein Mal der Erste ist und begibt sich vor den Tisch. Er sieht seine Tasks, und verschiebt diejenigen Tasks, die er abgeschlossen hat nach „Erledigt“. Dies kommentiert er mit eigenen Worten und trägt auch die Zeit nach, die er dafür aufgewendet hat. Aus den noch offenen Tasks wählt er zwei aus, die er heute erledigen möchte und zieht sie nach „In Arbeit“. Einen Task hat er noch von gestern, den er nicht ganz beenden konnte. Bei diesem hat er nur kurz die Zeit angepasst, lässt ihn aber im Status „In Arbeit“.

6.8 Scrum Poker durchführen

Marco befindet sich am Sprint Planning zusammen mit vier Entwicklern. Als nächster Punkt sollen die User Stories bewertet werden. Dazu gibt er den Entwicklern je einen Stapel von Scrum Poker Karten in einer anderen Farbe. Der eine Entwickler kriegt grüne Karten, der andere blaue, und so weiter.

Die erste User Story kommt aus dem spezialisierten Arbeitsgebiet eines Entwicklers. Alle Mitglieder legen ihre Scrum Poker Karte mit der Schätzung verdeckt auf den Scrum Table. Sobald jeder seine Karte abgelegt hat, wählt Marco auf ScrumTable durch Knopfdruck zum Karten-aufzudecken aus. Jeder Entwickler kann nun seine Karte vom Tisch nehmen und sieht die abgelegten Karten und erkennt anhand der Farbe, wer welche Karte gelegt hatte. Der Spezialist und ein weiterer Entwickler haben 5 Stunden geschätzt, die übrigen Mitglieder enthalten sich mit der „?“-Karte einer konkreten Schätzung. Marco und das Team beschliessen nach kurzem diskutieren, diese User Story mit 5 Story Points zu bewerten und wählt den Wert der gelegten Karte aus. Automatisch wird dieser nun der User Story zugewiesen.

Nun nimmt er die zweite User Story aus dem Stapel der nicht bewerteten User Stories. Diese kommt aus einem allgemeineren Gebiet. Nach dem Aufdecken der Karten wird klar, dass hier unterschiedliche Meinungen existieren. Die Entwickler haben unterschiedliche Karten abgelegt. Es liegen die Schätzungen 2, 3, 8, und 13 vor. Als Durchschnitt wird der aufgerundete Wert 8 angezeigt. Nach einer etwas längeren Diskussion wird die Unstimmigkeit beseitig und es wird bestimmt, die User Story mit 5 Story Points zu versehen. Marco wählt deswegen manuell die 5.

Nach diesem Muster werden noch sechs weitere User Stories bewertet. Danach fährt Marco mit der Erstellung der Tasks weiter.

7 System-Sequenzdiagramme

Die System-Sequenzdiagramme sollen dem Leser eine Übersicht geben, welche Anfragen konzeptionell zum Microsoft Team Foundation Server gesendet werden und welche Daten konzeptionell auf dem Surface Tisch verarbeitet werden.

Diese Diagramme werden nur konzeptionell beschrieben. Die vollständige Interface-Beschreibung zwischen ScrumTable und Microsoft Team Foundation Server befindet sich im Teil II - SW-Projektdokumentation im Kapitel Implementation.

7.1 UC2-5: Daten mutieren / CRUD

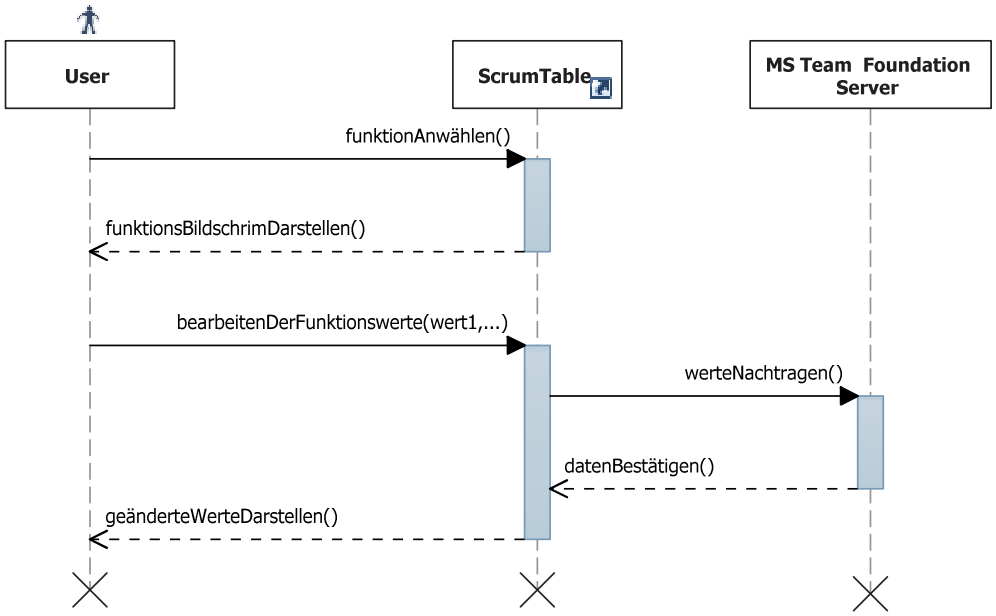


Abbildung 2 SSD fürs Bearbeiten von Daten auf dem Team Foundation Server

7.2 UC1: Anmelden an MS Surface

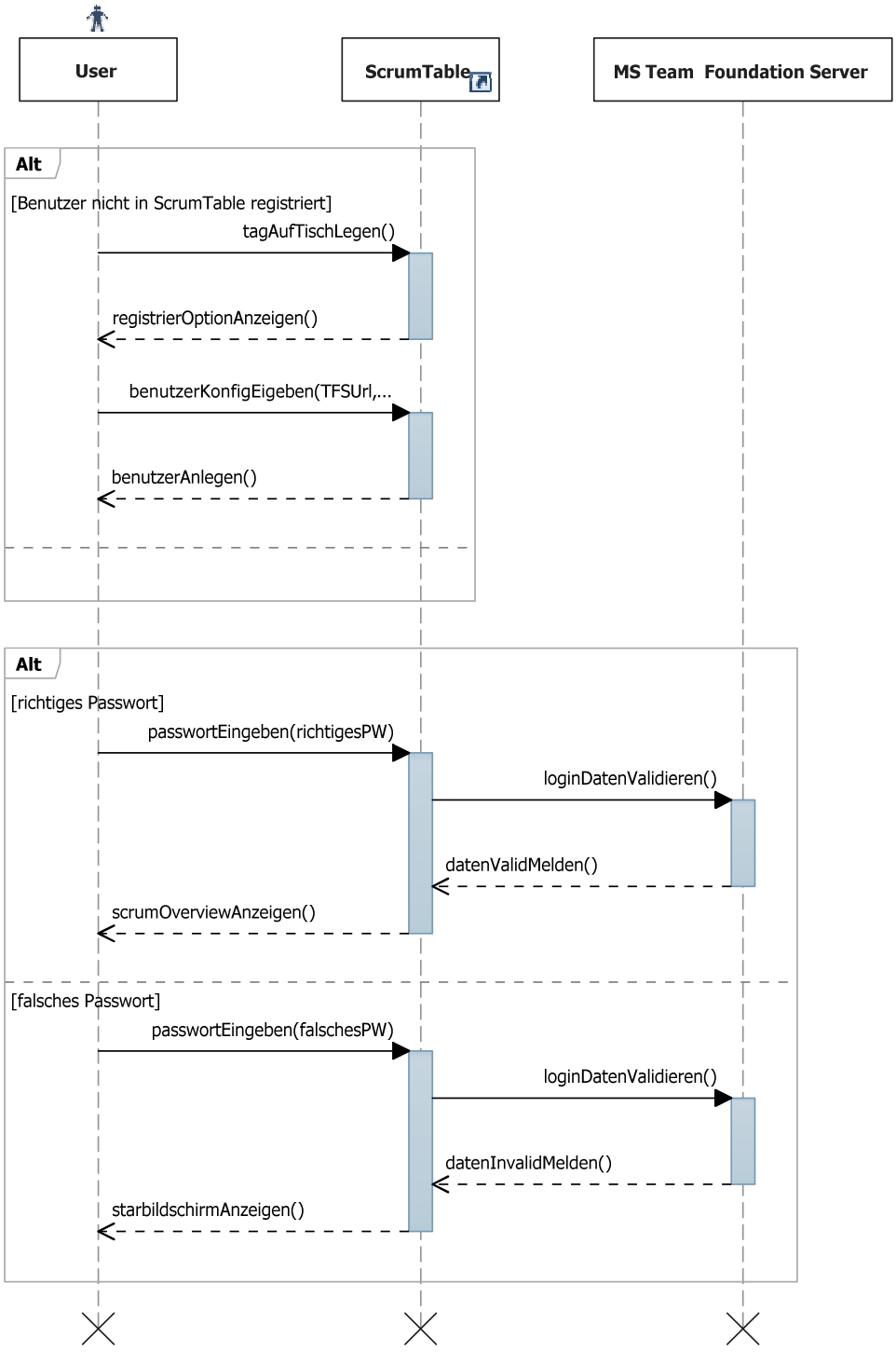


Abbildung 3 SSD fürs Einloggen auf dem Team Foundation Server

8 Nicht-funktionale Anforderungen

Die Anforderungsspezifikationen von ScrumTable folgen der ISO/IEC 9126[5] Norm. Dieser ISO-Standard enthält eines der renommiertesten Modelle um Softwarequalität sicherzustellen. Mehr über ISO 9126 kann auf http://de.wikipedia.org/wiki/ISO/IEC_9126 nachgeschlagen werden.

8.1 Funktionalität

8.1.1 Richtigkeit

ScrumTable besteht aus den typischen Elementen des Scrums[6]. So sollen beim zuweisen dieser Elemente die Kind-Elemente richtig und vollständig sowie konsistent zugewiesen werden. Namentlich ist dies:

Element	Unterelement
Product Backlog	Stories
Sprint Backlog	Sprint
Sprint	Stories
Stories	Tasks

Tabelle 1 Korrekte Verlinkungsmöglichkeiten

Die Verklingen sollen anschliessend in ein spezifisches Zielsystem (zum Beispiel Microsoft Team Foundation Server) eingespiesen werden können und darin korrekt und konsistent gespeichert werden.

8.1.2 Interopabilität

ScrumTable soll über die Möglichkeit verfügen, mit diversen Ziel- und Endsystemen zu kommunizieren. Namentlich soll eines davon der Microsoft Team Foundation Server bilden, es sind allerdings auch weitere Endsysteme, beispielsweise XML als Datenspeicher denkbar. In der ersten ScrumTable Version soll der Microsoft Team Foundation Server 2010 RC1 unterstützt werden.

8.1.3 Sicherheit

Die Sicherheitsanforderungen an ScrumTable sind hoch, da ScrumTable mit Microsoft Team Foundation Server zusammenarbeiten soll. Dieser von Microsoft entwickelte Server enthält nicht nur eine fundamentale Scrum-Unterstützung, sondern darüber hinaus auch den gesamten Program Source, welcher als besonders schützenswertes Gut gilt.

Deshalb soll die Verbindung zum Microsoft Team Foundation Server optional mit einem besonders gesicherten Protokoll aufgebaut werden. Dies soll im Falle von Microsoft Team Foundation Server das TLS/SSL-Protokoll sein. Die TLS/SSL Funktion ist optional und in der Version 1.0 noch nicht enthalten.

Die Benutzerpasswörter, welche fürs Login auf Server-Seite benötigt werden, sollen nicht in Klartext zwischengespeichert werden. Dies wäre eine Sicherheitslücke mit potentiell fatalen Auswirkungen.

[5] (www.wikipedia.ch IEC_9126, 2010)
[6] (Scrum Alliance, 2010)

Ein automatischer Login mit Statusspeicher der aktuellen Sicht soll nur über eine Two Factor Authentication[7] möglich sein. Das angestrebte Vorgehen soll nach Möglichkeit auf den allgemein bekannten und veröffentlichten Verschlüsselungsalgorithmen basieren.

8.1.4 Angemessenheit

Die aufgabenorientierte Zusammensetzung von ScrumTable wird detailliert in der Analyse beschrieben.

Zusätzlich sei an dieser Stelle erwähnt, dass das Ziel- und Endsystemen grundsätzlich einem beliebigen Datenspeicher entsprechen soll. In der ersten Version von ScrumTable soll dies der Microsoft Team Foundation Server sein. Spätere Releases sollen ohne architektonische Änderungen neue Verbindungsarten zu weiteren Servern unterstützen.

8.1.5 Ordnungsgemässheit

ScrumTable verhält sich gemäss den Ausführungen beschrieben in den weiteren Kapiteln dieser Anforderungsspezifikation. Grundsätzlich ist das Verhalten von ScrumTable auf den Regeln des Scrums basierend und soll entsprechend für Scrum-Kenner intuitiv sein.

Die Basis dieser Arbeit bildet das Dokument Aufgabenstellung, welches der Arbeit beiliegt. Dieses Dokument beschreibt die grundsätzlichen Aspekte, welche ScrumTable ordnungsgemäss unterstützen soll.

8.2 Zuverlässigkeit

8.2.1 Reife

ScrumTable soll in 98% der Fälle die ScrumTable Daten aus dem Ziel- und Endsystemen einlesen und darstellen können. Es sollen beim zurückspeichern der Daten in ein Endsystem bei 98% keine Inkonsistenzen auftreten.

ScrumTable ist nicht das führende Datenverarbeitungs-System. Dies bildet das Ziel- und Endsystem (Server). Es können also Inkonsistenzen aufgrund von veränderten Daten auf dem Server entstehen. In diesem Falle kann ScrumTable die Daten nicht mehr sauber aus dem Server auslesen bzw. auf dem Server speichern. Mehr zur Fehlerbehandlung siehe Fehlertoleranz.

Auf Netzwerkverbindungsfehler hat ScrumTable keinen Einfluss und daher besteht kein zusätzlicher Behandlungsbedarf.

8.2.2 Fehlertoleranz

ScrumTable zeigt dem Endbenutzer Fehlermeldungen betreffend Login aus dem Ziel- und Endsystem an. Bei Auftreten eines Fehlers wird dieser in ein Log-File gespeichert und der Benutzer wird wieder auf den Start-Screen, mit einer Ausgabe des entsprechenden Fehlers, geleitet.

Fehler können aufgrund von weiteren Benutzern, welche die Daten auf dem Ziel- und Endsystem von einem anderen Computer zum selben Zeitpunkt mutieren, verursacht werden. Dabei kann der aktuelle Bearbeitungsfortschritt auf ScrumTable verloren gehen. Es gilt der folgender Grundsatz: Das Hauptsystem, also das System mit den aktuellen Daten, entspricht dem Ziel- und Endsystem

[7] (www.wikipedia.ch Two-Factor Authentication, 2010)

(beispielsweise Microsoft Team Foundation Server). Es werden also nur die letzten Änderungen im System gespeichert oder bei der Mutation eines gelöschten Datensatzes die neuen Daten durch eine Ausnahme verworfen.

Benutzer-Inputs sollen von ScrumTable direkt bei deren Eingabe auf Gültigkeit überprüft, d.h. validiert werden.

8.3 Benutzbarkeit

8.3.1 Verständlichkeit

Die Applikation soll eine gute interne und externe Konsistenz aufweisen. Die Prototypen des externen Designs sollen durch mehrere Reviews auf deren Verständlichkeit überprüft werden. Die Verständlichkeit soll für die Microsoft Surface Multi-Touch-Umgebung optimiert werden, das heisst den Microsoft Surface Guildlines[8] entsprechen.

Die Oberfläche der Applikation wird also durch die Microsoft Developer Network Surface Guildlines beeinflusst und ist daher in dessen Komplexität nicht beliebig wählbar.

8.3.2 Erlernbarkeit

Die Hauptnavigationselemente sollen schnell erlernbar und übersichtlich gestaltet werden. Die interne Komplexität der Applikation soll vor dem Benutzer verborgen werden. Der Benutzer soll sich also auf einer einfach erlenbaren und für eingeschulte Scrum User intuitiven Oberfläche befinden. Die oben beschriebenen Punkte sollen durch die Reviews des Paper Prototypes und des User Interfaces mit echten Benutzern bestätigt werden.

Die Einarbeitungszeit fürs Beherrschen der grundsätzlichen Navigationselemente und Funktionalitäten der Applikation soll mit einem Einschulungsvideo maximal 15 Minuten dauern. Dabei entsprechen grundsätzliche Navigationselemente und Funktionalitäten den folgenden Szenarien: Hauptnavigation bedienen (siehe Teil II – SW Projektdokumentation Kapitel Paper Prototype), Login tätigen, User Stories mutieren, Tasks erfassen, Tasks Personen zuweisen, Sprints planen.

Tiefergehende Systeminstruktionen und Bedienelemente sollen der Benutzerhilfe entnommen werden können.

8.3.3 Bedienbarkeit

Der Benutzer soll die ScrumTable Oberfläche grundsätzlich komplett ohne Maus bedienen können. Die Bedienung erfolgt über den Microsoft Surface TouchScreen, welcher mit den Fingern bedient wird. Die Tastatur (physisches Keyboard) kann bei Eingaben aus Effizienzgründen zur Hilfe genommen werden.

Folgende Mechanismen zur Sicherstellung der Bedienbarkeit sind geplant:

- Keine durch Applikationsfehler verursachten Wartezeiten.
- Schneller Programmstart: Maximal 60 Sekunden vom Programmstart bis zur benutzbaren Oberfläche.

[8] (MSDN User Experience Guidelines for Surface, 2010)

- Intuitive Bedienelemente für Touch-Screen optimiert: Keine Einarbeitung in die einzelnen Bedienelemente, möglichst Synergien von bestehenden GUI Komponenten übernehmen.
- Einfache, prägnante und aussagekräftige Grafiken für Buttons und Navigation: Der Benutzer soll auf der Oberfläche die Buttons klar von den Hintergrundbildern und Titeln unterscheiden können.
- Der Benutzer soll sich nicht um Netzwerk- und Kommunikationsprotokolle kümmern müssen.
- Auf Basis der Paper-Prototypen wird das Benutzerinterface angefertigt. Die Paper-Prototypen durchlaufen mehrere Sprints, welche durch die Ergebnisse der Tests mit den Testbenutzern beeinflusst werden.
- ScrumTable wird durch fortlaufende Usability-Tests verbessert. Die Reviews werden dokumentiert und fliessen ins Programm ein.

8.3.4 Wiederherstellbarkeit

Ein Programmabsturz führt zum Verlust sämtlicher sich im Speicher befindlichen Daten. Nicht vom Datenverlust betroffen sollen die getätigten Systemeinstellungen sowie die bereits auf das Ziel- und Endsystem gespeicherten Werte sein.

Ein Programmabsturz kann zur Folge haben, dass die gesamte Touch-Applikation im Surface Shell neu gestartet werden muss. Dieses Verhalten wird von Microsoft Surface Tisch vorgegeben und kann deshalb nicht verändert werden.

8.4 Effizienz

8.4.1 Zeitverhalten

Das Ziel des Zeitverhaltens in Bezug auf die Netzwerk-Verbindungsperformance von ScrumTable ist, dass ScrumTable die Informationen über ein möglichst übliches Netzwerk (100-1000MBit/s) performant, das heisst innerhalb von 20 Sekunden, darstellen kann. Bedingt durch das verwendete Netzwerk-Protokoll kann die Fehler-Rückmeldung aufgrund eines Verbindungsunterbruchs bis zu 90 Sekunden dauern.

Bei verschlüsselter Verbindung und hoher Netzwerkauslastung sowie grossen Transfermengen, kann die Kommunikation mit dem Microsoft Team Foundation Server auch bis zu maximal 90 Sekunden dauern. Bei einer sehr komplexen Netzwerkstruktur, zum Beispiel bei VPN basierenden Kommunikationstunnels, kann die Dauer für den Datentransfer noch höher liegen.

8.4.2 Verbrauchsverhalten

Bei ScrumTable soll der Benutzer keine merkbaren Performance-Unterschiede (d.h. nicht-reagieren des Benutzerinterfaces) während dem Verwenden der Applikation spüren. Dies gilt explizit für die Übergänge zwischen den verschiedenen Screens sowie für das Aufbauen der Elemente auf dem Table Screen.

Eine Ausnahme zum beschriebenen Verhalten bildet die Netzwerk-Kommunikation. Diese kann aufgrund Änderungen in der Konnektivität sowie Änderungen der Netzwerklast variieren.

8.5 Änderbarkeit

8.5.1 Modifizierbarkeit

Weitere Ziel- und Endsysteme sollen entsprechend der Angemessenheit flexibel erweiterbar sein. Des Weiteren sind folgende Massnahmen geplant:

- Zusätzliche Ziel- und Endsysteme sollen keine Architekturumbauten zufolge haben.
- Weitere Sprachen des User Interfaces sollen einfach zu implementieren sein.
- Einfache Anpassungen an den Views sollen durch XML-Modifikationen realisierbar sein. Diesem Punkt kann mit dem konsequenten Einsatz von WPF entsprochen werden.

8.5.2 Stabilität

Änderungen betreffend Interfaces in der Applikation sind vor allem im Bereich der Ziel- und Endsysteme zu erwarten. Diese Änderungen werden entweder verursacht durch die Hersteller der Ziel- und Endsysteme oder durch Hinzufügen neuer Konnektoren zu Ziel- und Endsystemen.

Unerwartete Änderungen erfolgen also vorwiegend durch Änderungen vom Microsoft Team Foundation Server 2010 *Release Candidate*, auf welchem ScrumTable aufbaut, zur *Final Version*. Auf welchem Niveau die entsprechende Änderungs-Wahrscheinlichkeit liegt, kann nicht qualifiziert bestimmt werden.

Falls die Ziel- und Endsysteme komplett ändern, muss allenfalls der entsprechende Konnektor neu geschrieben werden.

8.5.3 Prüfbarkeit

Die Stabilität soll durch geeignete Unit Tests überprüfbar gemacht werden. So sollen also grundsätzlich die Konnektoren sowie die darauf basierenden Business Komponenten auf ihre Gültigkeit überprüft und getestet werden. Der entsprechende Aufwand soll vor der Implementation der Konnektoren geschätzt werden.

8.6 Übertragbarkeit

8.6.1 Anpassbarkeit

Die Software soll auf dem Zielsystem, das heisst dem auf Microsoft Surface Tisch, funktionsfähig sein. Aufgrund der Hard- und Software-Vorgaben und Microsoft Surface kann ScrumTable nur mit grossem Aufwand auf eine weitere Plattform portiert werden. So enthält der Touch-Table diverse Elemente und Features, welche nur auf der Surface Hardware einsatzbereit sind. Somit ist eine erweiterte Anpassbarkeit der Benutzerschnittstelle nicht geplant.

8.6.2 Installierbarkeit

ScrumTable wird mit einer installierbaren Setup Datei ausgeliefert. Dieses kann anschliessend Wizard-gesteuert auf dem Surface Tisch installiert und deinstalliert werden. Die Installation soll zur Folge haben, dass die ScrumTable Applikation in der Surface Shell, dem Basisprogramm für alle Surface Applikationen, registriert wird.

Auf dem Surface Table muss das Surface Shell sowie die dazu benötigten Grundframeworks bereits installiert sein.

8.7 Schnittstellen

8.7.1 Benutzerschnittstelle

ScrumTable setzt auf folgenden Benutzerschnittstellen auf:

- Microsoft Surface Tisch mit Touch-Screen Funktionalität
  - Die Bedienung erfolgt mit dem Finger oder einer separaten Tastatur, mehr dazu siehe Bedienbarkeit.
  - Microsoft Surface bietet separate, für den Touch-Screen optimierte Komponenten an. Diese sollen wenn immer sinnvoll eingesetzt werden.
- Microsoft .NET Framework 3.5 mit Windows Presentation Foundation
  - Basis-Komponenten für den Microsoft Surface wird im .NET Framework 3.5 abgebildet.

8.7.2 Hardwareschnittstelle

Als Hardware wird der Microsoft Surface Tisch vorausgesetzt. Die Hardware wird von Microsoft vorgegeben und hat die folgende minimalen Spezifikationen:

- Intel Core 2 Duo @ 2.13GHz
- 2GB DDR2 RAM
- 250GB SATA Hard Drive
- ATI Radeon X1650
- Maximale Auflösung 1024x768 Pixel

Diese Angabe wurden der Microsoft Surface Hardware Specifiactions entnommen, welche unter http://go.microsoft.com/?linkid=9686793 veröffentlich sind.

8.7.3 Softwareschnittstelle

- Windows Vista Business SP 1+
- Microsoft .NET Framework 3.5 SP1
- Windows Presentation Foundation 3.5
- Log4NET[9]
- Surface Platform Runtime
- Microsoft Team Foundation Server 2010
- Microsoft Unity Application Block 1.1[10]
- WPF Localization - On-the-fly Language Selection[11]

8.7.4 Lizenzanforderung

Um ScrumTable zu verwenden, werden folgende Software-Lizenzen benötigt:

- Surface Software License
- Surface Platform Runtime SP 1
- Windows Vista Business SP 1+

[9] (log4net, 2010)
[10] (Microsoft Application Block, 2010)
[11] (Shamam, 2009)

Server-Seitig werden folgende Software-Lizenzen benötigt:

- Microsoft Team Foundation Server 2010
- Microsoft SQL Server 2008 mit Reporting Services und Analysis Services
- Microsoft Windows Server 2008

Für die Entwicklung (Entwicklungsarbeitsplätze) werden zusätzlich folgende Lizenzen benötigt:

- Microsoft Visual Studio .NET 2008
- Surface Platform SDK 1.0 SP1
- Surface SDK 1.0 SP1 Workstation Edition

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

User Centered Design

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

Quellenverzeichnis

Garrett, J. J. (2002). *Elements of User Experience.* Peachpit Press.

Hochschule für Technik Rapperswil, Prof. Dr. Stolze. (17. März 2010). *UInt2 - FS 2010*. Abgerufen am 17. März 2010 von UInt2 - FS 2010: \\vf3\skripte\Informatik\Fachbereich\User_Interfaces_2\UInt2\Vorlesungsfolien\

1 Einführung

1.1 Zweck

Dieses Dokument soll für das Projekt ScrumTable das Vorgehen im Bereich des Requirement Engineerings und User Interface Design dokumentieren. Zuerst wird kurz auf die grundlegenden Modelle eingegangen, anschliessend werden die entstandenen Artefakte und deren Schlussfolgerungen erläutert. Viele Schlussfolgerungen sind in weiteren Dokumenten detailliert behandelt und werden deshalb nur kurz erwähnt und referenziert.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

2 Vorgehen

Das User Centered Design Vorgehen bei ScrumTable zielt darauf ab, die Software möglichst gebrauchstauglich für die Alltagsarbeit der Zielgruppe von ScrumTable zu gestalten. Dies versucht ScrumTable durch die Anwendung des 5 Schichten Models von Garret zu erreichen.

2.1 Das 5 Schichten Models nach Garret

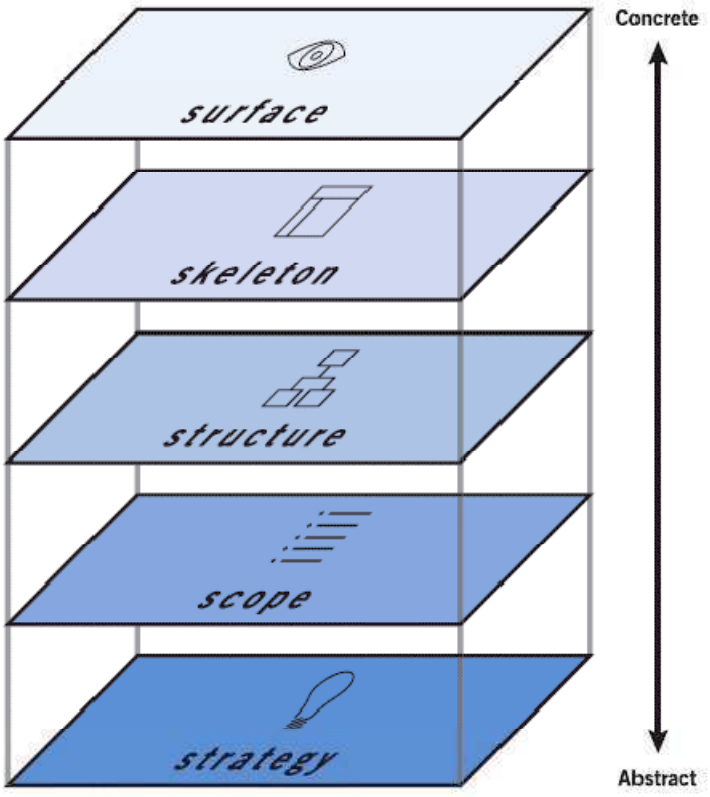


Abbildung 1: Methoden im 5 Phasen Model von Garret[1]

Bei den 5 Schichten nach Jesse James Garrett[2] wird ein Bottom-Up Ansatz angewendet. Dies bedeutet, dass die Applikation von Grund auf im interaktiven Prozess mit realen Benutzern der Applikation erstellt wird. Dieser Buttom-Up Ansatz zielt darauf ab, die Software von Grund auf den Wünschen der Endkunden (User) auszurichten: Der Endbenutzer steht im Zentrum – er bestimmt die Wünsche und die wunden Punkte der aktuellen Lösung (also das WAS als Teil der Strategie und Scope Ebene). Die Lösung ist ihm auszurichten und mittels der zur Verfügung stehenden Technologien umzusetzen (das WIE als Teil der Ebenen Structure, Skeleton und Surface). Technische Abklärungen und Finesse, welche dem Benutzer keine Wertsteigerung bringen, sind zu vermeiden.

[1] (Hochschule für Technik Rapperswil, Prof. Dr. Stolze, 2010)
[2] (Garrett, 2002)

Die folgende Tabelle zeigt die 5 Schichten nach Garret in einer Übersicht mit einer kurzen Beschreibung (zur besseren Veranschaulichung vom ersten Schritt zum Letzten):

Schicht	Beschreibung
Strategy	Zielgruppe und Vision der Applikation bestimmen.
Scope	Features mit Business-Values erfassen.
Structure	Navigation innerhalb der Applikation planen, konzeptionelles Modell und User Interface Architecture aufnehmen.
Skeleton	Layout der Applikation mittels Grid planen.
Surface	Farben und Schriftarten der Applikation definieren.

Tabelle 1 Beschreibung des 5 Schichten Modells (Garrett) nach Prof. Dr. M. Stolze[3]

2.2 Erkenntnisse von Goodwin

Garrett beschreibt den strukturellen 5 Schichten Ansatz. Dieser zeigt die verschiedenen Bereiche, in denen die Erkenntnisse erlangt werden sollten. Diese Ergebnisse können mittels eines Vorgehens nach Goodwin erarbeitet werden, welches auch die zu entstehenden Artefakte (Dokumente) genauer definiert:

Schicht	Beschreibung
Strategy	Die **Vision** des Programmes wird in dieser Phase erstellt. Anschliessend definiert man die Zielgruppe (**Personas**) sowie deren Umgebung (**Context**) mit der Applikation. Diese Informationen können aus Interviews und **Contextual Inquiries** mit den Endanwendern gewonnen werden.
Scope	Die funktionalen Anforderungen werden aus den Interviews gewonnen und als **Szenarios** abgelegt. Die Features werden also in dieser Phase in Form von **Ist- und Sollszenarien** erfasst.
Structure & Skeleton & Surface	Die Navigationsstruktur und die User Interface Architecture werden mittels **Grid** aufgesetzt und zusammen mit dem Endbenutzer in **Interviews** validiert. Die ersten Validationen erfolgen mittels des erstellten **Paper Prototypes**. Fortlaufende **Usability Tests** zusammen mit dem Endbenutzer ermöglichen es, eine dem Benutzer möglichst zugeschnittene Applikation zu erschaffen.

Tabelle 2 Vorgehen nach Goodwin auf Garrets 5 Schichten Model angewendet

[3] (Hochschule für Technik Rapperswil, Prof. Dr. Stolze, 2010)

Die folgende Grafik beschreibt den Kontext des Usability Testing zusammen mit der Entwicklungsmatrix nach Garret:

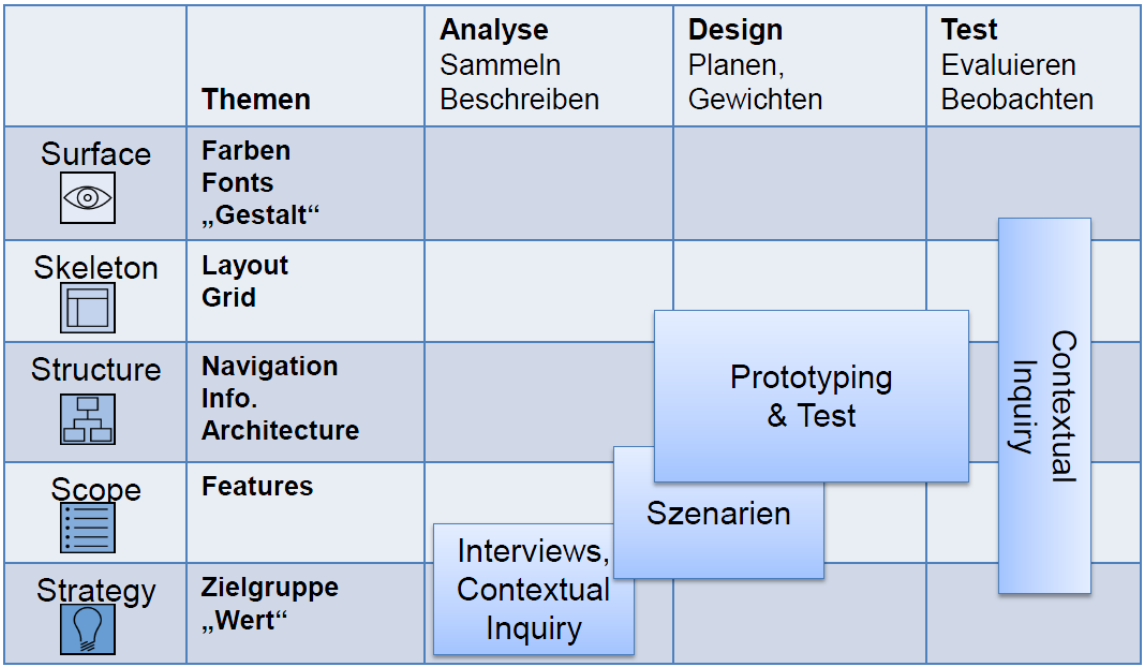


Abbildung 2: The Elements of User Experience nach Prof. Dr. M. Stolze[4]

Die Entwicklung sowie die Validation des ScrumTable User Interfaces wurde nach dem oben definierten Schema durchgeführt. Dies ermöglicht eine genaue Aufnahme der Zielgruppe sowie eine möglichst auf die Zielgruppe zugeschnittene Applikation.

Ein grosser Teil des oben beschriebenen Vorgehens wurde für die Bachelorarbeit im Zusammenhang mit dem Modul User Interface 2, welches an der HSR Rapperswil von Prof. Dr. Markus Stolze unterrichtet wird, erstellt. In diesem Rahmen erhielt das Projektteam tatkräftige Unterstützung durch Andreas Zollinger.

[4] (Hochschule für Technik Rapperswil, Prof. Dr. Stolze, 2010)

3 Strategy und Scope

Im Rahmen der Strategy Phase wurde die Zielgruppe für ScrumTable definiert. Der erste Schritt in diese Richtung gestaltete sich im Erfassen von hypothetischen Personas und Szenarios. Dies diente vor allem für eine erste Betrachtung der jeweiligen Mustereigenschaften bzw. der Behavioural Variables. Ein weiterer positiver Aspekt dieser Vorgehensweise besteht darin, dass man sich intensiv mit der gewünschten Zielgruppe auseinander setzt. Da die Personas nicht real sind, haben wir auf ein Bild, welches eine reale Persona suggerieren könnte, verzichtet.

3.1 Hypothetische Personas

3.1.1 Macro Maestro

Macro Maestro

42 Jahre

Dipl. Ing. FH

Familienvater von 3 Kindern (12, 15, 17)

Seit 4 Jahren als Scrum Master tätig

Guten Kenntnisse im Managen vom Projekten

Absolvierte mehrere Weiterbildungen im Bereich Agile Software-Entwicklung

Beherrscht Scrum

Macro leitet seit mehreren Jahren Projekte. Er koordiniert seine Mitarbeiter mit grosser Vorliebe und delegiert, erfasst und priorisiert Aufgaben für seine Teammitglieder. Die ständige Kontrolle des Projektes und des Projektfortschrittes ist eine seiner grossen Aufgaben.

Das Scrum Meeting findet jeden Tag am Morgen, sowie wöchentlich an einem runden Tisch in einem abgeschiedenen Raum statt. Dabei werden die diversen Tasks besprochen und zusammen mit dem gesamten Team genauer analysiert. Falls erforderlich, können die Prioritäten sowie Risiken der Tasks angepasst werden. Dafür ist jedes Teammitglied zum Mitdenken aufgefordert, da die Fachkompetenzen bei den einzelnen Teammitgliedern liegen.
Macro verfügt über einen Laptop, auf welchem er die Tasks erfasst hat und diese nun mutiert. Zusätzlich hat er am eigenen Arbeitsplatz eine Dockingstation, mit welcher er auf das Netzwerk zugreifen kann.

Für Macro ist der reibungslose Projektablauf sehr wichtig. Er ist sehr ehrgeizig und möchte jedes noch so schwierige Projekt in einem angenehmen Massstab durchbringen.

Durch die diversen Weiterbildungen im Fachbereich Agile Software-Entwicklung hat sich Macro viel Wissen über Scrum erarbeitet. Dieses setzt er auch tatkräftig in die Praxis um. Neuerungen geht er nicht aus dem Weg und versucht jeweils die positiven Aspekte einer neuen Technologie gleich anzuwenden.

Macro ist nicht glücklich über die Art, wie er mit seinem Team die Meetings durchführen muss. Seine Teammitglieder möchten gerne interaktiv Einfluss nehmen können auf die Gestaltung der Prioritäten, Risiken und auf die verschiedenen Backlogs. Auch sehen während des Meetings nicht alle Teammitglieder auf seinen Laptop, was die Zusammenarbeit und das Verständnis häufig erschwert.
Nach dem Meeting ärgert sich Macro über das Abgleichen der besprochenen Tasks & Backlogs zurück auf das gemeinsame Projektportal. Wie schnell doch etwas vergessen geht und dann wieder zu Missverständnissen und unnötigen Diskussionen führt!

3.1.2 Debian Develo

Debian Develo

24 Jahre

Informatiklehre Applikationsentwickler

Verheiratet und Vater eines Kindes (1)

Seit 2 Jahren als Software Test Entwickler tätig

Sehr gute Kenntnisse im SW Bereich

Ist mit Scrum ein wenig vertraut

Debian arbeitet in einem sechs köpfigen Team direkt unter seinem Teamchef. In diesem Team ist er verantwortlich für die Erstellung, Prüfung und Wartung von Software Tests.

Gearbeitet wird dabei in einem grossen Raum, welcher Debian sich mit zwei weiteren Arbeitskollegen teilt. Die anderen drei Teammitarbeiter sind im benachbarten Raum tätig. Seine Arbeitsstation umfasst einen Platz für einen Rechner mit zwei Bildschirmen sowie einen Platz für Papierarbeiten. Jedes Teammitglied besitzt ein eigenes stationäres Telefon. Besprechungen im Team werden in einem ebenfalls anliegenden Konferenzraum abgehandelt.

Debian organisiert alle Unterlagen nach den Vorgaben der Firma und besitzt meist einen aufgeräumten Arbeitsplatz. Seine technische Arbeit erledigt er immer korrekt und mit vollstem Elan.
Scrum kennt Debian erst seit diese Technik vor kurzem in diesem Team für eine neues Projekt eingeführt wurde. Die Grundlegenden Fakten zum Thema Scrum hat er schnell verstanden und kann sich selber dabei auch schon einbringen.

Am meisten ärgert sich Debian über grossen Papierkrieg. Wenn während einer Besprechung jemand Protokoll führt und dies am Whiteboard macht, so muss dies einer später abschrieben damit es protokolliert ist. Wird das Protokoll direkt elektronisch mitgeschrieben, haben die anderen Teammitglieder nur beschränkt Sicht auf das Erfasste.

3.2 Hypothetische Szenarios

3.2.1 Macro Maestro: Sprint-Planing User Stories in Tasks aufteilen

Macro steht mit seinem Team vor dem Multitouch-Tisch. Die Prioritäten der einzelnen User Stories hat er mit dem Team bereits vergeben. Nun teilt zusammen mit dem Team die User Stories in Tasks auf. Für jeden Task soll eine fachkundige Person eine Zeit schätzen, oder zumindest mit dem Schätzen bzw. mit dem Bearbeiten beauftragt werden.

Das Team kann auf dem Tisch auch erkennen, ob eine Person bereits über- oder unterbelastet ist. Dies hilft Macro beim Planen und dem Team beim Abarbeiten der Tasks.

3.2.2 Macro Maestro: Sprint-Planing Interruptions erfassen

Macro hört während des Meetings, dass ein Mitglied seines Teams nächste Woche Ferien hat. Ups! Hätte er dies doch beinahe vergessen, oder hat das Mail von der Personalabteilung verloren? Sofort trägt er auf dem Tisch unter Interruptions die Abwesenheit bzw. Ferien seines Mitarbeiters nach.

Leider bemerkt er danach, dass nun ein anderer Mitarbeiter, welcher nur 80% arbeitet, in der nächsten Woche überbelastet ist.

3.2.3 Macro Maestro: Sprint-Planing Kapazitäten anzeigen

Im Team von Macro gelten die generellen Regeln beschrieben im Agilen Manifest. Jeder Mitarbeiter soll also „energized“ Arbeiten können. Überstunden sollen nur die Ausnahme bilden, da die Mitarbeiter dadurch ausgelaugt und demotiviert werden. Daher soll der Multitouch-Tisch eine Übersicht über die reservierten und offenen Kapazitäten geben.

In dieser Übersicht sieht Macro seine Mitarbeiter mit den entsprechenden Projekten und Kapazitäten.

3.2.4 Macro Maestro: Daily Scrum Meeting durchführen

Das 15-minütige Daily Scrum Meeting leitet Micro mit grossem Elan. Während des Scrum Meetings berichtet jeder Mitarbeiter über den Status seiner gegenwärtigen Arbeiten. Diese Status trägt Micro auf den einzelnen Tasks nach, falls diese die Mitarbeiter noch nicht eingetragen haben. Das Burndown Chart zeigt dem Team den Fortschritt in einer grafischen Art und Weise.

3.2.5 Debian Develloo: User Stories priorisieren und Story Points vergeben

Debian sitzt in einem Project Planing Meeting. Der letzte Sprint wurde gerade abgeschlossen und nun wird der Product Backlog neu betrachtet. Im Raum sind der Scrum Master, der Product Owner und die anderen Entwickler anwesend. Er weiss, was er im letzten Sprint abschliessen konnte und was noch offene Punkte sind.

Sobald der Scrum Master sich am Tisch vorbereitet hat, meldet jeder seinen Fortschritt. Dieser wird auf dem Tisch erfasst. Nach dieser Runde werden neuen Userstories erfasst, welche in der Zwischenzeit dazugekommen sind. Die einzelnen User Stories werden im Team priorisiert, indem sie in verschiedene Priorisierungsgefässe verschoben werden. Den Aufwand wird abgeschätzt und dementsprechend werden Story Points vergeben.

3.2.6 Debian Develloo: User Stories für folgenden Sprint planen

Debian sitzt in einem Project Planing Meeting. Der letzte Sprint wurde gerade abgeschlossen und nun wird der Product Backlog neu betrachtet. Im Raum sind der Scrum Master, der Product Owner und die anderen Entwickler anwesend. Debian kann gut abschätzen welche Art von Aufgabe einem entsprechenden

Sobald die einzelnen User Stories am Table priorisiert und Story Points vergeben wurden, wird die nächste Iteration geplant. Dazu wechselt der Scrum Master die Ansicht auf die Iterationsplanung. Alle noch abzuschliessenden User Stories werden angezeigt. Das Team beschliesst zusammen, welche User Stories im nächsten Sprint abgearbeitet werden sollten. Dank der Ansicht der Auslastung merkt der Scrum Master, dass einer der Entwickler zu fest ausgelastet wird und Debian im Moment nur sehr wenig Arbeit für den kommenden Sprint zugewiesen bekommen hat. Sie beschliessen zusammen das Debian einen Auftrag des Kollegen übernimmt. Dadurch sind alle Mitarbeiter für den folgenden Sprint ideal ausgelastet.

Nachdem die Sitzung geschlossen wurde, weiss Debian was er im nächsten Sprint zu erwarten hat.

3.3 Hypothetische Rollen

Rolle	Beschreibung
Master	Die Master-Rolle entspricht dem ScrumMaster, welcher fürs Leiten eines Projektes verantwortlich ist. Der ScrumMaster ist auch die berechtigte Person fürs Einloggen auf dem Tisch und leitet die jeweiligen Meetings.
Developer	Der Developer ist ein Mitarbeiter des Scrum-Teams. Er nimmt an den Meetings teil. Die Fachkompetenz liegt bei ihm.
Owner	Diese Rolle, für den Scrum zwar benötigt, ist für ScrumTable irrelevant.

3.4 Hypothetische Kontexte

Kontext (Informatik-Ausrüstung)	Beschreibung
Minimal	Die Entwickler der Software haben einen Rechner, auf welchem sie arbeiten. Im restlichen Bereich des Unternehmens hat meist nur Abteilungschef einen Rechner. Einen gemeinsamen Team-Server wurde erst nach langen Überredungskünsten zugelegt.
Casual	Jeder Entwickler besitzt einen Rechner mit zwei Bildschirmen. Ein gemeinsamer Server ist Pflicht, sowie auch die Einhaltung der Richtlinien im Bezug auf Datensicherheit. In jedem Arbeitsraum sind verschiedene Möglichkeiten vorhanden um das weitere Vorkommen zu besprechen, wie zum Beispiel Whiteboards.
High-End	Jeder Entwickler besitzt mindestens einen Rechner mit zwei Bildschirmen. In der Regel besitzen sie auch ein Notebook welches einfach synchronisiert werden kann. Auch ein Smartphone/Blackberry ist keine Seltenheit. Gemeinsame Server sind „State of the Art“. Richtlinien sind kein Thema und werden von jedem Wahrgenommen und eingehalten. In jedem Raum (auch Gänge, Mensa, Toilette) hat es digitale Whiteboards wo sich ein Mitarbeiter schnell und einfach mit einem Tag anmelden und Kurznotizen erfassen kann.

3.5 Behavioural Variables

Die Personas können nach folgenden Eigenschaften und Kriterien eingeordnet werden:

Eigenschaft	Attribute für Skala	
Führungsqualität	gut	schlecht
Motivation	viel	wenig
Entscheidungsbefugnis	viel	wenig
Kontrollfreudig	organisiert	chaotisch
Fachwissen: Scrum	viel	wenig
Fachwissen: Technik	viel	wenig
Sozialkompetenz	viel	wenig
Sozialverhalten	extrovertiert	introvertiert
Geduld	viel	wenig
Innovationsfreude	viel	wenig
Pedanterie	sehr genau	ungenau

Diese Eigenschaftsbetrachtung wurde aufgrund folgender Kriterien durchgeführt. Diese sollen vor allem eruieren, welche Einstellungen und Eigenschaften eine Persona gegenüber Scrum und ScrumTable hat.

Eigenschaft	Eigenschaft
Führungsqualität	Die Führungsqualität soll vor allem bei Projektleitern vorhanden sein. Sie müssen die Projektmeetings (Scrum Meetings) am ScrumTable leiten.
Motivation	Motivation ist in allen Geschäftsbereichen eine sehr wichtige Eigenschaft. Allerdings ist bei den Führungspersonen eine erhöhte Motivationsbereitschaft zu erwarten, da diese die Mitarbeiter zu höchster Effizienz während der Arbeit und den Sitzungen anspornen müssen.
Entscheidungsbefugnis	Die Entscheidungsbefugnis ist vor allem für Projektleiter relevant. So können Personen mit mehr Entscheidungsbefugnissen an Scrum Meetings mehr Einfluss nehmen.
Kontrollfreudig	Diese Eigenschaft soll eruieren, ob eine Person eher direkt auf ein Ziel zugeht oder vermehrt den chaotischen Ansatz wählt.
Fachwissen: Scrum	Das Scrum-Fachwissen ist für die Verwendung und das Verständnis von ScrumTable wichtig. Deshalb soll diese Eigenschaft eruieren, wie tiefgehend die Hilfestellungen sein müssen.
Fachwissen: Technik	Die Domain von Scrum-Table ist sehr technisch. Daher ist das Fachwissen in der Technik erforderlich.
Sozialkompetenz	Die Sozialkompetenz ist für die Bedienung eines Multitouch-Systems wie ScrumTable vor allem bei der Bedienung erforderlich.
Sozialverhalten	Bei Scrum Meetings ist es wichtig, die Leute mit eher introvertierteren Haltungen zu fördern, oder extrovertierte Leute zu bremsen. Ansonsten besteht die Gefahr, dass die Scrum Meetings nichts nützen, oder zu lange dauern.
Geduld	Vor allem Projektleiter benötigen in Scrum viel Geduld, da bei Scrum gewisse Meetings zeitlich begrenzt sind und sich alle

	Mitarbeiter an einen zeitlichen Rahmen halten müssen.
Innovationsfreude	Um alte Prozesse durch neue, innovative Agile Entwicklungsprozesse zu ersetzen, benötigen die Mitarbeiter für ScrumTable und Scrum Innovationsbereitschaft.
Pedanterie	Die Effizienz im Scrum kann leiden, falls die Personen zu pedantisch auf strukturelles anfertigen von Dokumenten pochen.

3.6 Interview Matrix

	Master	Developer	(Owner)	Total Kontexte
Minimal	0	0	0	0
Casual	0	2	0	1
High-End	1	0	0	1
Total Rollen	1	1	0	2

3.7 Interviews

Die Interviews wurden mit externen und internen Personen der HSR Rapperswil durchgeführt. Die Befragungen sind mit den Schlussfolgerungen in einem Separaten Teil unter Interviews in dieser Dokumentation abgelegt. Auch wurde in den Interviews klar, wie die Momentanen Ist-Szenarien aussehen und in welche Richtung die Soll-Szenarien gehen müssen. Die Szenarien sind in den Anforderungsspezifikationen abgelegt. Somit wurde die Scope Phase nach Garret abgedeckt.

Durch die Interviews konnten die hypothetischen Personas fundiert überprüft werden. Daraus entstanden die konkreten Personas, welche in den Anforderungsspezifikationen niedergeschrieben sind.

3.7.1 Neue Erkenntnisse: Behavioural Variables

Die Interviews an sich haben folgende, zusätzliche Behavioural zum Vorschein gebracht:

Eigenschaft	Attribute für Skala	
Durchführen der Scrum-Meetings	spontan	regelmässig
Infrastruktur vorhanden	ja	nein
Agilität (agiles Vorgehen erwünscht)	ja	nein
Teamgrössen Optimum	klein	gross
Kommunikation Fähigkeit	gut	schlecht
Arbeitstrennung	allrounder	sehr spezialisiert

Wobei diese Eigenschaftsbetrachtung wieder aufgrund folgender Kriterien durchgeführt wurde:

Eigenschaft	Eigenschaft
Durchführen der Scrum-Meetings	Die Meetings werden nicht immer und an jedem Tag gleichzeitig durchgeführt. Dies ist allerdings für Scrum erforderlich.
Infrastruktur vorhanden	Ein Multi-Touch Tisch und Team Foundation Server (TFS) muss zur Verfügung stehen. Falls die Persona über keinen solchen Tisch verfügt, kann ScrumTable nicht verwendet werden.
Agilität (agiles Vorgehen erwünscht)	Der Entwicklungsprozess, welcher von einer Persona im Team eingesetzt wird, kann beispielsweise agil sein. Je nach Vorlieben, Erfahrungen und Vorschriften können nicht agile Prozesse eingesetzt werden.
Teamgrössen Optimum	Scrum Teams sollten nicht zu gross (> 8 Personen) sein, da ansonsten die Meetings nicht mehr effizient durchgeführt werden können.
Kommunikation Fähigkeit	Die Fähigkeit im Team zu kommunizieren und sich gegenseitig abzustimmen ist in Scrum sehr wichtig.
Arbeitstrennung	Scrum Mitglieder sollten wenn möglich Allrounder sein. Scrum sieht vor, dass jedes Mitglied zu einer bestimmten Anforderung eine Schätzung abgeben kann.

3.7.2 Behavioural Diagram

Aufgrund der folgenden Behaviour Patterns wurden die Personas verfeinert:

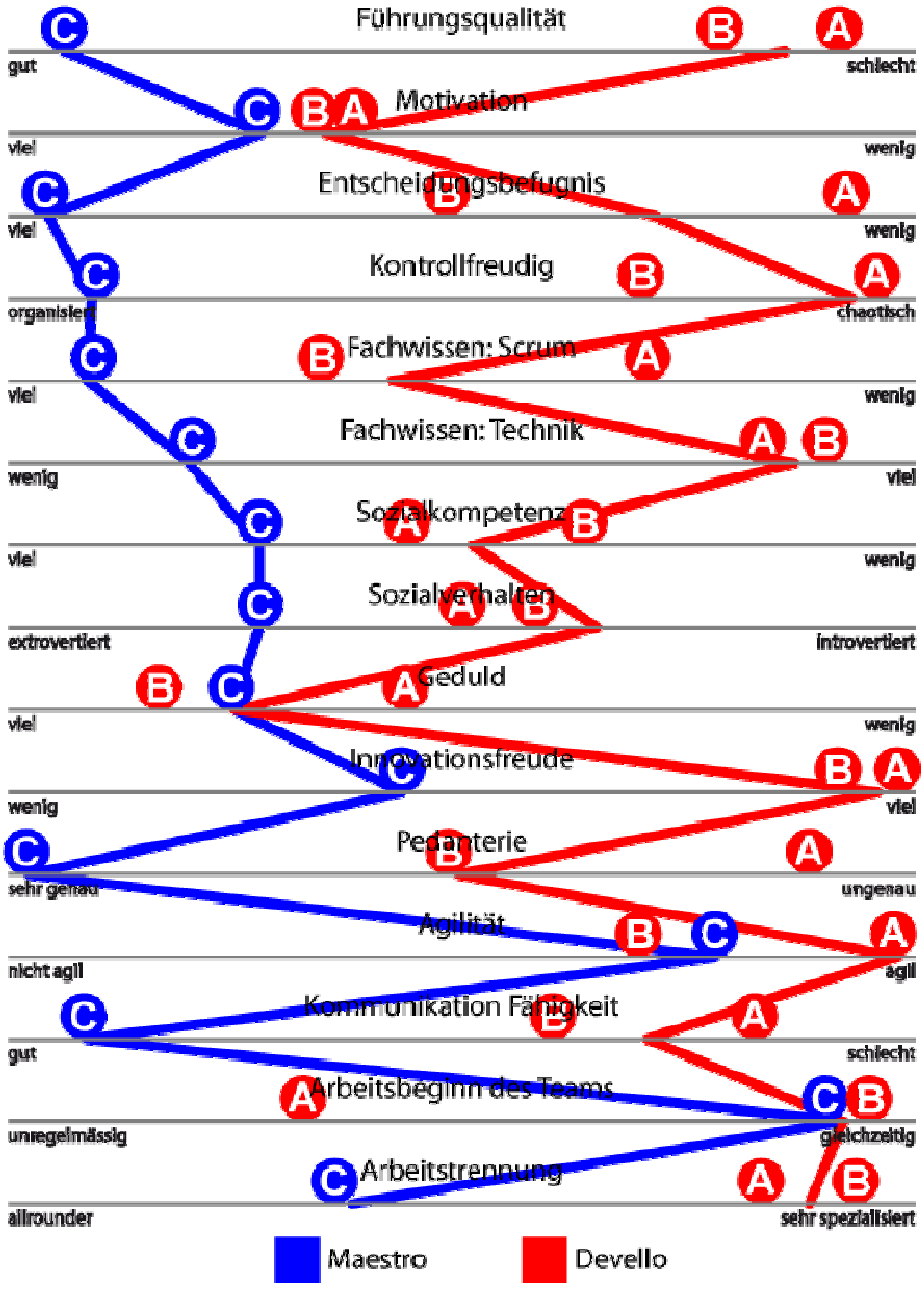


Abbildung 3 Verhaltensmuster der Interviewpartner

Das Bild Verhaltensmuster der Interviewpartner zeigt folgendes: Für ScrumTable wurden Interviews mit drei Personen geführt. Zwei Personen konnten der Zielgruppe (Persona) Devello zugewiesen werden, eine Person der Persona Maestro. Die durchgezogenen Linien zeigen eine Interpretation der typischen Eigenschaften (siehe Behavioural Variables) einer Persona in Zusammenhang mit ScrumTable.

Folgende Entscheide führten zum Diagramm in Abbildung 3 oberhalb. Vielfach wurde auch ein Wert in die Grafik interpretiert, da oft eine fundierte Auswertung teils eine zusätzlich psychologische Untersuchung erfordern würde.

Eigenschaft	Erkenntnis aus Interviews	
	Maestro	**Devello**
Führungsqualität	Leitet die Meetings und versucht, das Team im Griff zu haben.	Lässt sich vom Team-Leiter die Arbeit zuweisen.
Motivation	Viele Überstunden zeugen von der Arbeitsbereitschaft.	Die Motivation für technische Herausforderungen und Erneuerungen ist jederzeit da.
Entscheidungsbefugnis	Der Teamleiter beschreibt die Arbeit und klärt mit dem Kunden die wirklich wichtigen Arbeiten ab.	Erhält die Aufgaben vom Team-Leiter.
Kontrollfreudig	Leitet die Meetings und versucht möglichst geordnet die Meetings zu beenden.	Sieht überall Lösungsmöglichkeiten. Denkt sehr vernetzt und reisst entsprechend alle Arbeiten gleichzeitig an.
Fachwissen: Scrum	Ist als Scrum Master zertifiziert.	Muss sich zuerst noch in Scrum einarbeiten, was allerdings kein Problem darstellt.
Fachwissen: Technik	Das technische Verständnis ist von der Hochschule her gegeben, hat allerdings keine Technologie-Kenntnisse mehr.	Arbeitet täglich mit modernen Technologien.
Sozialkompetenz	Lässt andere Leute ausreden und versucht, die beste Lösung aus allen Meinungen zu eruieren.	Seine Meinung ist normalerweise die technisch genauste und daher richtige.
Sozialverhalten	Geht offen auf die Leute zu und ist sich an Kundenkontakt gewöhnt.	Muss zuerst nach seiner Meinung gefragt werden.
Geduld	Weist die Team-Mitglieder oft an, die zeitlichen Richtlinien betreffend der Meetings einzuhalten.	Hält sich nicht immer an die Scrum Abmachungen. Vor allem Themen aus seinem Fachbereich bespricht er gerne sehr ausführlich.
Innovationsfreude	Möchte bestehende Entwicklungsprozesse sind nach wie vor unterstützen	Arbeitet gerne immer sofort mit den neusten Technologien und stellt sich gerne neuen Herausforderungen.
Pedanterie	Möchte das Projekt Management für die Software möglichst genau und sauber gestalten.	Projekt Management ist für ihn keine Kernaufgabe. Der Entwickler soll sich auf seine Fachgebiete konzentrieren können.
Agilität	Agile Prozesse sind ihm wichtig.	Agile Software-Entwicklung ist

	Nur so kann das Team auf neue Anforderungen schnell reagieren.	im Trend. Er möchte wenn möglichst dies ausprobieren und ist offen für neues.
Kommunikation Fähigkeit	Kommuniziert offen die für ein Projekt relevanten Daten.	Häufig geht die Kommunikation im Projektstress unter. Und auch Dokumentationen schreiben findet er nicht ansprechend.
Arbeitstrennung	Ist sehr breit abgestützt und hat den Überblick über die Technologien.	Möchte sich je länger je tiefer in eine spezielle Technologie einarbeiten. Er ist der Spezialist!
Durchführen der Scrum-Meetings	Möchte, dass das Team jeden Morgen zu selben Zeitpunkt mit Arbeiten beginnt. Nur so kann der Scrum Prozess eingehalten werden.	Möchte am liebsten jeden Morgen um eine andere Zeit mit Arbeiten beginnen, da er abends auch häufig länger im Büro ist.

4 Structure & Skeleton & Surface

Aus den Sezenarien wurde für die beiden Personas einen Prototypen erstellt. Da beide Personas an den Meetings gleichermassen teilnehmen und über keine speziellen Funktionen verfügen, gestalteten wir nur eine Oberfläche.

4.1 Paperprototyp

Der Paperprototyp mit den Learnings aus den verschiedenen Usability Tests sind im Teil II - SW-Projektdokumentation im Kapitel Paper Prototype zu finden.

Anfänglich wurde der Paperprototype mit den in der Zielgruppe erfassten Endkunden validiert. Auf die Navigation innerhalb der Applikation und die User Interface Architecture wurde besonders geachtet. Auch wurde zu diesem Zeitpunkt das Grid der Applikation gestaltet, welches wie folgt aussieht:

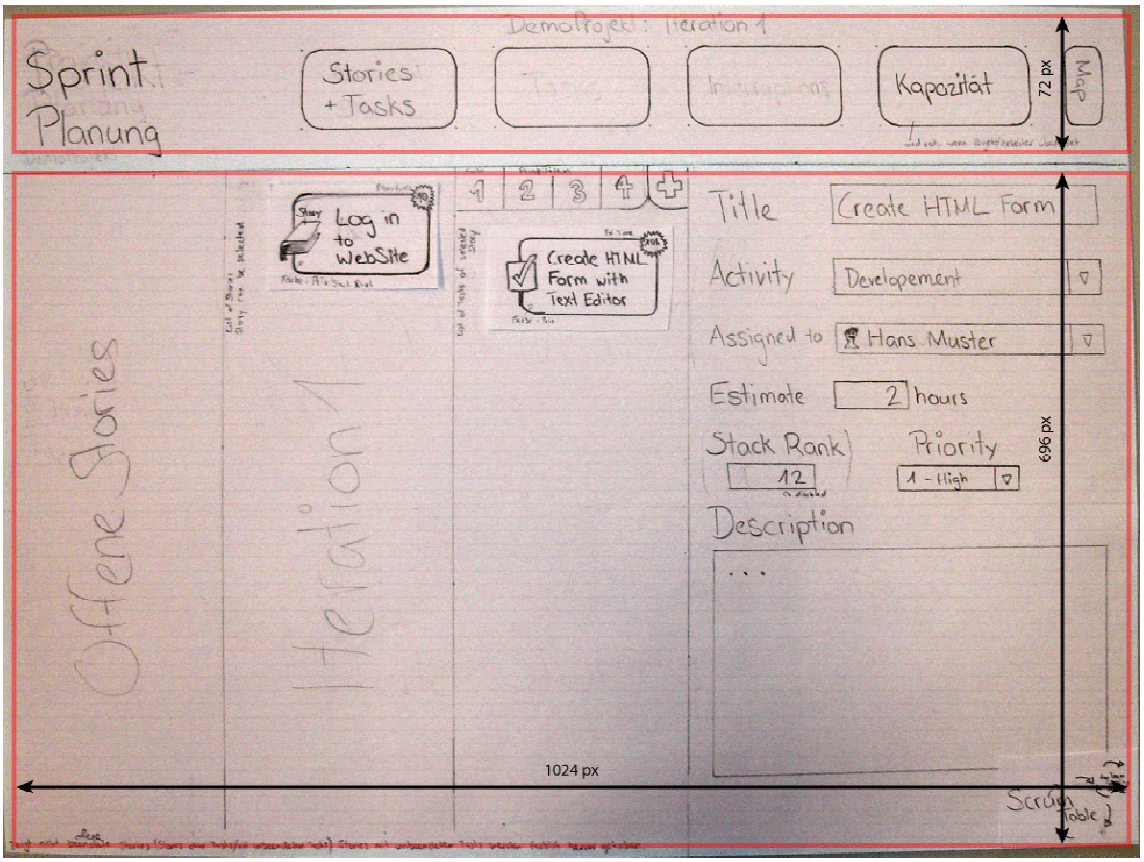


Abbildung 4 Visual Disign (Grid) von ScrumTable am Beispiel des Sprint Planning Screens

Das Grid beschränkt sich auf einen oberen und auf einen unteren Teil der Applikation. Im oberen Teil wird die Navigation dargestellt, im unteren Bereich die Inhalten zu entsprechend ausgewählten Navigationspunkt.

Das Grid zieht sich durch die gesamte Applikation. Es gibt allerdings zwei Ausnahmen (Log-In Screen und ScrumPoker), wo auf den oberen Teil aufgrund von Gestaltungs- und Platzvorteilen verzichtet wird.

4.2 Usability Tests & Mockups

Der Usability Test fand mit einem realen potentiellen Endkunden, welcher auch bereits für die Anforderungs-Interviews zur Verfügung stand, statt. Der gesamte Test wurde für spätere Analysen auf Video aufgezeichnet.

Beim erstellen der Test Szenarien wurde darauf geachtet, dass nur Funktionalitäten mit Business-Value überprüft werden. Für die Test Szenarien wurden absichtlich die Aufgaben so gestellt, wie die Anwender diese im realen Leben erhalten und nun mit dem Program lösen sollen. Die Lösungswege mussten die Anwender also selbst finden.

Die Aufzeichnungen und Schlussfolgerung befinden sich in einer separaten Stelle in dieser Dokumentation. Mehr Informationen finden Sie also im Teil II – SW-Projektdokumentation im Kapitel Interviews.

4.3 Farben und Gestaltung

In der letzten Phase (Surface) der ScrumTable Entwicklung wurden die Farben und Bilder sowie Icons dem Design hinzugefügt. Diese sollen in sich konsistent sein, so dass die externe Konsistenz der Applikation gegeben ist.

So wurden beispielsweise alle verschiebbaren Elemente mit einem leichten Schatten hinterlegt. Das ausgewählte und verschiebbare Element sollte speziell hervorgehoben werden und erhielt einen tieferen Schatten.

Abbildung 5 Aktiviertes Element mit Schatten

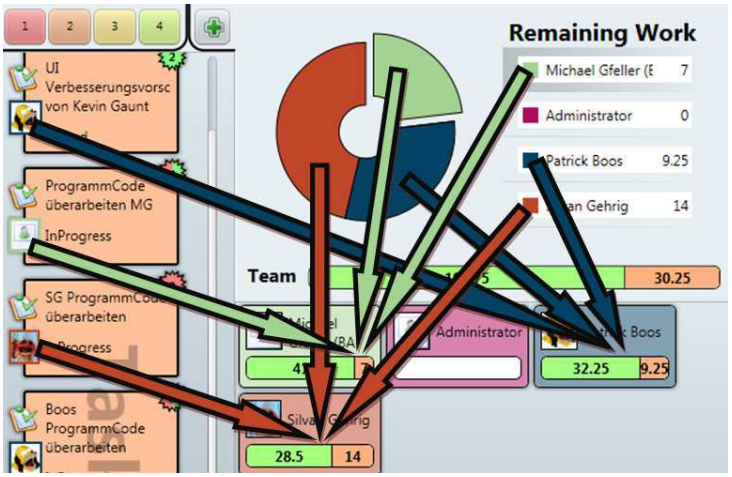
Abbildung 6 Button mit schwarzem Rand

Die Buttons für die Navigation wurden mit einem schwarzen Rand markiert.

Jeder Button, bis auf einige sehr wenige Ausnahmen, wurden mit einem Icon versehen, welche den Benutzer grafisch in der Navigation unterstützt.

Die farbliche Gestaltung von ScrumTable soll das User Interface abrunden und dem Endkunde das Gefühl geben, eine aktuelle und leistungsfähige Applikation zu benutzen.

Jedem Entwickler wird eine eigene Farbe zugewiesen, die durch die ganze Applikation gleich bleibt. Sie kommt an verschiedenen Stellen zum Einsatz. In dem Bild sieht man die Sprint Planung, wo man Tasks den Entwicklern zuweist. Man sieht die Farbe des Entwicklers an allen Stellen, wo dieser Angesprochen ist.



User Stories und Tasks sind vom Aufbau gleich, da sie vom Typ her ähnlich sind. Sie unterscheiden sich durch das Symbol Links. So zeigt ein Buch eine Geschichte an und eine Mappe mit Check-Haken ein Task. Der Restliche Aufbau ist ähnlich. Erst etwas unklar, aber durch Benutzung wird dies schnell und einfach klar.

So wird rechts oben entweder die Story Points oder die geschätzte Zeit angezeigt und als Pie Chart, wie viel bereits abgearbeitet wurde.

Abbildung 8 Task und User Story in ScrumTable

Beschreibung von „Behältern“ oder einem Status ist konsequent wie rechts im Bild gemacht. Dies damit der kleine vorhandene Platz auf dem Surface Tisch nicht noch weiter verkleinert.

Damit der Text sichtbar ist, und trotzdem nicht zu stark stört, wird dieser halb-transparent vor die Items gelegt. Dadurch kann man zu Beginn der Nutzung den Text lesen um zu wissen, was die Behälter darstellen. Wenn man die Texte kennt, bemerkt man zwar, dass sie da sind, aber sie stören nicht in der Benutzung.

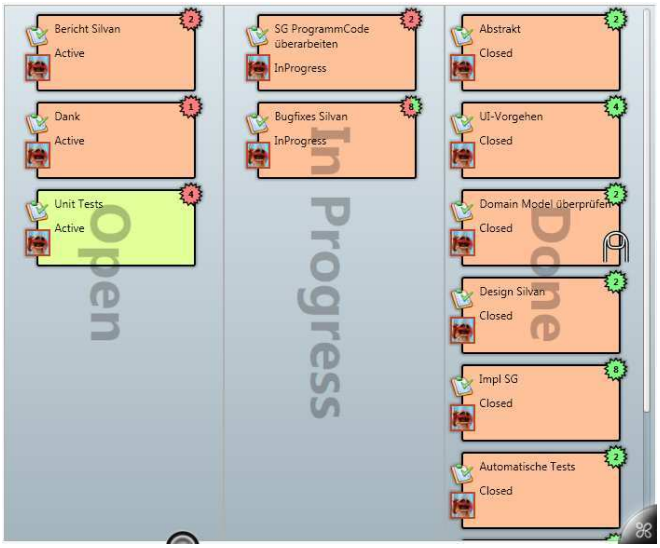


Abbildung 9 Beschriftung der Behälter in ScrumTable

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Interviews

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Ziel dieses Dokument ist es, die Eigenschaften und Spezialitäten der Benutzer durch ein Interview zu erfassen. Entwickler, welche im Nachhinein zum Projekt stossen, sollen durch diese Dokumentation über die bereits errungene Erfahrung über die Benutzer informiert werden.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

2 Interview-Fragen

Dieses Kapitel dient zur Vorlage für ein Interview.

2.1 Einführung

Guten Tag, unser Name ist Patrick Boos, Michael Gfeller und Silvan Gehrig. Wir entwickeln zurzeit in unserer Bachelorarbeit ein Programm für den Microsoft Surface Tisch, welches Scrum Meetings vereinfachen soll. Deswegen freut es uns sehr, dass Sie uns erlaubt haben, Sie zu interviewen und mit Ihnen in diesem Rahmen konkrete Fälle durchzusprechen, in denen ein solches System womöglich von Nutzen sein könnte. Nachdem wir die grundlegenden Daten erfasst haben, möchten wir mit Ihnen möglichst detaillierte und konkrete Fälle (Szenarios) durchsprechen. Da Sie genau unserem typischen User entsprechen, wäre es ideal, wenn Sie während den Tätigkeiten die jeweiligen Schritte erklären würden und welche Dinge die Aufgabe erschweren oder vereinfachen. Wir werden dieses Gespräch aufzeichnen und zur gegebenen Zeit wieder löschen. Ihre Vertraulichkeit ist zu jeder Zeit gegeben. Zur besseren Verständlichkeit werden wir Sie möglicherweise unterbrechen, um den Prozess besser zu verstehen. Ansonsten lassen Sie sich bitte nicht von unserer Anwesenheit irritieren.

2.2 Zur Person

- Wie heissen Sie und welche Position haben Sie bei Ihrer Tätigkeit?
- Welchem Aufgabengebiet gehört ihre Tätigkeit an?
- Welche Informationen werden benötigt für Ihre Arbeit?
- Welche Arbeitsumgebung und Werkzeuge benutzen sie in Ihrer Tätigkeit?
- Wie kommunizieren Sie am liebsten? (Face-to-Face, E-Mail, Telefon, Notiz)
- Können Sie kurz Ihren Werdegang schildern?
- Was denken Sie, wie Erfolg gemessen werden kann?
- Was hindert den Erfolg, wie geht unnötige Zeit verloren?
- Wo möchten sie in Zukunft hingelangen?
- Wie sehen die die Trends in der Softwareentwicklung?
- Welche Erfahrung im Softwarebereich hat sie in letzter Zeit besonders positiv überrascht.
- Und was haben Sie eher negativ in Erinnerung?

2.3 Beobachtung

- Welche Einstellung haben Sie zu Scrum?
- Wie viele Scrum Mitglieder hat ihre Gruppe und welche Art von Personen umfasst diese?
- Haben Sie bereits eine Scrum-Unterstützung?
    - Welches Produkt bietet Ihnen eine Lösung?
    - Wo sehen Sie Optimierungsbedarf an Ihrer momentanen Scrum Lösung?
    - Welche weiteren Scrum-Unterstützungen sind Ihnen bekannt?

2.3.1 Projekt-Planung

- Wie führen sie das Project Planning durch? (In Bezug auf Product Backlog und Priorisierung)
    - Falls Sie eine Scrum- Unterstützung verwenden: Wie geben Sie die Daten entsprechend ein?
- Eine neue User Story soll erfasst werden. Wie geht Ihr Team vor?
    - Falls Sie eine Scrum-Unterstützung verwenden: Wie geben Sie die Daten entsprechend ein?
- Infrastruktur/Architektur der Applikation in welche User Story?
    - Erste User Story einfach riesig, oder dafür eigene User Story? Wie wird dies gemacht?
- Wann teilen Sie User Stories in Tasks auf?
    - Kommt es vor, dass Sie dies auch nicht in der Projekt-Planung machen?
- Welche Bedeutung haben Story Points für Sie?
    - Wann weisen Sie diese zu?
    - Wie wählen Sie die Story Points? Welche Überlegungen?
        - Kennen Sie Scrum Poker?
            - Benutzen Sie Scrum Poker?
            - Benutzen Sie Scrum Poker während Projekt-Planung oder Sprint-Planung?
            - Was sind Ihre Erfahrungen mit Scrum Poker?
                - Welche Probleme tauchten damit auf?
                - Was waren die Vorteile?
                - Teilen Sie bei dieser Diskussion die User Stories auf?
- Welche Bedeutung hat der Stack Rank für Sie?
    - Wie weisen sie den User Stories die Stack Ranks zu?
    - Benutzen Sie dazu eine spezielle Visualisierungstechnik?
- Räumen Sie das Story-Board für die Projekt-Planung nach dem Meeting wieder ab?
    - Wäre es sinnvoll das Story-Board für die Teams stehen zu lassen, damit die Team-Mitarbeiter sich daran orientieren können?

2.3.2 Sprint-Planung

- Wie planen sie die Iterationen & Sprints?
    - Falls Sie eine Scrum-Unterstüztung verwenden: Wie geben Sie die Daten entsprechend ein?
- Wann weisen Sie User Stories den Iterationen zu?
    - Kommt es vor, dass Sie dies auch zu einem anderen Zeitpunkt machen?

2.3.3 Daily Sprint

- Wie sieht Daily Scrum Meeting aus?
- Wie verwenden Sie ein Burndown Chart?

2.4 Abschluss

- Verfügen sie über ein Microsoft Team Foundation Server 2008?
- Arbeiten Sie mit Microsoft Technologien? (Spezifisch TFS)

- Besitzen Sie einen Scrum Tisch oder haben Sie vor einen anzuschaffen?

Ich bedanke mich herzlich bei Ihnen für Ihre Zeit und wünsche Ihnen noch einen schönen Tag.

3 Interview Log 1

3.1 Interview

Vertraulich	Ja, garantiert
Lokalität	HSR
Aufnahmetechnik	Audio & Mitschreiben
Länge	60 Minuten

3.2 Interviewpartner

Hauptbefrager	Silvan Gehrig
Schreiber	Patrick Boos

3.3 User Profile

Funktion	Developer (Team Member)

3.4 Erkenntnisse

3.4.1 Zur Person und Gedanken zu Scrum

Hat bereits in einem Projekt teilgenommen, welches unter Verwendung von Scrum geführt wurde.

Welche Informationen sind wichtig

- Hauptsächlich Burndown Chart und Task-Board (Guter Überblick)
- Wiki und Trac war auch sehr nützlich

Verwendete Werkzeuge

- Excel: Product Backlog und Sprint Planning
  - Für diese Teamgrösse war dies ganz ok.
  - Wurde für Meeting jeweils für jeden ausgedruckt
- Story-Board mit Burndown Chart
- Planning Poker mit Karten
- Wiki: User Stories, Dokumente und Punkte von Retrospektive
- Daily Builds (Hudson Server) mit Report wenn etwas fehlerhaftes eingecheckt wurde
- Trac

Bevorzugte Kommunikationsart

Daily Scrum ohne Product Owner. Ansonsten E-Mail, da Leute etwas verteilt waren. Teilweise auch per Instant Messaging.

Gedanken zu Scrum

Erfolgreich war man, wenn man ungefähr richtig geschätzt hat und alle User Stories, welche man geplant hat, erfüllt. Scrum übt generell mehr Druck auf die Entwickler aus, was jedoch auf den Scrum Master drauf ankommt, welcher hier schauen muss, dass der Druck nicht zu gross wird.

Für KMUs ist Scrum auf jeden Fall gut. Für grosse Firmen ist es schwierig den Umstieg auf Scrum zu machen.

Scrum mit eXtreme Programming zusammen ist eine optimale Mischung.

3.4.2 Beobachtung

Excel

Nur der Scrum Master hat das Excel Dokument bearbeitet. Deswegen brauchte es kein super Tool. Schön wäre ein Filtern oder Sortieren. Mit Excel Filter möglich, wurde aber nicht gemacht. Praktisch wäre es gewesen wenn User Story gleich mit einer Skizze oder weiteren Informationen verlinkt wäre. Dies wurde in diesem Fall auf Wiki separat geführt.

Story-Board

Hing immer und war etwas vom wichtigsten. Man musste dadurch jeweils beim ins Zimmer kommen, wie das Projekt steht.

Ist nicht durch ein Desktop Programm ersetzbar. Braucht etwas, das immer hängt und sichtbar ist. Dies wäre auch als LCD denkbar. Praktisch ist natürlich das automatische Generieren des Boards. Noch besser wäre, wenn man auf diesem LCD auch per Touch die Sachen verschieben/editieren kann.

3.4.3 Project Planing

User Stories kamen von Product Owner bereits Priorisiert. Wurden in Project Planning nicht geschätzt. Jedoch dann in Sprint Planning.

User Stories wurden gleich Personen zugeiwesen. Tasks dann jeweils auch diesen Personen.

User Stories wurden bereits am Anfang in etwa auf die geplanten Sprints verteilt.

Stack Rank der User Stories

Ist die Wichtigkeit. Wurde von Product Owner geschätzt und teilweise wieder angepasst. Höchste Nummer = höchste Priorität (1-100).

3.4.4 Sprint Planning

Bei Sprint Planning wurde Product Backlog genommen und daraus die wichtigsten User Stories und diese dann in Tasks unterteilt.

User Storeis wurden gemeinsam geschätzt. 1 Punkt = 1 Arbeitstag. Anfangs war es schwer zu schätzen, wurde dann aber besser. Maximal 5min pro User Story.

3.4.5 Daily Scrum

Wurde immer im Stehen bei der Tafel, wo sich auch das Story-Board mit den Stories/Tasks und Burndown Chart hing, abgehalten. Die Dauer war maximal 10min.

Probleme wurden teilweise nachher in kleinerer Gruppe besprochen oder teilweise auch alle gemeinsam. Aber in dem Meeting wurde nie gross darauf eingegangen. Tasks auf dem Board wurden

auf den neusten Stand gebracht (Zeiten anpassen und verschieben). Nach dem Meeting hat der Scrum Master jeweils das Burndown Chart von Hand nachgetragen.

3.4.6 Review und Retrospektive

Oft wurde aus der Demo vom Product Owner eine neue User Story erkannt, die er möchte. Diese wurde dann im Product Backlog erfasst und im Sprint Planning für den nächsten Sprint eingetragen. Oft machte der Product Owner eine Skizze, wie er sich die User Story vorstellt. Diese wurde dann gescannt und mühsam auf Wiki geladen.

3.4.7 Team Foundation Server

Von Team Foundation Server gehört, aber nicht damit gearbeitet.

3.5 Wichtige Erkenntnisse aus Interview

- Skizze zu User Story hinzufügen
- ScrumPoker wird von Team zu Team zu anderen Zeiten gemacht. Nicht fix während Produkt Planung.
- Story-Board sehr wichtig! Gross und immer präsent.

4 Interview Log 2

4.1 Interview

Vertraulich	Ja, garantiert
Lokalität	Ostschweiz
Aufnahmetechnik	Audio & Mitschreiben
Länge	60 Minuten

4.2 Interviewpartner

Hauptbefrager	Michael Gfeller
Schreiber	Patrick Boos

4.3 User Profile

Funktion	2 Scrum Master

4.4 Erkenntnisse

4.4.1 Zur Person und Gedanken zu Scrum

Haben in der Firma die Einführung von Scrum vorangetrieben und finden, dass Scrum als Agile-Methode im Hype ist. Dieser soll noch eine Weile anhalten und sich dann festigen.

Welche Informationen sind wichtig

- Hauptsächlich das User-Story/Task-Board .
    - Bugs werden auch auf dem Task-Board platziert

Verwendete Werkzeuge

- NetResults Tracker
    - User-Stories
    - Bugs
    - Zusätzliche Informationen werden als Word Files hinterlegt

Bevorzugte Kommunikationsart

Daily Scrum. Product Owner ist eine spezielle Abteilung welche den Kunden vertritt. Ansonsten durch Gespräche, da sich alle Leute im gleichen Raum befinden.

4.4.2 Beobachtung

Produkt Owner

Der Produkt Owner ist eine Abteilung innerhalb der Firma welche die Wünsche aller Kunden sammelt und in den Meetings vertritt.

Burndown Chart

Wird von Hand nachgetragen. Wird nur vom Kunden/Chef verlangt um einen Überblick zu erhalten. Für die Entwickler weniger wichtig. Der Burndown basiert auf User-Story Points.

4.4.3 Project Planing

Der Produkt-Owner erfasst die User-Stories mit einem Rank und Prüfungskriterien. Der Rank ist aufsteigend und einmalig. Einmal pro Woche wird Scrum Poker durchgeführt. Dabei werden nur so viele User Stories geschätzt damit es für 1 Woche reicht. Dabei gilt: 1Tag = 0.6-0.8 Story Points. Es sind alle Team-Mitglieder dabei um den Wissenstransfer zu gewährleisten.

Für kleine User-Stories kann auch via Mail Gepokert werden. (Maximal 5 Story-Points).
Am Anfang wird schon entschieden in welchem Sprint ungefähr welche User Story bearbeitet wird. D.h. es wird eine Art Release-Plan erstellt.

4.4.4 Sprint Planning

Kapazität wird am Anfang bestimmt. Kapazität wird durch folgende Werte vermindert:

- Bugs
- Ferien/Absenzen

4.5 Wichtige Erkenntnisse aus Interview

- Scrum != Scrum
- Scrum Poker während dem Sprint
- Schnelles und unkompliziertes Task-Board ist wichtig.
- Nicht Business Aufgaben mindern die Kapazität

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Analyse

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einführung

1.1 Zweck

Die Domain-Analyse diskutiert die Rolle der wichtigsten Entitäten aus ScrumTable und deren Abhängigkeiten untereinander.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Als erstes wird die Nomenklatur des Domain Modells beschrieben und anschliessend die Rolle der einzelnen Entitäten.

2 Domain Analyse

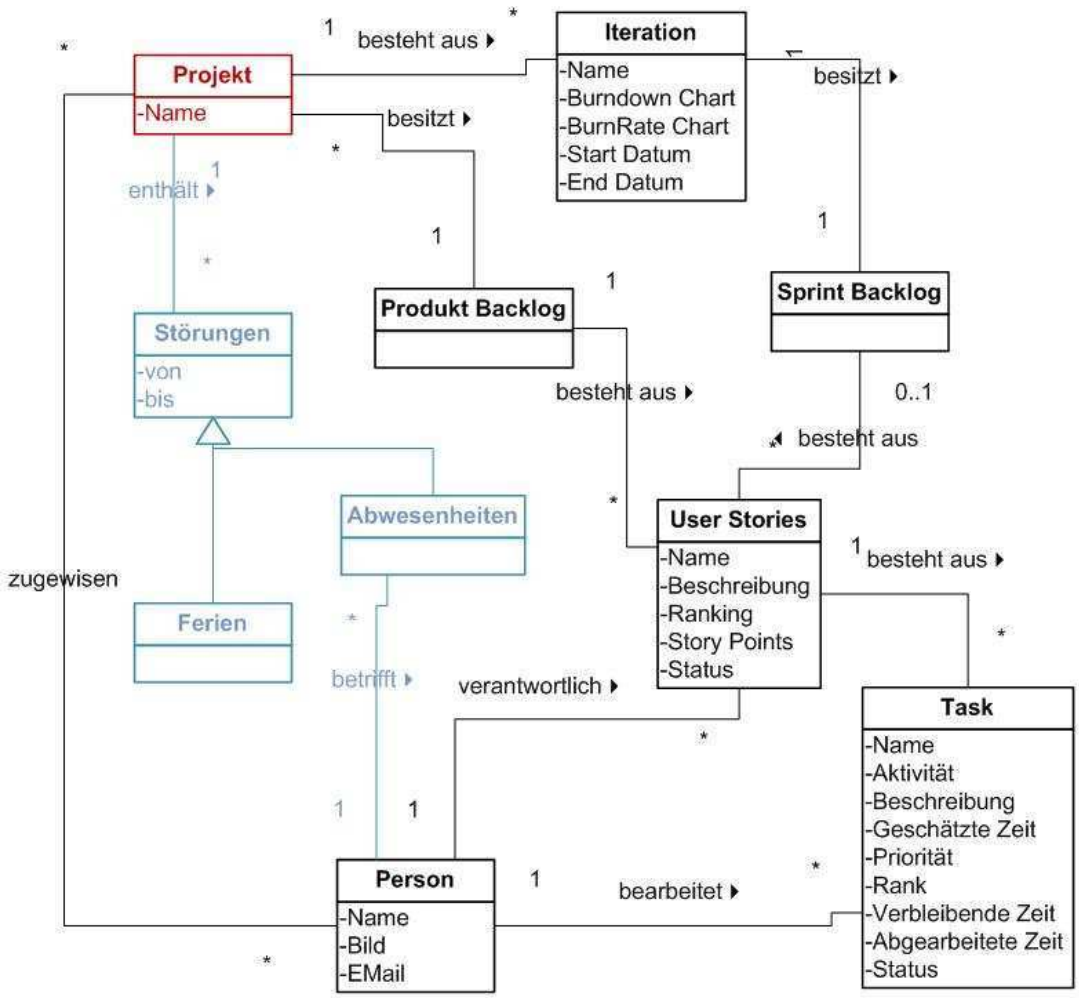


Abbildung 1 Domain Model von Scrum Table

2.1 Domain Model Beschreibung

2.1.1 Projekt

In einem Projekt wird die Anzahl Iterationen definiert. Das Projekt gilt als zentrales Element im Domain Model und ist deshalb speziell gekennzeichnet.
Störungen gehören zu einem Projekt.
Ein Projekt besitzt ein Produkt Backlog.
Ein Projekt besteht aus mehreren Iterationen.

Attribut	Beschreibung
Name	Enthält eine lesbare und sprechende Identifizierung eines Projektes.

2.1.2 Iteration

Eine Iteration besitzt einen Sprint Backlog. Eine Iteration gehört immer zu einem Projekt.

Attribut	Beschreibung
Name	Enthält eine lesbare und sprechende Identifizierung einer Iteration.
Burndown Chart	Eine grafische Darstellung, welche den Arbeitsfortschritt der aktuellen Iteration darstellt.
BurnRate Chart	Eine grafische Darstellung mit der aktuellen und benötigten Entwicklungs-Geschwindigkeit.
Start Datum	Enthält das Start Datum einer Iteration (oder Sprint im Scrum-Jargon).
End Datum	Enthält das End Datum einer Iteration (oder Sprint im Scrum-Jargon).

2.1.3 Störungen / Abwesenheiten / Ferien

Diese Domain Objekte bilden optionale Komponenten und sind in der Version 1.0 des Programms nicht enthalten. Deshalb sind diese Objekte hellblau gekennzeichnet.
Die Störungen / Abwesenheiten / Ferien beschreiben die Unterbrechungen im Bereich der Arbeitsbereitschaft der Team-Mitglieder (Personen) eines Projektes. Dies können sein:

- Ferien (oder auch freie Tage)
- Abwesenheiten von Mitarbeiter (Krankheiten, Militär…)

Attribut	Beschreibung
Von/Bis	Enthält den Zeitraum der Störung.

2.1.4 Person

Eine Person ist einem oder mehreren Projekten zugeordnet.
Personen können abwesend sein.
Eine Person kann für eine oder mehrere User Stories verantwortlich sein.
Eine Person bearbeitet Tasks.

Attribut	Beschreibung
Name	Enthält den vollständigen Name einer Person. Falls dieser nicht eindeutig ist, wird der Name um eine eindeutige Kennzeichnung aus dem System ergänzt.
Bild	Enthält das Portrait-Bild einer Person.
E-Mail	Enthält die Mail Adresse einer Person.

2.1.5 User Stories

Eine User Story ist ein Wunsch bzw. eine Anforderung des Produkt Owners.

Attribut	Beschreibung
Name	Enthält eine lesbare und sprechende Identifizierung einer User Story.
Beschreibung	Enthält die Beschreibung der User Story im Stile von „As a <type of user> I want <some goal> so that <some reason>".
Ranking	Enthält die Wichtigkeit der aktuellen User Story gegenüber allen anderen User Stories.
Story Points	Enthält die zu schätzende Komplexität der User Story.
Status	Die User Story kann im Status „Aktiv", „Gelöst" und „Geschlossen" sein.

2.1.6 Task

Task beschreibt eine Arbeit welche ausgeführt werden muss um eine User Story zu vollenden.
Ein Task wird von einer Person bearbeitet.

Attribut	Beschreibung
Name	Enthält eine lesbare und sprechende Identifizierung eines Tasks.
Aktivität	Enthält die Art und Zugehörigkeit einer Tätigkeit, beispielsweise „Entwicklung“, „Design“, „Dokumentieren“, „Testen“ oder „Anforderungen spezifizieren“.
Beschreibung	Enthält eine kurze Beschreibung des Tasks.
Rank	Enthält die Wichtigkeit des aktuellen Tasks gegenüber allen anderen Tasks in einer Iteration.
Priorität	Enthält eine Prioritätseinstufung eines Tasks von 1-4.
Geschätzte Zeit	Dieses Feld enthält die ursprünglich geschätzte Zeit fürs Abarbeiten des Tasks.
Verbleibende Zeit	Dieses Feld enthält die noch benötigte Zeit bis zum Beenden eines Tasks.
Abgearbeitete Zeit	Dieses Feld enthält die bereits investierte Zeit in einen Task.
Status	Der Task kann im Status „Aktiv“ und „Geschlossen“ sein.

2.1.7 Produkt Backlog

Das Produkt Backlog beinhaltet alle User Stories vom assoziierten Projekt.

2.1.8 Sprint Backlog

Das Sprint Backlog beinhaltet alle User Stories welche zur assoziierten Iteration gehören.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Design

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

Quellenverzeichnis

Marguerie, F. (12. Mai 2009). *Localization in WPF.* Abgerufen am 9. Juni 2010 von Fabrice's weblog: http://weblogs.asp.net/fmarguerie/archive/2009/05/12/localization-in-wpf.aspx

Peter Sommerlad and Marcel Rüedi, E. 9. (1998). *Design Patterns Library and Host of the PLoP Conferences*. Abgerufen am 28. 4 2010 von Design Patterns Library and Host of the PLoP Conferences: http://hillside.net/europe/EuroPatterns/files/DIY_Reflection.pdf

Shaman, T. (6. Februar 2009). *WPF Localization - On-the-fly Language Selection.* Abgerufen am 9. Juni 2010 von http://blogs.microsoft.co.il/blogs/tomershamam/archive/2007/10/30/wpf-localization-on-the-fly-language-selection.aspx

1 Einführung

1.1 Zweck

In diesem Dokument wird das Design von ScrumTable erläutert.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Am Anfang werden die verschiedenen Strukturen von ScrumTable erklärt. Darauf folgend das Design vom User Interface, Data Layer und Business Layer.

2 Systemstruktur

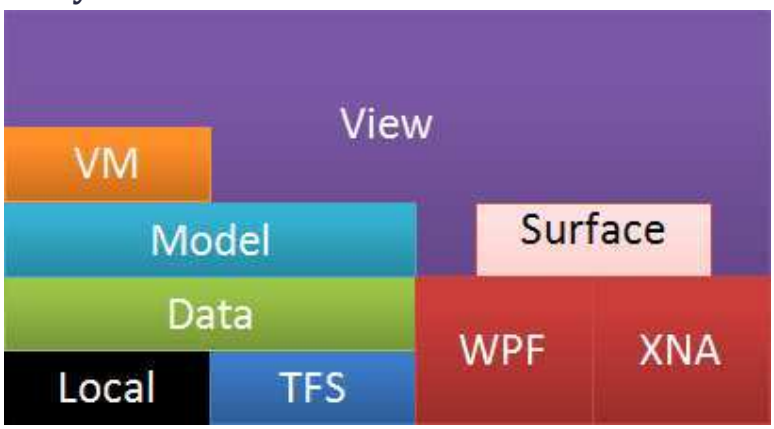


Abbildung 1 Systemstruktur

2.1 Local & TFS

Stellt eine Anbindung zu einer physischen Datenquelle her und ist verantwortlich für das korrekte speichern und laden der Daten.

2.2 Data

Die Datenschicht bildet den Zugriff auf die Datenquelle und wird vom Model angesteuert. Die Datenschicht enthält ebenfalls die Architektur fürs dynamische Einbinden (Laden) von Bibliotheken zur Ansteuerung physischer Datenquellen.

2.3 Model

Das Model repräsentiert die Daten welche vom Programm benötigt werden.

2.4 View-Model (VM)

Kapselt das Model für die View. Fügt Funktionalitäten dem Model hinzu welche von der View benötigt werden und stellt die Daten so dar, damit die View einfach (z.B. per Binding) auf diese zugreifen kann.

2.5 View

Beinhaltet die grafische Darstellung.

2.6 WPF & XNA

GUI-Frameworks welche von der Surface-SDK erweitert werden.

2.7 Surface

Stellt für die beiden Frameworks Elemente und Controls zur Verfügung welche benötigt werden.

3 Physische Struktur

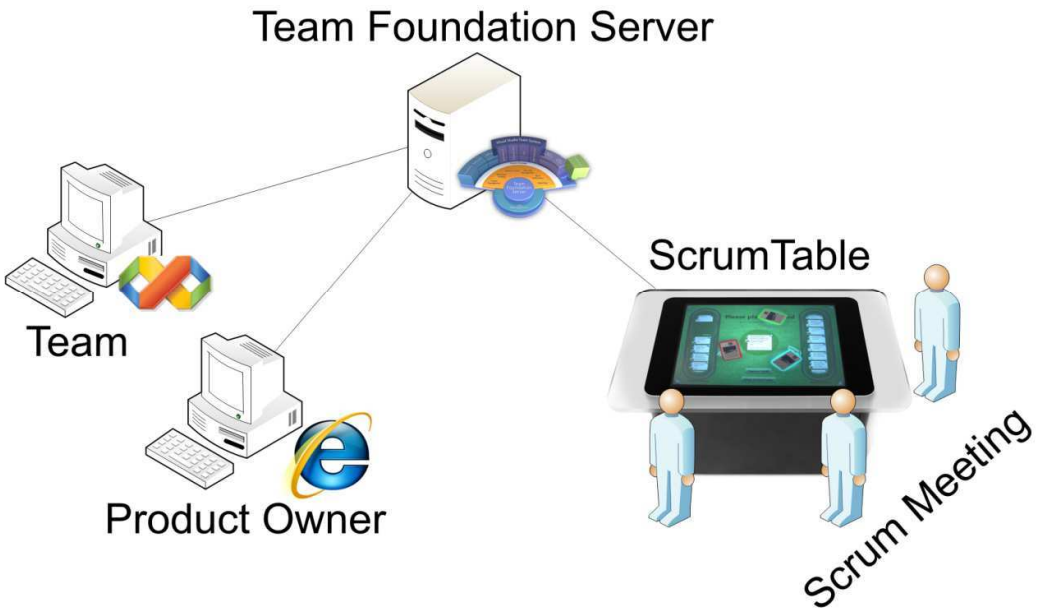


Abbildung 2 Physische Struktur

Die Physische Struktur ist geprägt vom zentralen Server in diesem Fall der Team Foundation Server

3.1 Microsoft Team Foundation Server (TFS)

Speicher für die Aufgaben und alle andern anfallenden Daten. Stellt Scrum Items zur Verfügung. Ermöglicht das Erfassen und Bearbeiten von allen notwendigen Artefakten von Scrum. Stellt ebenfalls für Scrum wichtige Reporting-Funktionen zur Verfügung.

3.2 ScrumTable

Stellt die TFS-Daten in einem interaktiven Format dar und unterstützt dadurch das Scrum Meeting.

3.3 Produkt Owner

Erfasst vor dem Meeting für ihn wichtige Geschichten (User Stories) welche vorhanden sein sollten.

3.4 Developers

Erfasst vor dem Meeting für ihn relevante Daten.

3.5 Scrum Meeting

Die Parteien kommen zusammen und führen das jeweilige Scrum Meeting durch.

4 Logische Struktur

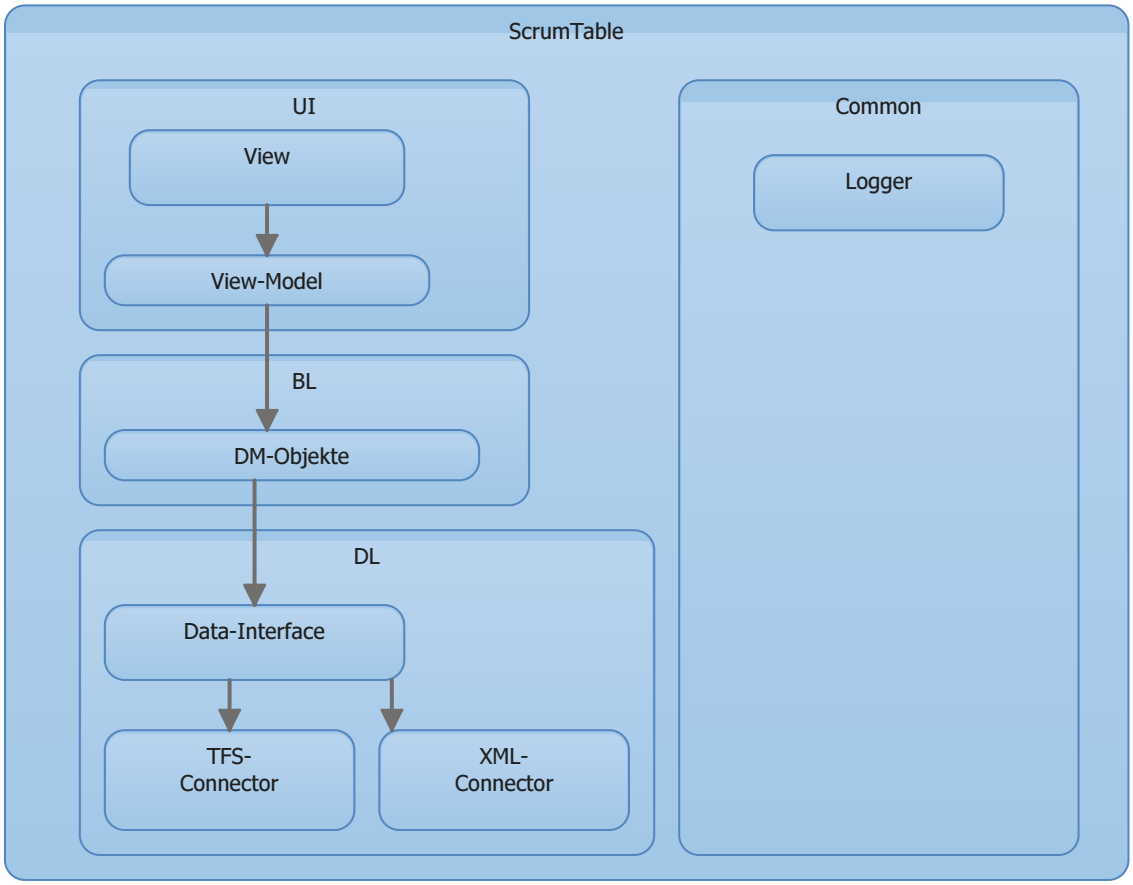


Abbildung 3 Logische Struktur

4.1 UI View

Beinhaltet die grafische Darstellung.

4.2 UI View Model

Beinhaltet die GUI-Logik. Bearbeitet die DM-Objekte für das GUI auf.

4.3 BL DM-Objekte

Beinhaltet die Business Objekte.

4.4 DL Data-Interface

Stellt die Interfaces zu Verfügung, welche nötig sind um die Daten aus einer Quelle zu laden.

4.5 DL XML/TFS-Connector

Stellen eine Verbindung zu einer Datenquelle her und ist für Speicherungen und Laden der Daten verantwortlich.

4.6 Common Logger

Applikations-Logger welcher von überall angesteuert werden kann.

5 MVVM Architektur

Die View benötigt ein View-Model.
Das View-Model benötigt ein Model.
Das Model benötigt eine Daten-Quelle.

Die MVVM-Architektur verlangt die Verdrahtung mehrere Elemente.
Wir haben 3 Möglichkeiten analysiert um die Verdrahtung zu ermöglichen.

1. Das Übergeben eines „Service-Locator“ oder ein Kontext welcher durch alle Instanzen durchgeschleust werden müsste.
2. Die Injection der Klassen mit Hilfe eines Dependency Injection Frameworks.
3. Das nutzen von Singletons um die Verdrahtung statisch zu lösen.

Entscheidung für Lösung Nummer 2 Aufgrund:

Service-Locator:

Pro: Kein zusätzliches Framework, wenig Komplexität, bekannt
Contra: WPF verlangt ein Default Konstruktor

Dependency Injection Framework:

Pro: Dynamische Verdrahtung; Einfache Testbarkeit
Contra: Zusätzliche Komplexität; Zusätzliches Framework

Singelton:

Pro: Einfach zu implementieren
Contra: Starke Bindung, Testbarkeit schlecht

Entscheid:

Aufgrund das Service Locator nicht einsetzbar ist und Singelton ein unschönes Design zur Folge hätte, fällt die Auswahl auf ein Dependency Injection Framework.

5.1 Dependency Injection Framework als Service Locator

Ein Vorteil vom Container ist, dass er auch einem Service Locator entspricht. Der Container kann auch Injected werden. Falls dies getan wird, kann der Container auch als Service Locator genutzt werden.

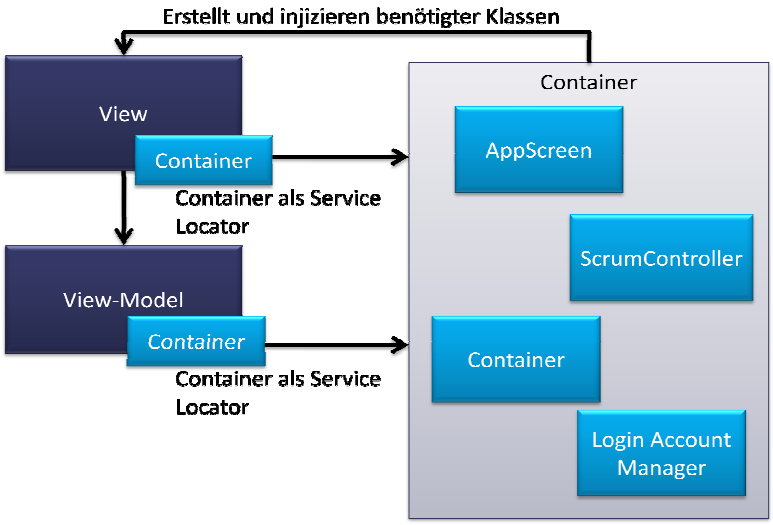


Abbildung 4 Container als Service Locator

5.2 Testen

View-Model ist konzeptioniert um diese Funktionalität zu testen. Durch das „Dependency Injection Framework“ lassen sich verschiedene Umgebungen aufsetzten um die Klassen besser testen zu können. In der Testumgebung werden Mocks registriert und in der Realenumgebung die echten Klassen.

Die Verdrahtung der Objekte wird durch die Container gewährleistet. Die im Container registrierten Objekte werden von Container in die zu erstellende Klasse eingeschleust.

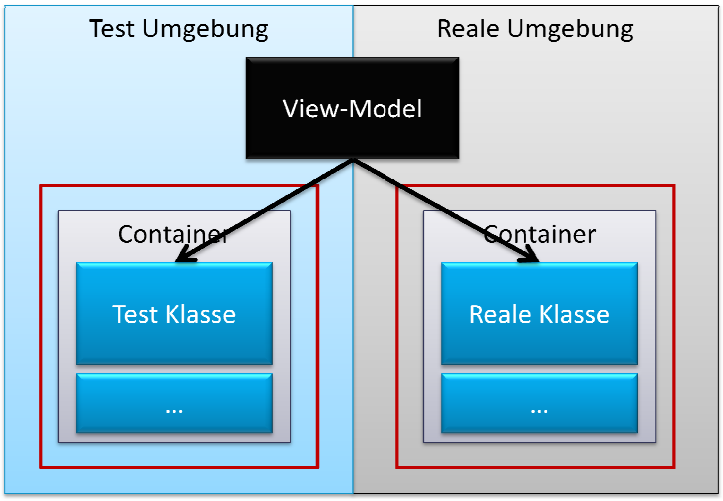


Abbildung 5 Testen mit Dependency Injection

6 Security Architektur

Für die Verbindung an den TFS wird ein Passwort verlangt. Wenn der Benutzer sich über den Tisch am TFS anmelden muss, ist es nicht verantwortbar, dass der Benutzer immer das komplexe Passwort eingeben muss. Deshalb werden die Einstellungen für den TFS lokal gespeichert und durch einen Pin gesichert.

Um die Sicherheit des Passwortes zu gewährleisten wird folgende Struktur festgelegt:

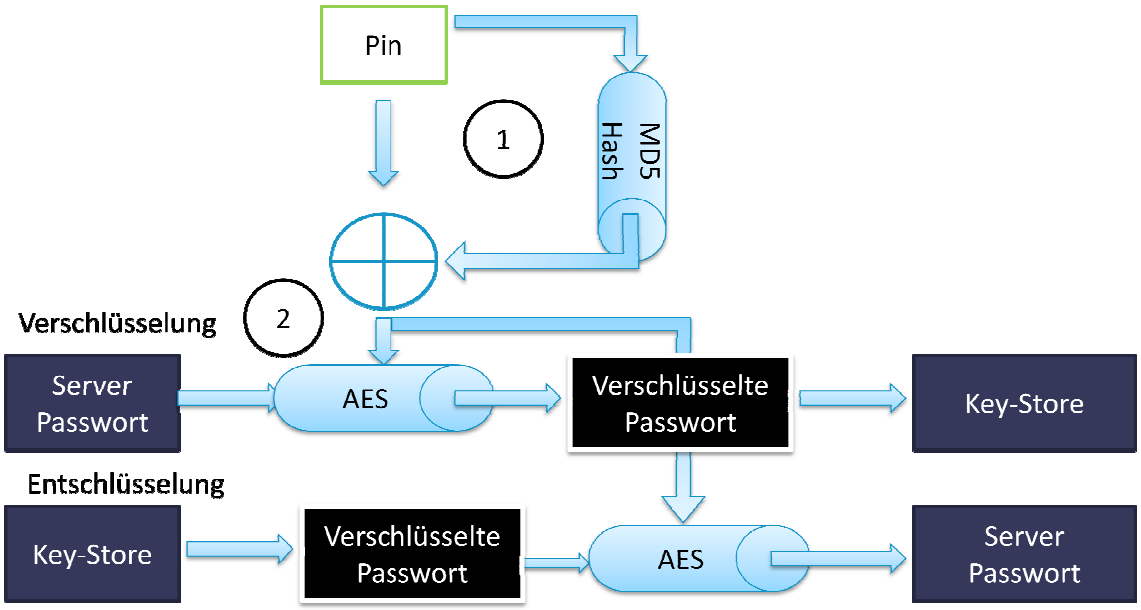


Abbildung 6 Verschlüsselung und Entschlüsselung von Passwörter

Der User verschlüsselt sein Server-Passwort mit seinem persönlichen Pin. Zusätzlich wird der Pin durch seinen eigenen MD5-Hash verlängert.

Typische Angriffe werden mit dieser Struktur erschwert:

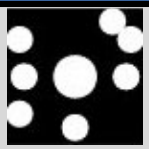
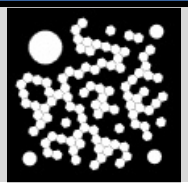
1: **Rainbow Table** MD5 Hash erschwert eine Rainbow-Table Angriff.

2: **Brute-Force:** Der entschlüsselte Wert kann nur durch einen Login-Versuch an der Datenquelle überprüft werden.

Hinweis:
Es wird angenommen, dass die Datenquelle den User sperrt, falls dieser zu viele Versuche mit einem falschen Passwort unternommen hat.

7 Surface Tags

7.1 Unterschiede

	Byte Tags	Identity Tags
Aussehen		
Datenbits	8 bit	2 x 64 bit = 128bit
Anzahl möglicher Werte	256	Fast unlimitiert 340,282,366,920,938 * 10^24
Grösse	1.905cm X 1.905cm	2.54cm * 2.54cm
Verfolgung	Kann schnell bewegt warden und wird trotzdem gut erkannt.	Eher für Objekte, die sich nicht oder nur langsam bewegen.
Datentyp	byte ByteTag.Value	long IdentityTag.Series long IdentityTag.Value

7.2 Tags für Login

Die Karte mit dem Tag wird jeweils auf den Tisch gelegt und dann eigentlich nicht mehr gross bewegt. Die Anzahl dieser Karten ist eher klein, könnte jedoch in einem Ausnahmefall 256 überschreiten. Dies in einer grösseren Firma, wo auch jeder Mitarbeiter eine Karte haben soll um selbstständig ScrumTable kurz nutzen zu können.

Aus diesem Grund wird vorsichtshalber ein IdentityTag verwendet von der Series 0 x 1111 1111 1111 1111. Jeder Value in dieser Serie ist ein eigener Login Tag, welcher registriert werden kann.

7.3 Tags für ScrumPoker

Die ScrumPoker Karte wird normalerweise auf den Tisch gelegt und nicht mehr bewegt. Es gibt 13 ScrumPoker Karten. Dann gibt es diese in verschiedenen Farben. Also theoretisch wird der Tisch nicht von mehr als 10 Personen benutzt. Deswegen würden ByteTags genügen. Jedoch wurde als Team beschlossen hier ebenfalls Identity Tags zu verwenden. Bei Byte Tags müsste man jeweils ein Byte einer Farbe und einer Karte (Zahl) zuweisen. Dazu müsste dann im Programm eine Tabelle geführt werden mit allen Werten. Dies wäre etwas mühsam und unflexibel. Deswegen werden ebenfalls IdentityTags verwendet. Dabei wird die Series 0 x 3333 3333 3333 3333 verwendet welche in ScrumTable für ScrumPoker Karten reserviert ist.

7.3.1 Aufbau der Value

0 x 0000 000C CCCC CBBA

A — Typ der Karte
Zahl = 0, Fragezeichen = 1, Kaffeetasse = 2

BB — Wert der Karte wenn es eine Zahlkarte ist. Dieser Wert ist multipliziert mit zwei. Das heisst der aus dem IdentityTag ausgelesene Wert muss durch 2 gerechnet werden.

CC CC CC — RGB Farbcode der Karte

8 Data Layer Plug in Architektur

In ScrumTable ist es grundsätzlich möglich, die Datenverbindung zum Ziel- und Endsystem beliebig um ein weiteres Ziel- und Endsystem zu erweitern. Die Version 1.0 enthält die folgenden Plug ins:

Name (Technischer Name)	Ziel- und Endsystem	Beschreibung
TFS (ScrumTable.DL.Data.TFS)	Microsoft Team Foundation Server	Die Daten für das Team Foundation Server Plug in werden aus den folgenden, mit dem Team Foundation Server installierten Produkten, gelesen: - Microsoft Windows SharePoint Service - Microsoft Reporting Service - Microsoft Team Foundation Server Die mutierten Daten werden anschliessend in die folgenden Endsysteme zurückgeschrieben: - Microsoft Team Foundation Server - Microsoft Windows SharePoint Service Die Synchronisation und das Updaten der Daten erfolgt automatisch.
Local (ScrumTable.DL.Data.Local)	Lokaler Speicher	Die Daten sind flüchtig, d.h. sie gehen beim Beenden des Programmes verloren, falls Sie nicht explizit in ein lokales XML File zurück gespeichert werden.

8.1 Plug in Komponenten

Die Basisfunktionalitäten mit dem Interfaces sowie Default-Implementierungen werden durch die Basis Library (technischer Name ScrumTable.DL.Data) zur Verfügung gestellt. Das folgende Diagramm stellt die Architektur entsprechend mit den benötigten Komponenten dar:

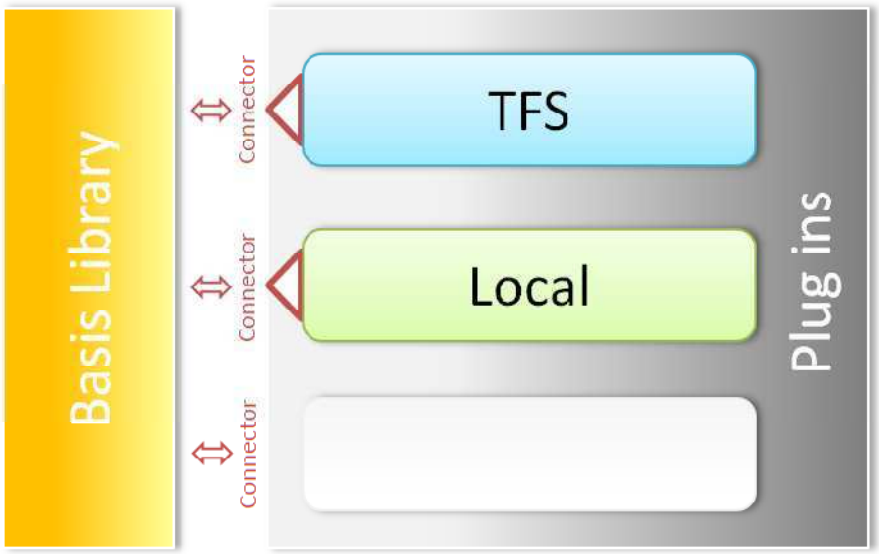


Abbildung 7 Plugin Architektur in der ScrumTable Datenschicht

8.2 Basis Library

Die Basis Library enthält den Mechanismus um zusätzliche Komponenten zur Laufzeit einzubinden. Dies geschieht über die Connectors (Konnektoren), welche eine Schnittstelle zwischen der Basis Library und den eingebundenen Plug ins (TFS / Local) definieren.

Das Laden von Libraries zur Laufzeit bietet folgende Vorteile:

- Die Basis Library enthält keine Abhängigkeiten zu den eingebunden Plug ins. Die einzige Schnittstelle besteht aus dem Connector.
- Es können zur Laufzeit neue Plug ins dazu geladen werden, ohne dass die Applikation neu gestartet werden muss.
- Die Applikation muss nicht neu kompiliert werden, falls ein neues Ziel- und Endsystem angesprochen werden soll.
- Drittanbieter haben die Möglichkeit, eigene Implementationen für ein spezifisches Ziel- und Endsystem anzubieten.
- Durch entkoppeln der einzelnen Plug ins vom Hauptsystem, können diese abgetrennt getestet und gewartet werden.

Folgende Nachteile birgt dieser Architektur- Entscheid:

- Eine weitere Indirektion mehr im Programm Code, daraus folgen langsamere Ausführzeiten.

- Fehler in der Datenschicht eines Drittanbieters können die ganze Applikation zum Absturz bringen.
- Mehr Komplexität im Programm Code, viele Interfaces müssen implementiert werden.

8.3 Plug ins

Die Plug ins referenzieren auf die Basis Library, damit die Plug ins die Interfaces der Basis Library implementieren können. Durch den Connector werden die Plug ins instanziiert. Die Plug ins können in ScrumTable beliebig ausgetauscht werden, damit ergibt sich eine sehr grosse Flexibilität und Skalierbarkeit.

In ScrumTable Version 1 wird die Basis-Architektur der Plug ins entsprechend der Architektur des TFS Connectors aufgebaut. Somit ist ein relativ geringer struktureller Realisationsaufwand fürs TFS Plug in, welches primär dem Ziel- und Endsystem entspricht, zu erwarten.

9 Data Layer Architektur als Anything

Das Kapitel Logische Struktur beschreibt das Zusammenspiel der verschiedenen Schichten als Ganzes. Darin steht beschrieben, dass der Business Layer auf dem Data Layer aufbaut. Die Interfaces im Data Layer sind dabei sehr generisch aufgebaut, damit möglichst viele Ziel- und Endsysteme angesprochen werden können.

Der Data Layer richtet sich in seiner Architektur nach dem Muster, auf welchem die Microsoft Team Foundation Server API's aufgesetzt wurden. Das folgende Schema soll dieses Muster, in der Fachliteratur allgemein als *Anything* bekannt, beschreiben:

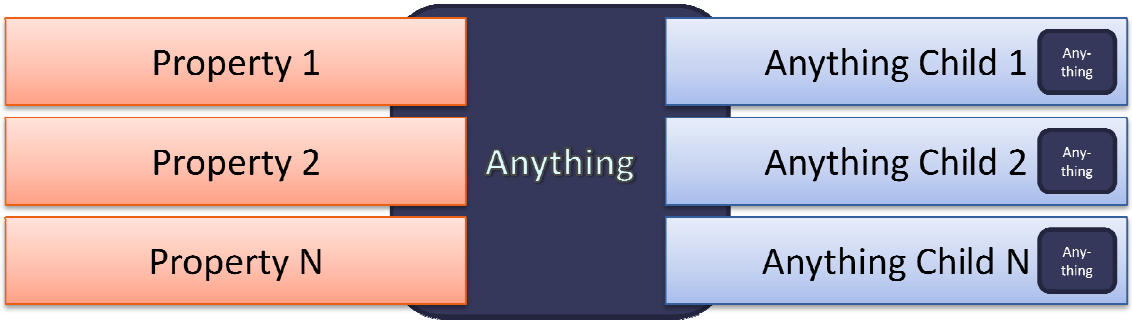


Abbildung 8 Das Anything-Muster

Ein Anything-Objekt (technische Bezeichnung *Anything Object*) besteht also aus Eigenschaften (Properties) sowie aus untergeordneten Anything-Objekten. Ein Anything-Objekt kann also mehrere Anything-Objekte enthalten, die wiederum Anything-Objekte enthalten können. Somit kann eine hierarchische Struktur gebildet werden.

- Ein Anything-Objekt kann mehrere, bis N (theoretisch unendlich) Eigenschaften enthalten. Alle Eigenschaften enthalten eine Identität sowie der zu speichernde Wert. Dies wird mittels einer Verzeichnisstruktur, welche die Identitäten zum Wert zuordnet, im Anything erreicht. Die Werte selbst sind typlos, die Typ-Informationen gehen beim Einfügen der Werte ins Anything-Objekt verloren und müssen beim auslesen wieder abgefragt werden.
- Ein Anything-Objekt kann mehrere, bis N (theoretisch unendlich) untergeordnete Anything-Objekte enthalten. Dies wird mittels derselben Verzeichnisstruktur, wie im ersten Punkt beschrieben, erreicht.

Das Anything-Muster bietet folgende Vorteile:

- Die Schnittstellen zwischen dem Data Layer und Business Layer sind sehr stabil, da neue Eigenschaften (Properties) ohne Änderungen an den Anything-Objekten eingefügt werden können.
- Ein Anything-Objekt bietet eine sehr grosse Flexibilität und kann theoretisch überall in der Applikation, wo entsprechende Datenstrukturen erwünscht sind, verwendet werden.

Folgende Nachteile birgt dieser Architektur- Entscheid:

- Sämtliche Typinformationen gehen verloren. Zum Zeitpunkt der Kompilation können sämtliche Typinformationen nicht mehr validiert werden, das *Error Left* Prinzip wird verletzt.

- Fehleranfällige Software, was sehr genaues Testing erfordert.
- Informationen über die gespeicherten Properties müssen irgendwo global abgelegt oder dupliziert werden.

9.1 Data Layer Architektur

Der Data Layer setzt auf dem Anything[1] Pattern auf. Das folgende Bild visualisiert die Zusammenhänge des Anyhting-Musters im Data Layer. Die farbliche Gestaltung ist entsprechend der Abbildung oben (Das Anything-Muster) gewählt.

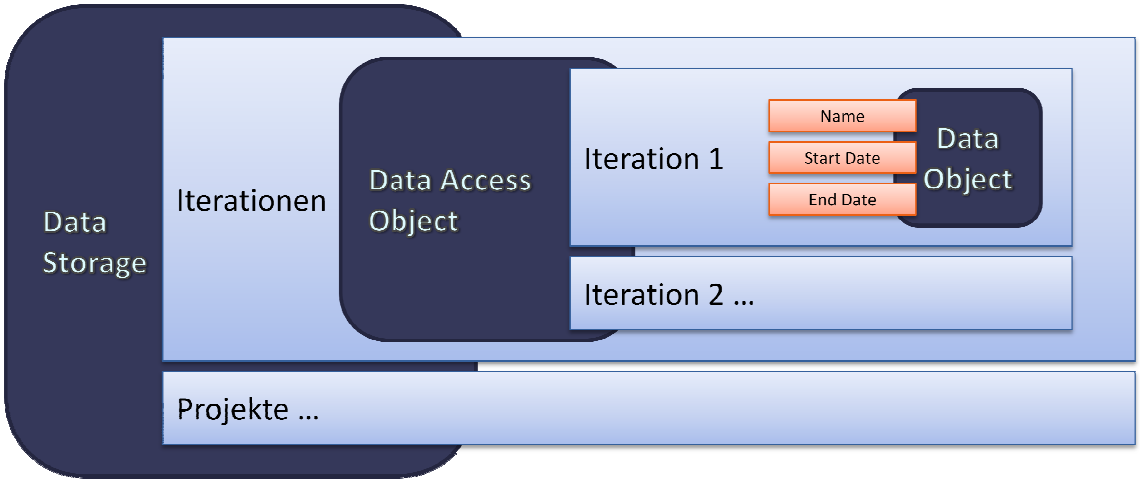


Abbildung 9 Objekthierarchie im Data Layer

Die Hierarchie der Objekte (Objekte) im Data Layer gestaltet sich wie folgt:

- Der Data Storage entspricht dem äussersten Objekt und enthält weitere Anythings für das Speichern der Iterationen, Projekte, User Stories, Members und Tasks, welche Datenzugriffsobjekte als Anything repräsentieren.
  Im Sinne einer Datenbank entspricht dies der Menge aller Datenbanktabellen.
- Ein Datenzugriffsobjekt (Data Access Object) enthält die Objekte, welche seinem Typ entsprechen. So enthält das Iterationen-Datenzugriffsobjekt Iterations-Objekte.
  Im Sinne einer Datenbank entspricht dies einer Datenbanktabelle.
- Eine Iteration (also ein Iterations-Objekt) entspricht wiederum einem Anything-Objekt als Daten-Objekt (Data Object) mit den Werten als Properties der Iteration.
  Im Sinne einer Datenbank entspricht dies einer Datenbanktabellen-Reihe mit den Kolonnenwerten.

Wie bereits erwähnt, entspricht diese Struktur im Ansatz einer In-Memory-Datenbank. Für derartige Zwecke hat Microsoft mit dem .NET Framework 1.0 das Datenmanagement-Framework ADO.NET eingeführt. Dieses könnte im Grundsatz für ScrumTable auch verwendet werden, enthält aber aufgrund von des Designs keine Möglichkeit, Daten in einem Datenzugriffsobjekt *Lazy* (also sobald benötigt) nachzuladen.

[1] (Peter Sommerlad and Marcel Rüedi, 1998)

Die folgende Abbildung zeigt die Data Layer Architektur aus Sicht einer Datenbank. Mehr dazu siehe Erläuterungen oberhalb.

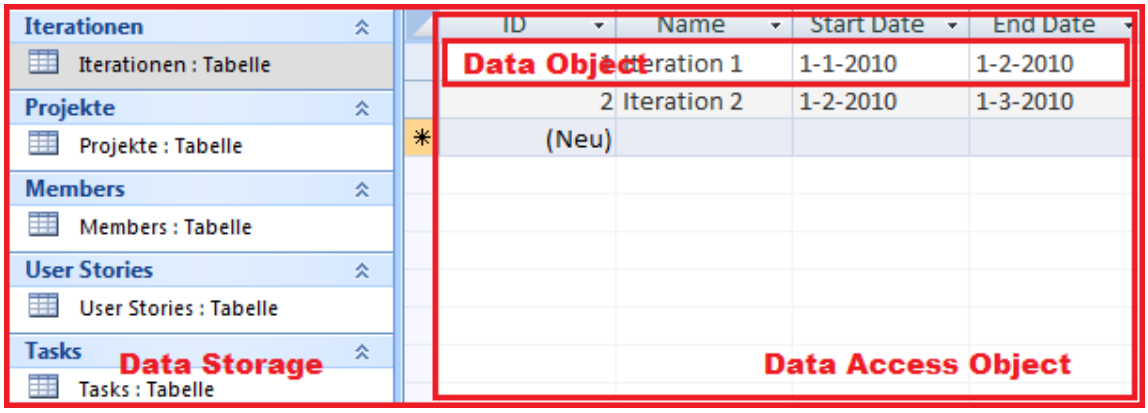


Abbildung 10 Schematische Darstellung des Data Layers als Datenbank

10 Data Layer Schema Architektur

Das Kapitel Data Layer Architektur als Anything beschreibt die Architektur des Data Layers auf Basis des Anything-Musters. Ein sehr grosses Problem dieses Musters bildet die Typenfreiheit, was allerdings auch der Hauptvorteil des Pattern ist (siehe Ausführungen in im vorherigen Kapitel). Beim Anything-Muster wird also die Flexibilität gegen Stabilität aufgewogen.

Der grösste Nachteil, der Verlust der Typensicherheit und damit eine geringere Programmcode-Stabilität, kann mittels eines Tricks reduziert werden. Allerdings handelt man sich durch diesen Trick wiederum nicht intuitive Komplexität ein. So werden zur Laufzeit die Strukturen auf ihre Gültigkeit auf dem Business Layer und Data Layer überprüft. Dies geschieht Mittels eines Schemas, was die folgenden Kapitel beschreiben sollen.

10.1 Schreibendes Arbeiten mit Schema

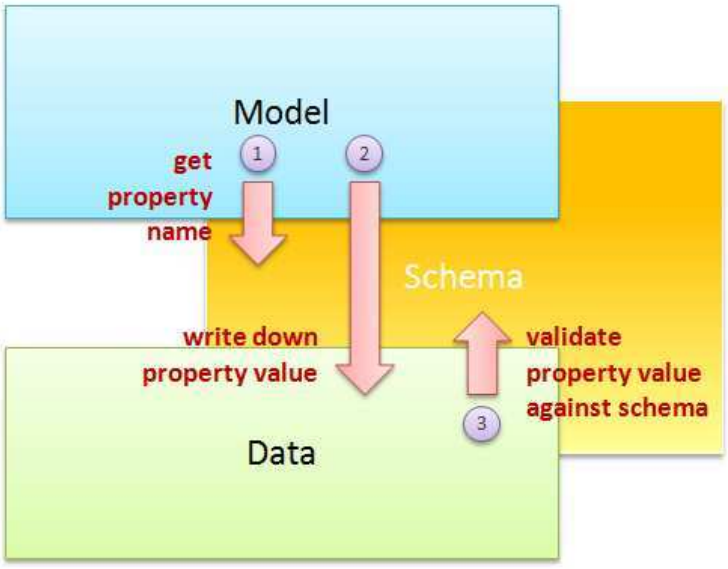


Abbildung 11 Zusammenarbeit zwischen Domain Model und Data Layer mittels Schema beim Schreiben

Das Arbeiten mit einem Schema hat sich als sinnvoll herausgestellt, da somit die erwarteten Eigenschaften der Anything-Objekte definiert sind. Dabei wird wie folgt vorgegangen:

Szenario: Domain Model möchte Daten auf Data Layer schreiben (siehe Abbildung oben).

1. Das Domain Model Objekt holt sich den Namen für das Property im Anything, welches geändert hat.
2. Der geänderte Wert wird ins entsprechende Data Layer Objekt geschrieben.
3. Das Data Layer Objekt (Anything-Objekt) validiert die Daten gegen das Schema und prüft, ob die angegebenen Daten in den im Schema vorgeschriebenen Datenbereich passen

10.2 Lesendes Arbeiten mit Schema

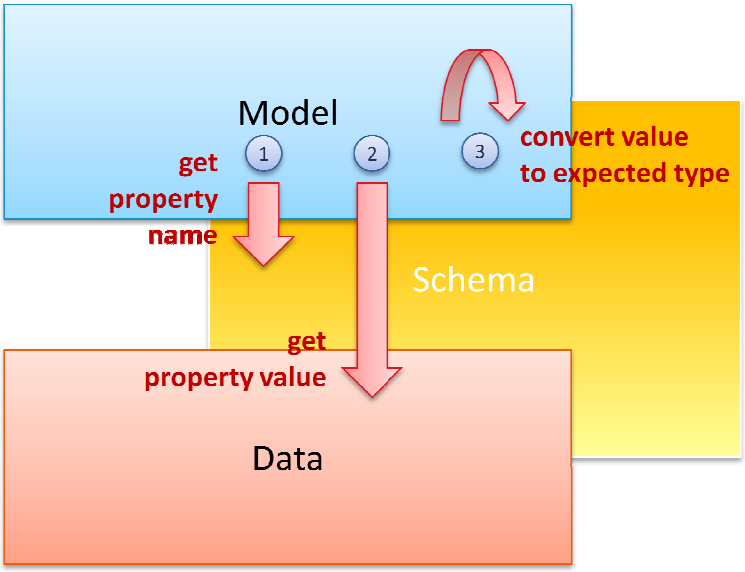


Abbildung 12 Zusammenarbeit zwischen Domain Model und Data Layer mittels Schema beim Lesen

Szenario: Domain Model möchte Daten aus Data Layer lesen (siehe Abbildung oben).

1. Das Domain Model Objekt holt sich den Namen für das Property im Anything, welches es lesen möchte.
2. Mit den aus dem Schema stammenden Namen wird das Property aus dem Anything im Data Layer ausgelesen.
3. Der Wert wird auf der Domain Layer Ebenen in den gewünschten Datentyp konvertiert. Dabei ist ein Type-Check nicht erforderlich, da dieser beim Schreiben geschieht.

11 Business Layer Architektur

Wie im Kapitel Logische Struktur beschrieben, ist ScrumTable in verschiedene Layer unterteilt. Jeder dieser Layer soll eine weitere Abstraktionsebene bieten. Da der Business Layer auf dem Data Layer von ScrumTable aufsetzt, veredelt dieser die Strukturen des Data Layers. Dies geschieht über die hierarchische Aufbereitung der flachen Daten aus dem Data Layer in eine für ScrumTable interne Objektstruktur.

Die hierarchische Aufteilung bietet folgende Vorteile:

- Die Daten können aus dem View Model hierarchisch auf den Business Layer gebunden werden. Da die Business Layer Objekte das fürs WPF-Binding sehr wichtige Interface INotifyPropertyChanged beinhalten, wäre sogar ein direktes Binden auf die Business Komponenten möglich.
- Referenzierte Objekte können mittels deren ID's direkt aufgelöst und in den Baum integriert werden.
- Daten verschiedener Hierarchie-Stufen können explizit und rekursiv nachgeladen werden.
- Jedes Objekt ist durch die hierarchische ID eindeutig im ganzen Baum identifiziert. Dies ermöglicht ein sehr einfaches Auffinden referenzierter Daten.
- Die im Schema (siehe Data Layer Schema Architektur) beschriebenen Eigenschaften können an einer zentralen Stelle an die Data Objects gebunden werden. Somit sind Tippfehler nur an einer Stelle im Business Layer Programmcode anzutreffen.
- Die hierarchischen Elemente können mittels Unit Tests auf deren Korrektheit im Baum überprüft werden.
- Die Ereignisse einzelner Elemente im Baum können weitergeleitet werden und so auf einer überliegenden, zusammenfassenden Stufe abgefangen werden.
- Status-Codes, welche als Werte aus dem Data Layer geliefert werden, können in den Business Layer Objekten auf kompilierbare Einheiten umgewandelt werden.

Folgende Nachteile birgt dieser Architektur- Entscheid:

- Die zusätzliche Indirektion kann mit Performance-Einbussen verbunden sein.
- Im Business Layer kommt einiges an Komplexität zusammen. Diese kann schwer zu beherrschen sein. Dies kann allerdings durch Unit Tests abgefedert werden.
- Das Interface des Business Layer muss auf die Elemente im View Model übertragen werden, falls diese ans View gebunden werden sollen. Dadurch können Lazy Classes und somit zusätzlicher Wartungsaufwand bei Änderungen entstehen.
- Es entstehen vermehrt Kosten im Bereich Design und Implementation. Diese können allerdings durch eine erhöhte Software-Qualität wettgemacht werden.

Das folgende Schema zeigt den strukturellen Objektaufbau von ScrumTable.

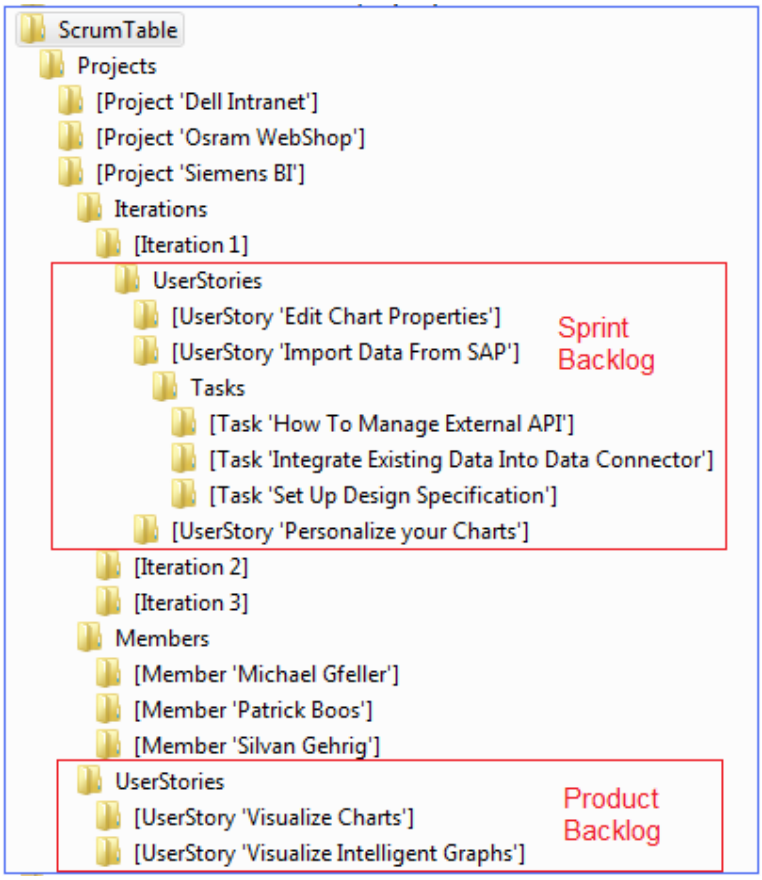


Abbildung 13 Objektgraph im ScrumTable Business Layer

11.1 ScrumTable Business Layer Context

Das oberste Element im Baum entspricht dem ScrumTable Business Layer Context. Dieser beinhaltet die Projekte auf erster Stufe. Jedes der folgenden Objekte wird in einem separaten Data Access Object (siehe Data Layer Architektur) gespeichert.

11.2 Project

Jedes Projekt seinerseits ist ausgestückt mit den zugehörigen Team Mitgliedern (Members) sowie dem Product Backlog (UserStories) und Iterationen (Iterations). Dies muss der Benutzer über das angeschlossene System erreichen. Falls eine User Story noch keiner Iteration zugewiesen wurde, landet diese im Product Backlog.

Das Projekt an sich kann in ScrumTable nicht verändert werden. Dies geschieht beispielsweise über den Team Explorer von Microsoft.

11.3 Member

Ein Member enthält Informationen über ein Team Mitglied, wie z.B. dessen Name, E-Mail Adresse und Bild. In Scrum Table können die Mitglieder nicht verändert werden, sie sind read-only.

11.4 Iteration

Die Iteration enthält die bereits zugewiesenen UserStories. Von der Iteration können nur die Start- & Enddaten verändert werden.

11.5 UserStory

Eine UserStory enthält die ihr zugewiesenen Tasks.

11.6 Task

Der Task bildet das unterste Leaf (Blatt) Element. Ein Task enthält keine weiteren Business Layer Objekte.

12 LibraryBar / SurfaceListBox

Für einige Ansichten wird eine Liste benötigt, von welcher Elemente (User Stories oder Tasks) in eine andere Liste verschoben werden können. Dazu bieten sich in Surface die LibraryBar und SurfaceListBox an. Beide haben ihre Vor- und Nachteile.

Hier wird kurz auf die Anforderungen an eine für ScrumTable brauchbare Liste eingegangen. Anschliessend werden Unterschiede der LibraryBar und SurfaceListBox erwähnt und abschliessend ein Entscheid getroffen und begründet.

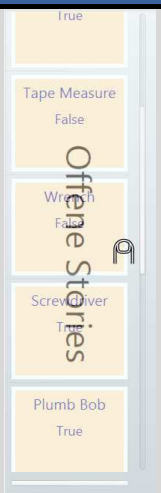
12.1 Anforderungen

Alle folgenden Anforderungen sind notwendig. Wenn diese von einer Komponente nicht gegeben sind, muss ein Weg gefunden werden um dies dennoch zu erreichen.

	Beschreibung
Horizontal / Vertical	Die Ausrichtung des Contents soll horizontal sowie vertikal möglich sein.
Styling	Das Control soll so gestylet werden können, wie man möchte.
ItemsSource	Vom DataContext her soll eine ObservableCollection angezeigt werden können.
ItemTemplate	Die in der ItemsSource enthaltenen Daten sollen per ItemTemplate gestylet werden können.
SelectedItem	Ein Item muss ausgewählt werden können und grafisch als ausgewählt erkennbar sein. Dieses ausgewählte Item muss über ein Property zugänglich sein.
Scrolling	Es soll per Flick-Geste in die Scrollrichtung gescrollt werden können.
Drag and Drop	Durch ziehen eines Items in die 90° verschobene Scrollrichtung soll dieses Item gedragt werden können. Ebenfalls soll ein Item in die Liste gedropt werden können.
Items statisch oder nicht	Zwei verschiedene Anwendungsmöglichkeiten - Möglichkeit Items mehrmals zu ziehen ohne, dass es gelöscht wird - Items nur einmal ziehen (wird inaktiv) und dann bei Drop aus der gedragten Liste gelöscht
Mehrere Spalten/Zeilen	Es soll möglich sein mehrere Spalten anzuzeigen, je nachdem wie viel Platz vorhanden ist. Dass z.B. eine Liste Benutzer in zwei Spalten und mehreren Zeilen angezeigt wird.
Animation	Animationen beim Drop und beim Canceln eines „Drag and Drop“.

12.2 Vergleich

Der Vergleich geschieht dadurch, dass geschätzt wird, wie gross der Aufwand ist um die Funktionalität in dem Control zu erhalten. Dabei bedeutet ein 0-Aufwand, dass das Control dies bereits bietet. Eine 10 bedeutet grosser Aufwand.

	LibraryBar	SurfaceListBox
Aussehen		
Horizontal / Vertical	2	0
Styling	10 (mit momentaner SDK nicht möglich)	3
ItemsSource	0	0
ItemTemplate	0	0
SelectedItem	10	0
Scrolling	0	0
Drag and Drop	0	10
Items statisch oder nicht	5	5
Mehrere Spalten/Zeilen	0	3
Animation	0	5

12.3 Entscheid

Wir haben uns als Team entschieden die SurfaceListBox zu verwenden und eine für uns brauchbare Ableitung davon zu entwickeln. Grund dafür ist, dass es einfacher ist die fehlende Funktionalität der SurfaceListBox hinzuzufügen als die LibraryBar anzupassen.

Ebenfalls gibt es die Möglichkeit das Drag and Drop wie gewünscht zu implementieren. Es fehlt dabei jedoch an Animationen, welche zusätzlichen Aufwand benötigen.

13 I18n

13.1 Problematik

Nach dem ein wenig im Internet gestöbert wurde, was WPF bietet wurde schnell erkannt, dass die Lokalisierung, die von Haus aus geboten wird nicht ganz zufriedenstellend ist. Deshalb gibt es bereits viele Vorschläge[2], wie die Lokalisierung alternativ umgesetzt werden kann.

13.2 Entscheid

Aus Zeitgründen wurden nicht alle bis ins Detail angeschaut. Jedoch überzeugte die Lösung von Tomer Shaman[3] uns sehr schnell. Diese Lösung hat folgende Vorteile:

- On-the-fly: Es bietet die Möglichkeit die Sprache während der Laufzeit umzuschalten
- Es hat eine bessere Performanz als langsames XML, XPath-basierte Binding-Lösung
- Es kann via Style, Control Templates und Data Templates genutzt werden
- Es übersetzt einen formatierten Text mit Parametern und kann default oder custom formatters nutzen
- Bietet ein “Translate” custom markup um schönes XAML zu schreiben
- Verschiedene Sprachen in separaten XML-Dateien
- Keine unnötigen Funktionalitäten (kompakt)
- Einfache Nutzung

[2] (Marguerie, 2009)
[3] (Shaman, 2009)

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Paper Prototype

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.

Quellenverzeichnis

www.wikipedia.ch. (3. 3 2010). Abgerufen am 3. 3 2010 von wikipedia: www.wikipedia.ch

1 Einführung

1.1 Zweck

Das Dokument dient dazu, aufzuzeigen, welche Gedanken sich beim Planen des User Interfaces gemacht wurden. Dabei wird auch erwähnt, was Reviews des PaperPrototypes mit anderen Personen aufgezeigt hat und was dadurch in das User Interface eingeflossen ist.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Das Dokument besteht aus den verschiedenen Versionen des PaperPrototypes. Ebenfalls beinhaltet es dessen Reviews. Zur Übersicht der verschiedenen Ansichten dient eine GuiMap.

2 Version 1

2.1 Ansichten

2.1.1 Anmeldung

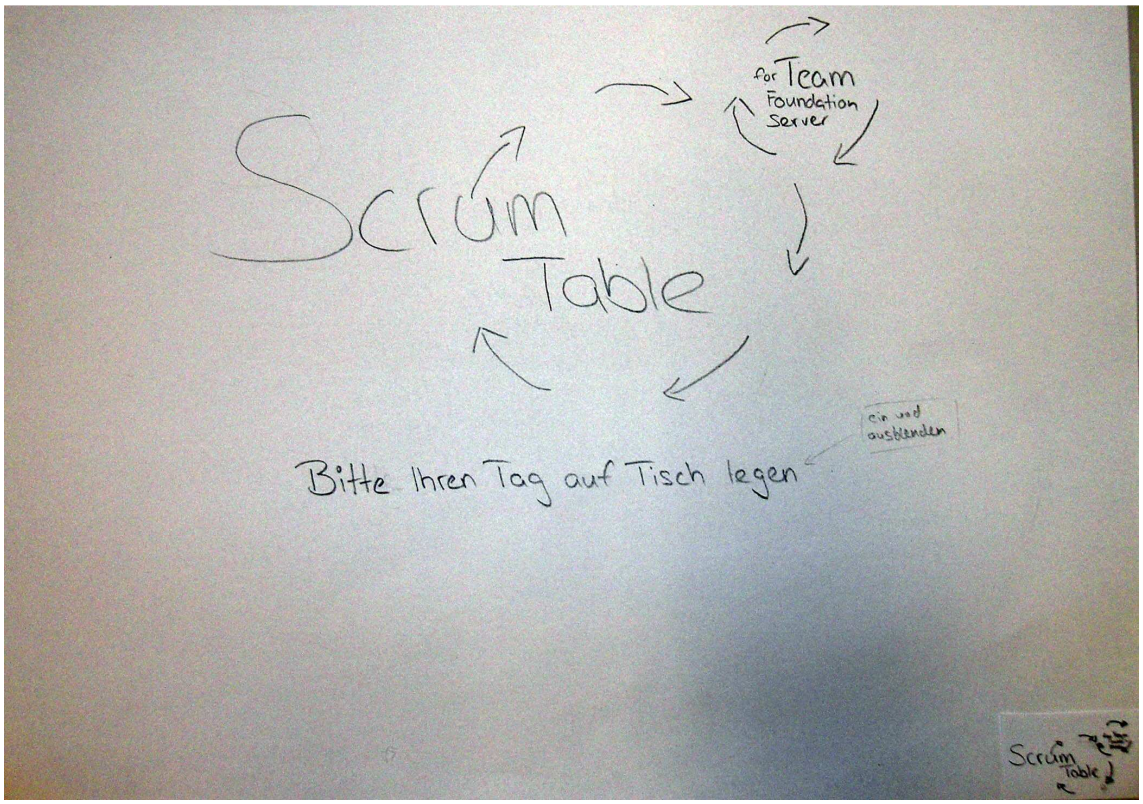


Abbildung 1: Version1 Übersicht

Diese Ansicht erscheint gleich wenn die Applikation gestartet wird. Hier muss sich der Benutzer anmelden indem er seine Tag-Karte auf den Microsoft Surface Tisch legt und seinen PinCode eingibt. Falls die Tag-Karte noch nicht registriert ist, bietet die Applikation die Möglichkeit diesen zu konfigurieren.

2.1.2 Tag-Konfig: TFS

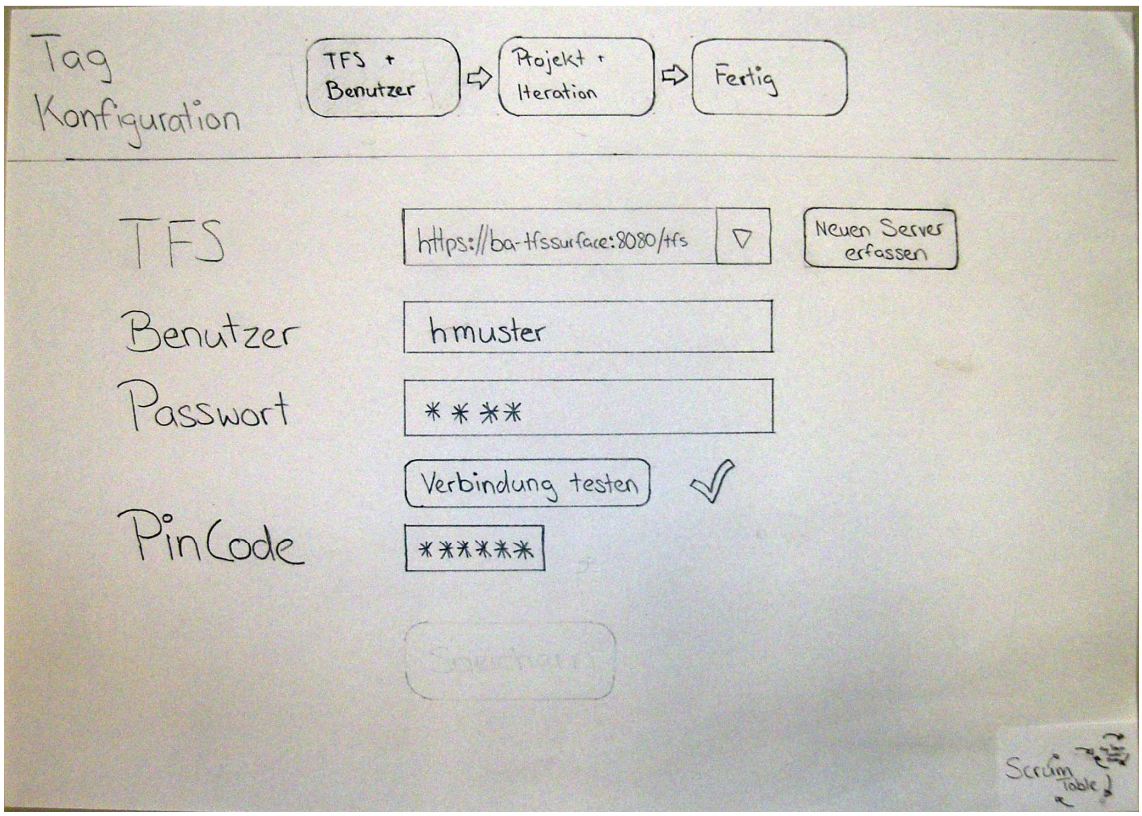


Abbildung 2: Version1 Tag-Konfig: TFS

In dieser Ansicht richtet der Benutzer seine Tag-Karte auf sich ein. Er wählt den Microsoft Team Foundation Server aus, mit welchem er sich verbinden möchte. Dann gibt er seinen Benutzernamen und sein Passwort ein, mit welchem er Zugriff auf den Microsoft Team Foundation Server hat.

Die Verbindung lässt sich testen um sicherzustellen, dass die eingegebenen Informationen einen Zugriff auf den Server zulassen.

Der PinCode ist nötig um sich dann mit diesem auf der Anmeldung anzumelden.

Die Daten werden gespeichert sobald diese Ansicht verlassen wird.

Weiter wird gegangen indem auf Projekt + Iteration geklickt wird. Fertig ist momentan deaktiviert. TFS + Benutzer ist farblich hervorgehoben um anzuzeigen, dass dies die aktive Ansicht ist.

2.1.3 Tag-Konfig: Projekt/Iteration

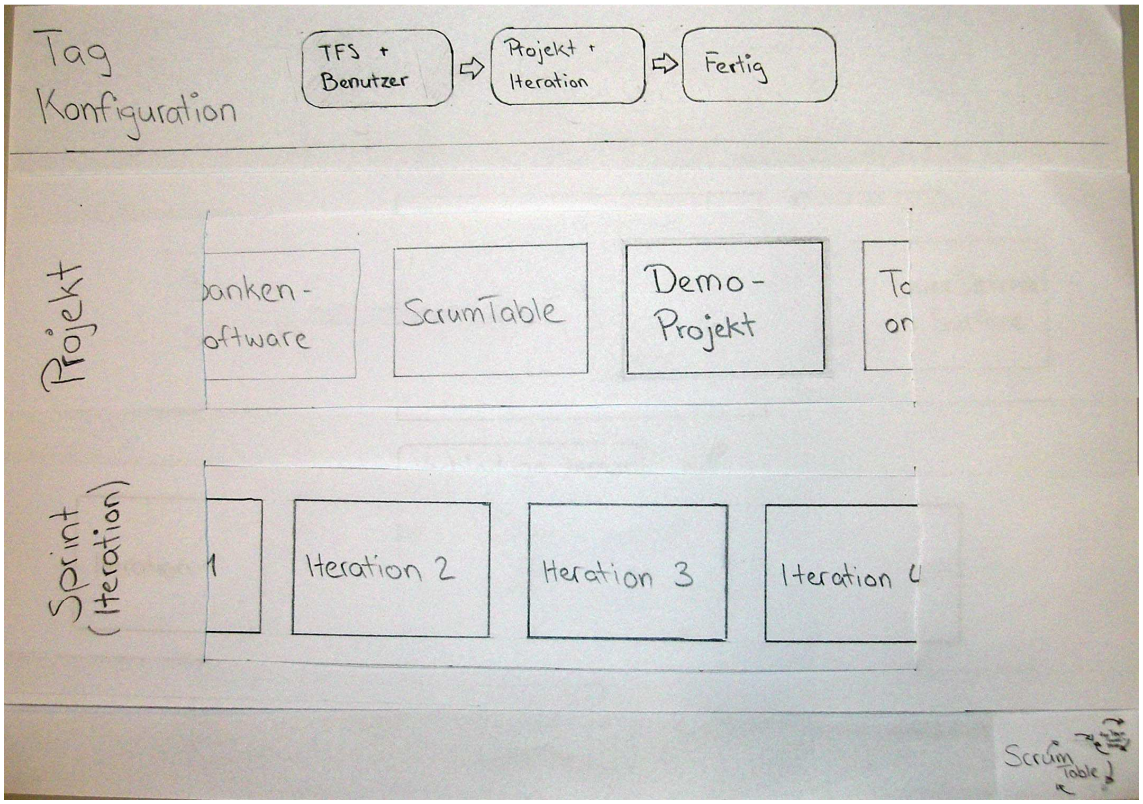


Abbildung 3: Version1 Tag-Konfig: Projekt/Iteration

Anhand der Anmeldeinformationen konnte nun eine Verbindung zum Microsoft Team Foundation Server aufgebaut werden und die Projekte ausgelesen werden. Erst werden die Iterationen nicht angezeigt. Der Benutzer wählt sein Projekt aus, worauf dieses farblich hervorgehoben wird. Dann erscheinen auch die zu diesem Projekt gehörenden Iterationen. Der Benutzer kann nun eine Iteration auswählen und dann auf Fertig klicken.

Der obere Knopf Projekt + Iteration ist farblich hervorgehoben um zu zeigen, dass dies die momentan aktive Ansicht ist.

2.1.4 Übersicht

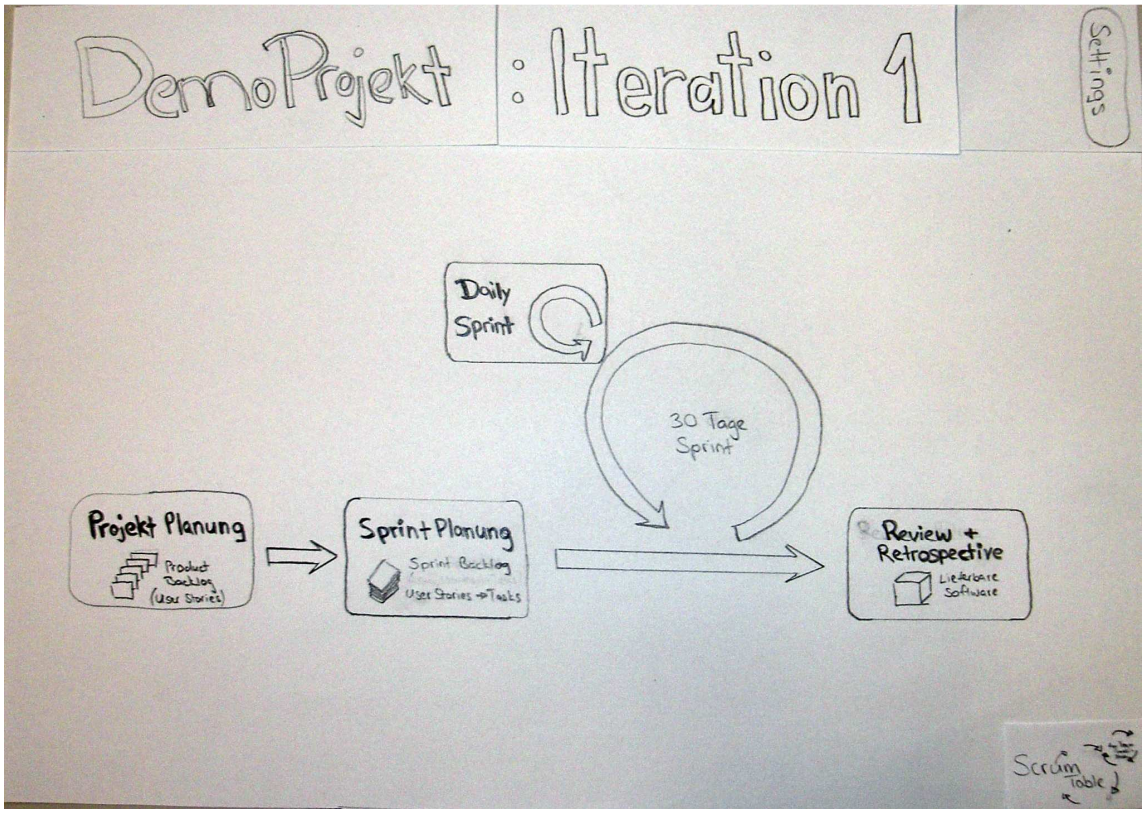


Abbildung 4: Version1 Übersicht

Durch klicken auf die Entsprechenden Sitzungen wird zu der entsprechenden Ansicht gewechselt.

Über Settings gelangt man zur Tag-Konfig: TFS.

2.1.5 Projekt Planung: User Stories

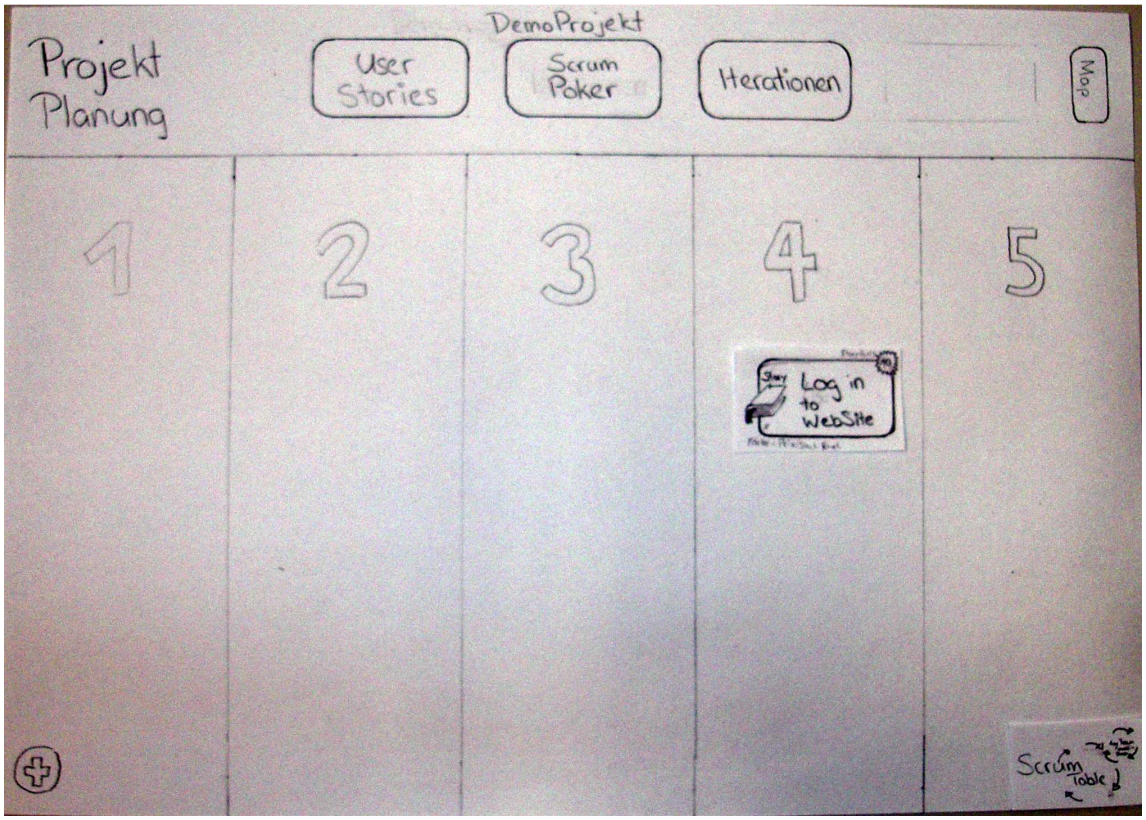


Abbildung 5: Version1 Projekt Planung: User Stories

Hier können User Stories priorisiert werden. Dabei können sie von Priorität zur anderen verschoben werden. Jede Priorität hat eine Farbe. Der Hintergrund der User Story besitzt jeweils die Farbe der Priorität.

Per Tappen der User Story kann diese editiert werden. Mit + links unten kann eine neue User Story hinzugefügt werden.

Im Menu kommt man mit dem Map Knopf wieder in die Übersicht.

2.1.6 Projekt Planung: ScrumPoker

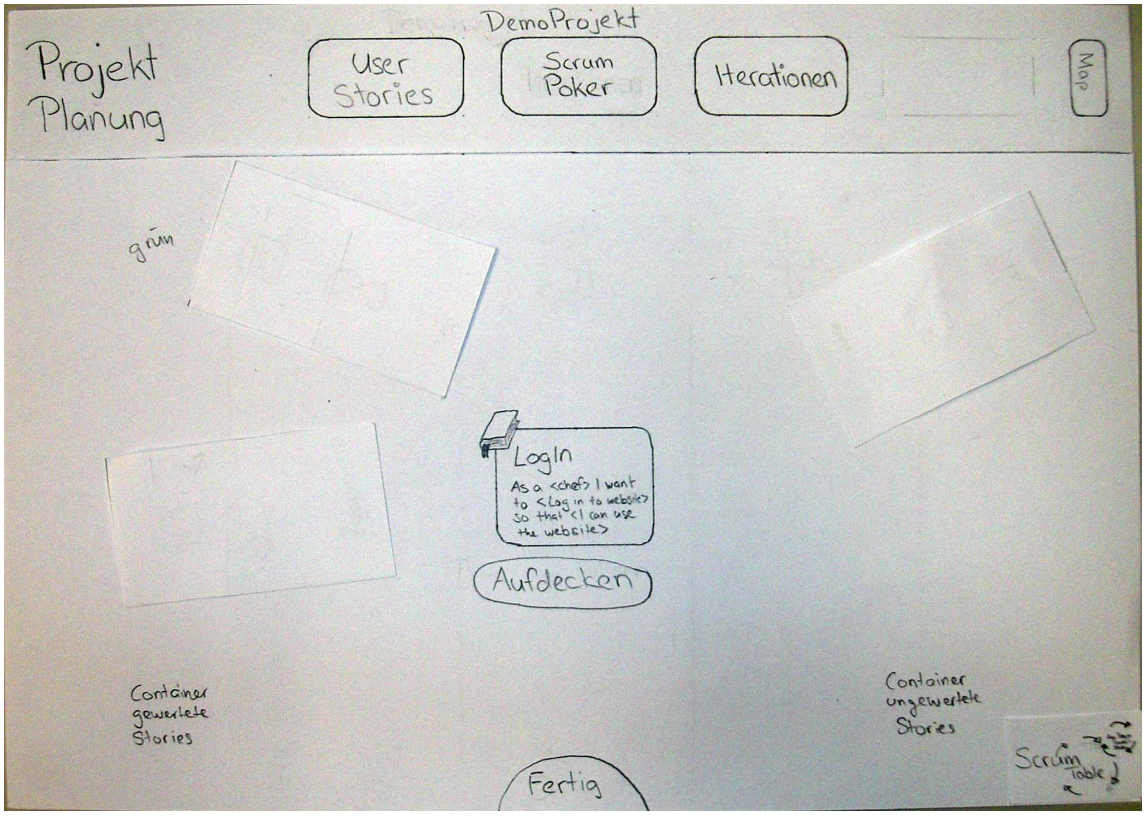


Abbildung 6: Version1 Projekt Planung: ScrumPoker

Man nimmt jeweils eine nicht gewertete User Story von rechts unten aus dem Stapel und legt sie in die Mitte. Jeder wählt eine der eigenen Karten aus und legt sie verdeckt auf den Tisch. Sobald alle ihre Karte abgelegt haben wählt der Scrum Master Aufdecken. Die Applikation zeigt darauf an, welche Karten gelegt wurden und was der Durchschnitt ist. Es kann dann danach auf den Durchschnitt geklickt werden um diesen zu wählen oder manuell ein Wert eingegeben werden. Die Karte wird dann auf den Stapel links gelegt, auf welchem sich die gewerteten User Stories befinden.

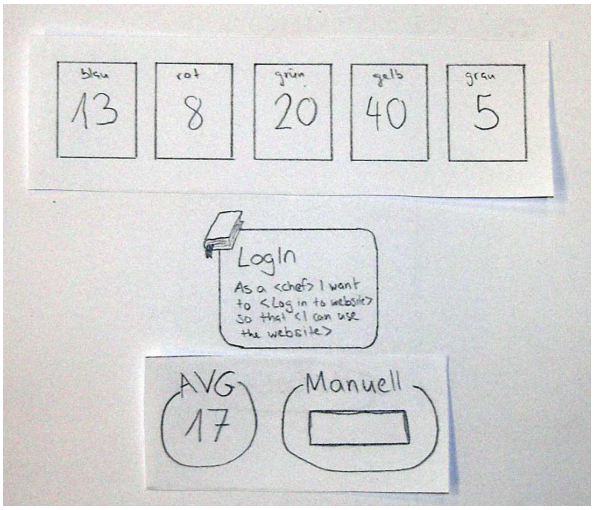


Abbildung 7: Version1 Projekt Planung: ScrumPoker aufgedeckt

2.1.7 Projekt Planung: Iterationen

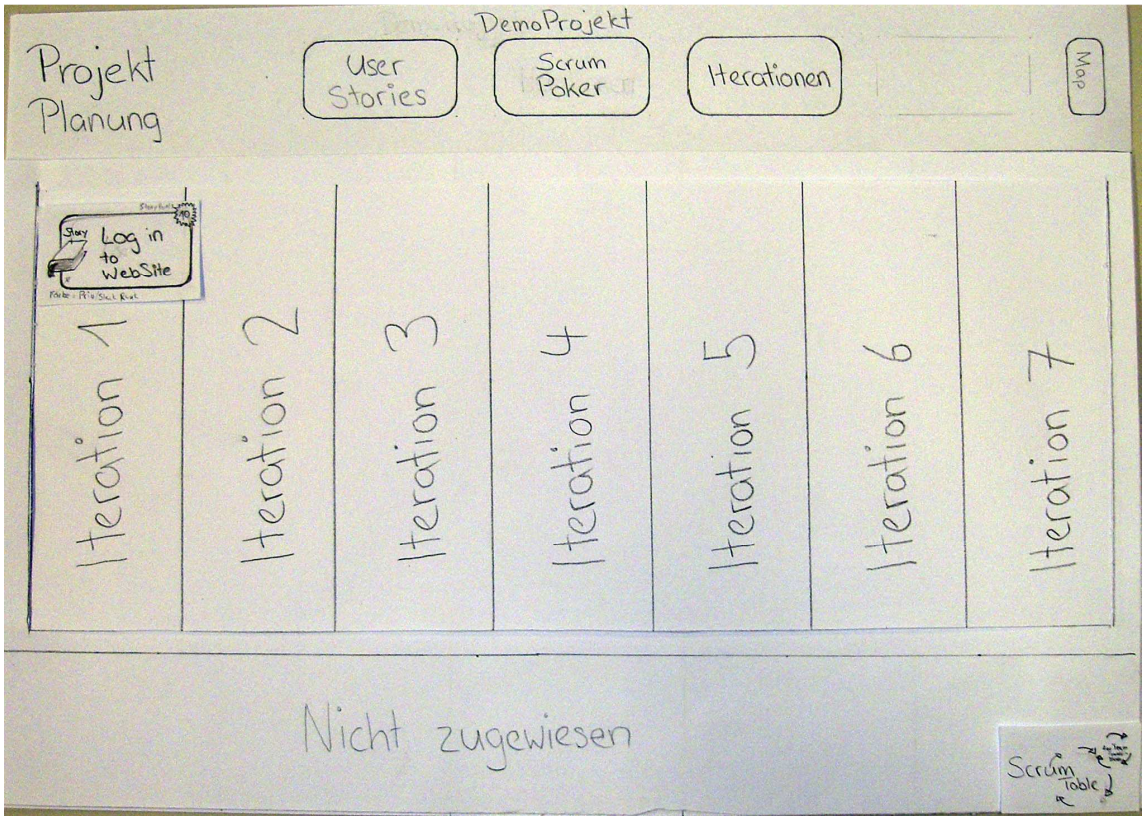


Abbildung 8: Version1 Projekt Planung: Iterationen

Hier können bereits vor dem Start einer Iteration User Stories an Iterationen zugewiesen werden. Die User Stories, welche noch keiner Iteration zugeordnet sind befinden sich unten in dem Feld der Nicht zugewiesenen User Stories.

2.1.8 Sprint Planung: Stories + Tasks

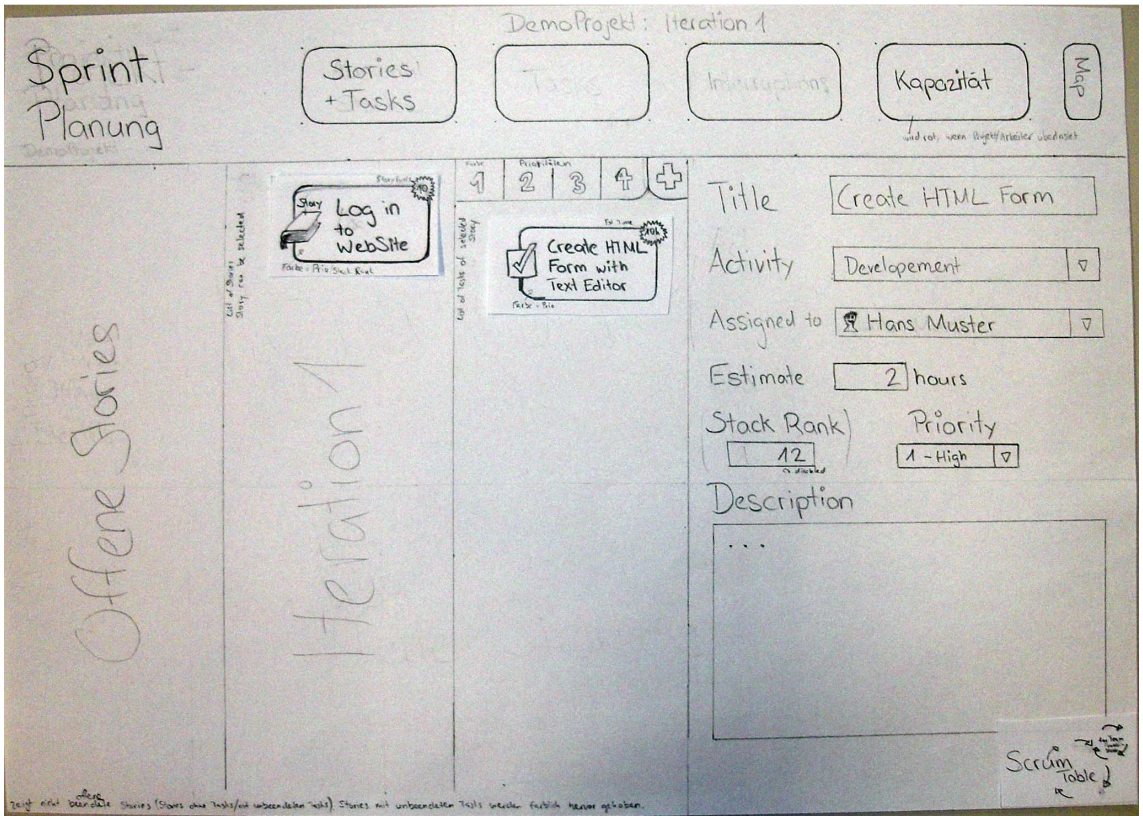


Abbildung 9: Version1 Sprint Planung: Stories + Tasks

Stories, welche noch nicht abgeschlossene Tasks haben oder welche noch gar keiner Iteration zugeordnet wurden befinden sich links unter Offene Stories. Diese können in die momentane Iteration gezogen werden und da ausgewählt werden. Darauf sieht man die zu dieser User Story gehörenden Tasks.

Die Prioritäten oberhalb der Tasks kann man durch ziehen auf einen Task dem Task zuordnen. Die Tasks lassen sich untereinander verschieben um so den Ablauf (Stack Rank) festzulegen.

Durch das + kann man einen neuen Task der momentan ausgewählten User Story hinzufügen.

Wenn man einen Task auswählt sieht man die Konfiguration rechts davon und kann daran Änderungen vornehmen.

Falls ein Entwickler oder ein Teammitglied zu viel Arbeit hat leuchtet der Kapazität Knopf rechts oben rot auf um dies dem Benutzer/Team mitzuteilen.

2.1.9 Sprint Planung: Kapazität

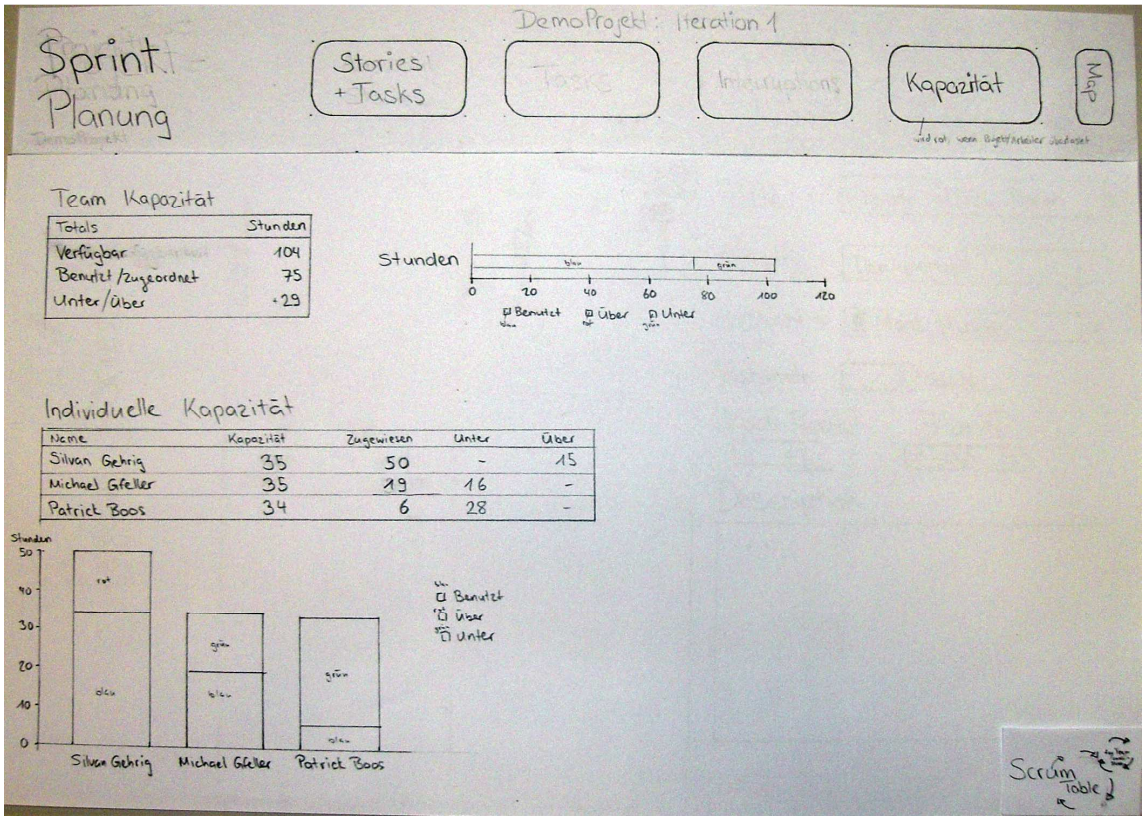


Abbildung 10: Version1 Sprint Planung: Kapazität

Auf dieser Ansicht kann man nichts verändern. Man sieht lediglich die Auslastung des gesamten Teams sowie der einzelnen Teammitglieder.

2.1.10 Daily Scrum: Individual Report

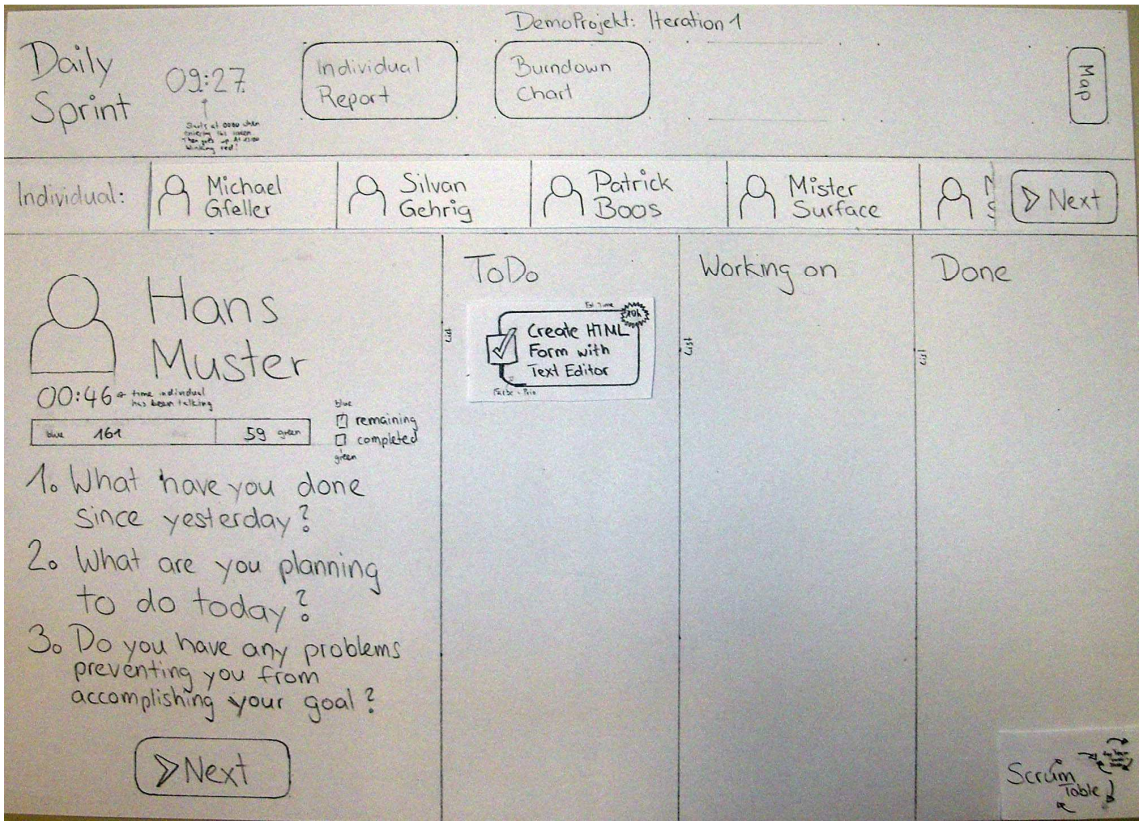


Abbildung 11: Version1 Daily Scrum: Individual Report

Jedes Teammitglied hat im Daily Scrum die Zeit kurz über seine Arbeit zu berichten. Dabei unterstützt diese Ansicht dies. Es wird angezeigt wie viel Zeit der Daily Scrum bereits dauert und wie lange die momentane Person am sprechen ist. Ist die Person fertig klickt die Person auf Next oder auf die nächste Person oben in der Liste.

Unter der Person wird dessen Fortschritt angezeigt. Dieser Fortschritt zeigt an wie viel Arbeit insgesamt während dem Sprint gemacht werden muss, und wie viel bereits abgeschlossen wurde.

Das Teammitglied kann die Tasks von ToDo nach Working on oder von Working on nach Done ziehen und dadurch die ersten zwei Punkte in seinem Bericht abarbeiten.

2.1.11 Daily Scrum: Burndown Chart

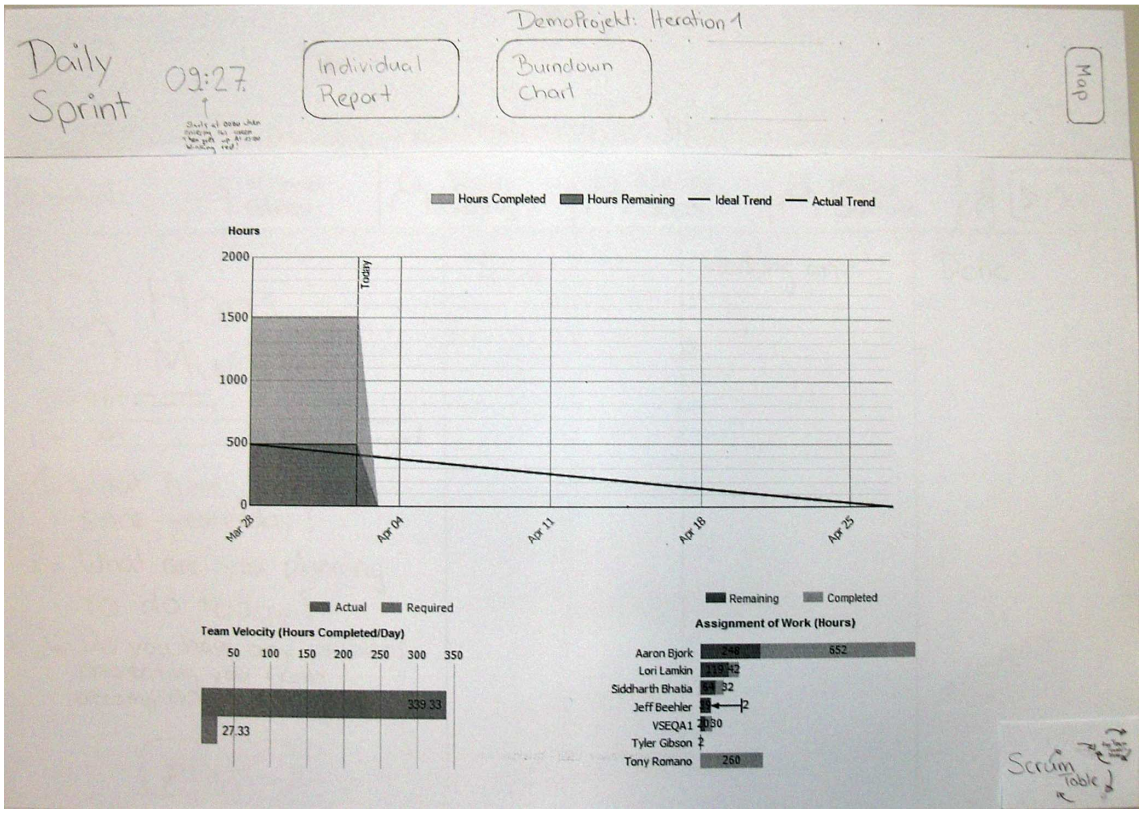


Abbildung 12: Version1 Daily Scrum: Burndown Chart

Diese Ansicht ist nur zur Darstellung des Burndown Chart. Verändert werden kann in dieser Ansicht nichts.

2.1.12 Review + Retrospective

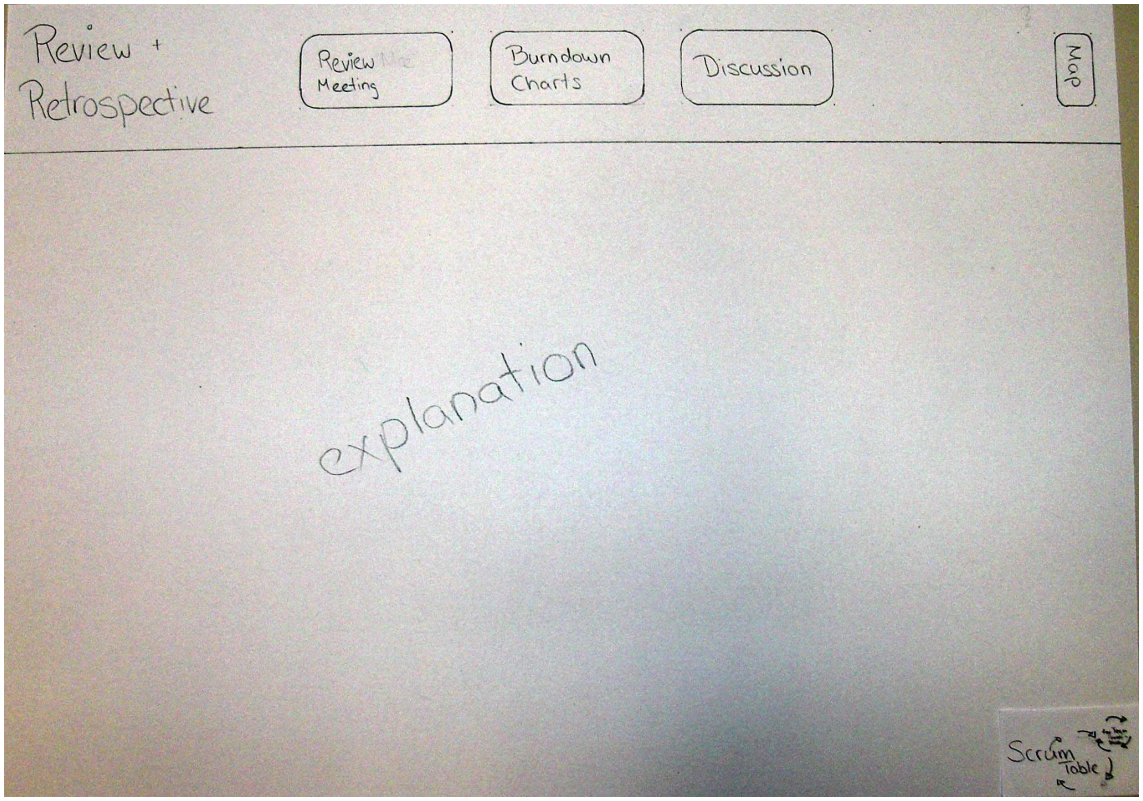


Abbildung 13: Version1 Review und Retrospective

Diese Ansicht wird eine statische Ansicht sein, in welcher nur eine Erklärung für das Review und die Retrospective angeboten wird. Diese Ansicht ist fraglich, ob sie überhaupt benötigt wird. Aber vollständigkeitshalber wird sie eingebunden. Im Review wird diese Ansicht wahrscheinlich nicht benutzt werden. Jedoch für die Retrospective könnten die Burndown Charts benutzt werden.

2.2 Elemente

2.2.1 User Story

Die User Story zeigt als Symbol ein Buch an womit die Story symbolisiert wird. Rechts oberhalb befindet sich die Angabe der Story Points, damit diese jeweils ersichtlich sind. Der Hintergrund der User Story ist die Farbe der Priorität.

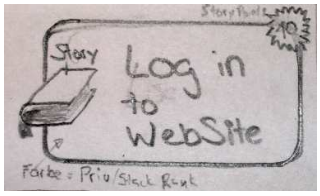


Abbildung 14: Version1 User Storv

2.2.2 Task

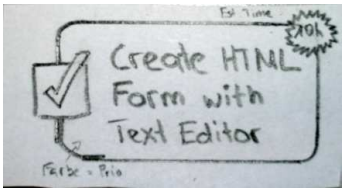
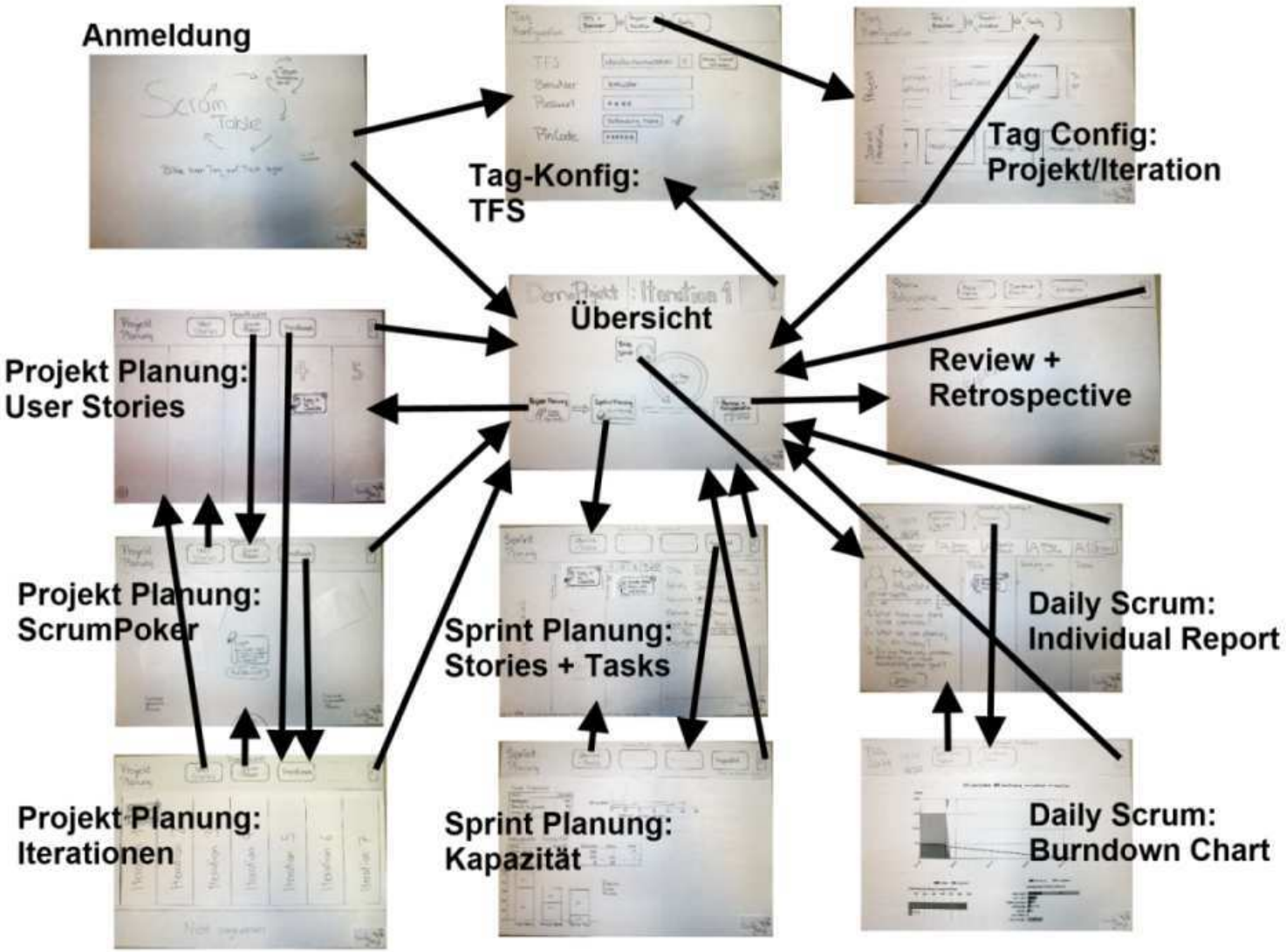


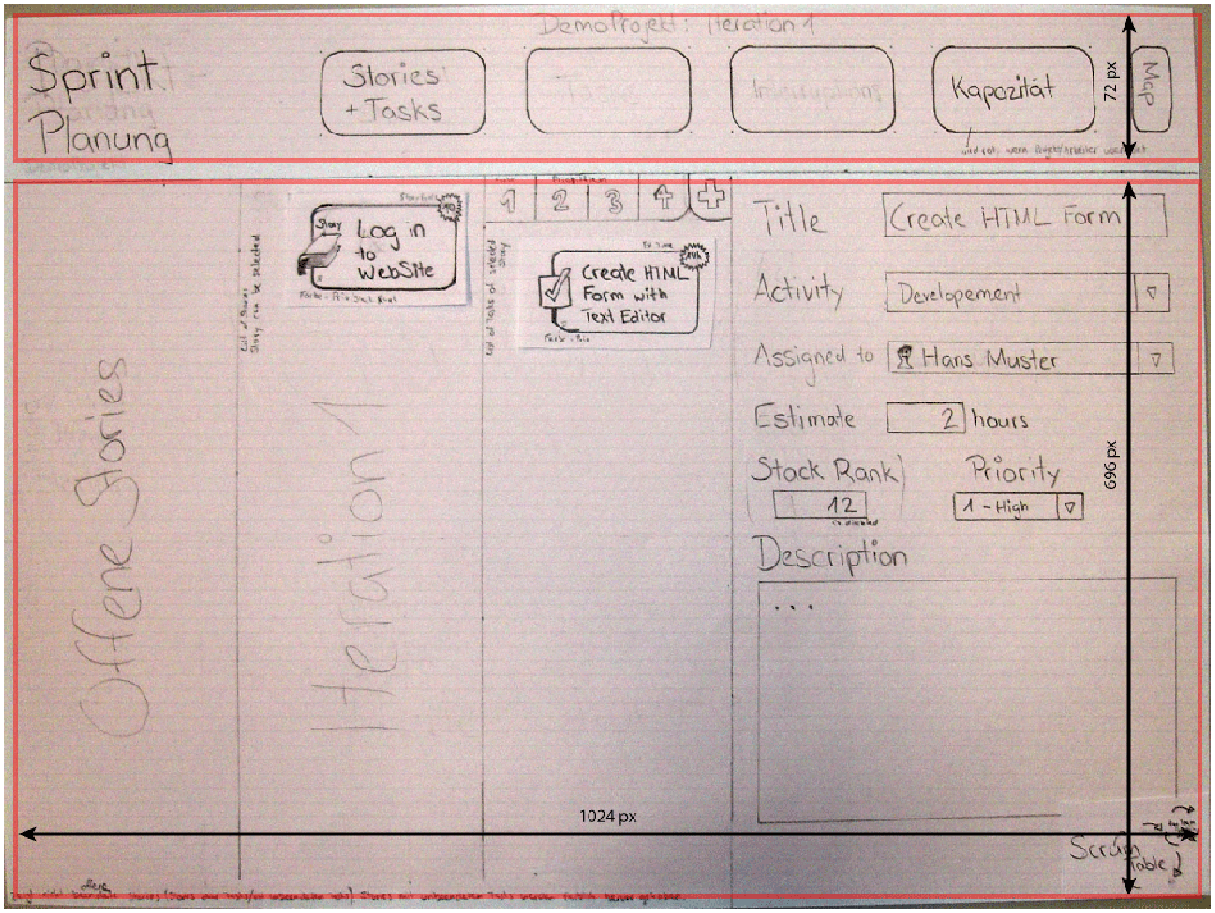
Abbildung 15: Version1 Task

Der Task ist vom Prinzip her gleich wie die User Story. Die Checkbox symbolisiert den Task. Rechts oberhalb wird angezeigt wie gross die geschätzte Dauer des Tasks ist. Mit der Farbe des Hintergrundes wird die Priorität ersichtlich.

2.3 GuiMap


Anmeldung
Tag-Konfig:
TFS
Tag Config:
Projekt/Iteration
Übersicht
Projekt Planung:
User Stories
Review +
Retrospective
Projekt Planung:
ScrumPoker
Sprint Planung:
Stories + Tasks
Daily Scrum:
Individual Report
Projekt Planung:
Iterationen
Sprint Planung:
Kapazität
Daily Scrum:
Burndown Chart

2.4 Visual Design



Das Visual Design (Framework) beschränkt sich auf einen oberen Navigations-Abschnitt und einen unteren Content-Abschnitt.

2.5 Reviews

2.5.1 Review mit Herr Stolze am 12. März 2010

Startup:
Was macht die Karte? Ich richte damit das Projekt ein? Tisch initialisieren?

Tag Konfiguration:
Wenn ich auf der falschen Seite gestanden, was passiert jetzt?
Wie drehe ich das Bild?
Wenn ich im TFS-Dropdown reingehen, dann sehe ich die TFS-Daten?
Man wird nicht selber tippen, der Benutzer ist bereits eingetragen?
Beim PinCode-Eingeben habe ich alles eingetragen?
Für was ist der PinCode? Hilfestellung?
Im BreadCrumb sehe ich, wo ich bin? Darf man da auch hinklicken?
Was passiert, wenn alle Werte Eingetragen werden? Ein „weiter"-Button wäre schöner?
Nachdem alles eingegeben wurde, was ist sichtbar? Was ist highlighted?
Im Projekt-Scroller sehe ich was? Dieses kann man verschieben?
Die Projekte werden high-lighted?
Was passiert, nachdem die zweite Iteration ausgewählt wurde? Geht's gerade weiter, sobald man auf die Iteration getippt hat?
Was heisst es, wenn keine Iteration ausgewählt wurde? Welche Bedeutung kommt diesem zu?
Macht es überhaupt sinn, eine Iteration auszuwählen? Wäre es sinnvoll, vor der ersten Iteration „Vor Projektstart" darzustellen?
Was wäre, wenn ich mit dem Projekt losgelegt habe und bin in der Iteration 3 und wähle die Iteration 1 an? Macht dies sinn?
Ich frage mich, ob es nicht sinnvoll wäre, nach dem Projekt wählen direkt auf die Übersicht zu gehen?
Wäre es sinnvoll, wenn ich nach der Auswahl des Projektes direkt auf die Iteration komme, welche zum aktuellen Datum aktiv ist?
Nach dem Verlassen des Tables und beim neuen Auflegen der Karte komme ich direkt wieder auf die letzte Iteration?

Scrum Übersicht:
Für was ist die Sprint-Planung, ist diese nur für einen Sprint?
Wäre es sinnvoll, wenn man direkt auf den Titel klicken könnte und so die Iteration und das Projekt wechseln könnte?

Projekt planen:
Was sind die einzelnen Abschnitte auf dem Bildschrim? Sind diese Rankings / Prioritäten?
Wenn ich die Items loslasse (zwischen zwei Rankings), dann fallen diese zurürck auf die Urposition?
Wo werden die nicht zugewiesenen hingelegt?
Wäre eine Leerliste angebracht?
Wie mache ich den CRUD-UserStory Dialog wieder klein (zu)? Wäre ein okay-Button angebracht?
Ist es innutitiv, die Stories per doppelklick mit einer Flip-Animation gross (CRUD ready) zu machen?
Warum soll ich die Story-Points/Rankings auch noch editieren können? Besser alles draussen lassen

und die Story-Points und Rankings entsprechend der anderen Dialoge planen.
Prioritäten sollte ich mit dem Product Owner diskutiert haben? Sind also Business-Priorities und Technische Priorities gemischt?
Wo kommt das eigentliche Iteration planning rein? Ist dies erst beim Sprint-Panning, oder bei den Iterationen?

Scrum Poker:
Wo ist meine Story? Ich bin überrascht, dass ich von rechts nach links arbeiten muss, d.h. gewertete Stories & ungewertete Stories tauschen.
Die Karte kommt nach unten?
Der Tisch merkt das alle Entwickler mitmachen? Die Entwickler werden identifiziert?
Die Karten sind immer noch pyhsisch da? Wann kommen die Karten wieder vom Tisch weg?
Könnte der Poker-Vorgang auch nochmals durchgführt werden? Repoker als Abhilfe.
Könnte die Storie nun auch aufgeteilt werden, da dies die Diskussion ergeben hat? Ein Storie-Split-Screen könnte Name, Ranking,… übernehmen.
Das Zuteilen der Storie (mit Verantwortung) zu einem Entwickler, wo geschieht dies? Sollte es nicht bereits in der Poker-Diskussion stattfinden? Würde dies aus Sicht der Scrum-Theorie sinn machen?
Dürfen irgedwelche Stories liegen bleiben? Kann man diese separieren, auseinander nehmen aufgrund Übersichtlichkeit?

Iteration planen:
Sieht man auf der Itetarions-Übersicht keine Daten oder so?
Könnten auch Iterationen eines anderen TFS angezeigt werden? Zum Beispiel nur angucken, d.h. einblenden? Low Prio, technisch möglich.
Nun seh ich auch gerade die Stories points, welche ich in der Iteration habe?
Seh ich nun auch gerade die Ferien? Ist dies nur eine Gesamtübersicht?
Wer macht den dies hier nun, ist dies mit dem Product Owner zusammen und mit den Entwicklern?
Wie kann die Applikation nun beendet werden? Wäre ein Exit-Button von Vorteil? Gut wäre eine Map- UND ein Exit-Button.

Sprint planen:
Werden die Stories automatisch ausgerichtet?
Hat man beim Scrum-Pokern bereits schon Ideen, was es für Tasks gibt? Wie könnte man diese Tasks beim Pokern bereits erfassen?
Könnte man nicht bereits bei den Personen die Auslastung anzeigen? Oder wäre es sinnvoll, das Team ganz rechts in einem geeigneten Bereich darzustellen mit allen Personen?
Wenn ich eine neue Task mache, dann könnte diese ja Flip-Over analog User Stories, damit wäre der rechte Bereich offen?

Daily-Sprint planen:
Ist der Ablauf nicht eher Working On->TODO (links-nach rechts)?
Sollten die Individuals nicht eher nach Random aufgelistet werden? Dies wäre besser.

Wäre es sinnvoll eine BAR einzublenden mit der Total-Zeit?
Könnte man nicht die Benutzer per Drag&Drop aus der Liste ziehen (schmeissen)?

2.6 Änderungen nach Reviews

2.6.1 Abmelden
Möglichkeit sich abzumelden auf der Übersicht und in den verschiedenen Bereichen

2.6.2 Erklärung PinCode
Beim Konfigurieren des Tags eine Information zu dem PinCode, damit der Benutzer weiss, wozu dieser benötigt wird.

2.6.3 Iteration automatisch erkennen
Iteration wird beim Anmelden automatisch erkennt anhand des momentanen Datums und den Start- und Endterminen der Iterationen.

2.6.4 Tag-Konfig: Weiter Knopf
Damit der Benutzer weiss, wie es weiter geht nach dem Eingeben der Daten wird ein „Weiter“ Knopf eingeführt.

2.6.5 Projekt Planung: User Stories: Nicht-priorisierte Stories
In der Priorisierungsansicht der User Stories in der Projekt Planung wurde eine zusätzliche Kategorie angelegt, in welcher sich die User Stories befindet, denen keine Priorität zugewiesen wurde.

2.6.6 Projekt Planung: Iterationen: Story Points und Datum der Iteration anzeigen
Damit mehr Informationen über die Iterationen ersichtlich sind, wird neben dem Namen der Iteration auch oberhalb die Anzahl der Story Points sowie unter dem Namen noch die Zeitperiode angezeigt.

2.6.7 ScrumPoker: Bitte Wertung abgeben
Einen Text einblenden, der die Sitzungsteilnehmer auffordert nun ihre Wertung abzugeben indem sie die entsprechende Karte verdeckt auf den Tisch legen.

2.6.8 ScrumPoker: von links nach rechts
Noch nicht gewertete Stories werden links anstatt rechts abgelegt. Fertig gewertete Stories werden rechts in einen Container gelegt.

Der Container lässt sich „aufklappen“, damit man darin auch eine User Story suchen kann.

2.6.9 ScrumPoker: erneut spielen
Ein weiterer Knopf bei Average und Manuell soll eingeführt werden, damit nach einer Diskussion erneut für die selbe User Story eine Runde ScrumPoker gespielt werden kann.

2.6.10 Übersicht: Wechsel der Iteration durch Klick auf momentane Iteration
In der Übersicht wird, wenn momentan keine Iteration aktiv ist, ein Platzhalter (z.B. <keine Iteration>) angezeigt. Darauf oder auf die momentane Iteration kann geklickt werden um eine Iteration auszuwählen.

2.6.11 Daily Scrum: Abgelaufene Zeit der momentan Person oberhalb

Die abgelaufene Zeit der Person soll oberhalb des Bildes dargestellt werden anstelle unterhalb, damit diese schneller ersichtlich ist.

2.6.12 Sprint Planung: Tasks anders editieren und Auslastung in gleicher Ansicht

Der momentane Bereich für das Editieren des Tasks soll dafür genutzt werden die Auslastung darzustellen und alle Team Members darzustellen. Unter diesen wird auch gleich angezeigt, wie gross diese ausgelastet sind. Die Team Members können auf einen Task gezogen werden, wodurch der Task dann diesen zugeordnet wird. Per Klick auf ein Team Member werden kurz die Tasks hervorgehoben, die diesem zugewiesen sind.

Das Item kann umgedreht werden um dann editiert zu werden. Darin werden nur noch die nötigen Felder dargestellt (Title, Activity, Estimate, Description).

2.6.13 Daily Sprint umbenennen in Daily Scrum

In der Übersicht und in der Daily Scrum ansicht wurde jeweils Daily Sprint geschrieben, was falsch ist. Sollte überall auf Daily Scrum geändert werden.

3 Version 2

In dieser Version wurden hauptsächlich die im vorherigen Kapitel erwähnten Änderungen nach dem Review umgesetzt. Die Ansichten werden hier nur vollständigkeitshalber eingefügt jedoch nicht nochmals beschrieben.

3.1 Ansichten

3.1.1 Anmeldung

Bleibt gleich wie in Version 1.

3.1.2 Tag-Konfig: TFS

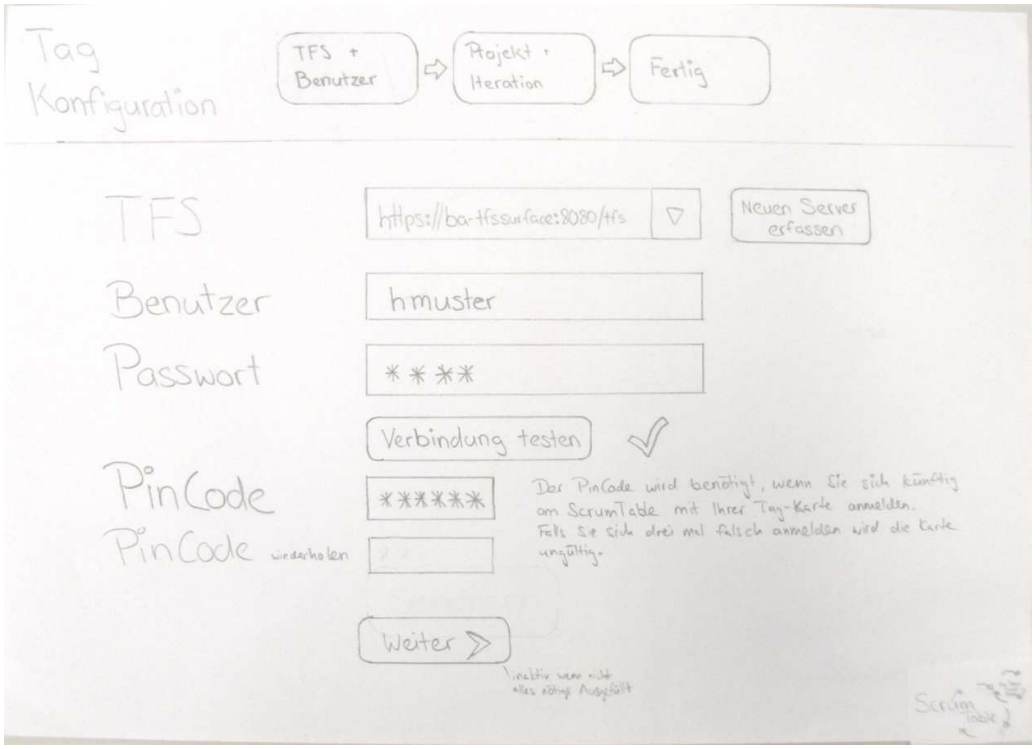


Abbildung 16: Version2 Tag Konfig: TFS

3.1.3 Tag-Konfig: Projekt

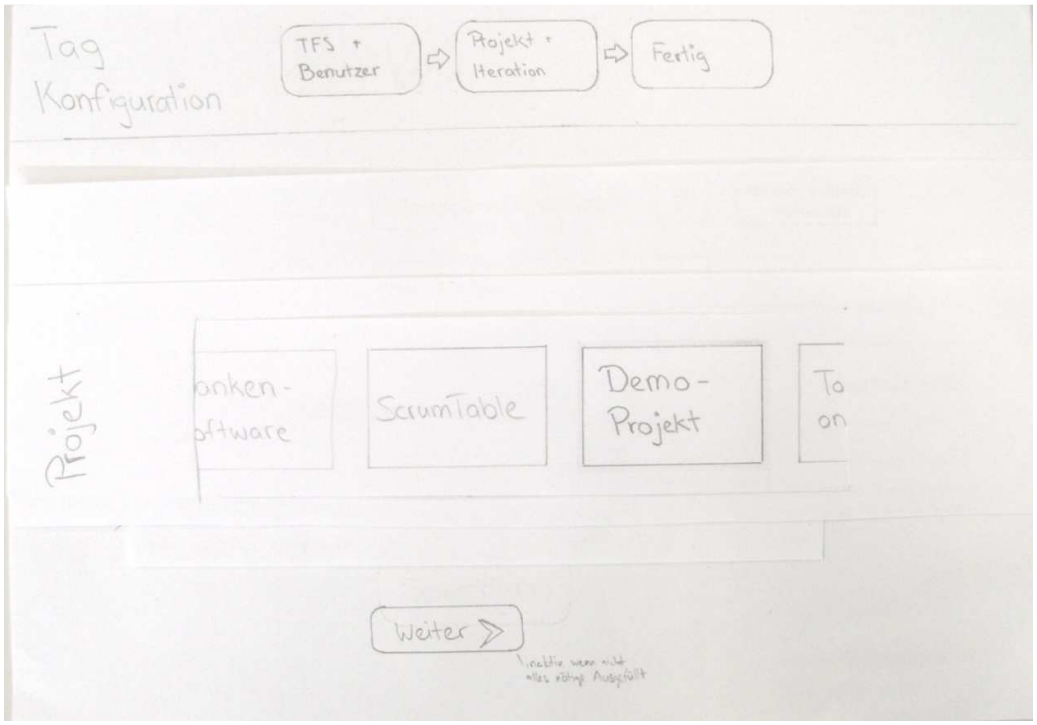


Abbildung 17: Version2 Tag Konfig: Projekt

3.1.4 Übersicht

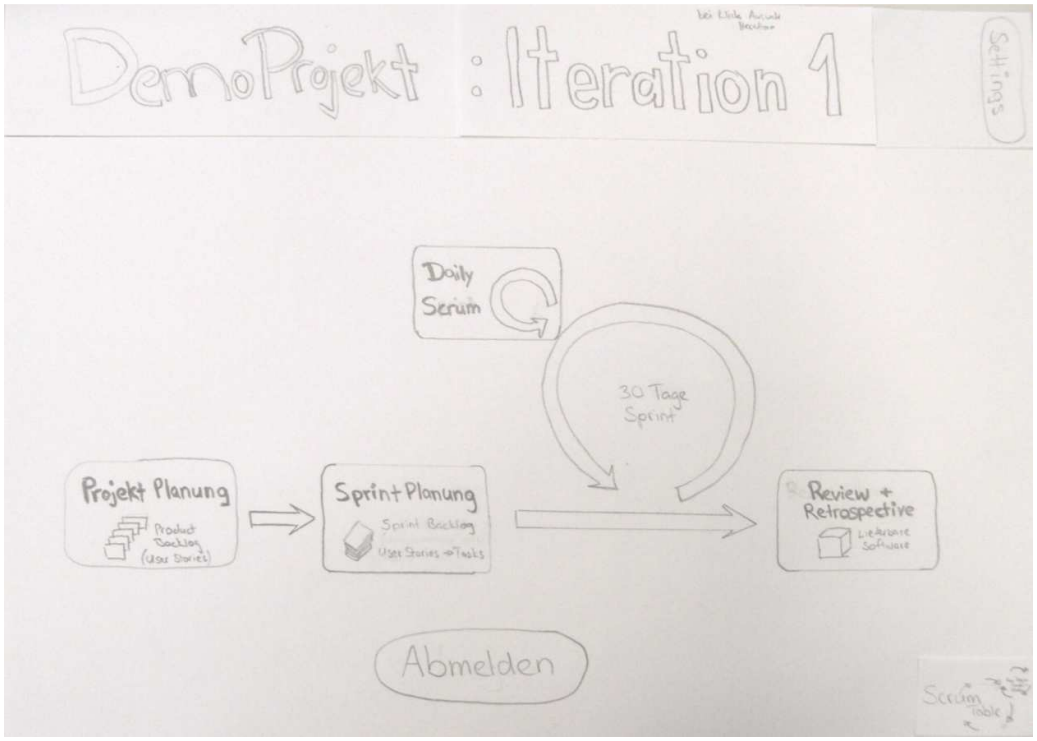


Abbildung 18: Version2 Übersicht

3.1.5 Projekt Planung: User Stories

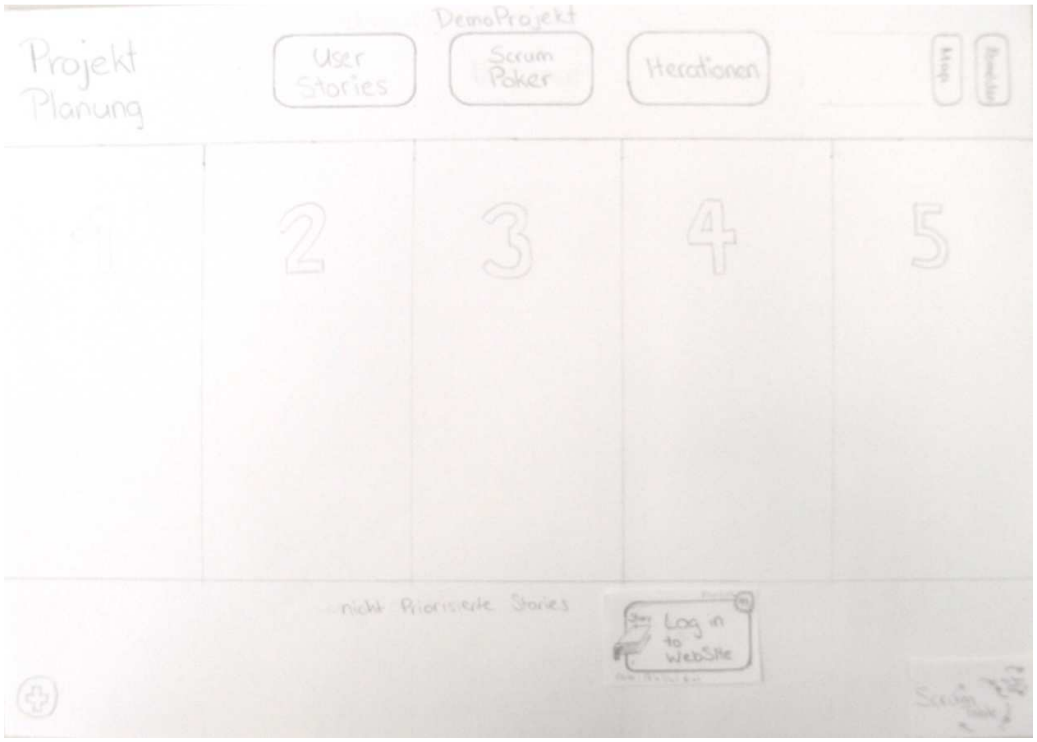


Abbildung 19: Version2 Projekt Planung: User Stories

3.1.6 Projekt Planung: ScrumPoker

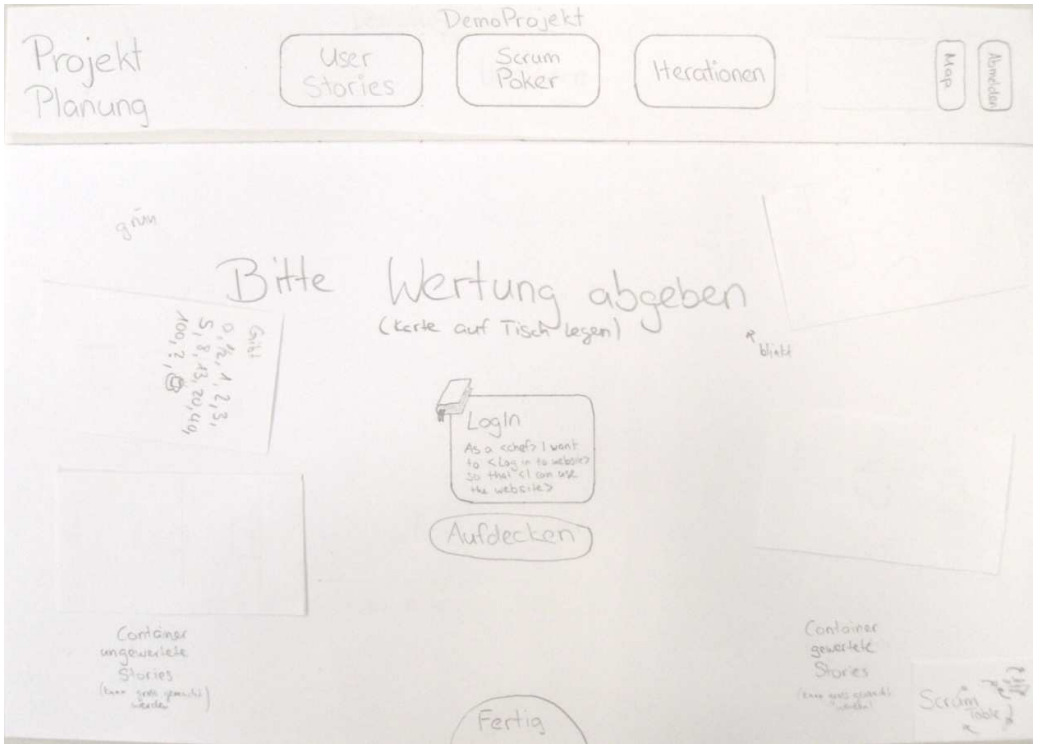


Abbildung 20: Version2 Projekt Planung: ScrumPoker

3.1.7 Projekt Planung: Iterationen

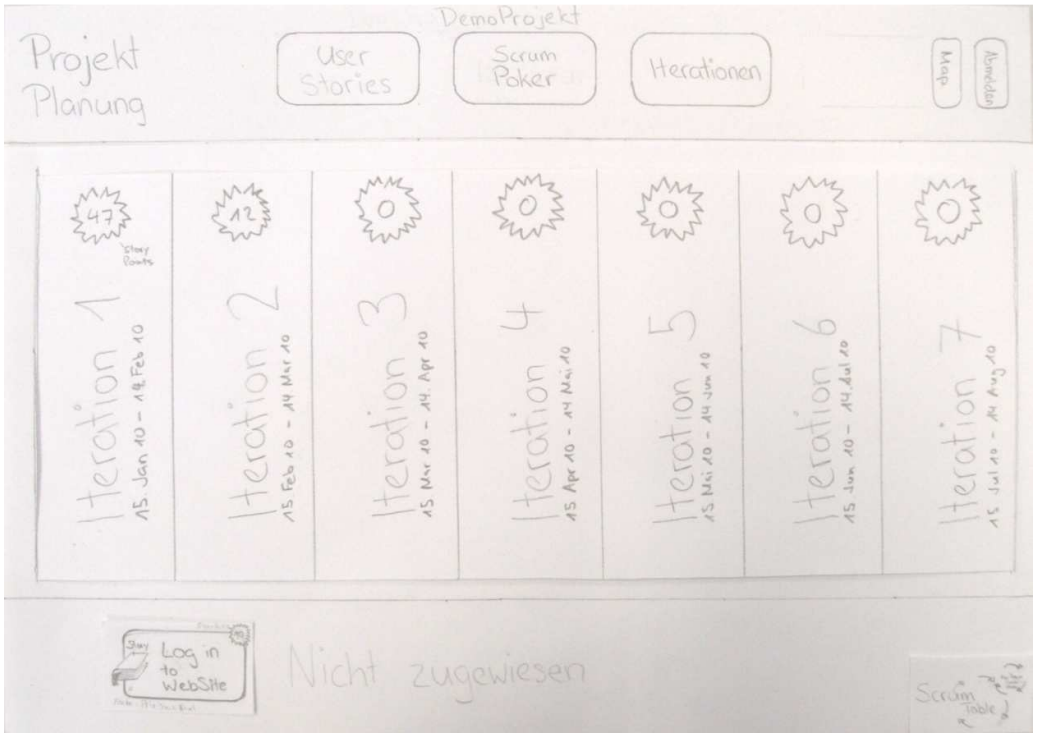


Abbildung 21: Version2 Projekt Planung: Iterationen

3.1.8 Sprint Planung: Stories + Tasks

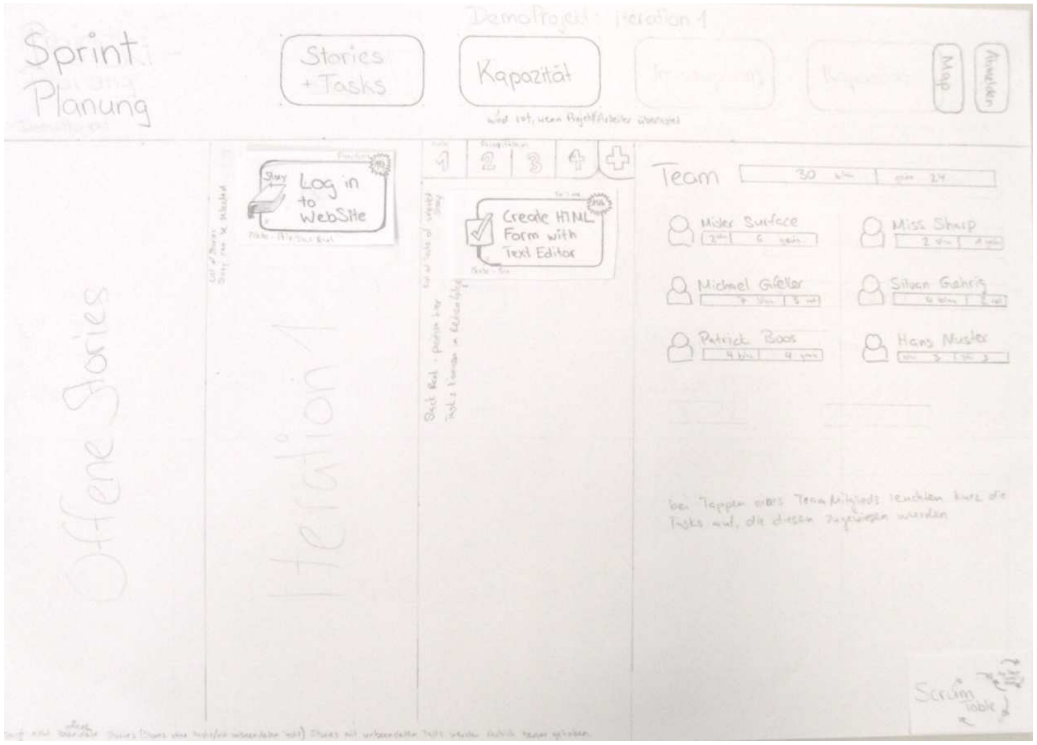


Abbildung 22: Version2 Sprint Planung: Stories + Tasks

3.1.9 Sprint Planung: Kapazität

Bleibt gleich wie Version 1.

3.1.10 Daily Scrum: Individual Report

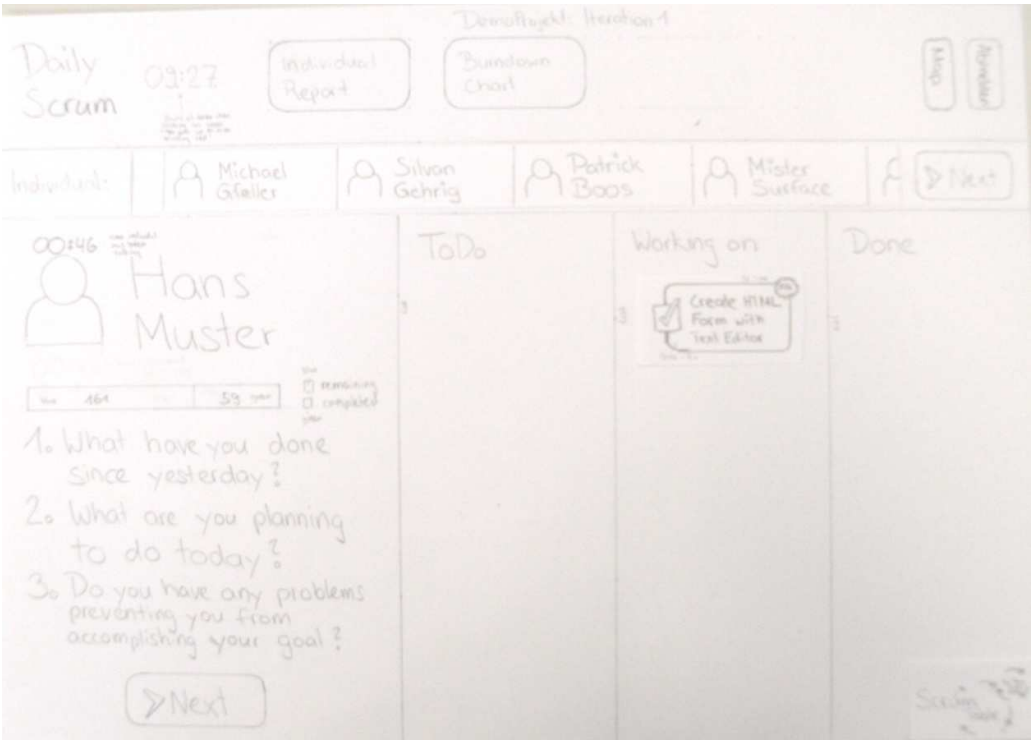


Abbildung 23: Version2 Daily Scrum: Individual Report

3.1.11 Daily Scrum: Burndown Chart

Bleibt gleich wie in Version 1.

3.1.12 Review + Retrospective

Bleibt gleich wie in Version 1.

3.2 Elemente

3.2.1 User Story

Bleibt gleich wie in Version 1.

3.2.2 Task

Bleibt gleich wie in Version 1.

3.3 GuiMap

Bleibt gleich wie in Version 1.

3.4 Visual Design

Bleibt gleich wie in Version 1.

3.5 Reviews

3.6 Änderungen nach Reviews

3.6.1 Knöpfe für Tools und Navigation

Zur einfacheren Handhabung des Programms sollen „Schnell-Knöpfe“ eingebaut werden. Diese sollen aus der Seite der Applikation herausragen und durch klicken aufgeklappt werden. Darin befinden sich Tools (z.B. Bildschirm drehen oder eine Liste aller Entwickler zum zuweisen) oder die Navigation, damit man gleich an eine Stelle im Programm springen kann.

3.6.2 Projekt Planung – User Stories - Stack Rank

Es hat sich bewährt die User Stories erst nach „Must“, „Should“ und „Could“ zu sortieren, wenn man diese priorisiert. Deshalb sollen in der Ansicht anstelle von Behältern für 1-5 nun 3 Behälter für „Must“, „Should“ und „Could“ eingesetzt werden. Ebenfalls soll dann innerhalb der Behälter die Positionierung durch schieben bestummen werden können. Dabei geht es von oben links nach rechts unten.

3.6.3 Skizze hinzufügen

Möglichkeit einer User Story eine Skizze hinzuzufügen.

Durch einen Knopf im Editierfenster der User Story soll es möglich sein eine User Story hinzuzufügen. Dabei soll diese gleich von Hand auf den Bildschirm gezeichnet werden können. Ebenfalls sollte die Möglichkeit geboten werden per Handy ein Bild von einer bestehenden Skizze zu machen und diese per Bluetooth an ScrumTable zu senden.

3.6.4 Retrospektive Items erfassen und auf Daily Scrum Ansicht anzeigen

Diese in dem Meeting erfassen können und dann während Scrum Meetings immer wieder sichtbar darstellen, damit diese nicht vergessen werden.

3.6.5 Bugs in Daily Scrum

An einem Ort sollen die wichtigsten Bugs sichtbar sein und einem Entwickler zuweisbar sein.

In der Ansicht der Personen sollen neben den Tasks auch die Bugs angezeigt werden, die der Person zugeordnet sind.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Implementation

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Quellenverzeichnis

Rational Software Corp., Philippe Kruchten. (14. Juni 2010). *The University Of British Columbia, Architectural Blueprints 4+1*. Abgerufen am 14. Juni 2010 von Architectural Blueprints 4+1: http://www.cs.ubc.ca/~gregor/teaching/papers/4+1view-architecture.pdf

1 Einführung

1.1 Zweck

Die Implementations-Details beschreiben die diversen technischen Vorgänge innerhalb von ScrumTable. Diese technisch tiefgehende Beschreibung dient dazu, den fachlich geschulten Leser die Weiterentwicklung des Projekts möglichst einfach zu gestalten.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Befasst sich zuerst mit der Benutzerschnittstelle dann mit dem Business Layer und zuletzt mit Data Layer.

2 User Interface Komponenten

2.1 Unity Container

Der Unity Container beinhaltet folgende Objekte:

IUnityContainer	Selbstregistration damit andere Klassen den Container als Service Locator nutzen können.
ApplicationArguments	Die Argumente, welche bei Applikationsstart übergeben wurden. Werden verwendet um Fehlermeldungen darzustellen, beim automatischen Neustart.
LoginAccountManager	Manager für die Login Accounts
ScrumViewModelData	Zentrale Komponente für die Daten des aktuellen Benutzers
ScrumController	Controller für die Navigation
IRunTaskAsyn	Verantwortliche Klasse für das Ausführen von asynchronen Aufrufen.
IDataSyncDispatcher	Wird benötigt um asynchrone Callbacks vom Datenlayer zu resynrchonisieren.
AppScreen	Das MainWindow stellt zentrale GUI-Funktionen zu Verfügung.

Der Container wird vor allem verwendet um die View-ViewModel Struktur aufzubauen.
Ablauf:

1. Container.Resolve<StartScreenUserControl>()
2. Neues StartScreenUserControl wird erstellt
3. [Dependency] Objekte werden Injiziert mit:
    a. Falls vorhanden mit einer regestierten Instanz
    b. Falls keine regestierte Instanz vorhanden ist, wird eine Neue erstellt. Wird nicht dem Container hinzugefügt.

```
[Dependency]
public IUnityContainer Container { get; set; }

[Dependency]
public DailyScrumIndividualReportViewModel Model
{
```

Für das Container- Property wird die regestierte Instanz injiziert.
Bei Model wird eine neue Instanz generiert.

Abbildung 1 ScrumUserControl verlangt den Container und sein ViewModel

2.2 Navigation-Handling

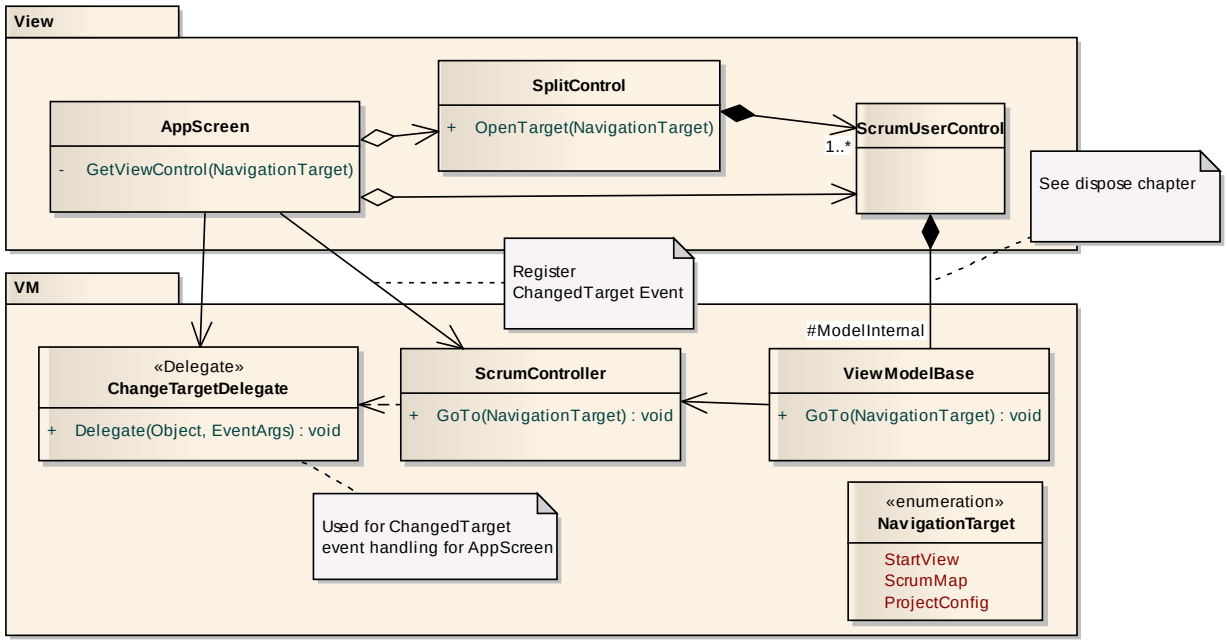


Abbildung 2 Navigationsübersicht

Der Ablauf sieht wie folgendes aus:

1. Der User klickt ein Navigationselement an (z.B. Button)
2. Via Property Binding wird im ViewModelBase das aktuelle Ziel umgestellt
3. Dies löst ein GoTo auf dem ScrumController aus. Dieser meldet dies als Event dem AppScreen.
4. Der AppScreen berechnet die neue Ansicht und stellt diese dar.

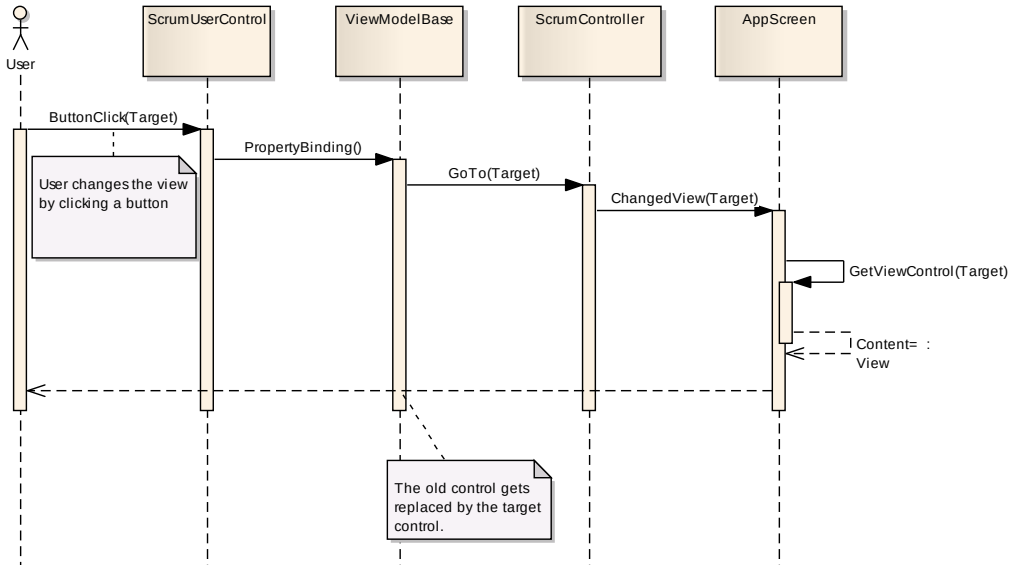


Abbildung 3 Navigationsablauf

2.3 Dispose von ViewModel

Um die Controls sauber aufzuräumen, implementieren die Basisklassen das Dispose-Pattern. Wenn der AppScreen ein ScrumUserControl austauscht, wird auf dem alten Control Dispose() aufgerufen. Dies ist vor allem nötig, um Events sauber aufzuräumen.

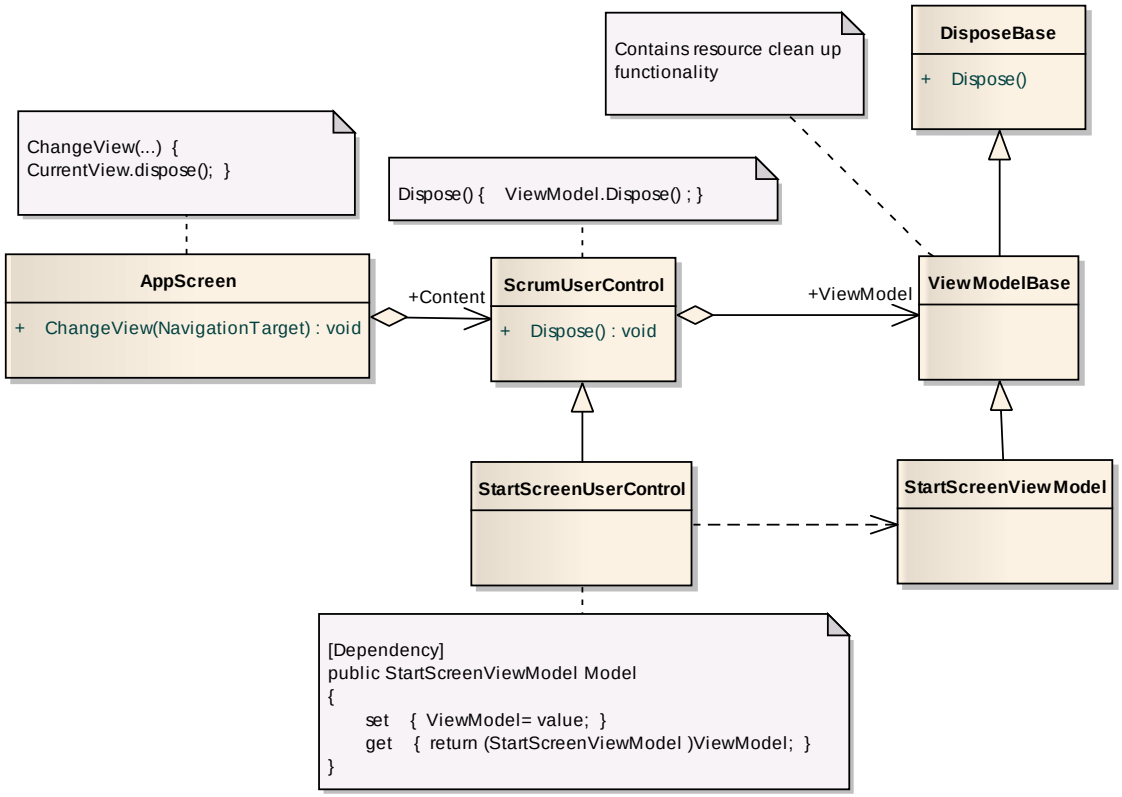


Abbildung 4 ScrumUserControl mit ViewModel erstellen

2.4 Overlay „Windows“ / Adorner

ScrumTable benötigt für verschiedene Funktionalitäten die „Overlay“ Funktion.
z.B. Darstellen von Tools / Editieren von Items / Darstellen von Informationen wie aktuelles Project / Iteration / Anzeigen der aktuellen Ansichten im Hintergrund.

Abbildung 5 Beispiel für Overlay

WPF unterstützt Overlay Windows als Adorner. Diese sind noch nicht in die Surface SDK integriert. Um dies mit der Surface SDK zu realisieren, werden auf den AppScreen drei Layers mithilfe von Grids erzeugt.

Items, welche den Grids hinzugefügt werden, übernehmen die Z-Order ihres Grids. Damit ist es möglich, die Overlays einfach zu realisieren:

```
<Grid>
    <Grid Width="1024" Height="768" Name="BackGrid">
        <Grid.LayoutTransform...>
    </Grid>
    <Grid  Width="1024" Height="768" Name="FrontGrid">
        <Grid.LayoutTransform...>
        <s:ScatterView Width="1024" Height="768" Name="ItemView"/>
    </Grid>
    <Grid Width="1024" Height="768" Name="ToolGrid">
        <Grid.LayoutTransform...>
        <Tool:ScrumToolbar Width="1024" Height="768" x:Name="Toolbar"/>
    </Grid>
</Grid>
```

Abbildung 6 Overlay mit Grids

BackGrid: Auf diesem Grid liegen alle ScrumUserControls. (Beispiel: Scrum Poker)

FrontGrid: Auf diesem Grid können Controls hinzugefügt werden, welche vor den ScrumUserControls liegen sollen. Ebenso stellt dieses Grid eine ScattertView zu Verfügung. (Beispiele: Control zum Editieren von User Story)

Zum Ansteuern dieses Grids stellt der AppScreen 4 Funktionen zu Verfügung. Siehe Abbildung 7 AppScreen.

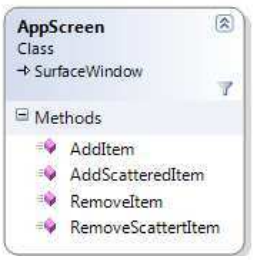


Abbildung 7 AppScreen

ToolGrid: Auf diesem Grid liegen die Scrum Tools und Informationen, welche überall sichtbar sein sollen. (Beispiel: Menü zum Navigieren)

2.5 Login /Registration

Der Ablauf des Logins läuft wie folgt ab:

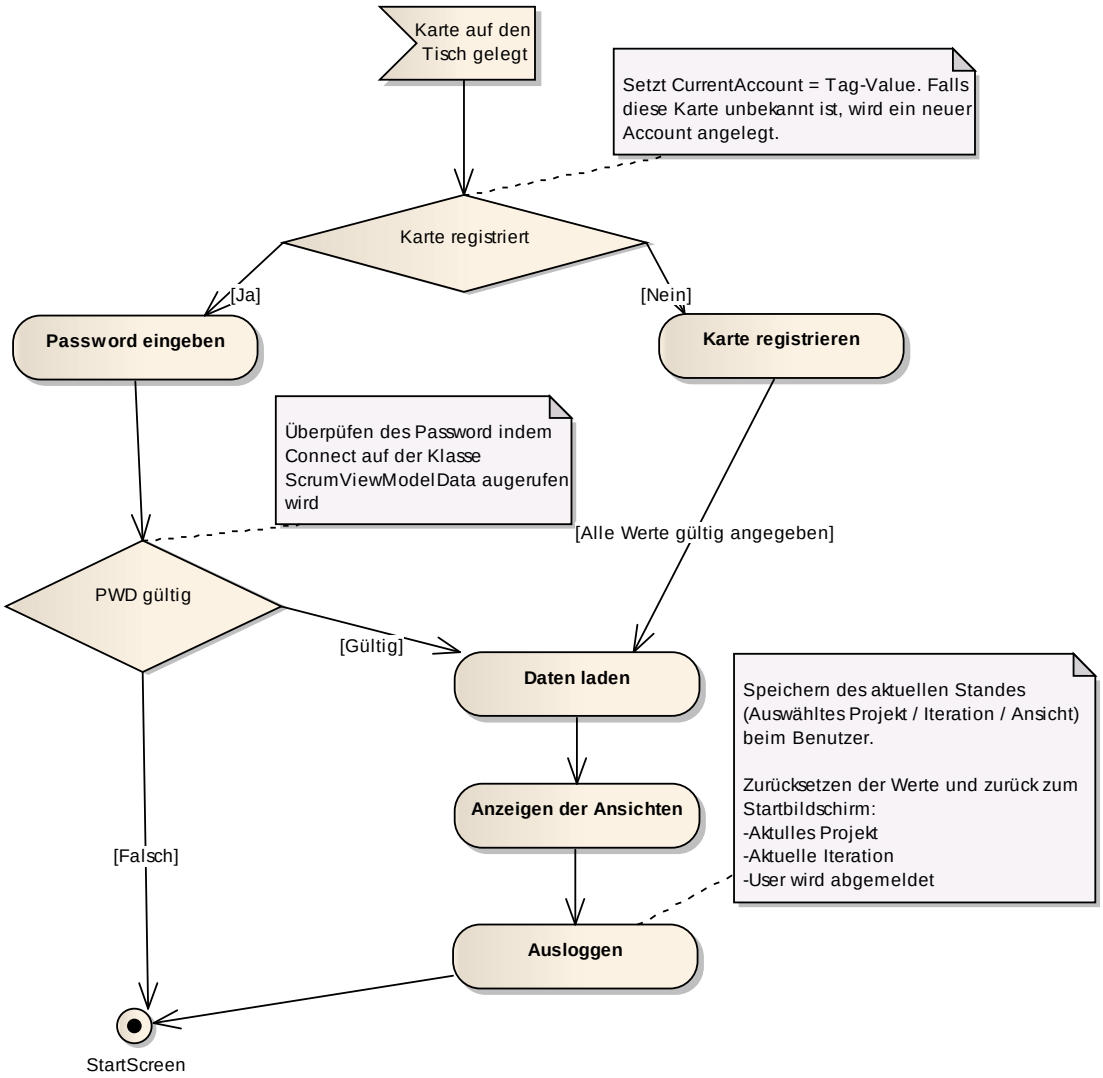


Abbildung 8 Login Ablauf

2.6 Drag & Drop allgemein

Folgend wird aufgezeigt wie der Ablauf einer Drag & Drop Operation aussieht.
Der weggelassene Drag & Drop Manager fragt nach dem starten der Drag & Drop Operation jedes Control mit AllowDrop=true beim eintreten dessen Zone, ob er Interesse an diesem Drag-Objekt habe. Falls ja, wird beim Fallenlassen dieses Control ausgewählt, um das Drag & Drop-Event zu bearbeiten. Falls kein Control gefunden wird, wird die Operation abgebrochen.

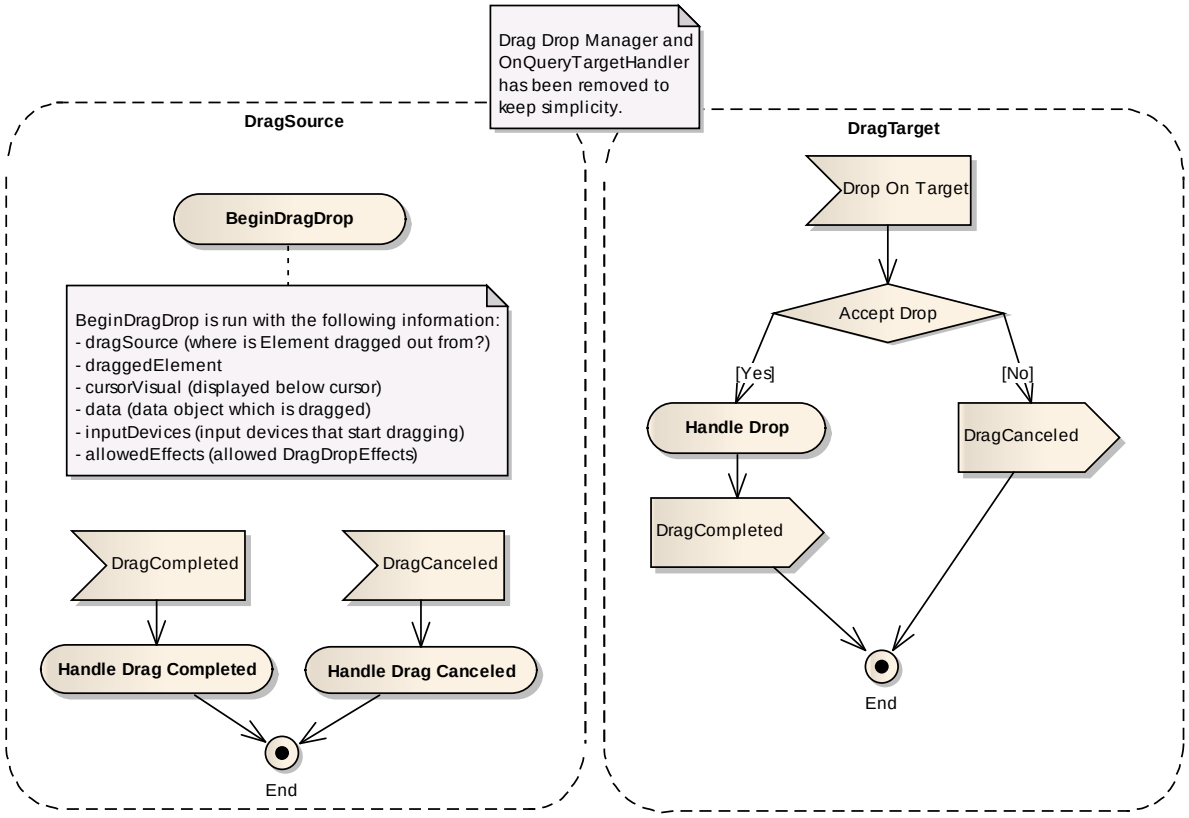


Abbildung 9 Drag & Drop

2.7 SurfaceComboBoxControl

2.7.1 Ausgangslage

Das Surface SDK bietet dem Benutzer viele Möglichkeiten für Controls, die per Touch bedient werden können. Dazu gehören:

- SurfaceButton
- SurfaceCheckBox
- SurfaceInkCanvas
- SurfaceListBox
- SurfaceMenu
- SurfaceRadioButton
- SurfaceTextBox, SurfacePasswordBox
- SurfaceSlider

Jedoch fehlt ein Element um ein ComboBox per Touch zu bedienen.

2.7.2 Anforderungen

Das Verhalten sollte gleich sein wie bei ComboBox. Dies umfasst folgende Punkte:

- ItemsSource (auch mit Binding)
- SelectedItem (auch mit Binding)
- ItemTemplate setzbar

Zusätzlich zur normalen ComboBox soll das ganze natürlich per Touch bedienbar sein. Aus diesem Grunde soll das Scrolling in der Liste so funktionieren wie in der SurfaceListBox.

2.7.3 Grundaufbau

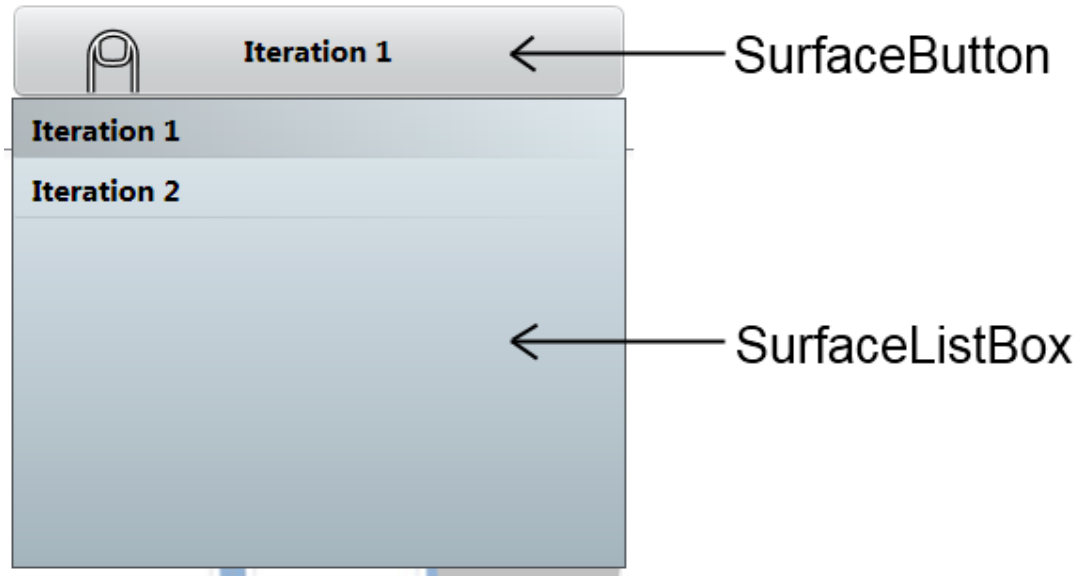


Abbildung 10 Aufbau SurfaceComboBoxControl

2.7.4 Funktionsweise

1. Klick auf den SurfaceButton.
2. Per Dialog wird die SurfaceListBox angezeigt.
3. Klick auf Item in SurfaceListBox.
4. Gewähltes Item wird auf SurfaceButton angezeigt.
5. Dialog wird geschlossen.
6. Über SelectedItem wird das momentan selektierte Item bekannt gegeben.

2.7.5 Benutzung in XAML

```
<DataTemplate x:Key="listTemplate">
   <TextBlock HorizontalAlignment="Center" FontSize="16" FontWeight="Bold">
      <TextBlock.Text>
         <Binding Path="Id" />
      </TextBlock.Text>
   </TextBlock>
</DataTemplate>

<UserControls:SurfaceComboBoxControl ListBoxHeight="250"
SelectedItem="{Binding Path=SelectedItemBindingPath, Mode=TwoWay}"
ItemsSource="{Binding Path=ThePathToObservableCollection}"
ItemTemplate="{StaticResource listTemplate}" />
```

2.7.6 Properties

Name	Beschreibung
SelectedItem	Momentan selektiertes Item (Auf Button angezeigt)
ItemsSource	Items, welche in der Liste angezeigt werden.
ItemTemplate	Wie die Items angezeigt werden sollen.
ListBoxHeight	Höhe der SurfaceListBox, welche im Dialog angezeigt wird. Wenn diese nicht gesetzt ist, wird die Grösse automatisch berechnet.

Tabelle 1 Properties der SurfaceComboBoxControl

2.8 SurfaceDragDropListBox

Anforderungen an dieses Control sowie die Alternativen können im Teil II – SW-Projektdokumentation im Design nachgelesen werden.

2.8.1 Aufbau

Aufbau ist der einer normalen SurfaceDragDropListBox.

Einige Verhaltensmerkmale der SurfaceListBox wurden übernommen, wobei viel zusätzliches Verhalten hinzugefügt wurde. Dies vereinfacht die Handhabung der ListBox.

2.8.1.1 Selektieren

Standardmässig kann ein Element mit einem kurzen Touch selektiert werden, wie dies bei einer SurfaceListBox möglich ist. Falls aber durch IsItemSelectable="False" konfiguriert, ist ein Selektieren nicht mehr möglich.

2.8.1.2 Drag per Hold Gesture

Durch das Halten eines Items für 0.5 Sekunden (Hold Gesture) wird das Objekt zum Draggen genommen, um es in einen anderen Container zu verschieben.

2.8.1.3 Drag per Item Herausziehen

Standardmässig kann ein Item durch das Ziehen in die entgegengesetzte Scrollrichtung aus der Liste gezogen werden, um es per Drag zu verschieben. Um dieses Verhalten auszuschalten, kann das Property DragOnMove auf false gesetzt werden.

Falls DragOnMoveOnlyIfOppositeOrientation gleich „true" ist, kann das Item durch ziehen in eine beliebige Richtung aus der Liste gezogen werden.

2.8.1.4 Inaktiv wenn bereits gedragt

Wenn die Liste nicht statisch ist, kann ein Item nur einmal gedraggt werden. Während dem Draggen ist das Item in der Liste aufgebleicht, damit man sieht, dass es nicht mehr gezogen werden kann.

2.8.1.5 Orientation

Bei dieser Listbox kann über das Property Orientation festgelegt werden, in welcher Orientierung die Items angeordnet werden. Dies ist bei einer normalen SurfaceListBox nicht so einfach möglich.

2.8.1.6 Droppen in die Liste

Es lassen sich Objekte in die Liste droppen ohne dafür EventHandlers schreiben zu müssen. Es muss angegeben werden, welche DropTypes gestattet sind (siehe dazugehöriges Property).

Das automatische Hinzufügen zur ItemsSource kann durch das Setzen von AddOnDrop auf false ausgeschaltet werden. Der Drag wird dann immer noch als completed angesehen, jedoch kann man dann selbst bestimmen, wie man das Item der Liste hinzufügen möchte. Dies ist unter gewissen Umständen nötig.

2.8.1.7 Löschen bei DragComplete

Ein Item wird aus der Liste gelöscht, wenn der Drag abgeschlossen wurde. Dies musste sonst per DragCompleteHandler jeweils selbst hinzugefügt werden.

Dies kann durch das Setzen von RemoveOnDragComplete auf false ausgeschalten werden.

2.8.1.8 Mehrere Spalten

Es können mehrere Spalten von Items angezeigt werden. Dies funktioniert vertikal sowie horizontal. Dazu wird ein AnimatedWrapPanel benutzt, welches dieses Verhalten bereits mit sich bringt. Dies wird über ein Property ein- und ausgeschalten.

2.8.2 Verwendung im Code

An und für sich gleich wie die SurfaceListBox. Man kann ItemsSource sowie ItemTemplate übergeben. Ebenfalls bietet das Control ein paar weitere Properties an.

Beispiel:

```
<UserControls:SurfaceDragDropListBox
            AllowDrop="True"
            MultipleColumnsRows="True"
            Orientation="Vertical"
            ItemTemplate="{StaticResource MyDataItemTemplate}"
            ItemsSource="{Binding Path=MyItems}"
            AllowedDropType="{x:Type View:DataItem}"/>
```

2.8.2.1 Automatisches Erkennen ob Scrollbar vorhanden

Das folgende Verhalten ist meist gewünscht:

1. Solange die Liste noch nicht voll von Elementen ist, und somit keine Scrollbar vorhanden ist, soll das Element durchs Ziehen in eine beliebige Richtung herausgezogen werden können.
2. Sobald die Liste mehr Elemente enthält als gleichzeitig angezeigt werden können, und somit eine Scrollbar angezeigt wird, sollen Elemente nur noch in die Richtung entgegengesetzt zur Scrollrichtung herausgezogen werden können.

Dieses Verhalten wird erreicht mit folgender Property-Zuweisung in der SurfaceDragDropListBox:

```
DragOnMoveOnlyIfOppositeOrientation="{Binding
RelativeSource={RelativeSource Self}, Path=IsScrollVisible}"
```

Dies ist der Standard beim Verwenden des Controls. Falls dieses Verhalten nicht gewünscht ist, kann man durch eigenes Setzen von DragOnMoveOnlyIfOppositeDirection dieses Binding überschreiben.

2.8.3 Properties

Name	Beschreibung
AddOnDrop	Ob Item bei Drop gleich der ItemsSource hinzugefügt wird. true (default)
DragOnMove	Wenn true (default), lässt sich durchs Ziehen eines Items aus der Liste ein Drag&Drop starten.
DragOnMoveOnlyIfOppositeOrientation	Wenn true, lässt sich der DragOnMove durchs Ziehen in eine beliebige Richtung starten. Wenn false, lässt sich der DragOnMove nur durchs Ziehen in die Entgegengesetzte Scrollrichtung gestartet werden. Default: Auf IsScrollVisible gebunden (Binding)
AllowedDropType	Gibt an, welche Datentypen automatisch in die Liste

	aufgenommen werden sollen, wenn sie auf die Liste gedropt werden.
IsItemSelectable	Spezifiziert, ob Items in der Liste selektiert werden können. Wenn true (default) können Items selektiert werden. Wenn false können Items nicht selektiert werden.
ItemsAreStatic	Spezifiziert, ob Items immer in der Liste bleiben. Wenn true bleiben Items immer in Liste und können mehrmals heraus gezogen werden. Wenn false (default) können Items nur einmal gezogen werden und werden bei Drop aus der Liste gelöscht.
MultipleColumnsRows	Ob mehrere Spalten angezeigt werden sollen. Wenn true wird ein AnimatedWrapPanel als Itemshost verwendet. Wenn false (default) wird ein AnimatedStackPanel als Itemshost verwendet.
Orientation	Gibt die Orientation des AnimatedStackPanel in der Liste an. Diese Orientation ist ebenfalls die Scrollrichtung. Default ist Vertikal.
RemoveOnDragComplete	Spezifiziert, ob ein Item nach einem erfolgreichen Herausziehen aus der Liste von der ItemsSource gelöscht werden soll. true (default)

Tabelle 2 Properties der SurfaceDragDropListBox

2.9 I18n

Die Internationalisierung befindet sich im Namespace ScrumTable.UI.View.Localization. Was dazu verwendet wurde, kann im Design unter Teil II – SW-Projektdokumentation im Kapitel Design nachgelesen werden.

2.9.1 Beispiel für Nutzung im Code

2.9.1.1 Laden der Sprachdateien (C#)

```
// Sprache aus xml hinzufügen
LanguageDictionary.RegisterDictionary(CultureInfo.GetCultureInfo("en-US"),
      new XmlLanguageDictionary("Localization/Languages/en-US.xml"));

// Momentane Sprache wählen (während Laufzeit)
LanguageContext.Instance.Culture = CultureInfo.GetCultureInfo("en-US");
```

2.9.1.2 XAML

```
<Label loc:Translate.Uid="Example_Label_1" Content="{loc:Translate}"
FontSize="{loc:Translate}" />


<TextBlock loc:Translate.Uid="Example_TextBlock_1" Margin="8"
HorizontalAlignment="Left">
   <TextBlock.Text>
      <loc:Translate>
         <Binding Path="FirstName" />
         <Binding Path="LastName" />
      </loc:Translate>
   </TextBlock.Text>
   <TextBlock.Foreground>
      <loc:Translate />
   </TextBlock.Foreground>
</TextBlock>
```

2.9.1.3 Sprachdatei (en-US.xml)

```
<?xml version="1.0" encoding="utf-8" ?>
<Dictionary EnglishName="English" CultureName="English" Culture="en-US">
  <Value Id="Example_Label_1 " Content="Text of the label" />
  <Value Id="Example_TextBlock_1 " Text="Welcome {0} {1}" />
</Dictionary>
```

2.10 DomainCollections als ObservableCollection nutzen

2.10.1 Problemstellung

Die Daten, welche vom BL kommen, bestehen aus vielen Listen, welche als DomainCollections angeboten werden. Diese Listen, z.B. eine Liste aller Tasks einer User Story, möchte man im User Interface anzeigen.

Wenn diese Listen nicht verändert werden würden, wäre dies kein Problem. Jedoch braucht es für das User Interface eine ObservableCollection, welche Events sendet, sobald sich der Inhalt ändert. Das User Interface reagiert darauf und zeigt den geänderten Inhalt an. Die DomainCollection sendet zwar Events, jedoch sind diese für die Controls von WPF nicht verständlich.

Wenn ein Element per Drop einer Liste hinzugefügt wird, soll dieses Element auch in der DomainCollection hinzugefügt werden.

2.10.2 Lösung durch eigene ObservableDomainCollection<T, TDomain>

Es bot sich an, eine Art Adapter zu bauen, welcher eine kompatible ObservableCollection anbietet, die auf der einen Seite die Events der inneren DomainCollection an die ObservableCollection weiterleitet und auch umgekehrt. Dies ist zwar nicht komplett ein Adapter, aber es geht in die Richtung Adapter, welcher zusätzlich um Funktionalität erweitert wurde.

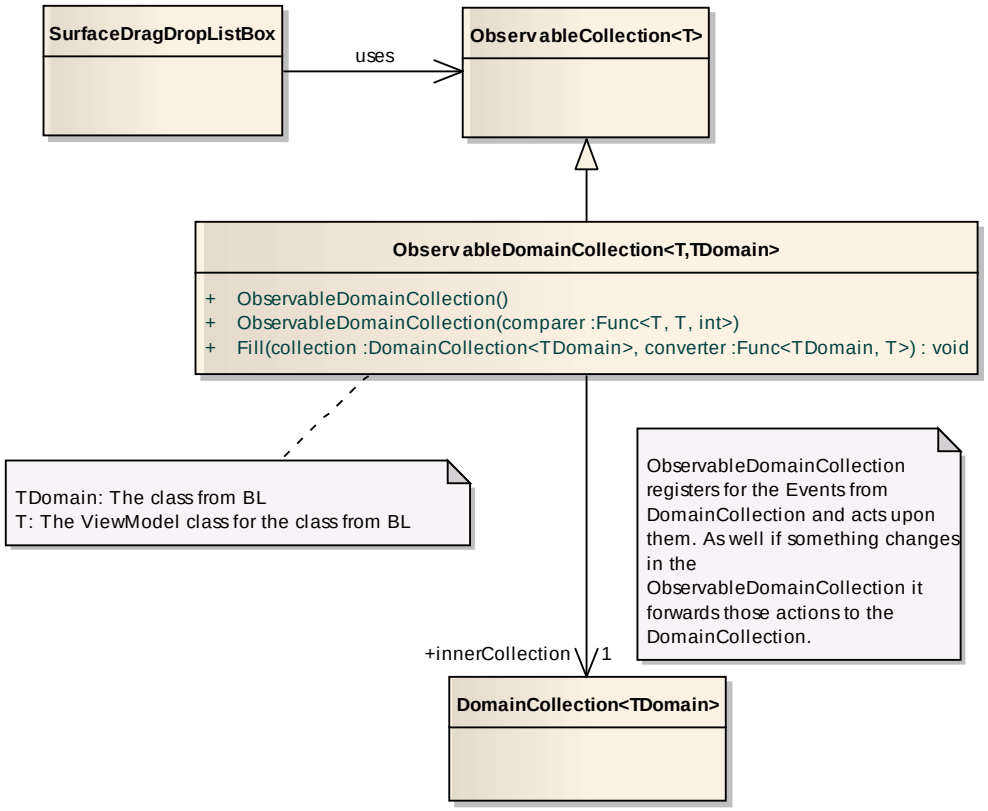


Abbildung 11 ObservableDomainCollection

Über den Konstruktor kann, falls gewünscht, ein Comparer mitgegeben werden, wodurch die Sortierung der Liste bestimmt wird.

Über Fill wird die Liste mit der DomainCollection gefüllt. Dabei gibt man mit, wie das BL-Objekt in ein ViewModel-Objekt konvertiert werden soll.

2.10.2.1 CodeBeispiel mit comparer für das Sortieren

```
ObservableDomainCollection<TaskViewModel, Task> _selectedStoryTasks = new
ObservableDomainCollection<TaskViewModel,
Task>((x,y)=>x.Priority.CompareTo(y.Priority));

_selectedStoryTasks.Fill(_selectedUserStory.Tasks, x => new
TaskViewModel(x));
```

3 Business Layer Komponenten

Die Grundlegende Architektur des Business Layers wird im Teil II – SW-Projektdokumentation im Kapitel Design unter Business Layer Architektur beschrieben. Die Implementation soll einen tieferen Einblick in die Internals des Business Layers geben.

3.1 Statische Struktur

Das folgende Diagramm zeigt die statische Struktur des Business Layers:

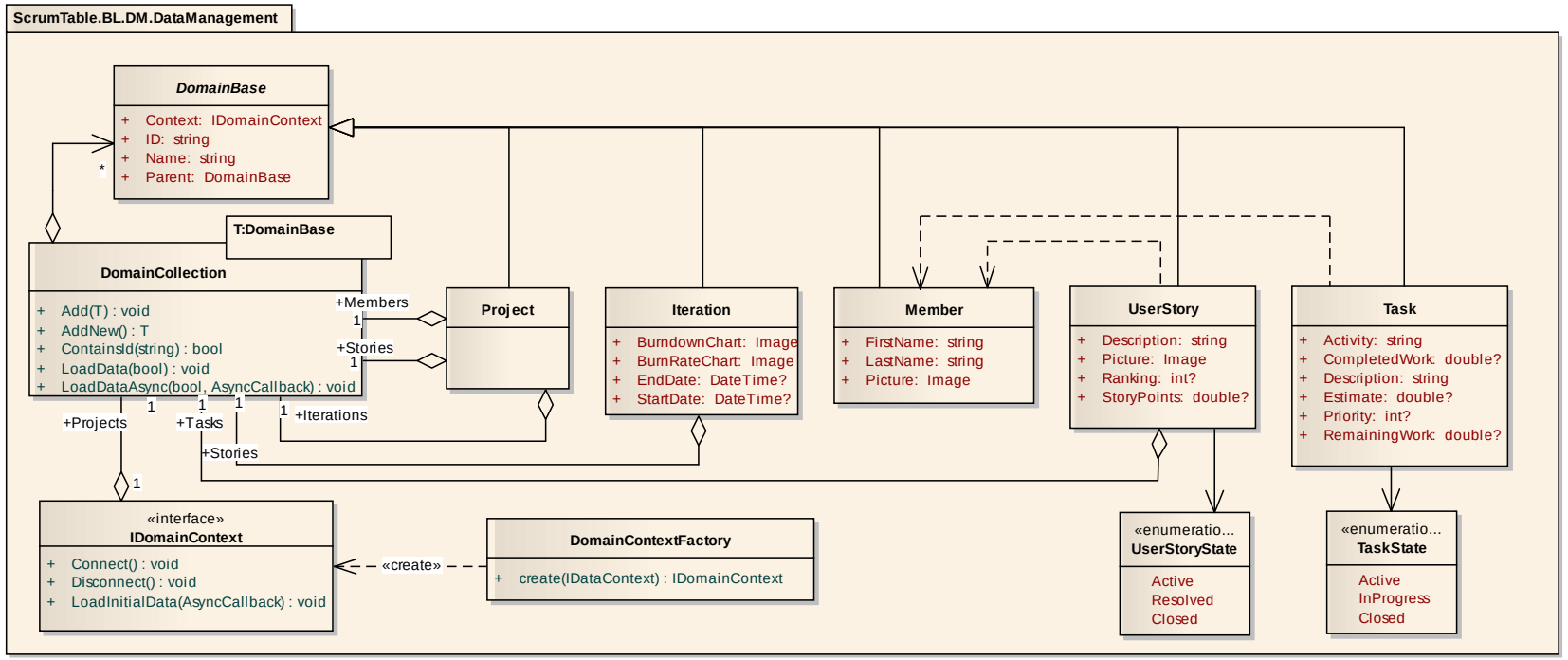


Abbildung 12 Domain Context Class Diagram

Der eigentliche Einstiegspunkt in die statische Struktur des Business Layers zeigt sich in der `DomainContextFactory`. Diese Klasse ist nach dem Simple Factory Pattern nach GoF implementiert. Die `DomainContextFactory` erzeugt neue `DomainContext` Instanzen, welche von `IDomainContext` abgeleitet sind. Die `IDomainContext` Instanz lädt wiederum eine `DomainCollection` mit den Projekten.

Als erstes muss auf ein Ziel- & Endsystem verbunden werden. Dies geschieht mittels der Connect(…) Methode der `IDomainContext` Schnittstelle, welche den Aufruf direkt an den Data Layer weiterleitet.

Das Laden der Daten kann synchron oder asynchron stattfinden. Um die Basisdaten (Daten, welche das Grundgerüst darstellen) des DomainContexts zu laden, bietet die `IDomainContext` Schnittstelle eine Methode LoadInitialData(…) an.

Eine `DomainCollection` enthält Instanzen von Domain Objekten. Die `DomainCollection` wird typifiziert, um unnötige Casts der Elemente in einer Collection zu vermeiden. Mit Hilfe der `DomainCollection` wird der Objektbaum, wie dieser im Teil II – SW-Projektdokumentation im Kapitel Design unter Business Layer Architektur ist, aufgebaut.

Beim Erstellen einer Verbindung mit einem Ziel- & Endsystem muss wie folgt vorgegangen werden:

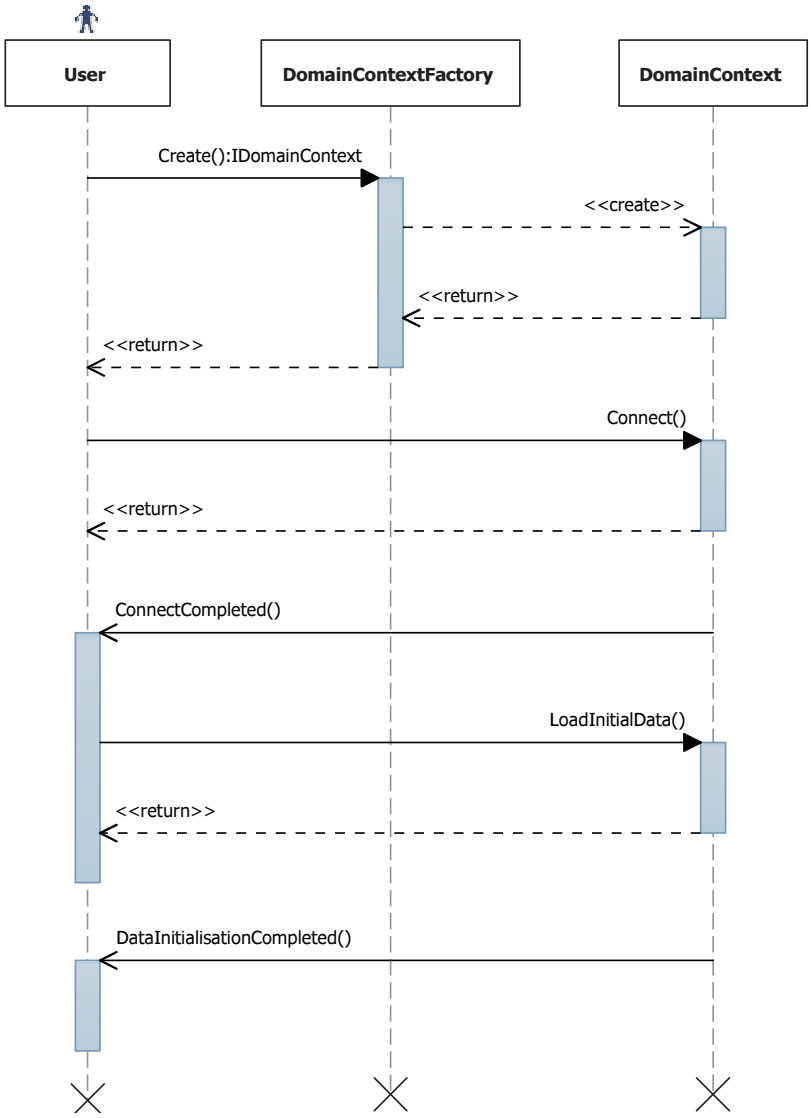


Abbildung 13 Domain Context Sequence Diagram

3.2 Asynchrone Aufrufe

ScrumTable arbeitet im Bereich des Daten-Ladens sowie beim Connecten auf das externe Ziel- und Endsystem mit mehreren parallelen Threads. Dies hat den Effekt, dass das User Interface bei Verbindungsproblemen mit dem Endsystem nicht einfriert. Die asynchronen Aufrufe allerdings finden nicht im Business Layer statt, sondern erfolgen im Data Layer und werden anschliessend im Data Layer mittels des Windows Message Queue Dispatchers in die Windows Message Queue zurück synchronisiert. Dazu steht im Data Layer das Interface `IDataSyncDispatcher` mit der Methode `Invoke(Action action)` zur Verfügung.

Mehr zu den Asynchronitäten erfahren sie im Kapitel Asynchrone Aufrufe.

4 Data Layer Komponenten

Im Data Layer Komponenten steht folgende statische Klassenstruktur zur Verfügung:

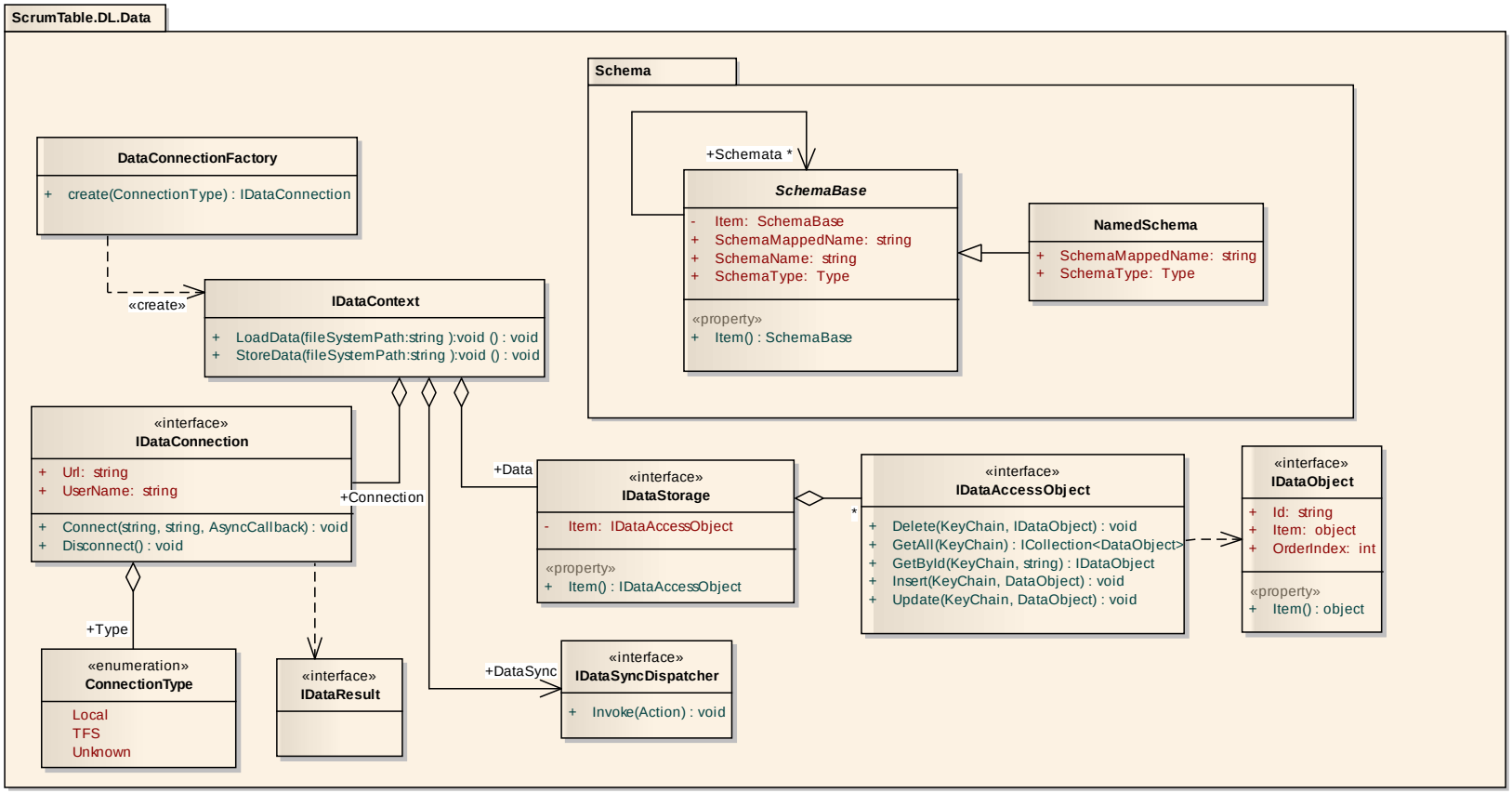


Abbildung 14 Data Layer Klassen in der Übersicht

Der eigentliche Einstiegspunkt in die statische Struktur des Data Layers zeigt sich in der `DataConnectionFactory`. Diese Klasse ist nach dem Simple Factory Pattern nach GoF implementiert. Die `DataConnectionFactory` erzeugt neue `DataContext` Instanzen, welche von `IDataContext` abgeleitet sind. Die `IDataContext` Instanz enthält wiederum eine `IDataConnection` mit den Funktionalitäten für die Verbindung zum externen System sowie eine Instanz mit dem `IDataStorage` für die Datenzugriffsobjekte. Die `IDataSyncDispatcher` Klasse bietet die Möglichkeit, asynchrone Aufrufe in die Main Message Queue zurück zu synchronisieren.

4.1 Asynchrone Aufrufe

Das folgende Aktivitäts-Diagramm soll die Asynchronitäten beim Connecten (Das Laden von Daten erfolgt analog) visualisieren, die Aufrufe erfolgen nach dem Fire and Forget Prinzip:

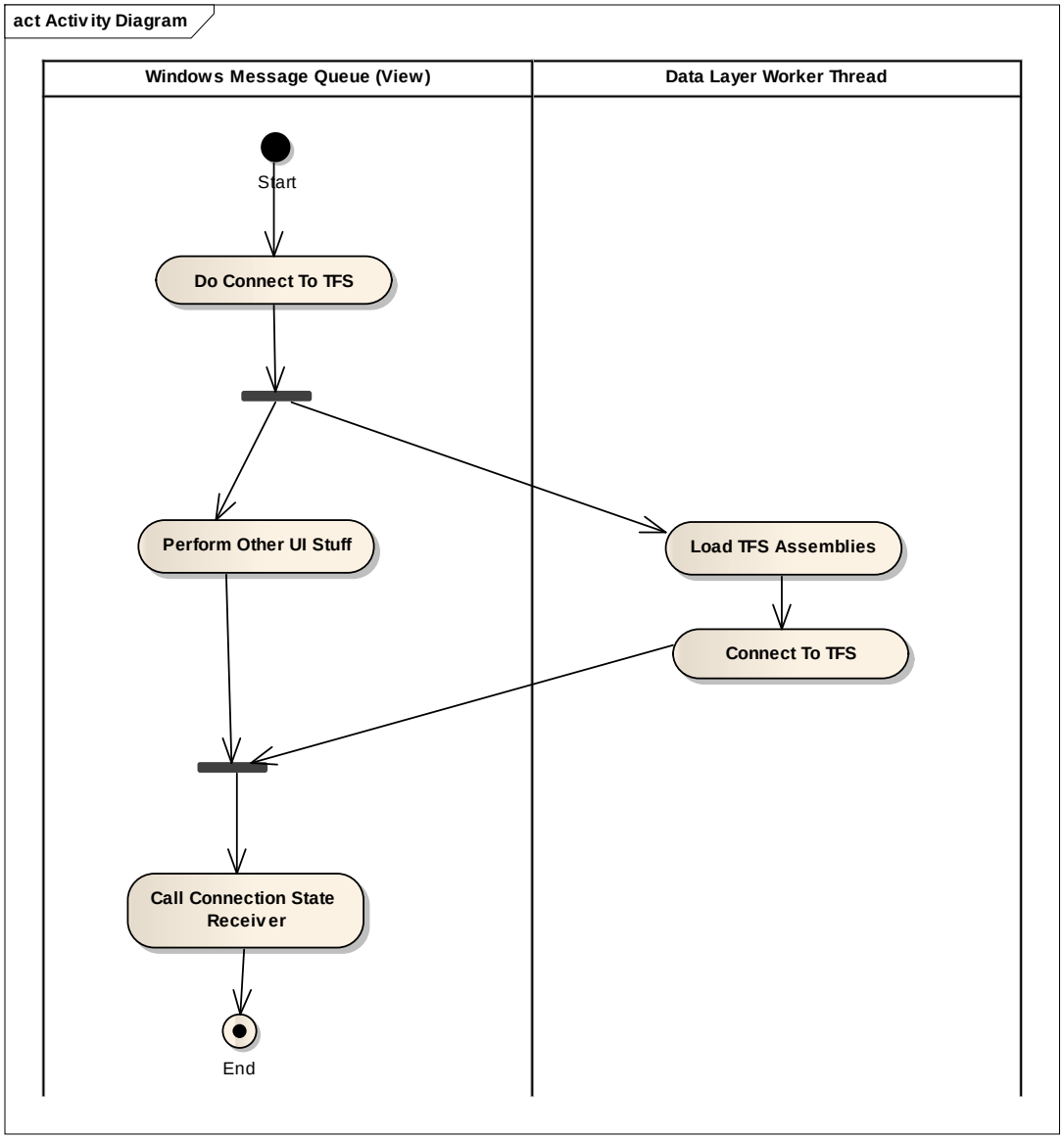


Abbildung 15 Asynchrones Connecten im Data Layer

Die asynchronen Aufrufe werden alle in ScrumTable über einen separaten Thread-Pool mit Worker Threads ausgeführt. Dies ermöglicht ein effizientes Wiederverwenden der Threads.

Da der Performance-Test mit dem Standard-Windows Thread-Pool kein Erfolg brachte, wurde für ScrumTable ein bereits existierender und von Silvan Gehrig für das Software Engineering 2 entwickelter und getesteter Thread-Pool eingesetzt.

4.1.1 Process Model

Für die Modellierung der lightweight Prozesse innerhalb von ScrumTable sind folgende Elemente zum Einsatz gekommen:

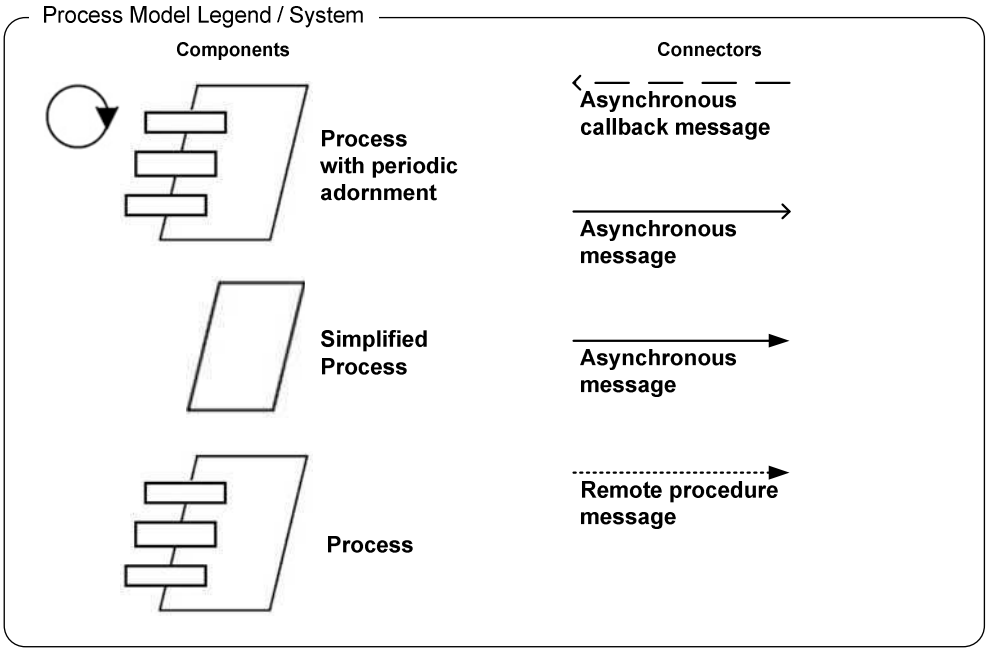


Abbildung 16 Bechreibung der Element im Process Model [1]

4.1.1.1 Prozesse in ScrumTable

Die Aufrufe gehen alle von der `Windows Message Queue` aus. Diese enthält und steuert die WPF Applikation.

Über WPF Bindings und Business Layer API Aufrufe auf die `Data Access Components` wird festgelegt, ob eine Anfrage in einem separaten Thread Pool asynchron ausgeführt werden soll. Andernfalls gehen die Aufrufe direkt synchron über das `External Service Interface` auf die externen Team Foundation Services. Die Aufrufe über die Systemgrenzen hinweg werden ähnlich zu Remote Procedure Calls durchgeführt.

Falls ein Aufruf asynchron erfolgen soll, wird dieser in die `Message Queue with Thread Pool` eingetragen. Ein Worker Thread führt die Anweisungen dann asynchron auf dem `External Service Interface` aus.

Das folgende Bild soll die Beschreibung oberhalb veranschaulichen und aufzeigen, wie die verschiedenen asynchronen Prozesse zusammenarbeiten.

[1] (Rational Software Corp., Philippe Kruchten, 2010)

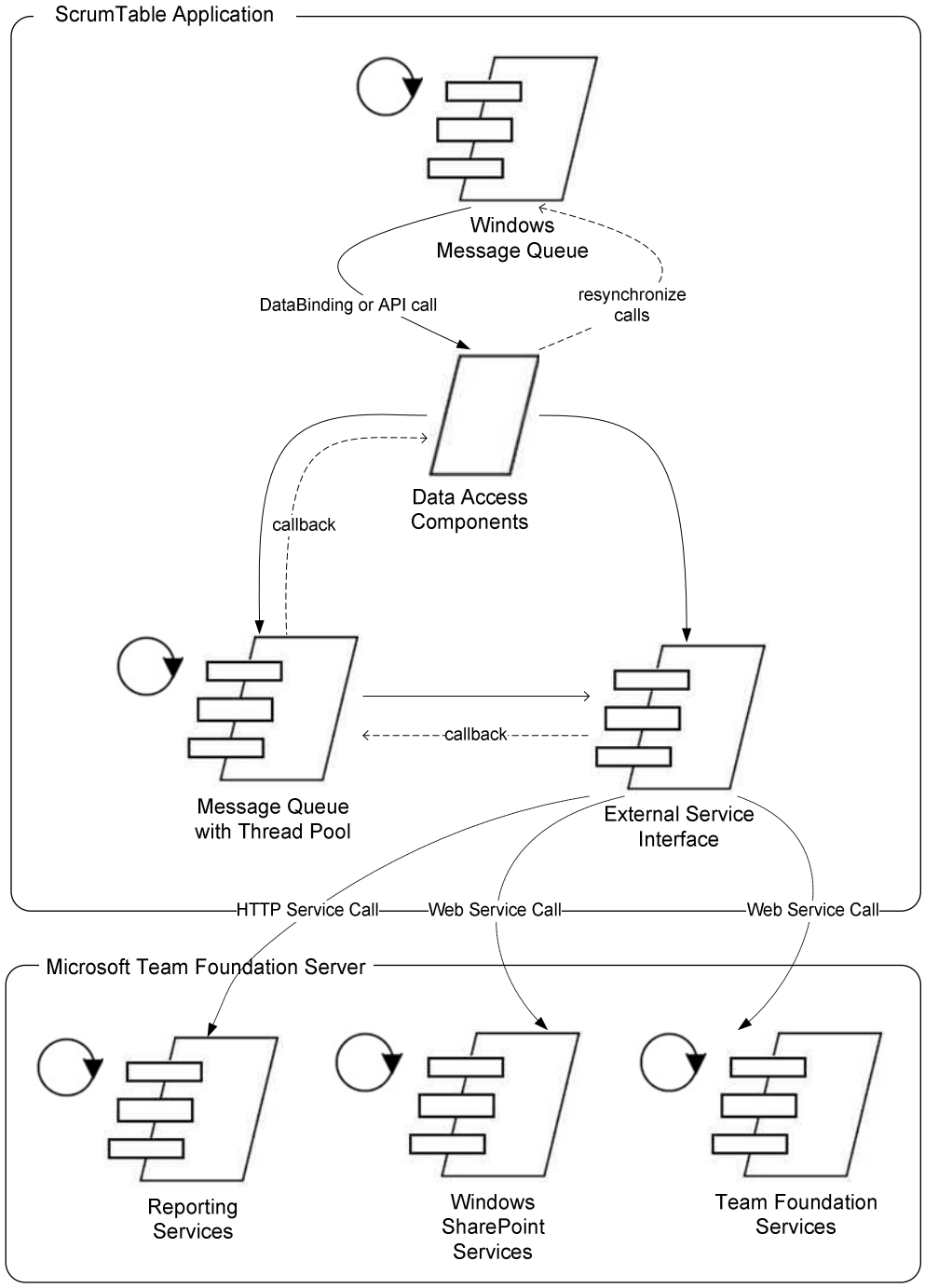


Abbildung 17 Process Model der Parallelitäten in ScrumTable

4.2 Plug in Implementation

Die Plug in Architektur von ScrumTable beinhaltet folgende Komponenten, welche für die Plug in Implementation relevant sind.

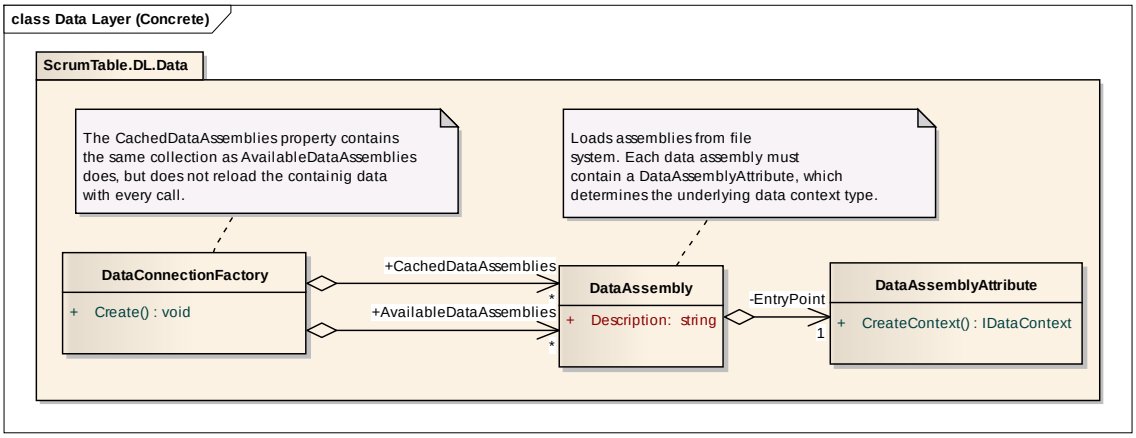


Abbildung 18 Data Layer Connection Implementation

Die `DataConnectionFactory` in der ScrumTable.DL.Data Library (auch DLL) durchsucht das Applikationsverzeichnis nach allen Files mit der Endung .DLL.

Sobald eine Datei als Library mit einem .NET PE Header identifiziert wurde, lädt ScrumTable diese Library in den Speicher und versucht das `DataAssemblyAttribute` auf Assembly-Ebene zu finden. Das `DataAssemblyAttribute` entspricht dem Einstiegspunkt in ein Data Layer Plug In.

Falls das `DataAssemblyAttribute` nicht vorhanden ist, werden die geladenen Daten verworfen. Ansonsten, speichert sich ScrumTable das `DataAssemblyAttribute` sowie die Beschreibung der geladenen DLL in den Speicher.

Das `DataAssemblyAttribute` enthält die Informationen über den `DataContext` (abgeleitet von `IDataContext`) innerhalb eines Plug ins. Wie die statische Klassenstruktur des Data Layers zeigt, entspricht der `IDataContext` dem zentralen Element mit der Einstiegspunktfunktionalität eines Daten-Connectors.

Die folgende Grafik illustriert **Plug in Load** Vorgang anhand der Plug ins TFS und Local:

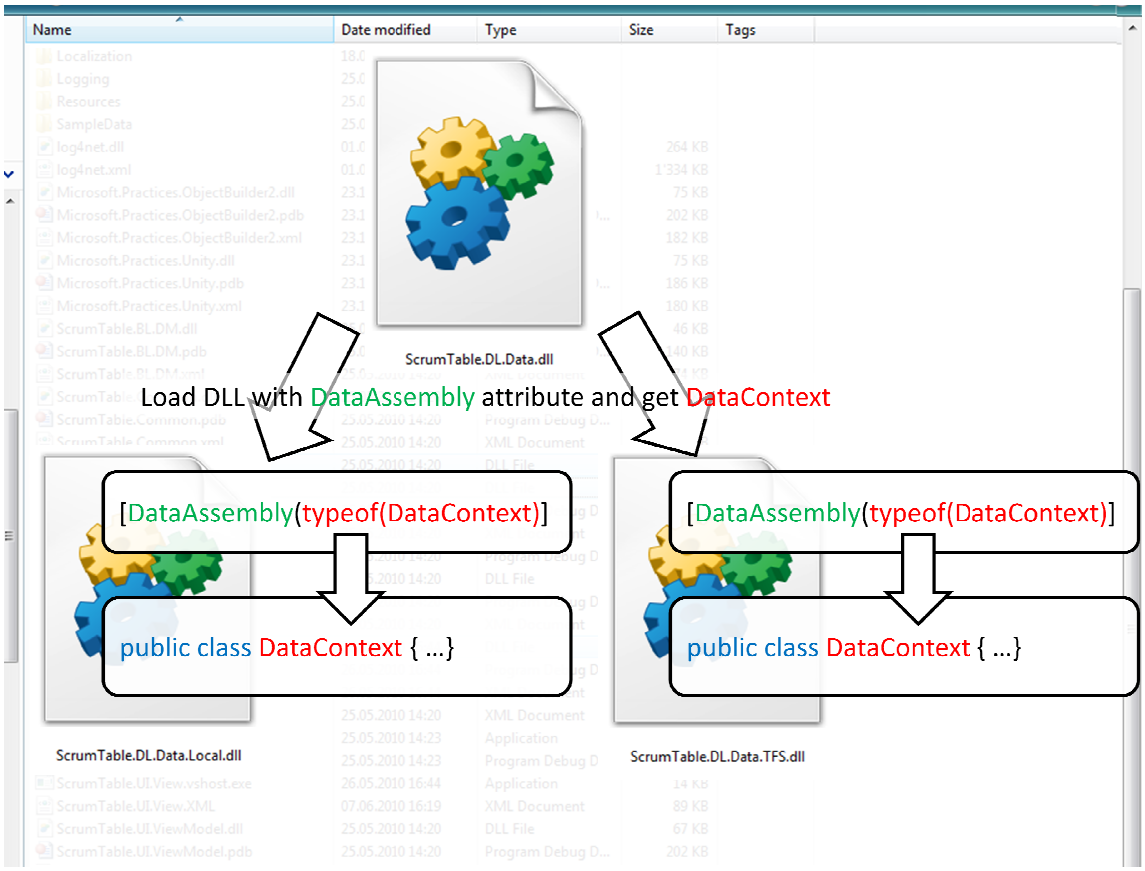


Abbildung 19 Plug in DLLs vom Dateisystem laden

4.2.1 Eigener Connector erstellen

Wenn ein eigener Connector für ein externes Ziel- und Endsystem programmiert werden soll, dann sollte vom Standpunkt des Local Data Connectors ausgegangen werden.

Der folgende Ablauf beschreibt, wie ein Connector für ScrumTable programmiert werden kann.

1. Anlegen eines neuer neuen .NET 3.x Library im Visual Studio als neues Projekt.
2. Hinzufügen der Referenzen „ScrumTable.DL.Data“ und „ScrumTable.DL.Data.Schema“ zum neuen Connector-Projekt.
3. Editieren der AssemblyInfo.cs Datei. Dabei muss das Assembly-Attribute `DataAssembly(typeof(DataContext))` eingetragen werden. Das Assembly-Attribute `AssemblyDescription` dient zur Kennzeichnung der Library in der ScrumTable Oberfläche bei den Einstellungen.
4. Anlegen der `DataContext` Klasse. Dabei muss beachtet werden, dass das Interface `IDataContext` mit allen Eigenschaften implementiert wird.

5. Kompilieren des Projektes. Anschliessend muss die neue Connector-DLL ins Verzeichnis der ScrumTable Applikation kopiert werden. Am besten geschieht dies mit einem Visual Studio Post-build event Script, welches in etwa wie folgt aussehen kann:
   ```
   xcopy "$(TargetDir)$(TargetName).*"
   "$(SolutionDir)ScrumTable.UI.View\$(OutDir)" /Y
   ```
   Eingepflegt wird diese Zeile in den Connector-Projekt-Eigenschaften im Visual Studio.

Das folgende Bild soll die Anleitung visualisieren.

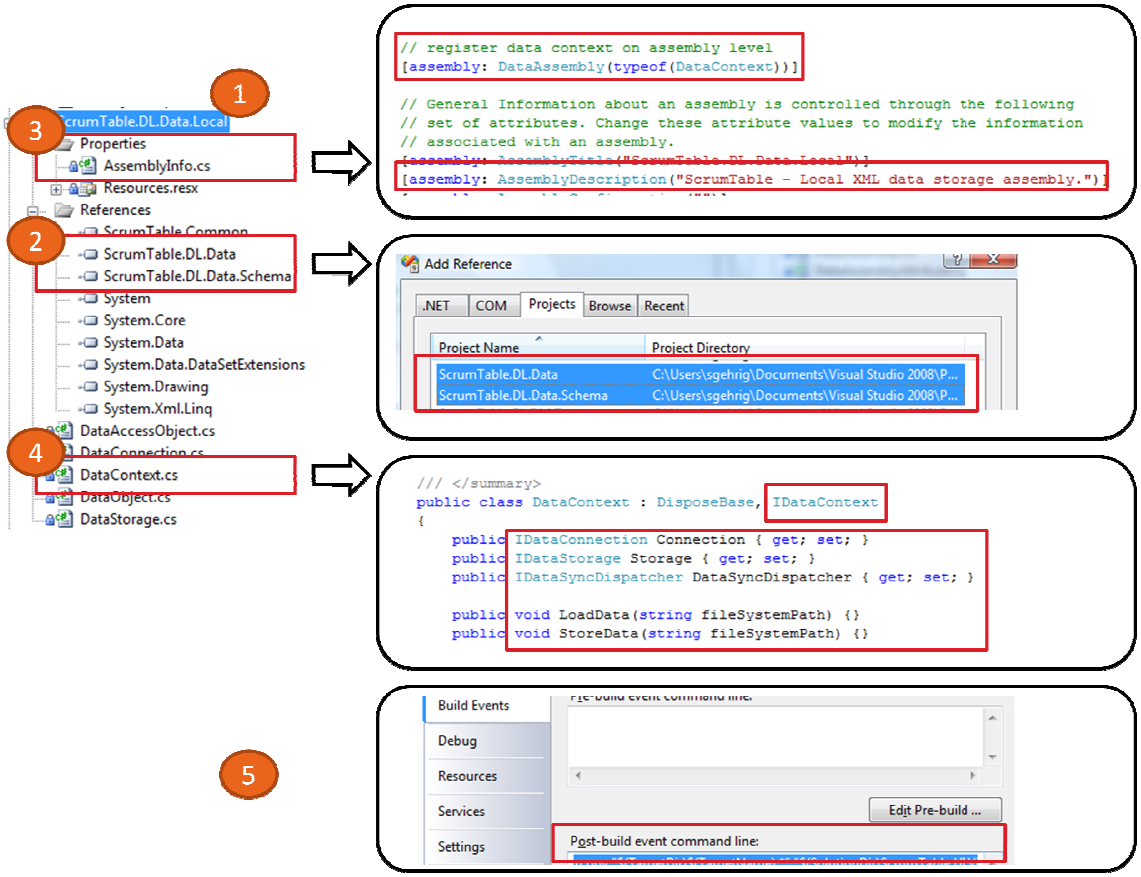


Abbildung 20 Anleitung zum Erstellen eines eigenen Plug ins

4.2.2 Local Data Connector

Der Local Data Connector realisiert die einfachste Art eines Data Connectors. Die Daten werden ins Memory, das heisst in Klassenfelder als Dictionary gespeichert. Nach dem Beenden des Programmes werden diese vom Garbage Collector gelöscht.

Die Daten, welche vom Business Layer gelesen oder geschrieben werden, befinden sich mittels der Keys aus dem Schema in den Klassenfelder-Dicitonary's.

Die statische Klassenstruktur des Local Data Connectors sieht analog zur statischen Struktur des Paketes ScrumTable.DL.Data aus.

4.2.3 Team Foundation Server Data Connector

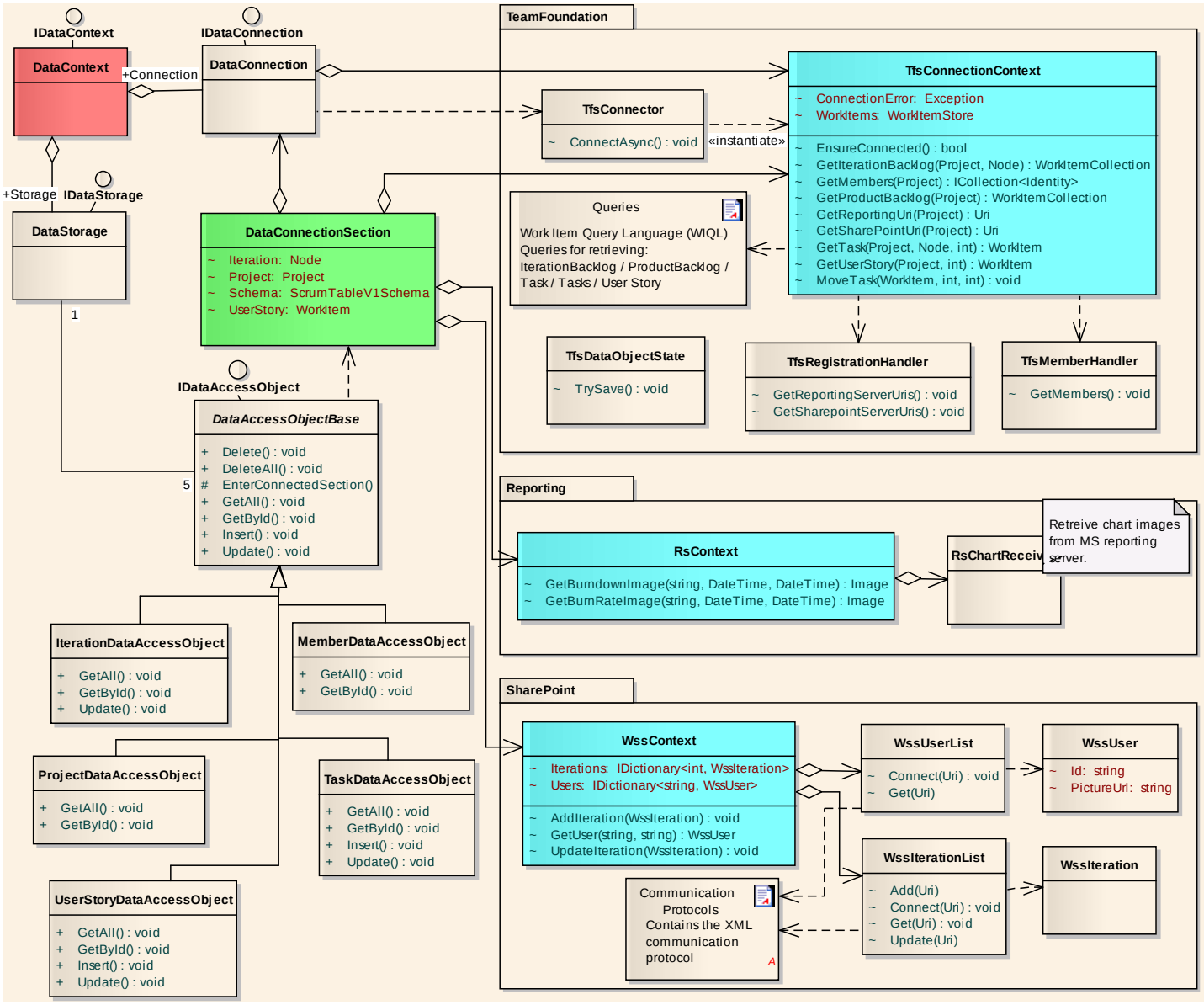


Abbildung 21 TFS Data Connector Klassenübersicht

Der Connector zum Microsoft Team Foundation Server besteht aus den Schnittstellen zu drei Microsoft Systemen:

1. Microsoft Team Foundation API
2. Microsoft Reporting Server Query API
3. Microsoft Windows SharePoint Services WebService API

Jede dieser Microsoft API's wird entsprechend mit einem Kontext (blau) als zentraler Zugriffspunkt gegen aussen gekapselt.

Der eigentliche Einstiegspunkt in den Connector wird durch die Klasse `DataContext` (rot) repräsentiert. Wie dieser instanziiert wird, kann in der Plug in Implementation nachgelesen werden. Der `DataContext` enthält eine Connection, welche wiederum asynchron über den `TfsConnecter` den `TfsConnectionContext` kreiert.

Der `TfsConnectionContext` kann mittels der in den Text-Files vorliegenden Work Item Query Language Files die gewünschten WorkItems über die Team Foundation API aus dem Team Foundation Server auslesen.

Der `DataContext` enthält ebenso den Storage, welcher die Datenzugriffsobjekte entsprechend der Schnittstellendefinition beinhaltet.

Die Basisklasse für alle Datenzugriffsobjekte im TFS Data Connector ist die `DataAccessObjectBase` Klasse. Diese kann mittels der EnterConnectedSection() Methode die `DataConnectionSection` (grün) erstellen, welche für den (Multithreading-) gesicherten Zugriff auf die drei verschiedenen Microsoft API's verantwortlich ist.

Die `DataConnectionSection` Klasse (grün) enthält alle Informationen, um auf die drei externen Systeme zugreifen zu können. So werden in der `DataConnectionSection` Klasse die Contexts (blau) der verschiedenen Systeme verlinkt.

Die DataAccessObject Klassen wie `IterationDataAccessObject`, `MemberDataAccessObject`, `ProjectDataAccessObject`, `TaskDataAccessObject` und `UserStoryDataAccessObject` überschreiben und implementieren diejenigen Operationen, welche für die gegebenen Datentypen erlaubt sind. Falls eine Operation nicht erlaubt ist, wird deren Aufruf einfach ignoriert. Jede DataAccessObject Klasse verwaltet DataObjects (`DataObjectBase`), welche die einzelnen Werte der Iterationen (`IterationDataObject`), Members (`MemberDataObject`), Projects (`ProjectDataObject`), Tasks (`TaskDataObject`) oder UserStorys (`UserStoryDataObject`) enthalten.

5 Test verfahren

In diesem Kapitel wird das Testvorgehen für ScrumTable beschrieben. Dabei wird zwischen funktionalen Tests, welche automatisiert durchgeführt werden, und System Tests, welche durch einen Benutzer erfolgen, unterschieden.

5.1 Funktionale Tests

Um die Business Funktionalität zu überprüfen, setzt ScrumTable auf automatisierte Unit Tests. Die Tests werden in Microsoft Visual Studio.NET angelegt und können auf einem Test-Sever kompatible Remote-Rechner ausgeführt werden. Bei den ScrumTable Unit Tests sollen folgende Funktionalitätsblöcke mit automatisierten Tests überprüft werden:

Funktionalitätsblock	Beschreibung
Data Layer	Das Data Layer Interface eines Ziel- und Endsystems soll auf dessen Funktionalität geprüft werden. Diese Testverfahren genügen nur für den lokalen DataStorage, welcher durch Unit Tests auf die Funktionalität geprüft werden kann. Falls ein externes System angesteuert wird, müsste die gesamte API des externen Systems „mocked“ werden, was bei einer API für Team Foundation Server, Windows SharePoint Services und Reporting Server praktisch unmöglich wäre.
Business Layer	Die Business Layer Komponenten werden einzeln auf deren Funktionalität getestet: • *UserManagement* Die Tests des LoginAccountManagers überprüfen, ob die Verschlüsselung / Entschlüsselung funktioniert. • *DataManagement* Der DataManagement Test sieht vor, die Domain-Elemente von ScrumTable auf deren Funktionalität zu prüfen. Bei diesem Testverfahren wird eine lokale In-Memory-Datenbank verwendet. Diese wird durch die Daten eines Domain-Layer-Test-Files gefüllt. Dies entkoppelt den Business Layer von möglichen Problemen mit dem Data Layer.

Wie in den Coding-Standarts beschrieben, muss das Programm vor dem Einchecken des Programmcodes ins gemeinsame Repository kompilieren und alle Unit-Tests müssen im Normallfall erfolgreich durchlaufen.

5.2 System Tests

Anhand der Use-Case Szenarios werden die Funktionalität des Programmes überprüft. Diese Überprüfung findet parallel zum Usability Test statt. Nicht erfüllte Punkte werden im Usability Feedback notiert.

UC1: Anmelden an ScrumTable	Erfüllt	N. Erfüllt
Anmelden mit einem registrierten Benutzer		
Anmelden mit einem nicht registrierten Benutzer		

UC2: Projektplanung durchführen	Erfüllt	N. Erfüllt
User Story Points schätzen		
User Story Stack-Rank setzten		
User Story erfassen/ändern		
User Stories den Sprints zuweisen		
Dauer des Sprints einstellen		

UC3: Sprint Planen	Erfüllt	N. Erfüllt
User Stories für aktuellen Sprint auswählen		
User Stories in Tasks aufteilen		
Einem Task einen Verantwortlichen zuweisen		
Einem Task die Stunden zuweisen		
Anzeigen der Auslastung der Team-Mitglieder		

UC4: Daily Scrum	Erfüllt	N. Erfüllt
Durchführung des Daily Scrums		
Burn-Down Chart anzeigen		
Task-Board		
User-Story-Board		

UC4: Scrum Poker	Erfüllt	N. Erfüllt
Durchführung von Scrum Poker um User-Stories zu schätzen		

5.3 Usability Tests

Der Usability Test erfolgte durch Michael Rüegg. Das Vorgehen ist in einem separaten Dokument beschrieben. Das Feedback wurde auf Video aufgenommen und in einem separaten Dokument dokumentiert. Die Inputs wurden verwendet um das Programm zu verbessern und eine höchstmögliche Usability zu gewährleisten.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Code Reviews

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument hält die Code-Reviews und deren Kritikpunkte fest.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Referenzen

http://msdn.microsoft.com/en-us/library/12a7a7h3(VS.71).aspx *[stand 20.02.2009]*

1.4 Übersicht

Zuerst wird in diesem Dokument auf die Code Review Kriterien eingegangen. Danach werden die einzelnen Code Reviews genauer beschrieben.

2 Kriterien

2.1 Code Style Analyse

	Erfüllt	N. Erfüllt
Die Code Richtlinien von ScrumTable wurden eingehalten		
.NET Common Language Specification Richtlinien wurden eingehalten		
Der Code wirkt durch seine Anordnung & Verschachtelung übersichtlich		
Die Scrum Headers sind in allen nicht generierten Sourcen vorhanden		
Die XML-Kommentar Kompilation wurde in den Projekten aktiviert		
Alle Public / Protected Members sind ausreichend Dokumentiert		
Die auskommentierten Programm-Stücke sind ausreichend erklärt		
Die Projekte enthalten keine toten Programm-Klassen		
Die fehlenden Programmstücke sind mittels TODO-Kommentar beschrieben		
Der Code übersetzt ohne Compiler Warnings		

2.2 Exception Handling

	Erfüllt	N. Erfüllt
Der Code enthält keine abgefangenen und ignorierten Ausnahmen		
Fehler in asynchronen Prozessen werden mittels Event weitergeleiltet		
Das Logging erfasst alle Fehler aus allen Funktionalitätsschichten		
Das Before/After Pattern wird, wo möglich, mittels using() { } angewendet		
IDisposable.Dispose() Methoden werden in jedem Fall aufgerufen		

2.3 Flow Control

	Erfüllt	N. Erfüllt
Es existieren keine Schleifen ohne Abbruchkriterien		
Es existieren keine toten Programmstücke (z.B. if(false) / while (false) / …)		
Rekursive Calls haben immer eine Verankerung und Abbruchbedingung		

2.4 Naming

	Erfüllt	N. Erfüllt
Die Namen der Klassen / Variabeln sind sprechend		
Verwirrende oder falsche Namen sind nicht vorhanden		
Interface-Klassen beginnen immer I (z.B. IDisposable)		
Klassen / Properties / Methoden werden mit *PascalCasing* geschrieben		
Lokale Variablen / Argumente werden mit *camelCasing* geschrieben		
Felder in Klassen werden mit *_camelCasing* geschrieben		
Der Code enthält keine Magic Numbers		

2.5 Tools

	Erfüllt	N. Erfüllt
Die Warnings von Resharper 4.5 werden wo sinnvoll behoben		
Die Errors von FxCop werden wo sinnvoll behoben		
Die Abhängigkeiten zwischen den Packages sind aufgrund der Aussagen NDepend und Metrics unidirektional.		

3 Durchgeführte Code Reviews

3.1 Review vom 27.05.2010

3.2 Code Style Analyse

	Erfüllt	N. Erfüllt
Die Code Richtlinien von ScrumTable wurden eingehalten		X
.NET Common Language Specification Richtlinien wurden eingehalten		X
Der Code wirkt durch seine Anordnung & Verschachtelung übersichtlich	X	
Die Scrum Headers sind in allen nicht generierten Sourcen vorhanden		X
Die XML-Kommentar Kompilation wurde in den Projekten aktiviert		X
Alle Public / Protected Members sind ausreichend Dokumentiert		X
Die auskommentierten Programm-Stücke sind ausreichend erklärt		X
Die Projekte enthalten keine toten Programm-Klassen	X	
Die fehlenden Programmstücke sind mittels TODO-Kommentar beschrieben	X	
Der Code übersetzt ohne Compiler Warnings		X

3.3 Exception Handling

	Erfüllt	N. Erfüllt
Der Code enthält keine abgefangenen und ignorierten Ausnahmen	X	
Fehler in asynchronen Prozessen werden mittels Event weitergeleiltet	X	
Das Logging erfasst alle Fehler aus allen Funktionalitätsschichten		X
Das Before/After Pattern wird, wo möglich, mittels using() { } angewendet	X	
IDisposable.Dispose() Methoden werden in jedem Fall aufgerufen	X	

3.4 Flow Control

	Erfüllt	N. Erfüllt
Es existieren keine Schleifen ohne Abbruchkriterien	X	
Es existieren keine toten Programmstücke (z.B. if(false) / while (false) / …)	X	
Rekursive Calls haben immer eine Verankerung und Abbruchbedingung	X	

3.5 Naming

	Erfüllt	N. Erfüllt
Die Namen der Klassen / Variabeln sind sprechend	X	
Verwirrende oder falsche Namen sind nicht vorhanden	X	
Interface-Klassen beginnen immer I (z.B. IDisposable)	X	
Klassen / Properties / Methoden werden mit *PascalCasing* geschrieben	X	
Lokale Variablen / Argumente werden mit *camelCasing* geschrieben	X	
Felder in Klassen werden mit *_camelCasing* geschrieben	X	
Der Code enthält keine Magic Numbers		X

3.6 Tools

	Erfüllt	N. Erfüllt
Die Warnings von Resharper 4.5 werden wo sinnvoll behoben		X
Die Errors von FxCop werden wo sinnvoll behoben		X
Die Abhängigkeiten zwischen den Packages sind aufgrund der Aussagen NDepend und Metrics unidirektional.	X	

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Usability Tests

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument dient dazu Usability Tests durchzuführen. Es geht darauf ein, wie die Usability Tests vorbereitet werden und was getestet werden soll. Die durchgeführten Usability Tests zeigen auf, was für Verbesserungspotenzial gibt.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Es gibt ein Inventar sowie eine Einweisung für den Usability Test. Danach werden die verschiedenen Szenarien aufgelistet, die durchgeführt werden sollen. Abschliessend werden die durchgeführten Usability Tests mit ihren Erkenntnissen aufgelistet.

2 Inventar

- Eine nicht registrierte Benutzerkarte
    - Papier: Zugangsdaten für TFS sowie Projektname und momentane Iteration
- Eine registrierte Benutzerkarte
    - Papier: PIN für die Karte
- Scrum Poker Karten
- Vorbereitete Daten:
    - User-Stories welche zu Iteration 1 gehören
    - User Stories welche nicht zugeordnet sind.
    - Tasks zu den User-Stories

3 Einweisung der Person

Die Person, mit welcher der Usability Test durchgeführt wird, soll auf folgende Punkte hingewiesen werden:

- Vertraulichkeit
- Gedanken laut aussprechen (Was erwarte ich? Bin ich verwirrt?)
- Mit dem Finger zeigen wo man hinschaut
- Grundinformationen zur Benutzung
    - Dual-Doppelklick zum öffnen von Items
- Bei Schätzungen (Story Points, Priorität, Zeiten) einfach etwas wählen

4 Test Szenarios

ID	Karte registrieren	
1	Rolle	ScrumMaster
	Vorbereitung	Eine nicht registrierte Karte, die Zugangsdaten zum TFS sowie der eigene Projektname und die momentane Iteration liegen vor. Die Applikation befindet sich auf dem Startbildschirm.
	Kontext	Sie sind der ScrumMaster und haben von Ihrem Chef die Karte erhalten. Sie sollen diese auf sich registrieren. Er hat ihnen gesagt, Sie sollen dazu die Karte auf den Login-Screen legen. Der Rest werde dann ersichtlich sein.
	Durchführung	Die Test-Person erhält die Karte und bekommt die Aufgabenstellung vorgelesen. Danach sollte keine Hilfe mehr notwendig sein. Nach der Registration soll die Anweisung gegeben werden, sich abzumelden.
	Ziel	Karte registrieren.
	Erfüllt	Die Karte wurde ohne weitere Hilfe registriert. Applikation befindet sich auf dem Startscreen.

ID	Projekt-Planung: Stack Ranking	
2	Rolle	ScrumMaster
	Vorbereitung	Die Applikation befindet sich auf der Übersichtskarte.
	Kontext	Sie sind der ScrumMaster und sollen die vorhandenen User-Stories nach Priorität sortieren.
	Durchführung	Die Aufgabenstellung wird vorgelesen. Danach sollte keine Hilfe mehr notwendig sein.
	Ziel	Die User-Stories sind zugeordnet.
	Erfüllt	Die Testperson hat folgendes automatisch erkannt: - Verschieben der User-Stories von Container zu Container - Verschieben der User-Stories innerhalb des Containers

ID	Projekt-Planung: Iterationen planen	
3	Rolle	ScrumMaster
	Vorbereitung	Die Applikation befindet sich auf der Übersichtskarte.
	Kontext	Sie sind der ScrumMaster und haben User Stories geschätzt und priorisiert. Sie wollen nun eine Vorverteilung der User Stories auf Iterationen vornehmen.
	Durchführung	Die Aufgabenstellung wird vorgelesen. Danach sollte keine Hilfe mehr notwendig sein. Beobachten: Scrolling der Iterationen.
	Ziel	Die User-Stories sind Iterationen zugeordnet.
	Erfüllt	Die Testperson hat folgendes automatisch erkannt: - Verschieben der User Stories von Product Backlog zu Iteration - Verschieben der User Stories von Iteration zu Iteration

ID	Sprint-Planung	
4	Rolle	ScrumMaster
	Vorbereitung	Die Applikation befindet sich auf der Übersichtskarte.
	Kontext	Sie sind der ScrumMaster und wollen den aktuellen Sprint planen. Es soll darauf geachtet werden, dass jede Person genügend ausgelastet ist aber nicht überlastet.
	Durchführung	Die Aufgabenstellung wird vorgelesen. Danach sollte keine Hilfe mehr notwendig sein. Falls eine User-Story mit keinem Task ausgewählt wird. Soll darauf hingewiesen werden, dass die Tasks vergessen wurden.
	Ziel	Jede Person hat genügend Stunden für den nächsten Sprint.
	Erfüllt	- Task wurde erstellt - Mitarbeiter sind gleichmässig ausgelastet.

ID	Daily Scrum	
5	Rolle	ScrumMaster
	Vorbereitung	Die Applikation befindet sich auf dem Startscreen.
	Kontext	Sie sind der ScrumMaster und wollen den Daily Scrum durchführen. Dazu gehört den Status der Tasks anzupassen und die Zeiten einzutragen. Am Ende wollen Sie dem Team den Burndown Chart anzeigen und gegebenenfalls besprechen.
	Durchführung	Karte und PIN aushändigen. Die Aufgabenstellung wird vorgelesen. Danach sollte keine Hilfe mehr notwendig sein. Es soll beobachtet werden ob die Testperson erwartet, dass sich die Zeiten vom Tasks automatisch verändern z.B. Active -> Done => Remaining Stunden= 0
	Ziel	Daily-Scrum durchgeführt und Burndown Chart angesehen
	Erfüllt	- Erfolgreich angemeldet - Tasks verschoben - Zeit eingetragen - Burndown Chart gefunden und verstanden

4.1 ScrumPoker Szenarios

ID	ScrumPoker: Alle UserStories bewerten	
6	Rolle	ScrumMaster
	Vorbereitung	Applikation befindet sich auf der ScrumMap.
	Kontext	Sie sind der ScrumMaster und Ihr Projekt befindet sich in der Projekt Planung. Sie wollen mit ihrem Team alle User Stories in dem momentanen Projekt schätzen.
	Durchführung	1. Scrum Poker Karten geben 2. Aufgabenstellung erläutern 3. Person machen lassen Zu einem Zeitpunkt, in dem sich eine User Story in der Mitte befindet, unterbrechen und sagen, dass der Name der User Story verwirrend ist und dieser deshalb angepasst werden soll.
	Ziel	Alle User Stories schätzen.
	Erfüllt	- Alle User Stories sind bewertet - Eine User Story wurde erfolgreich editiert

ID	ScrumPoker: UserStories der momentanen Iteration bewerten	
7	Rolle	ScrumMaster
	Vorbereitung	Applikation befindet sich auf der ScrumMap.
	Kontext	Sie sind der ScrumMaster und Ihr Projekt befindet sich am Anfang einer Iteration. Sie wollen mit ihrem Team alle User Stories der momentanen Iteration schätzen.
	Durchführung	1. Scrum Poker Karten geben 2. Aufgabenstellung erläutern 3. Person machen lassen Zu einem Zeitpunkt, in dem sich eine User Story in der Mitte befindet, unterbrechen und sagen, dass erkannt worden ist, dass der Entwickler „XYZ“ sich am besten damit auskennt und die User Story diesem zugewiesen werden soll. Hilfeleistung falls nötig: Nur erwähnen, dass es möglich ist Teammitglieder einzublenden über das Menu unten in der Mitte des Bildschirms.
	Ziel	User Stories der momentanen Iteration schätzen.
	Erfüllt	- User Stories der momentanen Iteration wurden bewertet - Keine anderen User Stories wurden bewertet - Eine User Story wurde erfolgreich einem Entwickler zugewiesen o Maximal oben erwähnte Hilfestellung - Der Ablauf und das Bewerten mit Scrum Poker Karten wurde selbst erkannt

5 Usability Test Durchführungen

5.1 Durchführung vom 26.05.2001

ID	Bemerkung
1	- Unknown Text im Startbildschirm ist verwirrend (Fehlermeldung) - Dass man Projekt / Iteration auswählen kann ist nicht erkennbar, die Elemente in der ListBox sehen nicht wie Buttons aus. - Dauer der Iteration nach einem festen Muster z.B. jede Iteration 4 Wochen lang. - Auf der Project Settings fehlte die Möglichkeit um weiter kommen. Menü nicht gesehen. Nicht klar ob bereits gespeichert. Offene Fragen: - Darf die Karte nach dem Anmelden weggenommen werden?
2	- Der „Unrated Container“ wurde als Commandbar wahrgenommen. Die Menü-Bar oben wurde nicht beachtet.
3	- Automatisch Verschieben von einer bewerteten Userstory in den richtigen Container wäre für den automatischen „Fluss“ hilfreich. - User-Story Points grösser darstellen; stärkeres Feedback
4	Keine Anmerkung
5	- Ziehen der User Story auf den Member ging nicht - Verschieben der Eingabe-Maske nicht verständlich. - Bestätigen der Eingabe mit einem „X“ ist verwirrend - Wenn Original Estimate gesetzt wird -> Remaining = Original Estimate (falls nicht 0) - Priorität beschriften - Farbe der Personen im Chart gleich wie die Member-Farben - Zeiteinheiten; Task = Stunden UserStorie-Points = ~Tage - Anzeigen des Parents beim Editieren des Tasks
6	- Runter zählen der Zeit - Zeiten eingeben war verwirrend (auf Closed verschieben und Remaining wurde nicht auf 0 gesetzt)
7	- Name der Gruppierung im Container farbig darstellen. - Verschieben des Tool-Fensters war verwirrend (geht nur am Rande)

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Resultate und Weiterentwicklung

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Dieses Dokument zeigt das Resultat von ScrumTable und was für Weiterentwicklungen während der Entwicklung vorgeschlagen wurden aber nicht Umgesetzt werden konnten.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Zuerst wird ausgewertet, wie weit ScrumTable die Anforderungen erfüllt. Daraufhin werden die Weiterentwicklungen beschrieben. Ebenfalls wird darauf eingegangen, wie man weiter Vorgehen könnte um ScrumTable noch attraktiver zu machen.

2 Resultate

Das Resultat wird in Teil I - Technischer Bericht im Kapitel Resultate, Bewertung und Ausblick behandelt.

3 Möglichkeiten der Weiterentwicklung

In diesem Kapitel werden auf Weiterentwicklungen an ScrumTable, welches auf Microsoft Surface läuft, direkt beschrieben. Ebenfalls werden wichtige und nützliche Tools oder Programme auf dem Desktop erläutert, welche für den Produktiven Einsatz von ScrumTable sehr hilfreich wären.

Übersicht mit Wichtigkeit und Aufwand kann in Teil I – Technischer Bericht im Kapitel Resultate, Bewertung und Ausblick entnommen werden.

3.1 Surface (ScrumTable)

3.1.1 Memory Leaks beheben

Momentan bestehen noch zahlreiche Memory Leaks, welche zu beheben sind. Besonders wenn ScrumTable für eine längere Zeitdauer und auf vielen verschiedenen Screens benutzt wird, kann die Applikation mit der Zeit langsam werden.

Dazu wäre ein „Memory Tracer“-Tool zu benutzen, welches aufzeigt wie die Objekte verbunden sind. So könnte man sehen, wieso Objekte vom Garbadge Collector nicht abgeräumt werden und noch am Leben sind. Man könnte diese Objekte in der Dispose-Methode besser aufräumen.

3.1.2 Performanz verbessern

Es gibt einige Bereiche, in welchen die Performanz optimiert werden könnte.

Als Beispiel das Laden der Elemente vom TFS. Wenn alles auf ein Mal geladen wird, wird dies Parallel geladen. Wenn die Elemente jedoch einzeln geladen werden, aufgrund des Lazy Loading, werden diese Daten sequentiell geladen. In den meisten Ansichten werden jedoch alle Stories geladen, welche dann auch die zugewiesenen Tasks mit laden. Dies könnte behoben werden indem man z.B. generell gleich nach dem einloggen alle Daten parallel einlädt.

Ein weiterer Ort wäre die Priorisierung in der Projekt Planung. Wird ein Element als erstes Platziert, wird die Priorität aller Stories in demselben Container um 1 erhöht. Somit wird auch auf jedem Save() aufgerufen. Das speichern über Netzwerk braucht so ziemlich viel. Dies kann durch ein Batch-Save (wird vom TFS angeboten) schneller gemacht werden. Dabei ist zu beachten, dass dasselbe noch auch in dem Container geschieht, aus welchem das Element gezogen wird (z.B. von „could“ zuoberst nach „must“ zuoberst).

3.1.3 Anzeigen des Task-/Story-Boards im Stand-By Modus

Wenn niemand an ScrumTable arbeitet und der Surface Tisch in den Stand-By Modus geht, soll das Task- oder Storyboard angezeigt werden.

3.1.4 Löschen von Items oder der Verlinkung zu Entwickler

ScrumTable unterstützt das Löschen von Items noch nicht. Dies sollte jedoch in einer zukünftigen Version möglich sein.

Falls einem Task oder einer User Story ein Entwickler zugeordnet worden ist, kann dies momentan nicht rückgängig gemacht werden. Man kann zwar einen anderen Entwickler zuweisen, jedoch bleibt die User Story oder der Task stetig jemandem zugewiesen.

3.1.5 Skizze per Zeichen zu User Story oder Task hinzufügen

Oft möchte man einer User Story eine Skizze hinterlegen, die kurz in der Sitzung gezeichnet wird. So soll ScrumTable es ermöglichen, die Skizze gleich in der Applikation hinzuzufügen. Man sollte für eine User Story, oder Task, eine solche Skizze gleich auf dem Bildschirm zeichnen können. Diese soll dann auf dem SharePoint abgelegt und mit der User Story, oder Task, verlinkt werden.

3.1.6 Skizze per Bluetooth zu User Story oder Task hinzufügen

Diese Funktionalität baut auf derjenigen im vorherigen Abschnitt auf. Es kann sein, dass ein Entwickler, oder der Product Owner, bereits eine Zeichnung zur Sitzung mitbringt. Diese soll man nun z.B. per Handy fotografieren und dann per Bluetooth an ScrumTable senden können um diese auf dem SharePoint abzulegen und auch gleich mit der User Story, oder dem Task, zu verlinken.

3.1.7 Automatisches Auslesen der IDs der Sprints (Iteration) aus dem SQL Server

Für das korrekte Darstellen der Burn Down Charts sollte jeweils nach den IDs der Sprints (Iteration) gefiltert werden, damit nicht zugehörige Items nicht mit einbezogen werden. Momentan lässt sich dies über einen Eintrag im SharePoint manuell machen. (Siehe Anleitung in Teil II - SW-Projektdokumentation in Kapitel Softwaredokumentation)

Um dies zu automatisieren, müsste ScrumTable auf dem SQL Server die Work Item Iteration Hierarchie selbständig durchgehen und nach der ID suchen.

3.1.8 Area-/Iterationen-Hierarchie (Bsp.: Construction -> Sprint 3, Sprint 4, …)

In der aktuellen Version kann nur die erste Stufe der Iterationen eingelesen werden. E s wäre wünschenswert dies auch für verschachtelte Areas/Iterationen zu ermöglichen.

3.1.9 Sprint Planning: Kapazitäten, Ferien und Abwesenheiten erfassen und editieren

Momentan ist in der Applikation nicht ersichtlich, wie viel Zeit ein Entwickler im Sprint zur Verfügung hat. Die Applikation weiss nichts von Feiertagen oder Ferien, die während dem Sprint auftreten. Momentan ist nur ersichtlich wie viel Zeit einem Entwickler zugeordnet ist und wie sich dies zu den anderen Entwicklern verhält.

Es sollte also möglich sein zu erfassen, wie viele Stunden ein Entwickler pro Tag an dem Projekt arbeitet. Denn der Entwickler könnte ja nur Teilzeit an dem Projekt arbeiten. Ebenfalls sollen Feiertage und Ferien der Entwickler eingetragen werden können. Dadurch liesse sich dann gut berechnen, wie viel Zeit ein Entwickler in einem Sprint wirklich für das Projekt verwenden kann.

Der TFS bietet die Möglichkeit, dies in einem Excel Sheet zu erfassen. Dieses Excel Sheet liegt auf dem SharePoint. Es wäre also gut, wenn ScrumTable dieses Excel Sheet aufsuchen würde und einliest. Die vorgenommenen Änderungen würden dann in das Excel Sheet zurück gespeichert.

3.1.10 Verschlüsselte Verbindung zum Team Foundation Server

Es bietet sich an eine verschlüsselte Verbindung zum Team Foundation Server aufzubauen. Dies ist natürlich sicherer und für viele Firmen ein Muss. Die Frage stellt sich hier dann, ob dadurch die Geschwindigkeit beeinträchtigt wird. Dies sollte getestet werden.

3.1.11 Weitere Zielsysteme neben dem Team Foundation Server MSF Agile 5.0

Da nicht alle Firmen, welche Scrum einsetzen, besitzen einen Team Foundation Server. Es wäre wünschenswert, wenn sich weitere Systeme mit ScrumTable verbinden lassen. So wäre es schlau eine Unterstützung für Jira und Trac anzubieten.

Ebenfalls wäre eine Unterstützung für das „Scrum for Team Systems“ Template, welches auf dem Microsoft Team Foundation Server 2010 installiert werden kann, wünschenswert.

ScrumTable wurde so aufgebaut, dass das Einbinden von weiteren Systemen möglich ist. Man muss für diese Systeme dazu einen entsprechenden Connector schreiben. Wie die Connectors aufgebaut sind, wird in Teil II – SW-Projektdokumentation in Kapitel Implementation beschrieben.

3.1.12 Undo und Redo

Damit ungewünschte Änderungen schnell rückgängig gemacht werden können, ist eine Unterstützung für Undo und auch Redo nötig.

3.1.13 Project Planning: Sprints: Sprints hinzufügen

Momentan lassen sich über ScrumTable keine Sprints hinzufügen. Dies muss manuell über den TFS gemacht werden. Es wäre wünschenswert, die Sprints gleich in der Projekt Planung zu machen. Dann wäre auch denkbar, den Start/End Termin des Sprints in dieser Ansicht anzupassen. Diese Termine werden aktuell in den Einstellungen gepflegt.

3.1.14 Kollaboration

Ganz praktisch für entfernte Teams wäre es, wenn man über die Raumgrenze hinweg gemeinsam eine Sitzung mit ScrumTable abhalten könnte. Dabei würden beide Orte einen Microsoft Surface Tisch besitzen wo die ScrumTable Applikation läuft. Ziel ist es nun, dass man die gleiche Ansicht auf Daten von ScrumTable hat und auch gleich sieht, was die Anderen darin machen. Nebenbei würde eine Videokonferenz stattfinden, über welche man miteinander spricht und sich gegebenenfalls auch sehen kann.

Die beiden Lösungsansätze werden nun kurz beschrieben:

3.1.14.1 Einfache Lösung

Hierbei würde an einem Ort die ScrumTable Applikation laufen und 100% wie gewohnt nutzbar sein. Eine Seite bildet den Server-Teil der Lösung. An einem anderen Ort würde eine Art Client-Applikation laufen, welche eine Verbindung zu ScrumTable aufnimmt (Client-Teil). Nach dem Aufbau der Verbindung würde auf dem Client-Teil eine Live-Übertragung des Screens vom Server angezeigt werden. Im Client kann somit keine Veränderung gemacht werden. Dies müsste dann dem entfernten Server oder den entfernten Leuten übertragen werden. Da sowieso eine Audio-Konferenz läuft, stellt dies kein Problem dar. Um die Audio-Konferenz aber doch etwas zu vereinfachen, soll auf dem Client die Möglichkeit bestehen, auf den Bildschirm zu zeichnen. Das gezeichnete würde nach

wenigen Sekunden wieder verschwinden. Diese Zeichnung würde dann zum Server übertragen und in einem Overlay angezeigt werden. So kann man einfach zeigen, was man wohin schieben möchte.

Mit dieser Variante ist nur bei einem Kommunikationspartner eine Verbindung zum Team Foundation Server nötig.

3.1.14.2 Schwierigere Lösung

In dieser Lösung soll es möglich sein, an beiden Standorten Veränderungen vorzunehmen. Dies erfordert jedoch eine Übertragung aller Aktionen, die vorgenommen werden. Dazu wird wahrscheinlich ein Server in der Mitte notwendig.

Ebenfalls stellt sich hierbei die Frage, was geschieht, wenn beide Seiten gleichzeitig am selben Item ziehen. Wird es gesperrt, sobald berührt? Oder werden einfach beide Touch Events dem Item weitergegeben? Durch diese Lösung könnte man beispielsweise ein Element gemeinsam über zwei Tische mit je einem Finger (einem Finger lokal, einem Finger entfernt) vergrössern.

Um einige Probleme zu beheben und das Ganze etwas zu vereinfachen wäre es denkbar, jeweils nur auf einem Tisch Veränderungen vornehmen zu können. Man muss dann um Erlaubnis bitten das „Editier-Recht“ zu erhalten. Wird einem dies erteilt, wird die Applikation auf der anderen Seite gesperrt.

3.1.15 Sprache der Login-Karte hinterlegen

Damit nicht jedes Mal die eigene Sprache ausgewählt werden muss, wäre es denkbar, in den Daten der Login-Karte die zuletzt benutzte Sprache zu speichern. Sobald die Login-Karte dann auf den Tisch gelegt wird, würde diese Sprache ausgewählt werden.

3.1.16 Änderungen vom Team Foundation Server als Ereignisse entgegennehmen

In der momentanen Version kann es geschehen, dass jemand über den Browser ein Element auf dem Team Foundation Server anpasst und diese Information nicht gleich auf dem Tisch sichtbar wird. Erst durch ein erneutes Laden wird diese Inkonsistenz sichtbar. Somit kann eine Exception auftreten, wenn zwei Personen nacheinander ohne neues Laden das Item editieren.

Der Team Foundation Server schickt Ereignisse um zu informieren, dass ein Item geändert wurde. Diese werden momentan noch nicht entgegengenommen. Jedoch wäre die Behandlung der Ereignisse praktisch um allfällige Probleme zu vermeiden. So könnte man dann über den Browser ein Element ändern und man sähe diese Änderung auch gleich in ScrumTable.

3.1.17 Bugs anzeigen und zuweisen

Erfasste Bugs sollen wenn möglich im Daily Scrum sichtbar sein. Kritische Bugs sollten so schnell wie möglich behoben werden. So ist es denkbar, diese Bugs im Daily Scrum gleich einem Entwickler zuzuweisen. Diese Bugs sieht man dann auch in den eigenen Tasks im Daily Scrum.

Momentan unterstützt ScrumTable keine Bugs.

3.1.18 Retrospektive: „Sprint-Verbesserungs“-Notizen

In der Retrospektive werden jeweils gute Ziele gesetzt, was man im nächsten Sprint am Prozess oder der Arbeitsweise verbessern möchte. Jedoch ist es gefährlich, dass man die Erkenntnisse schnell

wieder vergisst. So sollen die Erkenntnisse möglichst sichtbar sein, so dass man z.B. während dem Daily Scrum an diese erinnern kann.

So könnte man ScrumTable erweitern, dass man die Notizen oder Ziele in der Retrospektive erfassen kann und diese dann im Daily Scrum präsent hat.

MSF for Agile v5.0 unterstützt solche Items nicht von Haus aus. Man müsste dieses Template also erweitern, was zu Problemen führen könnte. Man müsste also gut abklären, wie man diese Items erfassen möchte. Möglich wäre es auch, diese auf dem Sharepoint oder nur lokal abzuspeichern.

3.1.19 Installer mit Mapping

Firmen haben mit Team Foundation Server jeweils die Freiheit, die Templates nach ihrem Belieben anzupassen. Wird dies gemacht, kann dies zu Problemen mit ScrumTable führen. So können Felder anders heissen, als von ScrumTable angenommen. Ein Installer könnte das Mapping-Problem lösen. Über den Installer können die ScrumTable Feldern den Team Foundation Feldern zugewiesen (also gemappt) werden.

3.1.20 Splitten einer User Story oder eines Tasks

Wird während einer Sitzung bemerkt, dass eine User Story oder ein Task zu gross ist und unterteilt werden sollte. Man könnte die Unterteilung so anbieten, dass ein weiteres Item mit denselben Informationen erstellt und zum Editieren gegeben wird. Dies würde dem Entwickler etwas Arbeit ersparen.

Die Frage stellt sich hierbei, ob dieses Feature wirklich auch genutzt werden würde. Grundsätzlich könnte einfach ein neuer Task erstellt werden ohne über das Splitten-Feature zu gehen.

3.1.21 Daily Scrum: Herausziehen von Team Members

Falls ein Entwickler zum Zeitpunkt des Daily Scrum fehlt, kann dieser per herausziehen aus der Liste entfernt werden. Dies könnte der Scrum Master am Anfang der Sitzung tätigen und man hätte dann nur noch die wirklich anwesenden Personen in der Liste der Entwickler für das Meeting.

Fraglich ist, ob das Entfernen mehr Aufwand oder mühsamer ist, als einfach kurz „Weiter“ zu klicken, wenn eine nicht Anwesende Person auf dem Bildschirm erscheint.

3.1.22 Release Backlog

Einige Firmen arbeiten mit Release Backlogs, welche über mehrere Sprints gehen. Dabei wäre es praktisch, wenn man solche über ScrumTable verwalten könnte.

Den verschiedenen Release Backlogs sollte dabei eine Farbe zugeordnet werden können. Diese Farbe ist dann bei den User Stories sichtbar um zu sehen, welche User Story zu welchem Release gehört.

Ebenfalls sollte pro Release ein Burn Down Chart angezeigt werden können.

3.2 Desktop

3.2.1 Gadget: Eigene Tasks

Dieses Gadget soll unter Windows Vista oder Windows 7 auf dem Desktop die eigenen Tasks eines Entwicklers anzeigen. Dabei besonders diejenigen, welche sich im Status „In Progress“ befinden. Dies

ermöglicht dem Entwickler gleich auf dem Desktop sehen zu können, was er im Daily Scrum gesagt hat, dass heute seine Arbeit sei.

Ein optionales und praktisches Feature wäre, wenn man beim Task gleich die Zeit editieren könnte.

Ein solches Gadget würde den Entwicklern die Arbeit sehr erleichtern.

3.2.2 Task-, Story-Board oder Burndown Chart

Da das Task- und Story-Board ein wichtiger Bestandteil von Scrum ist und man diese oft sehen möchte, sollte dafür eine Desktopapplikation geschrieben werden. Dabei gibt es zwei mögliche Anwendungsfälle:

1. Ein Entwickler oder der Scrum Master möchte auf seinem Desktop schnell ansehen, welche Tasks wo stehen und wie der Burn Down Chart aussieht.
2. Man möchte im Eingang zum Büro den Burn Down Chart gleich sehen. Ihn also auf einem aufgehängten Flat Screen anzeigen.

Der erste Punkt lässt sich durch ein Windows Vista oder Windows 7 Gadget einfach umsetzen, wenn man nur den Burndown Chart sehen möchte. Für das Task- oder Story-Board wäre wohl eine Applikation nötig. Für den zweiten Punkt wäre eine Applikation nötig, die den Burn Down Chart im Vollbild anzeigt.

3.2.3 Ausdrucken von Scrum Poker Karten für ScrumTable

Mit diesem Programm soll es möglich sein, vollständige Karten Sets für Scrum Poker auszudrucken. Auch sollten die Identity Tags für vorhandene Karten ausdruckbar werden.

4 Vorgehen

Als erstes sollten sicherlich die Memory Leaks, welche noch vorhanden sind, behoben werden.

Für das weitere Vorgehen kommt es darauf, was das Ziel ist. Dabei unterscheiden wir in diesem Fall Weiterentwicklungen im Stile von Bachelor- oder Masterarbeiten und Weiterentwicklungen um das Produkt Markttauglich zu machen.

4.1 Bachelor- oder Masterarbeiten

Für weitere Bachelor- oder Masterarbeiten wäre sicherlich das Thema Kollaboration sehr interessant wie auch eine Herausforderung. Ebenfalls könnten in diesem Rahmen eine Anbindung an andere Systeme oder das Erfassen und Auswerten von Kapazität, Feiertagen und Abwesenheiten interessant sein.

4.2 ScrumTable Markttauglich machen

Falls man möchte, dass Firmen die Software so schnell wie möglich benutzen können, wäre es von Nöten die wichtigsten Desktop Gadgets zu entwickeln. Evtl. könnte auch ein Tool für Team Foundation Server gefunden werden, welches mit ScrumTable zusammenarbeitet.

Je nach Firma, welche ScrumTable einsetzen möchte, müsste man zusätzlich noch einen weiteren Connector programmieren um auch deren Datenquelle als TFS anbinden könnte.

Bereits zu dem Zeitpunkt wäre ScrumTable für den Einsatz bereit. Klar sind nicht alle Funktionen dabei, die man sich noch vorstellen könnte. Jedoch wäre es dann soweit im Alltag einzusetzen und mehr oder weniger einfach damit zu arbeiten. Die weiteren Entwicklungen könnten dann nach und nach getätigt werden um den Gebrauch von ScrumTable immer weiter zu verbessern.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Projekt-Managment

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einführung

1.1 Zweck

In diesem Dokument wird das Projektmanagement beschrieben. Dieses umfasst die Projektübersichten und die Projektorganisation.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Beinhaltet die Projektorganisation, Risikoanalyse und Soll/Ist Vergleich.

2 Projektorganisation

In diesem Kapitel wird die Organisation des Projekts beschrieben. Diese umfasst die Beschreibung der teilnehmenden Parteien und eingesetzte Entwicklungswerkzeuge.

2.1 IFS: Institut für Software

Das Institut für Software stellt die Hardware- und Software-Ressourcen für das Projekt zur Verfügung. Dies umfasst vor allem den Microsoft Surface Tisch und die dazugehörigen Entwicklungslizenzen.

2.2 Lehrbeauftragte

Die Bachelorarbeit wird von Prof. Dr. Markus Stolze (HSR) betreut, welcher ein grosses Know-How im Bereich grafische Benutzeroberflächen und Usability besitzt.

Der Gegenleser (Experte) aus der Industrie ist Markus Flückiger von der Zühlke Engineering AG.

2.3 Bachelor-Team

Name	Funktionschwergewicht
Silvan Gehrig	QS-Chef & Interview-Chef Daten Layer, Domain Layer, Grafiken
Michael Gfeller	Team-Chef & Security-Chef View-Model, View, Security
Boos Patrick	Protokoll-Chef & Kommunikations-Chef View und View-Model, Icons

Tabelle 1 Team

2.4 Arbeitszeiten

Die Arbeitszeiten sind wie folgt definiert:

Montag:	Frei
Dienstag:	Nachmittag
Mittwoch:	Nachmittag
Donnerstag:	Nachmittag
Freitag:	Ganzer Tag anwesend

Dazu stellt die HSR einen Bachelorarbeitsraum zur Verfügung. Bei nicht erscheinen ist das Team zu informieren.

Bei Bedarf kann auch übers Wochenende am Projekt gearbeitet werden. Dies sollte entsprechend des Energized-Work Paradigma aus diversen agilen Software-Entwicklungsprozessen nur in Ausnahmefällen angewendet werden.

Die Projektmeetings mit Prof. Dr. Markus Stolze werden normalerweise Mittwochnachmittags um 15.00 durchgeführt.

Insgesamt stehen pro Student gemäss dem Bachelor-System rund 360 Stunden zur Verfügung. Das heisst, das Projekt hat einen Gesamtumfang von rund 1080 Stunden, das macht pro Student einen durchschnittlichen Aufwand von rund 21 Stunden pro Woche.

2.5 Entwicklungsprozess

ScrumTable wird nach dem Vorbild der agilen Software-Entwicklung realisiert. Dazu wird nach Möglichkeit das Agile Manifesto (beschrieben unter http://agilemanifesto.org/) eingehalten.

Es wird nicht nach einem konkreten Entwicklungsprozess vorgegangen. Um die Entwicklung zu unterstützen, sollte im späteren Teil Scrum eingesetzt werden.

2.6 Entwicklungswerkzeuge

Programm	Verwendung	Version
Microsoft Team Foundation Server	Zentrale Verwaltung des Programm-Codes.	2010 RC2
Microsoft Windows SharePoint Services	Portal für die Zusammenarbeit und das Management der Dokumente.	2007
Microsoft Office	Office Programm für das Erstellen der Dokumente.	2007
Microsoft Visual Studio Team Suite mit Team Explorer	Entwicklungsumgebung	2008
ReSharper	Entwicklungsunterstützung	4.5
Enterprise Architekt	UML-Designer	7.5.850
nDepend	Analysise des Codes; Code-Metrik	3.0.2.4769
FxCop	Überprüfung vom CIL-Code	1.36
Reflactor	.NET-Binaries dekompilieren	6 .1.0.11
TestDriven.Net	Unterstüzung für Testing im Visual Studio	3.0.2743
Windows Vista	Surface SDK unterstützt offizielle nur diese Version von Windows	SP2
CorelDRAW Graphics Suite X3	Bildbearbeitungs-Program	13

Tabelle 2 Entwicklungsumgebung

3 Meilensteine

Nr.	Woche-Nr.	Inhalt
1	3	Paper-Prototype vorführbereit Anforderungen spezifiziert
1a	5	Erste Versuche auf dem MS-Surface Tisch. Abklärung der GUI-Möglichkeiten vom MS-Surface
2	8	Interviews abgeschlossen Zwischenpräsentation
2a	13	Prototyp mit dem gesamten Scrum-Ablauf inkl. Security
3	17	Präsentation und Abschluss: - Die Dokumentation ist nachgeführt und abgabebereit

Tabelle 3 Meilenstein

3.1 Meilenstein-Feedback

3.1.1 Meilenstein 2a

Das folgende Feedback wurde vom Kunden für den Meilenstein 2a abgegeben:

Nr.	Beanstandung	Analyse
1	Navigationsführung fehlt / nicht verständlich	Dies ist eine kleine Verbesserung und wird für die Endversion wie vorgeschlagen angepasst.
2	Fehlende Automation; Standardwerte nicht immer logisch gesetzt.	Dies ist eine kleine Verbesserung und wird für die Endversion wie vorgeschlagen angepasst.
3	Release-Backlog fehlt	Ein Release-Backlog wäre wünschenswert aber nicht im Umfang dieser Arbeit. Wird als zukünftige Arbeit vermerkt.

4 Aufwandschätzung, Zeitplan

Aufwandschätzung und Zeitplan sind im separaten Projektplan Excel-File untergebracht.

5 Risiken

ID	Risiko	Auswirkungen	Massnahme	Kosten des Schadens [Std]	Wahrscheinlichkeit des Eintreffens	Gewichteter Schaden [Std]	Priorität
1	Ausfall vom TFS, welcher auf einem Schul-Rechner läuft	Ein Teil der Arbeit geht verloren	Backup vom TFS und tägliches Backup auf eine externe Festplatte	50	20%	10	Hoch
2	Keine Dokumentation über TFS SDK und Surface SDK	Längere Einarbeitungszeit	Abklärungen, ob Ressourcen im Internet vorhanden sind.	40	20%	8	Mittel
3	Die Instabilität der PCs verschlimmert sich.	Ausweichen auf die eigenen Laptops. Aufsetzten von der Umgebung auf dem Laptop	Keine spezielle.	100	10%	10	Niedrig
4	Umsetzung des Paper-Prototype nicht möglich wegen technischen Einschränkungen	Anpassen der Funktionalität, evtl. der Requirements	Mit dem ersten Prototyp abklären	50	20%	10	Mittel
5	Surface Controls funktionieren nicht wie erwartet	Anpassen der Funktionalität oder der Oberfläche/ Eigene erstellen.	Teil der technischen Abklärungen im Vorfeld tätigen	100	20%	20	Hoch
6	Security lässt sich nicht implementieren	Einsetzen im Produktiven Einsatz nicht möglich		100	10%	10	Hoch
7	Internationalisierung	Feature wird gestrichen		100	1%	1	Mittel
8	MVVM Umsetzungsschwierig-keiten	Komplizierte Struktur im Code		100	10%	10	Hoch
9	Dependency Injection Framework läuft nicht wie erwartet	Komplexe Struktur mit vielen Parameter die durchgereicht werden		50	10%	5	Mittel
10	TFS-API schwer nutzbar	Anbindung zum TFS wird erschwert		200	5%	10	Hoch
11	Auslesen der Teammitglieder	Überarbeiten der Oberfläche		20	50%	10	
	Totale Rückstellung			910		104	

5.1 Abklärung der Risiken

ID	Kommentar	OK
1	Tägliches Backup des TFS eingerichtet.	✓
2	Dokumentation ist vorhanden. Detaillierte Fragen werden nicht beantwortet.	✓
3		
4	Die Controls wurden geprüft und einige als schlecht befunden. Diese wurden nachgebaut oder optimiert.	✓
5	Der Paper Prototype wurde so einfach gehalten, dass dieser kein Problem darstellt.	✓

5.2 Technische Abklärungen

ID	Bezeichnung	OK
5	Probleme mit Surface Controls Kommentar: Diverse Probleme mit der Surface SDK 1.0. Bekannte Bugs/nicht vorhandene Features machen manche Controls nicht nutzbar. Diese wurden nachgebaut oder vorhandene Controls um die gewünschte Funktionalität erweitert.	✓
6	Security lässt sich nicht implementieren Kommentar: Die Security ist gut dokumentiert und diverse Codebeispiele sind vorhanden.	✓
7	Internationalisierung Abklärung Kommentar: WPF unterstützt ein eigenes Internationalisierungs-Konzept welches für diese Arbeit zu umfangreich ist. Anwenden eines anderen Konzepts.	✓
8	MVVM Umsetzungsschwierigkeiten Kommentar: WPF verlangt grundsätzlich ein Default- Konstruktor. Dies ermöglicht keine Übergabe von einem Applikation-Context. Als Lösung wird ein Dependency Injection Framework eingesetzt.	✓
9	Dependency Injection Framework läuft nicht erwartet Kommentar: Läuft nach ersten Erfahrungen wie erwartet.	✓
10	Abklären der Connection-Möglichkeiten zum Team Foundation Server. Kommentar: Der Zugang zum Team Foundation Server 2010 ist mittels der von Microsoft angebotenen API relativ einfach.	✓
10	Liste für Iterations-Werte aus dem ScrumTable im SharePoint anlegen. Kommentar: Der SharePoint bietet Web Services, welche Zugriffe auf die darin gespeicherten Daten und Schemas erlauben.	✓

11	Benutzer aus dem Team Foundation Server abfragen und mit Informationen aus dem Sharepoint Portal ergänzen. Kommentar: SharePoint verfügt über einen Web Service, welcher Listen-Daten verarbeitet. Die User-Daten können aus der UserList des SharePoints abgefragt werden.	✓

6 Projektmonitoring

Dieses Kapitel beinhaltet den Soll- und Ist-Zeitvergleich sowie die Code-Statistik.

6.1 Soll/Ist Vergleich

Der komplette Projektplan ist separat auf der CD hinterlegt.
Anzahl Stunden, welche für das Projekt aufgewendet werden sollten:

Bachelor-Punkte	Zeitaufwand pro Person
1 ECTS	25-30 h
12 ECTS	330 h (27h pro ECTS)
36 ECTS für 3 Personen	990.00 h

Zeitvergleich

Für die 990 Stunden wurde vor Projektbeginn die folgende Zeiteinteilung festgelegt.

- Viel Zeit für Usability und Validierung des User Interfaces.
- Für die Schätzung des Aufwandes der Dokumentation, haben wir uns an das SA-Projekt gehalten.
- Die Zeit für Meetings wurde vorauskalkuliert.
- Das Projekt ist sehr GUI-lastig und deshalb erhält die Implementation 50% der Zeit.

Daraus erfolgt folgende Einteilung der Zeit:

Erwartete Leistung für die Bachelorarbeit 990 h, davon geplant	
25% Dokumentation	297.00 h
25% QS & Tests & Meetings	297.00 h
50% Implementation	500.00 h

Der Vergleich zeigt, dass der Anteil der Dokumentation gut geschätzt wurde. QS & Test hat darunter gelitten, dass uns 2 Wochen abhanden gekommen sind. Die Funktionalität sollte fertig gestellt werden und deshalb wurden nur die primären Kern-Funktionalitäten getestet. Die Tests für die sekundären Funktionalitäten waren eingeplant aber durch Umstellung des Zeitplanes musste von QS & Tests Zeit abgezogen werden.

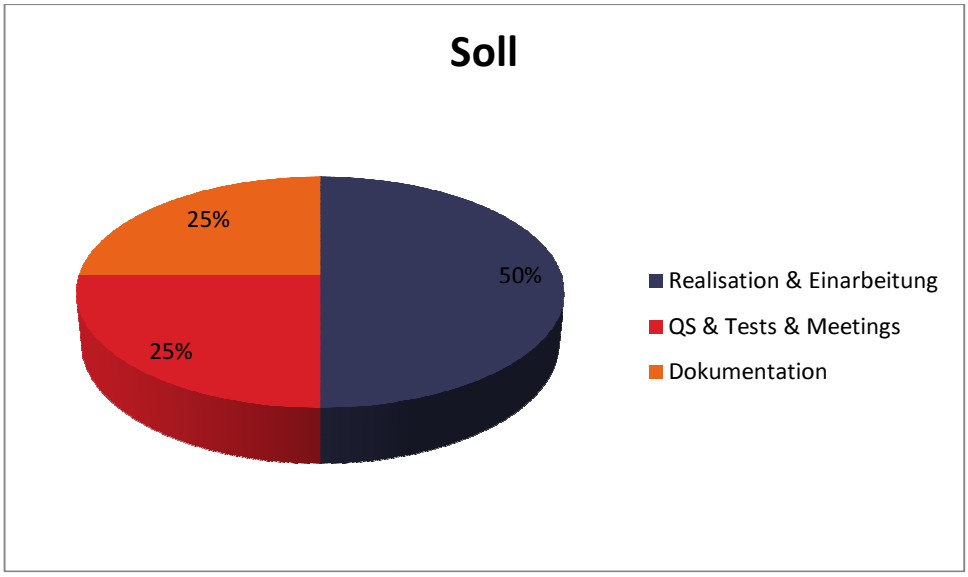


Abbildung 1 Geplante Soll-Zeit – Total ca. 1000 Stunden

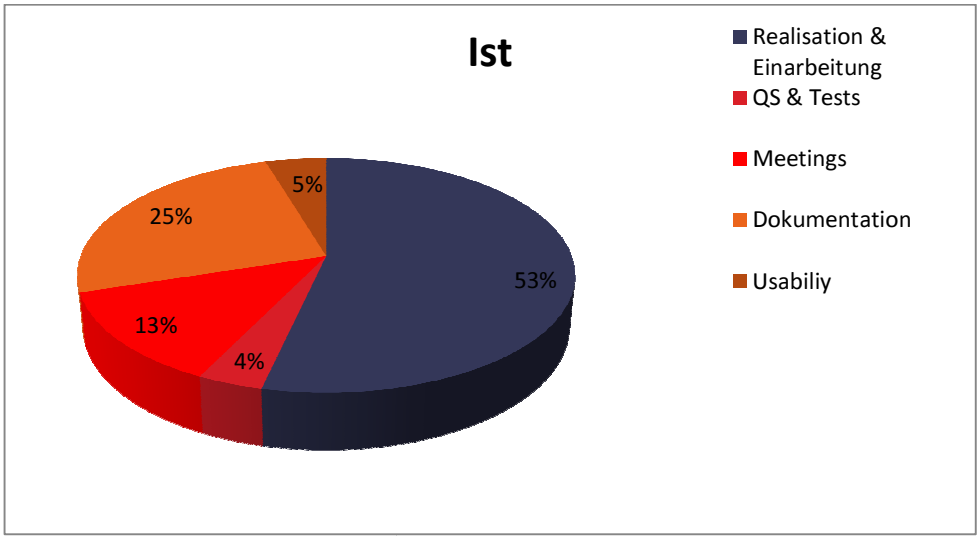


Abbildung 2 Soll Zeit: Effektive Ist-Zeit – Total ca. 1100 Stunden

6.2 Codestatistik

Dieses Kapitel beinhaltet die Codestatistik. Es werden die Abhängigkeiten innerhalb des Projektes analysiert.

6.2.1 Umfang des Projektes

Folgend der Umfang des Projektes an verschiedenen Werten gemessen:

Number of IL instructions	58246
Number of lines of code	7307
Number of lines of comment	14124
Number of assemblies	11
Number of classes	368

6.2.2 Abhängigkeiten innerhalb des Projektes

Bei der Analyse der Architektur wurde aufgezeigt, dass kein Strict Layering eingehalten wurde. Folgende Ausführungen sollen dies erklären:

- Erstellen des Injection-Containers bei Applikationsstart
- Überspringen eines Layer um nicht 1:1 Kopien zu generieren. Dies ist aus Sicht der Performance und des Memory-Verbrauchs wünschenswert.
- Das Plug in-Konzept hat in der statischen Analyse zur Folge, dass die Abhängigkeiten von den Konnektoren zum Plug in Loader gehen. Dies aufgrund der Gegebenheit, dass die Plug ins die Schnittstellen-Definition des Plug in Loaders kennen müssen. Aus der Sicht des Programmablaufs sind die Abhängigkeiten allerdings umgekehrt. Dies entspricht dem Dendency Injection Prinzip (auch Hollywood Prinzip), welches als Teil des grösseren Inversion Of Control Prinzips fungiert. Eine mögliche Lösung wäre die Separierung des Plug in Loaders mit den Interface Definitionen.

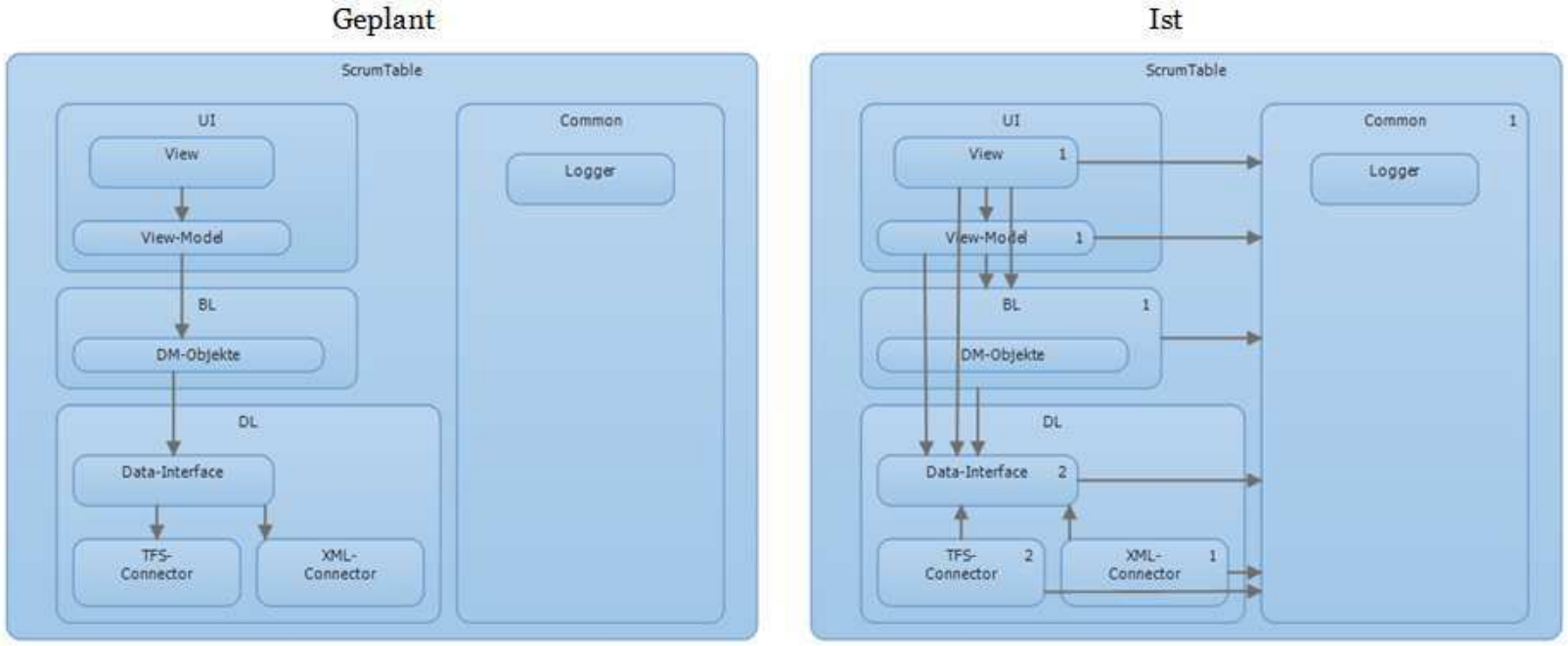


Abbildung 3 Geplante <-> Ist Architektur

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Software

Dokumentation

Bachelorarbeit Frühjahrssemester 2010

Autoren: Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer: Prof. Dr. Markus Stolze
Gegenleser: Prof. Stefan Keller
Themengebiet: Software
Abteilung: Informatik

Inhaltsverzeichnis

1 Einführung

1.1 Zweck

Ziele dieses Dokumentes sind folgende:

1. Ein Administrator kann ScrumTable anhand der Installationsanleitung auf dem Surface Tisch installieren.
2. Ein Scrum Master oder Team Member kann mit Hilfe des Benutzerhandbuches ScrumTable wie gewünscht bedienen.

1.2 Gültigkeitsbereich

Das Dokument ist für die komplette Dauer des Projektes gültig. Bei Änderungen am Dokument wird dies dem Team mitgeteilt.

1.3 Übersicht

Erst wird in diesem Dokument beschrieben, wie sich ScrumTable auf dem Surface Tisch installieren lässt. Weiter wird dem Benutzer ein Handbuch gegeben, welches ihm die nötigen Informationen gibt um ScrumTable zufriedenstellend zu nutzen.

2 Installation

Vor der Installation

Folgende Software muss bereits installiert sein:

- Microsoft Surface SDK 1.0
- Microsoft Surface Shell

Installation

1. Wechseln Sie ins Verzeichnis <CD-Laufwerk>\Installation\
2. Führen Sie Setup.exe aus
3. Folgen sie den Anweisungen auf dem Bildschirm

Nach der Installation

- Surface Shell muss neu gestartet werden, damit ScrumTable erkannt wird.
- ScrumTable ist als neues Symbol in der Shell auffindbar

Hinweis

Die Surface SDK ist auf der CD zu finden:

- <CD-Laufwerk>\Installation\SurfaceSDK\SurfaceSDKWE.msi

3 Benutzerhandbuch

3.1 Das Wichtigste in Kürze

3.1.1 Elemente und Symbole

User Story

- (1)
  - Name
- (2)
  - Stack Rank
- (3)
  - Story Points
  - Grün
    - Fertige Arbeit
  - Rot
    - Ausstehende Arbeit

Task

- (1)
  - Name
- (2)
  - Status
- (3)
  - Zugewiesener Entwickler
- (4)
  - Geschätzter Aufwand
  - Grün
    - Fertige Arbeit
  - Rot
    - Ausstehende Arbeit

Product Backlog

Task-/Story-Board mit Burndown und Burnrate Chart

Sprint Backlog

Review und Retrospektive

Daily Scrum

Scrum Poker

Individueller Report

3.1.2 Operationen

Öffnen / editieren
Um eine User Story oder Task zu öffnen, kann mit zwei Fingern gleichzeitig ein Doppelklick darauf gemacht werden. Somit öffnet sich das Editierfenster.
Ebenfalls kann so ein Burndown oder Burnrate Chart grösser geöffnet werden.

Ziehen / Verschieben
Elemente in Listen können gezogen werden um sie zu verschieben oder anderen Elementen zuzuweisen.
So können diese dann anderen Elementen zugewiesen werden oder in eine andere Liste verschoben werden, sofern dies zulässig ist.

Zuweisen / Droppen
Wenn man ein Element am ziehen ist, kann man dies einem anderen zuweisen. Dies Funktioniert nicht überall, aber an den jeweiligen Orten, wo eine Zuweisung auch gemäss Scrum Sinn macht. Dies geht oft in beide Richtungen. So kann man z.B. einen Entwickler auf einen Task ziehen, oder einen Task auf den Entwickler. Beides hat den Effekt, dass der Task anschliessend dem Entwickler zugeordnet ist.
Beispiele:

- Entwickler an User Story zuweisen
- Entwickler an Task zuweisen
- Task einer anderen User Story zuweisen
- Priorität an Task zuweisen

3.2 Navigation

3.2.1 Anmeldebildschirm

Wenn ScrumTable gestartet wird, erscheint der Anmeldebildschirm. Auf diesen kann man seine Login-Karte legen (1) und sich mit dem persönlichen PIN anmelden. Falls die Login-Karte noch nicht aktiviert ist, kann man diese aktivieren und konfigurieren.

3.2.2 ScrumMap

Sobald man sich angemeldet hat, sieht man die Übersicht. Oberhalb befindet sich eine Auswahlmöglichkeit für (1) Projekt und (2) Sprint. So kann man z.B. schnell in den neuen Sprint wechseln, ohne über die Einstellungen zu gehen.

3.2.3 Menu Bar

Hierbei handelt es sich um die Menu Bar, welche in den Ansichten der verschiedenen Sitzungen vorhanden ist.

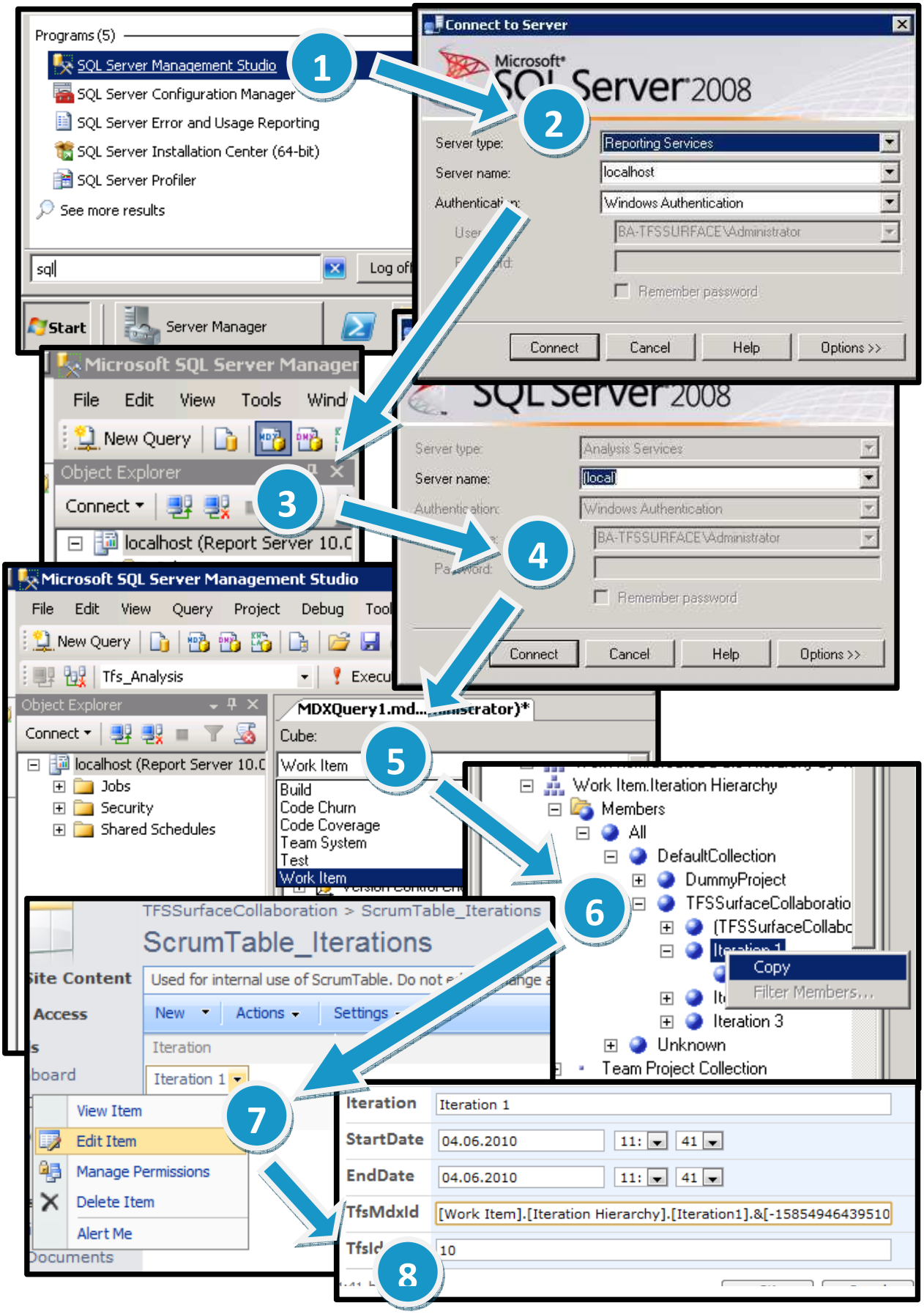
(1) Name des momentanen Projektes
(2) Name des momentanen Sprints
(3) Aktuelle Ansicht ist hervorgehoben
(4) Buttons zum wechseln zu der jeweiligen Ansicht
(5) Aktuelle Sitzungsansicht

3.3 Erweiterte Einstellungen

3.3.1 Burndown und Burnrate Charts auf Sprint limitieren

Standardmässig beachtet der Reporting Server beim auslesen der Burndown und Burnrate Charts nicht die Iteration zu welcher die Items gehören. Der dazu benötigte Parameter ist jedoch stark versteckt und kann mit der momentanen Version nicht automatisch ausgelesen werden. Die benötigte ID kann jedoch auf dem Sharepoint eingetragen werden, so dass ScrumTable diese dann findet. Dazu bitte die folgenden Schritte durchführen. Die entsprechenden Screenshots findet man auf der nächsten Seite dieser Anleitung.

1. Starte „SQL Server Management Studio“
2. Verbinde mit „Reporting Services“ (Connect)
    a. Server type: Reporting Services
3. Öffne eine MDX Query (Icon im Bild hervorgehoben. Dieses anklicken.)
4. Fenster so lassen und verbinden (Connect)
5. Unter Cube „Work Item“ auswählen
6. Unter „Work Item.Iteration Hierarchy“ kann nun zu der Iteration gebrowst werden. Auf der Iteration (Sprint) kann dann mit der rechten Maus drauf geklickt werden um die ID zu kopieren.
    a. Beispiel: Work Iterm -> Iteration Hierarchy -> Members -> All -> DefaultCollection …
7. Trage MdxId auf dem Sharepoint ein
    a. Öffne Web Browser
    b. Gehe zu:
    http://<server>/sites/<collection>/<projekt>/Lists/ScrumTable_Iterations/AllItems.aspx
    Beispiel:
    http://ba-tfssurface/sites/DefaultCollection/TFSSurfaceCollaboration/Lists/ScrumTable_Iterations/AllItems.aspx
    c. Wähle die Iteration, für welche die ID Kopiert wurde und klicke auf „Edit Item“
8. Im Feld „TfsMdxId“ Einfügen (Rechte Maustaste -> Einfügen oder CTRL + V).


Programs (5)
SQL Server Management Studio
SQL Server Configuration Manager
SQL Server Error and Usage Reporting
SQL Server Installation Center (64-bit)
SQL Server Profiler
See more results
sql
Start
Server Manager
1
Connect to Server
Microsoft SQL Server 2008
Server type:
Reporting Services
Server name:
localhost
Authentication:
Windows Authentication
BA-TFSSURFACE\Administrator
Remember password
Connect
Cancel
Help
Options >>
2
Microsoft SQL Server Manager
File Edit View Tools
New Query
Object Explorer
Connect
localhost (Report Server 10.0
3
SQL Server 2008
Server type:
Analysis Services
Server name:
(local)
Authentication:
Windows Authentication
BA-TFSSURFACE\Administrator
Remember password
Connect
Cancel
Help
Options >>
4
Microsoft SQL Server Management Studio
File Edit View Query Project Debug
New Query
Tfs_Analysis
Object Explorer
Connect
localhost (Report Server 10.0
Jobs
Security
Shared Schedules
MDXQuery1.md...
Cube:
Work Item
Build
Code Churn
Code Coverage
Team System
Test
Work Item
5
Work Item.Iteration Hierarchy
Members
All
DefaultCollection
DummyProject
TFSSurfaceCollaboratio
(TFSSurfaceCollabc
Copy
Filter Members...
Iteration 3
Unknown
Team Project Collection
6
TFSSurfaceCollaboration > ScrumTable_Iterations
ScrumTable_Iterations
Site Content
Access
Used for internal use of ScrumTable. Do not
New
Actions
Settings
Iteration
Iteration 1
View Item
Edit Item
Manage Permissions
Delete Item
Alert Me
Documents
7
Iteration
Iteration 1
StartDate
04.06.2010
11:
41
EndDate
04.06.2010
11:
41
TfsMdxId
[Work Item].[Iteration Hierarchy].[Iteration1].&[-15854946439510
TfsId
10
8

4 Referenzhandbuch

4.1 Fehler

4.1.1 Programm startet neu und zeigt rote Meldung an

Falls die Applikation automatisch neu gestartet wird und sich dann auf dem Anmeldebildschirm befindet heisst dies, dass ein Problem aufgetreten ist, welches nicht erwartet wurde. In rot wird jeweils die Meldung geschrieben, die aufgetreten ist.

Massnahmen zur Behebung:

- Kontakt mit Entwicklern aufnehmen und folgende Informationen weiter geben:
    1. Text der Meldung
    2. Was wurde gemacht

Mögliche Ursachen:

- Es wurde gleichzeitig dasselbe Item auf dem Team Foundation Server wie auch auf dem ScrumTable editiert. Dies wird in der momentanen Version nicht abgefangen.
- Ungültige Projektstruktur auf dem Team Foundation Server.

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

Glossar

Bachelorarbeit Frühjahrssemester 2010

Autoren:	Michael Gfeller, Silvan Gehrig, Patrick Boos
Betreuer:	Prof. Dr. Markus Stolze
Gegenleser:	Prof. Stefan Keller
Themengebiet:	Software
Abteilung:	Informatik

1 Glossar

Name	Beschreibung
!=, <>	nicht gleich
.NET	Software-Plattform und Framework von Microsoft.
BL	Der Business Layer in einer Software enthält die Kernfunktionalitäten der Applikation.
Bug	Ein Bug ist ein Fehler in einem Softwareprogram.
Bug-Tracker	Ist eine Software welche Bugs erfasst und ihren Lebenszyklus verfolgt.
Bugzilla	Bugzilla ist ein Bug-Tracking System
Burn Down Chart	Dieses Chart zeigt den Vorschritt während eines Sprints.
Burn Rate Chart	Zeigt aktuell/benötigte Stunden pro Tag an, um das Sprint Ziel zu erreichen.
C#	Microsoft entwickelte die Programmiersprache C# in Zusammenhang mit dem .NET Framework.
Contextual Inquiries	Eine Contextual Inquiry ist eine Beobachtung im Context d.h. in der natürlichen Umgebung der Zielgruppe mit anschliessender Diskussion.
Daily Scrum	Tägliches, 15minütiges, Meeting in Scrum. Während des Meetings sollten 3 Fragen beantwortet werden: - Welche Aufgaben hast du seit dem letzten Meeting fertiggestellt? - Welche Aufgaben wirst du bis zum nächsten Meeting bearbeiten? - Gibt es Probleme, die dich bei deinen Aufgaben behindern?
DB	DB ist eine Abkürzung für Datenbank.
DL	Der Data Layer in einer Software stellt die Verbindung zu einer Datenquelle sicher.
Drag&Drop	Drag&Drop ist die Möglichkeit ein Element durch ziehen zu verschieben.
Eclipse	Eclipse ist eine Softwareentwicklungs-Umgebung
Eclipse Plug In	Ein Plug In erweitert Eclipse um spezifische Funktionalitäten
ETH	Eidgenössische Technische Hochschule Zürich
Framework	Das Framework enthält einen Bausatz von Komponenten für Entwickler.
HSR	Hochschule für Technik Rapperswil
Iteration	Eine Iteration ist ein Teilabschnitt in einem Projekt. Eine Iteration ist in Scrum ein Sprint.
ITS 2009 Conference	Kongress für "Intelligente Transport Systems"
JIRA	JIRA ist ein Bug-&Task-Tracking System.
Mantis	Mantis ist ein Bug-Tracker.
MDX	Multidimensional Expressions (MDX) ist eine Datenbanksprache für OLAP-Datenbanken. Siehe: http://de.wikipedia.org/wiki/Multidimensional_Expressions
Microsoft Surface Tisch	Multi-Touch Tisch von Microsoft
Microsoft Test Manager	Der Test Manager ist eine Test-Umgebung von Microsoft um die Funktionalitäten von Software zu testen.
Model View ViewModel	Ist ein Architektur Pattern von Microsoft für WPF / Silverlight Weitere Informationen unter: http://en.wikipedia.org/wiki/Model_View_ViewModel
MS	Microsoft

MSF for Agile Software Development, Version 5.0	Scrum-Template für den Team Foundation Server von Microsoft
Multi-Touch	Multi-Touch ist eine Technologie welche ermöglicht, dass mit mehreren Fingern auf einen Eingabegerät (z.B. Bildschirm) gearbeitet werden kann.
MVVM	Siehe Model View ViewModel
Planning Poker	Siehe Scrum Poker
Product Backlog	Das Produkt Backlog ist die Sammlung aller User Stories für das jeweilige Projekt. (Begriff aus Scrum)
Produkt Owner	Legt die Ziele der Applikation fest und priorisiert diese. (Begriff aus Scrum)
Project Planning	Anfängliches Meeting um das Projekt zu planen. (Begriff aus Scrum)
Projekt Planung	Siehe Project Planning (Begriff aus Scrum)
Query / Queries	Ein Query ist eine Abfrage, meistens zu einer Datenquelle.
Reporting Service	SQL Server Reporting Services (SSRS) ist ein Server-basiertes Berichtgenerierungssystem von Microsoft
Scrum	Scrum ist ein Vorgehensmodell der agilen Softwareentwicklung.
Scrum Poker	Scrum Poker wird eingesetzt um den Aufwand/Komplexität einer User-Story zu schätzen. (Begriff aus Scrum)
Scrum Prozess	Scrum Prozess beschreibt den Ablauf der agilen Methode.
Scrum-Master	Der ScrumMaster hat die Aufgabe, die Aufteilung der Rollen und Rechte zu überwachen. Ebenfalls überwacht er, dass die Scrum Regeln eingehalten werden. (Begriff aus Scrum)
Scrum-Meetings	Siehe : Daily Scrum; Sprint Planning; Projekt Planning; Review; Review; Retrospective
Server	Ein Server (Software) ist ein Programm, das mit einem anderen Programm, dem Client (engl. = Kunde), kommuniziert, um ihm Zugang zu speziellen Dienstleistungen zu verschaffen. Siehe http://de.wikipedia.org/wiki/Server
SharePoint	Web-Applikation von Microsoft für die gemeinsame Zusammenarbeit
Sprint	Ein Sprint in Scrum ist eine Iteration. (Begriff aus Scrum)
Sprint Backlog	Das Sprint Backlog ist die Sammlung aller User Stories für den jeweiligen Sprint. (Begriff aus Scrum)
Sprint Planning	Meeting jeweils vor einem Sprint um den Sprint zu planen. (Begriff aus Scrum)
Sprint Retrospective	In diesem Meeting wird auf den Sprint zurück geblickt und dieser ausgewertet. Es wird besprochen, wie die gemeinsame Arbeitsweise verbessert werden kann und diesbezüglich Ziele gesetzt. (Begriff aus Scrum)
Sprint Review	In diesem Meeting wird das Produkt des aktuellen Sprints vorgeführt und Vorschläge für den nächsten Sprint gesammelt. (Begriff aus Scrum)
Story-Board	Darstellung von User Stories auf einem Board. Meistens unterteilt in Open/Resolved/Done. (Begriff aus Scrum)
Szenario	Ein Szenario definiert eine Interaktion in der Software.
Tag	Ein Tag ist eine Art Etikette um ein Objekt mit zusätzlichen Informationen zu versehen.
Task-Board	Darstellung von Tasks auf einem Board. Meistens unterteilt in Open/In Progress/Done. (Begriff aus Scrum)
Tasktop	Tasktop ist ein Eclipse Plug In welches Task-Managment Funktionaliäten hinzufügt.

Team Foundation Server	Der Team Foundation Server (TFS) ist eine Plattform für kollaborative Softwareprojekte.
TFS	Team Foundation Server
Touch	Touch ist eine Technologie welche ermöglicht, dass der Finger die Maus ersetzt. Ist ein englischer Begriff. Übersetzung: Kontakt, Berührung
Touch Screen	Ein Touch Screen ist ein Bildschirm der Berührungen erkennt.
Trac	Trac ist ein Task/Bug-Tracking System
Usability	Benutzbarkeit / Benutzerfreundlichkeit
User Centered Design	Dieses Vorgehen zielt darauf ab ein möglichst benutzerfreundliches (Usability) Produkt zu erschaffen
User Story	Eine User Story ist eine Anforderung an die Software.
User-Story Points	Die Komplexität einer User-Story wird anhand von User-Story Points festgelegt. Meist sind die Punkte eine bestimmte Zeitdauer z.B. 1 Punkt = 1 Tag Arbeit.
WIQL	Work Item Query Language: Abfragesprache für den Team Foundation Server
Work Item	Ein Datenspeicher-Element im Team Foundation Server
WPF	Windows Presentation Foundation ist ein Grafik-Framework von Microsoft.
YouTube	YouTube ist eine Video-Plattform im Internet.

Scrum

Scrum (deutsch: *Gedränge*) ist ein Vorgehensmodell der agilen Softwareentwicklung. Ken Schwaber, Jeff Sutherland und Mike Beedle haben Scrum entwickelt und etabliert. Als Software-Entwicklungsmethode wird Scrum das erste Mal in dem Buch *Wicked Problems, Righteous Solutions* beschrieben. Scrum in Produktionsumgebungen wird zum ersten Mal in dem Artikel *The New New Product Development Game* erläutert und später in *The Knowledge Creating Company* weiter ausgeführt von Ikujiro Nonaka und Hirotaka Takeuchi.

Grundannahmen

Scrum wurde zuerst als Methode zur Produkt-Entwicklung von Nonaka und Takeuchi entwickelt (*The New New Product Development Game*). Die Grundlagen von Scrum liegen im Wissensmanagement und wurden später von Jeff Sutherland und Ken Schwaber durch die Hinzunahme streng wissenschaftlicher Vorgehensweisen weiter ausgebaut.

Scrum kann besser verstanden werden, wenn man sich mit der Schlanken Produktion (engl. *lean production*) auskennt. Scrum überträgt keine Erfahrungen aus der Automobilbranche auf die Softwareentwicklung, aber es lässt sich zeigen, dass diese Erfahrungen zum Verständnis der grundlegenden Probleme in der Software-Entwicklung beitragen. In der Automobilbranche gilt das Unternehmen Toyota als ein Vorreiter der Schlanken Produktion. Das sich ständig in der Weiterentwicklung befindende Toyota Production System (TPS) umfasst dabei Methoden und Arbeitsmittel der Produktion und steht im selben Verhältnis zum Toyota Way, der Philosophie hinter dem TPS, wie die Scrum-Methodik zur agilen Softwareentwicklung. Im Mittelpunkt steht bei beiden Systemen die ständige Weiterentwicklung der Mitarbeiter, der Herstellungsprozesse, der Arbeitsmittel und Methoden, unter gleichzeitig konstantem Beibehalten der Grundannahmen, die dahinter stehen. Gemein ist den meisten agilen Vorgehensmodellen dabei die gleichzeitige Weiterentwicklung aller an dem Prozess Beteiligten, auch der Kunden und Partner. Ziel der Grundannahmen ist es, die Produktion ständig zu verbessern, um höchste Qualität bei niedrigstem Aufwand zu erreichen (Kaizen).

Bei Scrum wird grundsätzlich angenommen, dass Produktfertigungs- und Entwicklungsprozesse so komplex sind, dass sie sich im Voraus weder in große abgeschlossene Phasen noch in einzelne Arbeitsschritte mit der Granularität von Tagen oder Stunden pro Mitarbeiter vorher planen lassen. Somit ist es produktiver, wenn sich ein Team in einem festen äußeren Rahmen mit sehr grober Granularität selbst organisiert. Dieses selbstorganisierte Team übernimmt in diesem mit dem *Product Owner* abgestimmten Rahmen die gemeinsame Verantwortung für die Fertigstellung der selbstgewählten Aufgabenpakete. Dabei werden traditionelle Werkzeuge, z. B. zur Kommunikation oder Projektsteuerung sowie durch das Management „von oben“, für die Teamstrukturierung festgelegter Prozesse, die die Zusammenarbeit im Team kontrollieren und regulieren, abgelehnt.

Scrum erfüllt als agile Methode die Bedingungen der agilen Software-Entwicklung, die 2001 im Agilen Manifest u. a. von Ken Schwaber und Jeff Sutherland mitformuliert wurden:

1. *Individuen und Interaktionen* gelten mehr als *Prozesse* und *Tools*.
2. *Funktionierende Programme* gelten mehr als *ausführliche Dokumentation*.
3. Die stetige *Zusammenarbeit mit dem Kunden* steht über *Verträgen*.
4. Der Mut und die Offenheit für *Änderungen* steht über dem *Befolgen* eines festgelegten *Plans*.

Im Folgenden werden Kernpunkte der Scrum-Arbeitsmittel wiedergegeben. Sie sind nicht zu verwechseln mit den hinter diesen Arbeitsmitteln stehenden oben genannten Grundannahmen und gelten lediglich als bewährte Möglichkeiten zur Umsetzung dieser Grundannahmen.

Rollen

Bei Scrum gibt es drei klar getrennte Rollen, die von Mitarbeitern ausgefüllt werden, die im selben Projekt zusammen arbeiten und damit auch dasselbe Ziel haben.

Product Owner

Der *Product Owner* legt das gemeinsame Ziel fest, das das Team zusammen mit ihm erreichen muss. Zur Definition der Ziele dienen ihm User Stories. Er stellt das Budget zur Umsetzung dieser User Stories zur Verfügung. Er setzt regelmäßig die Prioritäten der einzelnen Product-Backlog-Elemente (siehe unten). Dadurch legt er fest, welches die wichtigsten Features sind, aus denen das Entwicklungsteam eine Auswahl für den nächsten Sprint trifft.

Team

Das Team besteht idealerweise aus 7±2 Personen.[1] Es schätzt die Aufwände der einzelnen Backlog-Elemente ab und beginnt mit der Implementierung der für den nächsten Sprint machbaren Elemente. Dazu wird vor dem Beginn des Sprints ein weiteres Planungstreffen durchgeführt, bei dem die höchstpriorisierten Elemente des Backlogs und konkrete Aufgaben aufgeteilt werden. Das Team arbeitet selbstorganisiert im Rahmen einer Time Box (dem Sprint) und hat das Recht (und die Pflicht), selbst zu entscheiden, wie viele Elemente des Backlogs nach dem nächsten Sprint erreicht werden müssen, man spricht dabei von *commitments*.

ScrumMaster

Der *ScrumMaster* hat die Aufgabe, die Aufteilung der Rollen und Rechte zu überwachen. Er hält die Transparenz während der gesamten Entwicklung aufrecht und unterstützt dabei, Verbesserungspotentiale zu erkennen und zu nutzen. Er ist keinesfalls für die Kommunikation zwischen Team und *Product Owner* verantwortlich, da diese direkt miteinander kommunizieren. Er steht dem *Team* zur Seite, ist aber weder *Product Owner* noch Teil des *Team*. Der *ScrumMaster* sorgt mit allen Mitteln dafür, dass das *Team* produktiv ist, also die Arbeitsbedingungen stimmen und die Teammitglieder zufrieden sind. Er tritt somit für die ordnungsgemäße Durchführung und Implementierung von *Scrum* im Rahmen des Projektes ein.

2003 legte Ken Schwaber ein Zertifizierungsprogramm für *ScrumMaster* auf. Das Ziel ist es, die Software-Entwicklung durch das Nutzen von Scrum zu professionalisieren.

Zusammenspiel

Bei der Rollenaufteilung wurde berücksichtigt, dass das Team sich selbst organisiert. Der *Product Owner* gibt nicht vor, welches Teammitglied wann was macht und wer mit wem zusammenarbeitet. Bei Scrum wird von der Annahme ausgegangen, dass das Team sich intuitiv selbst organisiert, und zu jeder Aufgabe dynamisch eine optimale innere Organisationsstruktur bildet, die sich relativ schnell an die sich wandelnden komplexen Aufgaben anpasst. Der *ScrumMaster* hat die Pflicht, darauf zu achten, dass der *Product Owner* nicht in diesen adaptiven Selbstorganisationsprozess eingreift und das Team stört oder Verantwortlichkeiten an sich nimmt, die ihm nicht zustehen.

Artefakte

Product Backlog

Das **Product Backlog** enthält die Features des zu entwickelnden Produkts. Es umfasst alle Funktionen, die der Kunde wünscht, zuzüglich technischer Abhängigkeiten. Vor jedem Sprint werden die Elemente des Product Backlogs neu bewertet und priorisiert, dabei können bestehende Elemente entfernt sowie neue hinzugefügt werden. Hoch priorisierte Features werden von den Entwicklern im Aufwand geschätzt und in den Sprint Backlog

übernommen. Ein wesentliches Merkmal des Backlogs ist die Tiefe der Beschreibung von einzelnen Features. Hoch priorisierte Features werden im Gegensatz zu niedrig priorisierten sehr detailliert beschrieben. Somit wird viel Zeit für die wesentlichen Elemente und wenig für unwesentliche verwendet.

Selected Backlog

Das **Selected Backlog** enthält alle Aufgaben, die notwendig sind, um das Ziel des Sprints zu erfüllen. Es wird von dem Product Owner zusammen mit dem Scrum Team erstellt und gilt somit als Vereinbarung und als Grundlage der späteren Abnahme.

Sprint Backlog

Das **Sprint Backlog** enthält alle Aufgaben, die notwendig sind, um das Ziel des Sprints zu erfüllen. Eine Aufgabe sollte dabei nicht länger als 16 Stunden dauern. Längere Aufgaben sollten in kurze Teilaufgaben zerlegt werden. Bei der Planung des Sprint werden nur so viele Aufgaben eingeplant, wie das Team an Kapazität aufweisen kann.

Die Kapazität berechnet sich gemäß folgender Formel: Kapazität (in Stunden) = Arbeitstage × Anzahl Personen × 7 h. (8h Tag inkl. 1h Puffer) Außerdem enthält das Sprint Backlog die Verantwortlichen der Aufgaben.

Burndown Chart

Das **Burndown Chart** ist eine graphische, pro Tag zu erfassende Darstellung des noch zu erbringenden Restaufwands pro Sprint. Im Idealfall fällt die Kurve kontinuierlich (daher Burndown) und der Restaufwand ist am Ende des Sprints Null. Das Chart lässt anhand der Verlängerung der negativen Steigung bereits während des Sprints erkennen, ob der anfangs geschätzte Aufwand umgesetzt werden kann.

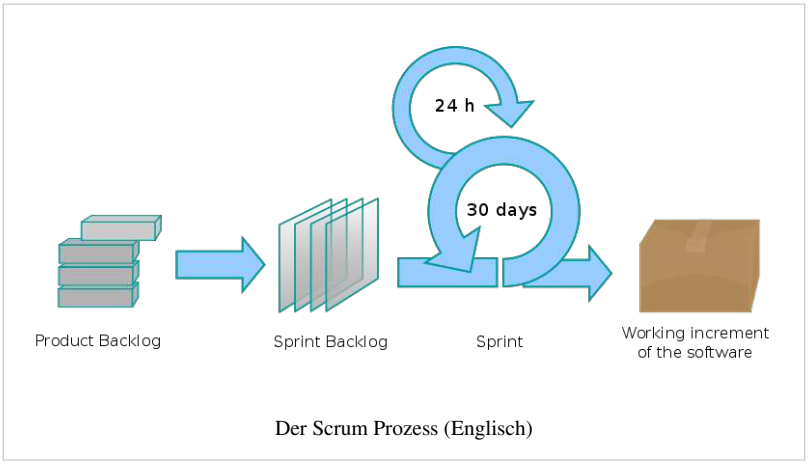
Impediment Backlog

In das **Impediment Backlog** werden alle Hindernisse des Projekts eingetragen. Der *ScrumMaster* ist dafür zuständig, diese gemeinsam mit dem Team auszuräumen.

Zyklusmodell

Sprint Planungstreffen 1

In diesem Treffen erklärt der Product Owner dem Team alle einzelnen Backlog–Einträge die für den nächsten Sprint benötigt werden. Außerdem einigt er sich mit dem Scrum Team auf das Sprint–Ziel. Dieses Sprint–Ziel bildet nach dem Sprint die Basis für die Abnahme. Die höchst priorisierten Einträge des Product Backlog werden entsprechend dem Ergebnis des Treffens in das Selected Backlog übernommen.



Der Scrum Prozess (Englisch)

Sprint Planungstreffen 2

Dieses Treffen organisiert das Scrum–Team eigenverantwortlich. Hier werden alle Arbeiten (also die Einträge im Selected Backlog) an die Mitglieder möglichst gleich verteilt. Dazu werden die Selected-Backlog-Einträge in Aufgaben zerlegt, die in weniger als einem Tag bearbeitet werden können. Aus den Aufgaben und deren Verteilung entsteht das sogenannte Sprint Backlog.

Sprint

Zentrales Element des Entwicklungszyklus von Scrum ist der **Sprint**. Ein Sprint bezeichnet die Umsetzung einer Iteration, Scrum schlägt ca. 30 Tage als Iterationslänge vor; in der Praxis liegt die Dauer (je nach Team) bei 1-4 Wochen. Vor dem Sprint werden die Produkt-Anforderungen des Kunden in einem Product Backlog gesammelt. Auch technische und administrative Aufgaben werden dort aufgenommen. Das Product Backlog muss nicht vollständig sein; es wird laufend fortgeführt. Die Anforderungen für den ersten Sprint sind meistens rasch aufgestellt. Die Anforderungen werden informell skizziert. Für einen Sprint wird ein Sprint Backlog erstellt. In diesen werden Anforderungen übernommen, die während des Sprints umgesetzt werden sollen. Die Entscheidung, welche Anforderungen umgesetzt werden, wird vom Team nach vom *Product Owner* und Kunden festgelegten Prioritäten getroffen. Dazu nimmt sich das Team aus dem Product Backlog die am höchsten priorisierten Aufgaben, die das Team in diesem Sprint für realistisch durchführbar hält. Zum Sprint organisiert sich das Entwicklungsteam selbst, braucht also keine detaillierten methodischen Vorschriften. Am Ende eines Sprints steht immer eine lauffähige, getestete, inkrementell verbesserte Software.

Daily Scrum

An jedem Tag findet ein kurzes (maximal 15-minütiges) **Daily Scrum** statt.

Scrum definiert keine konkrete Uhrzeit für das Meeting, das Meeting sollte jedoch täglich zur gleichen Zeit stattfinden. Empfohlener Zeitpunkt für das Scrum-Meeting ist die Zeit nach dem Mittagessen, da morgendliche Scrum-Meetings oft mit flexiblen Gleitzeitregelungen kollidieren und der müde Punkt nach dem Mittagessen bei einem Scrum-Meeting, das durchaus im Stehen abgehalten werden kann, nicht so stark ins Gewicht fällt wie bei anderen Tätigkeiten. Die Meetings sind kürzer als am Morgen, weil allgemeine Dinge und Neuigkeiten schon vorher diskutiert wurden und die Mitarbeiter mit dem Kopf ganz bei der Arbeit sind.

Das Team stellt sich gegenseitig die folgenden Fragen:

- „Welche Aufgaben hast du seit dem letzten Meeting fertiggestellt?“
- „Welche Aufgaben wirst du bis zum nächsten Meeting bearbeiten?“
- „Gibt es Probleme, die dich bei deinen Aufgaben behindern?“

Die Sitzung dient dem Informationsaustausch des Teams untereinander. Hier geht es darum, dass möglichst alle alles wissen. Falls neue Hindernisse erkannt wurden, müssen diese vom ScrumMaster bearbeitet werden. Dazu werden sie in das Impediment Backlog eingetragen. Im **Daily Scrum** haben nur die Teammitglieder und der *ScrumMaster* eigenständiges Rederecht; alle anderen Zuhörer (Interessierte, z.B. *Product Owner*, Vorgesetzte, Kunden, Gäste, andere Scrum Teams) reden nur, wenn sie gefragt werden.

Größere Projekte werden durch das Einführen von Scrum-of-Scrum Meetings, Product Owner Daily Scrums und ScrumMaster Weekly gesteuert.

Review

Nach einem Sprint wird das Sprint-Ergebnis einem informellen Review durch Team und Kunden unterzogen. Dazu wird das Ergebnis des Sprints (die laufende Software) vorgeführt, eventuell werden technische Eigenschaften präsentiert. Der Kunde prüft, ob das Sprint-Ergebnis seinen Anforderungen entspricht, eventuelle Änderungen werden im Product Backlog dokumentiert.

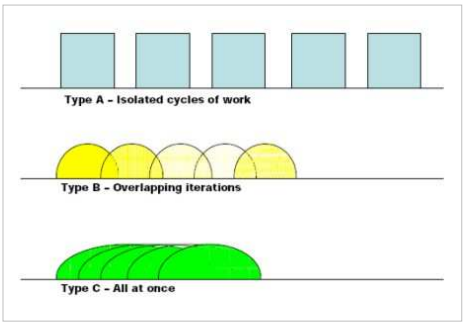
Retrospektive

In der Retrospektive wird die zurückliegende Sprint-Phase betrachtet. Es handelt sich dabei nicht um *Lessons Learned*, sondern um einen zunächst wertfreien Rückblick auf die Ereignisse des Sprints. Eine mögliche Vorgehensweise ist, dass alle Teilnehmer dazu die für sie wichtigen Ereignisse auf Zetteln notieren und sie dem Zeitstrahl des Sprints zuordnen. Anschließend schreiben die Teilnehmer alle Punkte auf, welche ihnen zu den Fragen „Was war gut?“ (Best practice) bzw. „Was könnte verbessert werden?“ (Verbesserungspotential) einfallen. Jedes Verbesserungspotential wird priorisiert und einem Verantwortungsbereich (Team oder Organisation) zugeordnet. Alle der Organisation zugeordneten Themen werden vom *ScrumMaster* aufgenommen und in das Impediment Backlog eingetragen. Alle teambezogenen Punkte werden in das Product Backlog aufgenommen. Egal, wie die Retrospektive gestaltet wird, das Ziel ist, den vergangenen Sprint zu beleuchten und daraus zu lernen.

Wird für die Retrospektiven und deren Vorbereitung nicht genug Zeit eingeräumt, bleiben die Erkenntnisse und Ergebnisse oberflächlich und die Resultate nach jedem Sprint ähneln sich. Dann läuft man Gefahr, dass die Retrospektiven an Stellenwert verlieren oder ganz gestrichen werden, weil die Ergebnisse der Retrospektiven vorhersehbar sind.

Parallelisiertes Pipelining von Sprints

Die Begründer von Scrum in Produktionsumgebungen Takeuchi und Nonaka präsentierten in einem Harvard Business Paper Review drei verschiedene Arten, wie Scrum in Firmen benutzt werden kann. Diese unterscheiden sich vor allem durch die unterschiedliche Lösung der Nullphasen zwischen zwei Sprintphasen. Nullphasen sind Zeiträume, in denen nichts produziert wird, weil verschiedene Regularien, wie Review, Retrospektive, Sprint Planning usw. durchgearbeitet werden müssen:



Typ A - Scrum

> Durchführung von Sprints mit absolutem Fokus auf eine Iteration im Prozess. Nullphasen zwischen den Sprints um Vor- bzw. Nachbereitungen für die Sprints durchzuführen.

Typ B - Scrum

> Entfernung der Nullphasen durch Durchführung der Vor- und Nachbereitungen am Ende des vorangegangenen bzw. Beginn des nachfolgenden Sprints. Reduktion der Aufwände für Vor- und Nachbereitungen der Sprints durch beispielsweise minimale und funktionale Spezifikationen.

Typ C - Scrum

> Aufteilung mehrerer Sprints, oft sogar aus unterschiedlichen Projekten zur gleichen Zeit auf ein Team. Dies erfordert von dem Team ein hohes Verständnis von Scrum-Arbeitsweisen und die Einführung von sogenannten Scrums von Scrums, übergeordneten Scrum Meetings an denen z.B. Product Owner und ScrumMaster von mehreren untergeordneten Scrums teilnehmen, um die groben Aufgaben global zu verteilen.

Bei der Einführung von Typ B und C ist darauf zu achten, dass zunächst Scrum gut eingeführt werden muss (was gut und gerne mehrere Monate dauern kann). Außerdem ist für die notwendige Steigerung der Produktivität ein automatisiertes Managing und Testing notwendig. Erst dann ist die Scrum – Arbeitsweise derart automatisiert, dass man nutzbare Ergebnisse bekommt.

Kritische Betrachtung

Auch das Vorgehensmodell Scrum kann unausgewogen eingesetzt werden und dann scheitern. Ein besonderes Risiko sind dominante Teamplayer, die den Prozess der Selbstorganisation stören, ohne einen gleichwertigen eigenen Beitrag in diesen Organisationsprozess und ohne einen angemessenen eigenständigen Beitrag in die Problemlösung einzubringen.

Zudem muss eine Entwicklungsabteilung generell bereits auf recht hohem Niveau mit einer modernen Programmiersprache arbeiten, um in kurzen Abständen lieferbare, qualitätsgetestete Software zu produzieren. Bei umfangreichen und vielschichtigen Projekten ist es in der Scrumtheorie erforderlich, dass alle Teammitglieder *alle* Aufgaben eines Sprints bearbeiten können, was häufig nicht in Projekten gegeben ist, die auf unterschiedlichen technologischen Basen bestehen. Die häufigen Änderungen erfordern Refaktorierung. Refaktorierung und häufige Lieferungen zum Kunden benötigen eine ausreichende Testabdeckung durch Unit- und Regressionstests. Gerade in alten, gewachsenen Programmen ist eine ausreichende Testabdeckung mit vertretbarem Aufwand oft nicht zu erreichen. Manuelle Tests generieren nach jedem Sprint erneut Testaufwand.

Software

Es gibt diverse Softwareanwendungen wie Agilo for Scrum, Icescrum, Greenhopper oder Scrumworks, um Scrum-basiertes Arbeiten zu unterstützen.

Literatur

- Ken Schwaber: *Agiles Projektmanagement mit Scrum.* Microsoft Press Deutschland, 4. Oktober 2007 (Originaltitel: *Agile Project Management with Scrum*, übersetzt von Thomas Irlbeck), ISBN 978-3866456310.
- Ken Schwaber, Mike Beedle: *Agile Software Development with Scrum.* Prentice Hall, 18. Februar 2002, ISBN 978-0130676344.
- Roman Pichler: *Scrum: Agiles Projektmanagement erfolgreich einsetzen*. dpunkt.verlag, 2008, ISBN 978-3-89864-478-5
- Holger Koschek: *Geschichten vom Scrum: Von Sprints, Retrospektiven und agilen Werten*. dpunkt.verlag, 2009, ISBN 978-3-89864-640-6
- Ken Schwaber: *Scrum im Unternehmen.* Microsoft Press Deutschland, 14. April 2008 (Originaltitel: *The Enterprise and Scrum*), ISBN 978-3866456433.
- Boris Gloger: *Scrum-Produkte zuverlässig und schnell entwickeln*. Hanser Verlag, 2008, ISBN 3-446-41495-9
- Ken Schwaber: *Scrum Development Process, Advanced Development Methods*, 131 Middlesex Turnpike Burlington, MA01803
- Jeff Sutherland, Ph.D.: *Future of Scrum: Parallel Pipelining of Sprints in Complex Projects.* In: Patientkeeper, Inc. (Hrsg.): *{{{Sammelwerk}}}*. Brighton, MA 2005 (http://jeffsutherland.com/SutherlandFutureOfScrumAgile2005.pdf, abgerufen am 24. März 2010).

Siehe auch

- Feature Driven Development
- Extreme programming (XP)
- Paarprogrammierung
- Testgetriebene Entwicklung
- Kanban in der IT
- ständige Refaktorisierungen
- Story-Cards
- schnelle Codereviews

Weblinks

- Die offizielle Website der Scrum Alliance (englisch) [2]
- Die offizielle Webseite der Scrum.org (english) [3]
- Einführung in Scrum [4]
- Artikel über Scrum (Original in IX 6/09 veröffentlicht) [5]
- Scrum auf einen Blick (englisch) [6] (PDF-Datei; 665 kB)

Referenzen

[1] Ken Schwaber, Mike Beedle: *Agile Software Development with Scrum.* Prentice Hall, Upper Saddle River 21. Oktober 2001, ISBN 978-0130676344, S. 36.

[2] http://www.scrumalliance.org

[3] http://www.scrum.org

[4] http://www.scrum-kompakt.de/einfuehrung-in-scrum/

[5] http://www.agile42.com/cms/articles/2009/6/16/Scrum-risks/

[6] http://media.agile42.com/content/Scrum_Cheat_Sheet_0909.pdf

Quellen und Bearbeiter der Artikel

Scrum *Quelle*: http://de.wikipedia.org/w/index.php?oldid=74767165 *Bearbeiter*: AV, Alexschwartz, Alfe, Amontillado, AndreasSchliep, AngeloSchneider, Astrobeamer, Avron, BKSlink, BSDev, Beek100, Ber, Bolondon, Borisgloger, Charly22, ChristophDemmer, DanielHepper, Das Ed, DasBee, Deino Wanthers, DirkDe, Dobberph, Fisfra, Fleshgrinder, Frank C. Müller, Fuinedhel, Gaussianer, Go4wiki, H005, HJH, HaLu, Habakuk, Haeber, Hhdw1, Holger Koschek, HorstHorst, Hutschi, Inkowik32, Is it now?, JMetzler, Jendryschik, Jergen, Jonathan Hornung, Joraal, Jpb24, Jpp, Jörny, Kaisersoft, Kaster, Klaus zinser, Krawi, Kriegaex, LKD, Lbirn, Lenny222, Ma-Lik, MadMike77, MarZilein, Mboehmer, Mcflashgordon, Michael Hüttermann, Mnh, Mschorer, Msommerlandt, Nameless23, Niemeyerstein, OecherAlemanne, Oemmler, Ollivander, Patchworker, Philipendula, Pilawa, Pittimann, Prettyprinter, Ralf5000, Renekaemmerer, RuedigerEckhard, Rufo, Saibo, Sebastian Wallroth, Sebastian.Dietrich, Seufert, Sinn, SixtyEight, Snipermatze, Sparti, Stachelfisch, Steevie, Sternenfaenger77, Sven.heyll, Teckmx5, Tettrino, TheDragon, Thomas 1609, Till.niermann, Tormod, Trublu, Trustable, Tschäfer, Tstagl, Tönjes, WalkerBoh, Wamito, Warhog, Zahnradzacken, 149 anonyme Bearbeitungen

Quellen, Lizenzen und Autoren der Bilder

Datei:Scrum process.svg *Quelle*: http://de.wikipedia.org/w/index.php?title=Datei:Scrum_process.svg *Lizenz*: GNU Free Documentation License *Bearbeiter*: User:Lakeworks

Datei:Scrum Parallelisiertes Pipelining.jpg *Quelle*: http://de.wikipedia.org/w/index.php?title=Datei:Scrum_Parallelisiertes_Pipelining.jpg *Lizenz*: unbekannt *Bearbeiter*: Dobberph, Suhadi Sadono

Lizenz

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- **A.** Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- **B.** List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- **C.** State on the Title page the name of the publisher of the Modified Version, as the publisher.
- **D.** Preserve all the copyright notices of the Document.
- **E.** Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- **F.** Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- **G.** Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- **H.** Include an unaltered copy of this License.
- **I.** Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
- **J.** Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- **K.** For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- **L.** Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- **M.** Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
- **N.** Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
- **O.** Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.
You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.
You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.
The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.
The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.
In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.
You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.
If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.
If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.
Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME.
>
> Permission is granted to copy, distribute and/or modify this document
>
> under the terms of the GNU Free Documentation License, Version 1.2
>
> or any later version published by the Free Software Foundation;
>
> with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.
>
> A copy of the license is included in the section entitled
>
> "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the
>
> Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.
If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Vereinbarung über Urheber- und Nutzungsrechte

1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Bachelorarbeit ScrumTable von Michael Gfeller, Silvan Gehrig, Patrick Boos unter der Betreuung von Prof. Dr. Markus Stolze geregelt.

2. Urheberrecht

Die Urheberrechte stehen der Studentin / dem Student zu.

3. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl von der Studentin / dem Student wie von der HSR nach Abschluss der Arbeit verwendet und weiter entwickelt werden

Rapperswil, den 15.06.10 .....
Michael Gfeller

Rapperswil, den 15.06.10 .....
Silvan Gehrig

Rapperswil, den 15.06.10 .....
Patrick Boos

Rapperswil, den.....
.....
Der Betreuer der Bachelorarbeit
Prof. Dr. Markus Stolze

Rapperswil, den.....
.....
Der Studiengangleiter Abt. Informatik

Erklärung über Selbständige Arbeit

Wir erklären hiermit,

- dass wir die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt habe, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.

Ort, Datum: 15.06.10

Name, Unterschrift: Michael Gfeller M Gfeller

Name, Unterschrift: Silvan Gehrig

Name, Unterschrift: Patrick Boos