**Naukanu Sailing School**

**Projektendbericht**

**Auftraggeber: Naukanu Sailing School**

Herr Prof. Dr. Dr. Neunteufel

**Auftragnehmer: Studs@Work AG**

Tobias Meyer (Projektleitung)

Benjamin Böcherer

Stefan Müller

Dominik Schumacher

# Dokumentenhistorie

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Datum | Autor(en) | Kommentar / Beschreibung |
| 0.1 | 07.02.2014 | Benjamin Böcherer, Stefan Müller, Dominik Schumacher, Tobias Meyer | Initiale Erstellung |
|  |  |  |  |
|  |  |  |  |

# Verantwortlichkeiten

|  |  |  |  |
| --- | --- | --- | --- |
| Kapitel | Seiten | Autor | Kommentar / Beschreibung |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Impressum

Dieses Werk und einzelne Teile daraus sind urheberrechtlich geschützt. Der Nachdruck sowie Verbreitung, auch auszugsweise, ist nur mit vorheriger schriftlicher Einwilligung der Autoren gestattet.

1. Auflage 07.2014

Herausgegeben von Studs@Work AG

© 2014 Studs@Work AG

[www.studsatwork.de](http://www.studsatwork.de)

Inhalt

[1 Dokumentenhistorie 2](#_Toc391832446)

[2 Verantwortlichkeiten 2](#_Toc391832447)

[3 Impressum 3](#_Toc391832448)

[4 Prolog [Dominik Schumacher] 8](#_Toc391832449)

[5 Die Ausgangssituation 9](#_Toc391832450)

[5.1 Der Auftraggeber 9](#_Toc391832451)

[5.2 Der Auftragnehmer 10](#_Toc391832452)

[6 IST – Analyse und SOLL-Zustand 12](#_Toc391832453)

[6.1 Definition 12](#_Toc391832454)

[6.2 Die Organisationsstruktur 12](#_Toc391832455)

[6.3 Die Geschäftsprozesse 13](#_Toc391832456)

[6.3.1 Anlage eines Teilnehmers 14](#_Toc391832457)

[6.3.2 Anlage eines Kurses 14](#_Toc391832458)

[6.3.3 Planen der Kurstermine 14](#_Toc391832459)

[6.3.4 Verwaltung des Materials 14](#_Toc391832460)

[6.3.5 Verwaltung der Kursleiter / freien Mitarbeiter 14](#_Toc391832461)

[6.3.6 Erstellen von Rechnungen / Mahnwesen 14](#_Toc391832462)

[6.4 Bestehende Probleme 14](#_Toc391832463)

[6.5 Die technische Ausstattung 15](#_Toc391832464)

[6.6 Zielsetzung (SOLL-Zustand) 17](#_Toc391832465)

[7 Projektmanagement 19](#_Toc391832466)

[7.1 Definition Projekt und Projektmanagement 19](#_Toc391832467)

[7.2 Nutzen des Projektmanagements 21](#_Toc391832468)

[7.2.1 Projektorganisation 22](#_Toc391832469)

[7.2.2 Projektplanung 26](#_Toc391832470)

[7.2.3 Projektstruktur 26](#_Toc391832471)

[7.2.4 Terminplan 27](#_Toc391832472)

[7.2.5 Projektüberwachung und -steuerung 28](#_Toc391832473)

[7.2.6 Projektrisiken 31](#_Toc391832474)

[7.2.7 Risikoanalyse 31](#_Toc391832475)

[7.3 Das V-Modell 32](#_Toc391832476)

[8 Verwendete Technologien 34](#_Toc391832477)

[8.1 Microsoft .NET-Framework 35](#_Toc391832478)

[8.2 Die Programmiersprache C# 36](#_Toc391832479)

[8.3 XAML 37](#_Toc391832480)

[8.4 Entity Framework 38](#_Toc391832481)

[8.5 Modern UI 38](#_Toc391832482)

[8.6 MVVM 38](#_Toc391832483)

[8.7 Microsoft SQL Server 40](#_Toc391832484)

[9 Softwareentwicklung 40](#_Toc391832485)

[9.1 Vorgehensmodell Scrum 41](#_Toc391832486)

[9.2 Prototyping 42](#_Toc391832487)

[9.3 Qualitätsmanagement 45](#_Toc391832488)

[9.4 Organisationswerkezeuge 47](#_Toc391832489)

[9.5 Datenbank 48](#_Toc391832490)

[9.6 UML 49](#_Toc391832491)

[9.7 Quellcodeverwaltung 49](#_Toc391832492)

[10 Die Prozesse der Software „Naukanu Sailing School Manager“ 50](#_Toc391832493)

[10.1 Die Softwarearchitektur 50](#_Toc391832494)

[10.2 Die graphische Oberfläche (GUI) 50](#_Toc391832495)

[10.3 Die Stammdatenverwaltung 50](#_Toc391832496)

[10.3.1 Kunden 50](#_Toc391832497)

[10.3.2 Kursleiter 50](#_Toc391832498)

[10.3.3 Material 50](#_Toc391832499)

[10.3.4 Boote 50](#_Toc391832500)

[10.3.5 Qualifikationen 50](#_Toc391832501)

[10.4 Die Kursverwaltung 50](#_Toc391832502)

[10.5 Die Materialverwaltung 50](#_Toc391832503)

[10.6 Die Rechnungsverwaltung 50](#_Toc391832504)

[10.7 Die Terminverwaltung 50](#_Toc391832505)

[11 Die Dokumentation 51](#_Toc391832506)

[11.1 Definition 51](#_Toc391832507)

[11.2 Merkmale einer Dokumentation 51](#_Toc391832508)

[11.3 Dokumentationstechniken 52](#_Toc391832509)

[11.4 Die technische Dokumentation 52](#_Toc391832510)

[11.5 Die Benutzerdokumentation 53](#_Toc391832511)

[12 Ausblick und zusätzliche Features 53](#_Toc391832512)

[13 Zusammenfassung 53](#_Toc391832513)

[14 Glossar 53](#_Toc391832514)

[15 Literaturverzeichnis 54](#_Toc391832515)

[16 Abbildungsverzeichnis 55](#_Toc391832516)

[17 Tabellenverzeichnis 56](#_Toc391832517)

[18 Abkürzungsverzeichnis 57](#_Toc391832518)

[19 Anhang 58](#_Toc391832519)

# Prolog [Dominik Schumacher]

Durch die Entwicklung der Mikroelektronik und der dadurch entstandenen Informations- und Kommunikationsbranchen, haben in den letzten Jahrzehnten immer mehr elektronische Geräte wie Computer, Mobiltelefone und ein Internetzugang in den Haushalten überall auf der Welt Einzug gehalten. Sie fördern nicht nur die private Kommunikation, unterstützen bei alltäglichen Dingen, wie z.B. Schreiben eines Briefes sondern ermöglichen uns auch eine komplett neue Freizeitgestaltung. Doch weit mehr als der Einzelne profitiert die weltweite Wirtschaft von den neuen Technologien. In den frühen 1960er und 1970er Jahren hielten die ersten Großrechner im Banken- und Logistiksektor Einzug, um dort die vorhandenen Prozesse zu unterstützen, erleichtern, erweitern und zu beschleunigen. Mit der fortschreitenden Entwicklung von IT-Systemen wurden Computer immer leistungsfähiger, kleiner und günstiger, sodass vor allem für immer mehr mittelständische Unternehmen elektronische Datenverarbeitung bezahlbar wurde. In diesem Bereich spielt die Entwicklung von Individualsoftware eine bedeutende Rolle, da für viele Unternehmen die vorhandene Standardsoftware Ihre Bedürfnisse nicht vollständig erfüllen konnte. Aus diesem Grund kontaktierte die Naukanu Sailing School die Studs@Work AG zur Entwicklung einer Anwendung für die ganzheitliche Verwaltung ihrer Segelschule.

Dieser Projektbericht beschreibt im Rahmen einer durchgeführten IST-Analyse die vorhandenen Geschäftsprozesse der Naukanu Sailing School. Des Weiteren wird detailliert auf die theoretischen Grundlagen sowie die im Rahmen dieses Projektes angewandten Methoden im Bereich der Softwareentwicklung eingegangen. Hierbei werden nicht nur die technischen Details der Software erläutert, sondern auch die damit verbundenen Prozesse und Vorgehensweisen, welche zu dem fertigen Produkt „Naukanu Sailing School Manager“ führten.

# Die Ausgangssituation

## Der Auftraggeber

Die Firma Naukanu Sailing School am Gardasee ist eine Segel- und Surfschule in Norditalien. Sie wurde 1928 von Felippe Santane gegründet. Im Sommer 2001 erwarb Stefan Marx die Segelschule und übernahm die Verantwortung für diese traditionsreiche Institution, in der schon Generationen von Seglern ausgebildet wurden. Die dort angebotenen Kurse können von Gruppen und Einzelpersonen gebucht werden. Die dafür eingesetzten Kursleiter sind freie Mitarbeiter, die saisonweise beschäftigt und kursweise bezahlt werden. Zum Nachweis entsprechender Kenntnisse der Teilnehmer gibt es eine Vielzahl unterschiedlicher Segel- und Surfscheine. Wer einen Segel- bzw. Surfschein machen möchte, hat bei der Naukanu Sailing School die Möglichkeiten diese unter kompetenter Anleitung zu absolvieren. Die Teilnehmer lernen das seemännische Handwerk vom Einsteiger bis zur Prüfungsreife für den Binnenführerschein.

## Der Auftragnehmer

|  |  |
| --- | --- |
| **Firmenname** | Studs@Work AG |
| **Gesellschaftsform** | Aktiengesellschaft |
| **Gegründet** | 1999 |
| **Gründer und Geschäftsführer** | Max Mustermann |
| **Mitarbeiter** | Derzeit 49 |
| **Adresse** | Max-von-Laue-Straße 9  60439 Frankfurt am Main |

Die Studs@Work AG, mit Sitz in Frankfurt, wurde 1999 gegründet und beschäftigt 49 Vollzeitbeschäftigte (davon 46 Entwickler) und 3 Teilzeitbeschäftigte. Der Fokus der Studs@Work AG liegt auf der Individualsoftwareentwicklung im Enterprise-Bereich, dem Applikationsbetrieb (Betrieb und Support von Enterprise-Applikationen) und der IT-Beratung. Die Enterprise-Applikationen werden in Java Enterprise und Microsoft .NET entwickelt. Zum Kundenkreis gehören u.a. Automobilhersteller, Energielieferanten, Landmaschinenhersteller, Banken, Heiztechnikhersteller sowie diverse IT-Dienstleister und –Systemhäuser. Unser junges Team entwickelt Lösungen basierend auf fundierten betriebswirtschaftlichen Branchen-Know-How in Verbindung mit erstklassiger technischer Expertise.

Die Studs@Work GmbH hat es sich zur Aufgabe gemacht, als verlässlicher und kompetenter Partner seinen Kunden zur Verfügung zu stehen, denn nur so kann eine fundierte und langandauernde Partnerschaft aufgebaut werden. Dabei verwenden wir modernste Technologien und sind einer der führenden Entwickler von individual Software für Dienstleister, mittelständische Unternehmen und Konzerne in allen Branchen.

Im Jahre 2010 wurde ein weiterer wichtiger Meilenstein in der Firmengeschichte erfolgreich abgeschlossen. Nach 24 Monaten Bauzeit entstand ein moderner Bürokomplex mit Arbeitsplätzen für 100 Mitarbeiterinnen und Mitarbeiter und einem großzügigen Konferenzbereich mit hervorragender Infrastruktur für Schulungen, Kundenbesprechungen und Tagungen sowie einer Kantine.

Der Einsatz moderner Arbeitsgeräte ist für die Studs@Work AG selbstverständlich. So setzen unsere Entwickler modernste Notebooks mit Microsoft Windows 7 - 64bit ein, die mit einem Intel Core i5 und mindestens 8GB Arbeitsspeicher ausgestattet sind. Die Notebooks werden spätestens alle 2 Jahre erneuert. Zudem nutzt die Studs@Work AG VMWare-Server (vSphere ESXi) für die Bereitstellung der Entwicklungsumgebungen. Im Java-Umfeld wird Eclipse und im .NET-Umfeld Visual Studio 2012 als Entwicklungsumgebung eingesetzt. Als CI-Server wird Jenkins verwendet, als (Sub)Version Control wird TortoiseSVN genutzt und als Buildtools werden Maven sowie Ant eingesetzt. Die Datenbanksysteme von Oracle, Microsoft SQL, mySQL und PostgreSQL sind bei verschiedenen Kundenprojekten im Einsatz. Für den Einsatz von Qualitätssicherungs-Maßnahmen wird EMMA eingesetzt. Die Projektorganisation wird agil in der Regel mit dem Vorgehensmodell SCRUM oder Adaptionen daraus umgesetzt.

# IST – Analyse und SOLL-Zustand

## Definition

Die IST-Analyse ist Teil des Projektmanagements und stellt die Phase eines Vorgehensmodells dar, in der die objektive, möglichst neutrale und wertungsfreie Erfassung eines bestehenden Problems bzw. des aktuellen (IST)-Zustandes stattfindet. Zur Datenerhebung kommen die Methoden der Primärerhebung zum Einsatz, welche aus Befragungen, Selbstaufschreibungen und Beobachtungen zum Einsatz. Zusätzlich werden im Zuge der Sekundärerhebung mittels der Dokumentenanalyse, d.h. dem Betrachten und Auswerten von Akten, Rechnungen, Berichten und Schreiben weitere Informationen gesammelt.

## Die Organisationsstruktur

Die Naukanu Sailing School beschäftigt aktuell sieben festangestellte Mitarbeiter und weißt folgende Unternehmensstruktur auf:



* Der Marketing-Leiter ist für alle Marketing-Aufgaben sowie für den Internetauftritt verantwortlich
* Eine Sekretärin, die sämtliche Bürotätigkeiten durchführt
* Zusammen mit einem Buchhalter übernimmt der Geschäftsführer die Aufgaben der Personal- und Finanzabteilung
* Der Schulungsleiter sowie dessen Stellvertreter sind für die Auswahl, Einarbeitung sowie Koordination der Kursleiter zuständig. Des Weiteren haben sie zusammen für die Einsatzbereitschaft des Materials Sorge zu tragen.
* Zwei Materialwarte, welche für die Wartung der Boote, der Surfbretter sowie deren Besegelung verantwortlich sind.

## Die Geschäftsprozesse

Die Segelschule besitzt keine zentrale Anwendung, mittels der die Kurstermine und den damit verbundenen Tätigkeiten verwaltet werden. Sämtliche Prozesse und die dazugehörigen Informationen sind in verschiedenen Excel Sheets festgehalten.

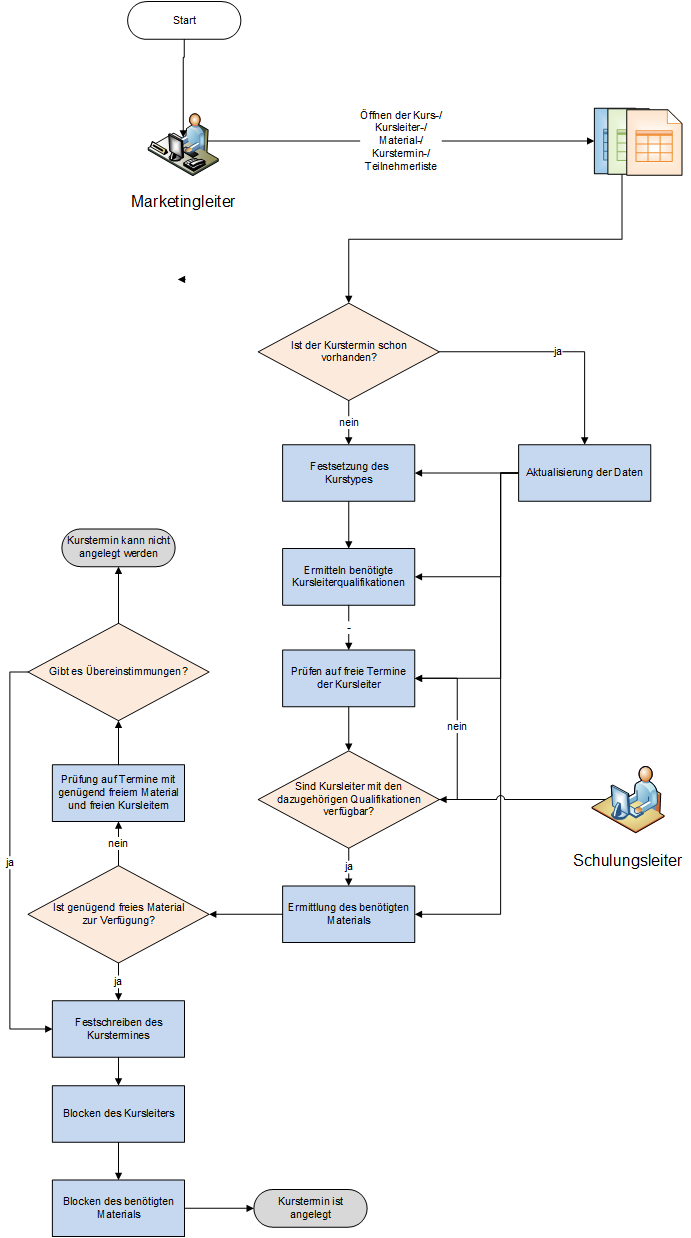
|  |  |
| --- | --- |
| **Name** | **Prozess-/Informationen** |
| Kunden.xslx | Verwaltung der Kunden- / Teilnehmerstammdaten (u.a. Adresse und Kontodaten) |
| Kurs.xlsx | Anlage aller Kurse, die angeboten werden |
| Kurstermine.xlsx | Eintragen der Kurstermine / Reservierung der Kursleiter sowie Material |
| Material.xlsx | Verwaltung der Materialeinzelteile, sowie kompletter Fahrzeuge (Boote, Surfbretter, etc.) |
| Kursleiter.xlsx | Auflistung sämtlicher freier Mitarbeiter / Kursleiter (Stammdaten wie Adresse / Kontodaten und Verfügbarkeitszeiten) |
| Rechnung.xlsx | Verwaltung gezahlter sowie zu stellende Rechnungen, inkl. Mahnungen |

### Anlage eines Teilnehmers

### Anlage eines Kurses



### Planen der Kurstermine



### Verwaltung des Materials



### Verwaltung der Kursleiter / freien Mitarbeiter



### Erstellen von Rechnungen / Mahnwesen



## Bestehende Probleme

Die dezentralisierte Datenhaltung, aufgeteilt in mehreren Dateien, hat mehrere Nachteile, die oft im Geschäftsbetrieb zu Problemen / Verzögerungen führen.

* Bankdaten (Stammdaten) der Kunden und der Kursleiter werden jeweils in getrennten Dateien gepflegt. Zusätzlich sind die Formate für beide Datensätze unterschiedlich.
* Rechnungen werden manuell erstellt. Die dazu nötigen Rechnungspositionen, d.h. absolvierte Kurse eines Teilnehmers müssen händisch eingepflegt werden. Diese Tätigkeit ist sehr zeitaufwendig und führt des Öfteren zu fehlerhaften Rechnungen.
* Eine Protokollierung über durchgeführte Reparaturmaßnahmen findet nicht statt. Defektes und instandgesetztes Material wird lediglich räumlich getrennt. Eine Nachvollziehbarkeit sowie Aussortierung von Material, welches seine maximale Verwendungsdauer überschritten hat, ist deshalb nicht gegeben.
* Es findet keine durchgängige Referenzierung von Daten zwischen den einzelnen Excel Sheets statt. Somit treten häufig Mehrfacheinträge auf und eine damit verbundene Fehlersuche sowie Beseitigung ist sehr schwierig.
* Das Finden von Übereinstimmungen hinsichtlich freien Kursleiter(zeiten) und dem zu Verfügung stehenden Material ist sehr zeitraubend und fehlerbehaftet. Aufgrund dessen mussten schon Kurse mehrfach abgesagt bzw. verschoben werden.

## Die technische Ausstattung

In dem zweistöckigen Hauptgebäude der Segelschule befinden sich neben den Schulungsräumen, die Büros der Geschäftsführung sowie die der Verwaltung. Das Unternehmen besitzt folgende EDV-Ausstattung.

* Jeder festangestellte Mitarbeiter sowie der Geschäftsführer besitzt einen eigenen Computerarbeitsplatz. Somit sind aktuell acht PCs auf Basis von Windows 7 in Verwendung.
* Microsoft Office 2013 dient als einzige Software zur Unterstützung des Geschäftsbetriebes.
* Ein Kyocera C85020 Multifunktionsdrucker ist als zentrale Scan-, Druck- sowie Faxstation im Einsatz
* WLAN-Empfang (Arbeit- und separater Gastzugang) im gesamten Gebäude
* Eine FritzBox 7390 übernimmt die Internet-Einwahl. Eine integrierte Firewall sorgt für den entsprechenden Schutz des Netzwerkes.
* Als zentraler Dateispeicherort / Fileserver (NAS) dient eine QNAP TS 459 PRO II mit 2 Terabyte Nutzkapazität.
* Für die Datensicherung wird eine externe USB-Festplatte verwendet, auf welche jede Nacht eine Synchronisation von dem NAS durchgeführt wird.



## Zielsetzung (SOLL-Zustand)

Die Naukanu Sailing School besitzt zum Zeitpunkt der IST-Analyse eine moderne EDV – Umgebung. Lediglich die Geschäftsprozesse sind wenig automatisiert bzw. mit vielen händischen Nacharbeiten verbunden. Um weiterhin wettbewerbsfähig zu bleiben und um das Management seines Unternehmens zu optimieren, benötigt der Geschäftsführer eine Anwendung für die komplette Kursverwaltung. Diese Applikation soll folgende Aufgaben bewältigen:

* Verwaltung der freien Mitarbeiter (Vertragsmanagement der freien Mitarbeiter, Bezahlung der freien Mitarbeiter, Aufnahme und Verwaltung der persönlichen und beschäftigungsrelevanten Daten, Einteilung für Kurse)
* Verwaltung der Kurse (Termine, Zuordnung zu Kunden, Gebühren, Bereitstellung des Materials, Kursleiter)
* Materialverwaltung (Einsatzbereitschaft, Aussonderung, Reparaturverwaltung, Neubeschaffung, Daten zu Material wie Merkmale, Marke, Kaufpreis, Reparaturkosten)
* Kundenverwaltung (Daten, gebuchte Kurse, Rechnungserstellung, Zahlungsverfolgung, Mahnwesen)

Mit der Einführung der neuen Software werden nachfolgende Grundsätze und Ziele verfolgt:

* Aktuellster technischer Stand (Datenbank, Softwareentwicklung)
* Zentrale Datenverwaltung
* Vermeidung von Datenredundanzen
* Benutzerfreundlichkeit (intuitive Bedienkonzepte und Selbsterklärbarkeit der Graphischen Oberfläche)
* Hoher Abdeckungsgrad der Anforderungen
* Flexible Erweiterbarkeit der Software (Hinzufügen neuer Programmfunktionen)
* Sicherheit und Integrität (kein unberechtigter Zugriff sowie versehentliches Löschen von Daten)
* Vereinfachung, Optimierung und Verkürzung der bestehenden Geschäftsprozesse

# Projektmanagement

## Definition Projekt und Projektmanagement

Ein **Projekt** ist ein “Vorhaben, das im Wesentlichen durch Einmaligkeit der Bedingungen in ihrer Gesamtheit gekennzeichnet ist, z.B. Zielvorgabe, zeitliche, finanzielle, personelle und andere Begrenzungen, Abgrenzung gegenüber anderen Vorhaben, projektspezifische Organisation“. (DIN 69901)

„**Projektmanagement** ist ein systematischer Prozess zur Führung komplexer Vorhaben. Es umfasst die Organisation, Planung, Steuerung und Überwachung aller Aufgaben und Ressourcen, die notwendig sind, um die Projektziele zu erreichen.“ (PM-Handbuch.com, 2014)

Um Projekte erfolgsversprechend abzuschließen, bedarf es einer konsequenten Steuerung und Kontrolle. Die Steuerung und Kontrolle wird im Allgemeinen als Projektmanagement bezeichnet. Durch folgende Punkte zeichnet sich Projektmanagement [[1]](#footnote-2)aus:

Abbildung : Aufteilung Projektmanagement

* Fähigkeiten

… entsprechend qualifizierte Projektleiter und –mitarbeiter

* Strukturen

… passende Einzel- und Multiprojektmanagementprozesse

* Kultur

… eine gemeinsam entwickelte und gelebte Projektmanagementkultur

* IT-Tools

… Software-Anwendungen, die Mitarbeiter und Prozesse unterstützen

## Nutzen des Projektmanagements

Ein korrektes und systematisch durchgeführtes Projektmanagement soll zum Gelingen eines Projektes beitragen. Es schafft Strukturen, die Prozesse bei der Erreichung von Zielen unterstützt. Hinsichtlich folgender Messkriterien führt ein erfolgreiches Projektmanagement zu positiven Effekten:

* **Effektivität**

Teamarbeit und die Einbeziehung von Betroffenen schafft wirkungsvolle und dauerhafte Problemlösungen – “Do the right things.”

* **Effizienz**

Die Gesamtkosten werden durch planvolleres Vorgehen und beschleunigter Prozessabläufe gesenkt. – “Do the things right”

* **Personalentwicklung**

Projektmanagement fördert die Kompetenz von Führungspersonen sowie der einzelnen Teammitglieder

* **Wissensmanagement**

Eine ordentlich geführte Dokumentation erweitert das bestehende Wissen und führt in nachfolgenden Projekten zu kürzeren Laufzeiten und höherer Qualität. Gleichzeitig verbessert dies die gesamtheitliche Transparenz des Projektes.

* **Kontrollierbarkeit**

Ein funktionierendes Projektcontrolling führt zu einer gezielten Projektsteuerung und lässt dem Team die Ziele nicht aus den Augen verlieren.

* **Plantreue**

Eine professionelle Planung und Steuerung fördert die Einhaltung von festgesetzten Abgabeterminen. (PM-Handbuch.com, 2014)

### Projektorganisation

Nach der Annahme des Angebotes durch die Firma Naukanu Sailing School wurden folgende Verantwortlichkeiten festgelegt:

* Auftraggeber: Naukanu Sailing School, vertreten durch Herrn Prof. Dr. Dr. Neunteufel
* Auftragnehmer: Studs@Work AG
* Projektleiter: Herr Tobias Meyer
* Projektteam: Herr Benjamin Böcherer, Herr Stefan Müller, Herr Dominik Schumacher

Der Projektleiter wurde am Anfang durch das Projektteam bestimmt.

Abbildung : Projektorganisation

### Projektplanung

Das potenzielle Risiko in einem Projekt versucht man durch eine detaillierte Projektplanung zu minimieren bzw. auszuschließen. Dabei ergeben sich aus der Planung Soll-Vorgaben für die einzelnen Aufgabenbereiche. An der Projektplanung war das gesamte Team beteiligt.

Die Folge der Aufteilung waren Aufgabenblöcke mit unterschiedlichen Schwerpunkten, z.B. die Entwicklung, die Dokumentation und der Test der Software. Eine Aufteilung der Gebiete auf die einzelnen Projektmitarbeiter wurde vom Projektleiter nach Abstimmung mit dem Team vorgenommen.

### Projektstruktur

Für den Begriff Projektstruktur gibt es verschiedene Definitionen, im Allgemeinen kann man den Begriff Projektstruktur als Menge aller Elemente und ihrer gegenseitigen Beziehungen in einem Projekt beschreiben.

Definition nach DIN 69901-5

Die DIN 69901-5 engt den Begriff der **Projektstruktur** bereits erheblich ein, indem sie nur die "wesentlichen Beziehungen" zu ihr zählt. Weiterhin nennt die DIN als Elemente des Projekts explizit lediglich "Teilprojekte, Arbeitspakete, Vorgänge". Die DIN 69901-5 geht somit von einer aktivitäts-orientierten Gliederung des Projekts aus. Andere Elemente eines Projekts, wie z.B. Produkte, Ressourcen, Kosten usw. zählt die DIN 69901-5 bei strikter Auslegung somit nicht zur **Projektstruktur**

Bei der Erstellung des Angebotes wurden die Hauptarbeitspakete festgelegt. Diese und die daraus resultierten einzelnen Arbeitspakete wurden in einem Projektstrukturplan (PSP) visualisiert. Der PSP dient zur Gliederung des Projektes in übersichtliche Einzelaufgaben. Der Projektstrukturplan wird auch als Work Breakdown Structure (WBS) bezeichnet. Dieser wurde an das Angebot vom 01.04.2014 angehängt.

### Terminplan

Der Terminplan gibt auf oberster PSP-Ebene eine grobe Übersicht über den möglichen Projektverlauf. Viele Aktivitäten stehen in engem Zusammenhang zueinander, andere können parallel abgearbeitet werden.

Es gab folgende fixe Meilensteine:

* 01.04.2014 Abgabe des Angebotes
* 24.05.2014 Zwischenpräsentation
* 20.07.2014 Endpräsentation
* 01.08.2014 Abgabe des Endberichtes und der Dokumentation

Anschließend wurde eine Terminübersicht erstellt, die die Meilensteine berücksichtigt und alle vorher definierten Arbeitspakete beinhaltet.



Abbildung : Terminplanung

### Projektüberwachung und -steuerung

Projektüberwachung und -steuerung ist das Überwachen des Projektfortschritts anhand von Soll/Ist-Vergleichen. Wenn Probleme auftreten müssen geeignete Korrekturmaßnahmen veranlasst werden.

* Die Planung ist iterativ
* Die Planung muss ebenso gewissenhaft angepasst werden, wie sie erstellt wurde

Abbildung : Projektüberwachung/-steuerung

**Projektüberwachung – Aufgaben und Termine**

Der Projektleiter prüft, ob die Arbeitspakete gemäß den Arbeitspaketbeschreibungen und zum vorgesehenen Termin ausgeführt wurden. Er stellt fest, ob die Meilensteine eingehalten werden können und ob das Projektteam die nächsten Aufgaben wie geplant in Angriff nehmen kann.

**Projektüberwachung – Kosten**

Der Projektleiter vergleicht die geplanten Kosten mit den effektiven Aufgaben. Bei Abweichungen ergreift er Maßnahmen, damit die Gesamtkosten des Projektes nicht oder nur minimal überschritten werden. Er muss Kostenabweichungen genau untersuchen, da diese unterschiedliche Ursachen haben können.

**Projektüberwachung – Ressourcen**

Der Projektleiter vergleicht den rapportierten Arbeitsaufwand mit den geplanten Arbeitstagen pro Projektmitarbeiter. Aufgrund des nachgeführten Balkenplanes erstellt er eine Prognose für die zukünftigen Arbeitsmonate. Auf diese Weise kennen die Mitarbeiter und ihre Vorgesetzten die effektiv bevorstehende Belastung durch die Projektarbeit.

**Projektüberwachung – Qualität**

Der Projektleiter prüft, ob die im Qualitätsplan definierten Prüfungen und Reviews sach- und termingerecht durchgeführt wurden. Damit kontrolliert er, ob die definierten Vorgaben angewendet werden und ob die Projektergebnisse die geforderten Qualitätsstandards erfüllen. Fehler oder Qualitätsmängel sind zu beheben und die Vorgaben sind so anzupassen, dass die Abweichungen nicht wieder vorkommen. Die Ergebnisse der Qualitätsprüfung und die initialisierten Maßnahmen sind zu dokumentieren.

**Projektüberwachung – Projektfortschritt kommunizieren**

Ziel ist, gemäß der Definition im Kommunikationsplan alle Beteiligten regelmäßig über den Fortschritt des Projektes zu informieren. Es empfiehlt sich, dass der Projektleiter das Ergebnis der wöchentlichen Meetings mit den Projektmitarbeitern in einem Kurzprotokoll festhält und zusätzlich eine so genannte Liste der offenen Punkte (LOP) führt. Sie enthält alles, was während dem Meeting nicht geklärt werden konnte, sowie Angaben darüber, welche Aufgaben bis zu einem bestimmten Zeitpunkt erledigt werden.

Der monatliche Projektstatusbericht an die Auftraggeberin muss den aktuellen Projektstand wahrheitsgetreu wiedergeben. Unstimmigkeiten und Abweichungen zum Plan sind offen zu kommunizieren. Nur bei einer ehrlichen Berichterstattung wird die Auftraggeberin den Projektleiter in der Problemlösungsfindung auch sachgerecht unterstützen. (Funke, 2014)

Zu überwachende Parameter:

Termine

Kosten

Leistung / Qualität

* Leistung / Qualität
* Termine
* Kosten

Abbildung : Projektziele

### Projektrisiken

Aufgrund des Umfanges des Projektes, können verschiedene Projektrisiken auftreten. Diese lassen sich in zwei große Kategorien unterteilen:

* Ablaufplanung verschiebt sich aufgrund
  + Späte Zusage seitens der Naukanu Sailing School
  + Projektmitarbeiter fallen wegen Krankheit bzw. Urlaub aus
* Softwareentwicklung
  + Einzelne Bereiche gestalten sich komplexer als vorher erwartet
  + Aufgrund verschiedener Abhängigkeiten können nicht alle Bereiche realisiert werden

### Risikoanalyse

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Risikobezeichnung | Wirkungs-bereich (K,Z,Q) | Wahr-scheinlichkeit | Priorität | Maßnahmen |
| Späte Zusage | Z | 7% | Niedrig | Nachfragen bei Naukanu Sailing School, bis wann die Zusage erwartet werden kann |
| Krankheit/Urlaub | Z,Q | 10% | Mittel |  |
| Komplexe Bereiche | Q,K | 15% | mittel | Funktionen einfacher gestalten |
| Abhängigkeiten | Z,Q | 20% | Hoch | Implementierung der notwendigen Funktionen ggf. Terminverschiebung |

Tabelle : Risikoanalyse

Das Arbeitsblatt zur Risikoanalyse enthält für jedes Risiko die folgenden Informationen:

* **Risikobezeichnung:** Name oder kurze Beschreibung des Risikos
* **Wirkungsbereich:** Beurteilt, wo das Risiko das Projekt betrifft und zwar in Bezug auf Kosten (K), Zeitplan (Z) oder Qualität (Q) des Arbeitsergebnisses
* **Wahrscheinlichkeit:** Geschätzte Wahrscheinlichkeit, mit der ein Risiko eintrifft
* **Priorität:** Beurteilung der Risiken, die schwerpunktmäßig zu beachten sind. Eingeteilt in die Kategorien hoch, mittel, niedrig
* **Maßnahmen:** Pläne zur Minderung der Wahrscheinlichkeit, dass das Risiko eintrifft

## Das V-Modell

Ein Vorgehensmodel in der Softwareentwicklung ist das V-Modell. Bei diesem Modell wird der Entwicklungsprozess in Phasen organisiert. Das V-Modell erläutert nicht nur die Entwicklungsphase sondern auch Ablauf zur Qualitätssicherung. Es basiert auf dem Wasserfallmodell und wurde von Barry Boehm (Grechenig & Bernhart, 2010) entwickelt.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| System-  anforderungs-  analyse |  |  | | | | | |  | Abnahme  und Nutzung |
| System-Architektur | |  |  | | | |  | Software-integration | |
| Systementwurf | | |  |  | |  | Integrationstests | | |
| Software-Architektur | | | |  |  | Unit-Tests | | | |
| Software-Entwurf | | | | | | | | | |

Abbildung : V-Modell

Das V-Modell verfolgt folgende Ziele und Maßnahmen:

* **Minimierung der Projektrisiken**

Durch standardisierte Vorgehensweise sowie die Beschreibung der zugehörigen Resultate und der verantwortlichen Rollen erhöht das V-Modell die Transparenz und verbessert die Planbarkeit von Projekten. Abweichungen von der Planung und Risiken werden bereits frühzeitig erkannt.

* **Verbesserung und Gewährleistung der Qualität**

Damit die Resultate vollständig und in der gewünschten Qualität erreicht werden, ist ein standardisiertes Vorgehensmodell nötig. Festgelegte Zwischenergebnisse können frühzeitig geprüft werden.

* **Eindämmung der Gesamtkosten über den ganzen Projekt- und Systemlebenszyklus**

Um den Aufwand für die Entwicklung, den Betrieb und die Wartung eines Systems kalkulieren, abschätzen und steuern zu können muss ein standardisiertes Vorgehensmodell angewendet werden. Die daraus resultierenden Ergebnisse sind einheitlich und verständlicher nachvollziehbar. Diese Maßnahme verringert die Abhängigkeit zwischen Auftraggeber und Auftragnehmer und führt zu weniger Aufwand in anschließenden Projekten.

* **Verbesserung der Kommunikation zwischen allen Beteiligten**

Um Reibungsverluste zwischen Benutzer, Auftraggeber und Auftragnehmer und Entwicklern zu reduzieren, bedarf es einer standardisierten und einheitlichen Beschreibung aller relevanten Bestandteile und Begriffe.

Mir gefällt der Abschnitt sehr gut, allerdings müssen wir das ein bisschen auf SCRUM anpassen. Das Team ist dort selbständiger. Bitte den SCRUM-Abschnitt dazu lesen.

Das V-Modell muss generischer formuliert werden. SCRUM ist dann die konkrete Umsetzung dieses Modells

# Verwendete Technologien [Benjamin Böcherer]

Der Kern der Anwendung „Naukanu Sailing School Manager“ basiert auf Microsoft Technologien im .NET-Umfeld. Die Daten werden in dem Datenbankserver Microsoft SQL Server persistiert. Die graphische Benutzeroberfläche wurde ebenfalls mit .NET-Technologien umgesetzt. Zusätzlich wird hier noch das UI-Framework „Modern UI“ eingesetzt, um die Oberfläche grafisch an aktuelle Softwareprojekte anzupassen. Die eingesetzten Technologien stehen kostenlos zur Verfügung. Davon profitiert auch die Segelschule als Kunde, die nur die reinen Entwicklungskosten der Anwendung und keine weiteren Lizenzkosten zahlt.

Die relevanten Technologien werden in dem folgenden Kapital genauer erläutert.

## Microsoft .NET-Framework

Die Anwendung „Naukanu Sailing School Manager“ benötigt das Microsoft .NET-Framework ab der Version 4.5. .NET bezeichnet eine von Microsoft entwickelte Software-Plattform zur Entwicklung und Ausführung von Anwendungsprogrammen. Es besteht aus einer Laufzeitumgebung (Common Language Runtime), in der die Programme ausgeführt werden, sowie einer Sammlung von Klassenbibliotheken, Programmierschnittstellen und Dienstprogrammen. .NET ist auf verschiedenen Plattformen verfügbar und unterstützt die Verwendung einer Vielzahl von Programmiersprachen. Die .NET-Programme werden zunächst in eine Zwischensprache (Common Intermediate Language) übersetzt, bevor sie von der Laufzeitumgebung ausgeführt werden. Diese Übersetzung geschieht in der Regel mithilfe eines sogenannten Just-In-Time-Compilers. Mit .NET löste Microsoft zuvor eingesetzte Softwareentwicklungskonzepte wie das Component Object Model (COM) ab.

Die folgende Tabelle gibt eine Übersicht der verschiedenen .NET-Versionen:

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Versionsnummer** | **Datum** | **enthalten in** |
| 1.0 | 1.0.3705.0 | 5. Januar 2002 | n/a |
| 1.1 | 1.1.4322.573 | 1. April 2003 | Windows Server 2003 |
| 2.0 | 2.0.50727.42 | 7. November 2005 | Windows Server 2003 R2 |
| 3.0 | 3.0.4506.30 | 6. November 2006 | Windows Vista, Windows Server 2008 |
| 3.5 | 3.5.21022.8 | 9. November 2007 | Windows Server 2008 R2 |
| 3.5 SP 1 | 3.5.30729.1 | 11. August 2008 | Windows 7 mit SP1 |
| 4.0 | 4.0.30319 | 12. April 2010 | n/a |
| 4.5 | 4.5.50501 | 15. August 2012 | Windows 8, Windows Server 2012 |
| 4.5.1 | 4.5.50938 | 12. Oktober 2013 | Windows 8.1, Windows Server 2012 R2 |

Die folgende Grafik gibt einen Überblick die verschiedenen Bausteine, die kontinuierlich in das .NET-Framework eingebaut wurden:



## Die Programmiersprache C#

C# (lies englisch c sharp) ist eine von Microsoft im Rahmen seiner .NET-Strategie entwickelte Programmiersprache und ist bei ECMA und ISO als Standard registriert.

C# greift Konzepte der Programmiersprachen Java, C++, Haskell, C sowie Delphi auf und zählt zu den objektorientierten Programmiersprachen. Es unterstützt sowohl die Entwicklung von sprachunabhängigen .NET-Komponenten als auch COM-Komponenten für den Gebrauch mit Win32-Anwendungsprogrammen.

Wir haben uns für diese Programmiersprache entschieden, da wir eine Windows-Desktop-Anwendung entwickeln wollte. Aufgrund von Vorkenntnissen der Projektteilnehmer im .NET-Umfeld (C#, VB .NET) bat sich diese Technologie hervorragend an.

## XAML

XAML ist die Abkürzung für „Extensible Application Markup Language“ und ist eine allgemeine Beschreibungssprache für die Oberflächengestaltung von Anwendungen die von Microsoft entwickelt wurde. Bei der Beschreibungssprache handelt es sich um eine XML-basierte Sprache. Es lassen sich damit grafische Elemente, Benutzeroberflächen, Verhaltensweisen, Animationen, Transformationen, Darstellung von Farbverläufen, Abspielen von Mediadateien und vieles mehr definieren. Bei einer üblichen Verwendung werden XAML-Dateien von optisch orientierten Design- und Entwicklerwerkzeugen wie Microsoft Expression Blend, Microsoft Visual Studio oder XAML Pad generiert.

Mit XAML kann man unter Verwendung jeder beliebigen anderen Programmierschnittstelle Anwendungen entwickeln, wobei XAML eine eigenständige Sprache darstellt. Der Hauptaspekt dieser Technologie ist die verringerte Komplexität, die Programme haben müssen, um XAML zu verarbeiten, weil es sich im Grunde um einfaches XML handelt. XAML-Dateien sind hierarchisch strukturiert. Ein oder mehrere Elemente können, abhängig von ihrer Ordnung, das Layout und Verhalten der Oberfläche beeinflussen. Jedes Element besitzt nur ein Elternelement. Jedes Element kann eine unbegrenzte Anzahl von Kindelementen besitzen, nur bei einigen wenigen ist die Anzahl eingeschränkt. In allen XAML-Anwendungen ist das Wurzelobjekt typischerweise ein Panel (oder eines seiner sechs Unterarten), das sich um Positionierung und Rendern jeglichen Inhaltes kümmert.

Eigenschaften und Einstellungen z. B. einer Schaltfläche werden wie bei XML bzw. HTML im Tag als Attribute aufgeführt. Wie jede XML-Datei besitzt XAML ein Wurzel-Tag. Bei XAML nimmt man ein Vaterobjekt (z. B. ein <Window>), in das man weitere Kinder (also Elemente) einfügen kann.

## Entity Framework

Das ADO.NET Entity Framework (kurz: ADO.NET EF) ist ein ORM-Framework von Microsoft, welches auf ADO.NET basiert. Objektrelationale Abbildung (englisch object-relational mapping, ORM) ist eine Technik der Softwareentwicklung, mit der ein in einer objektorientierten Programmiersprache geschriebenes Anwendungsprogramm seine Objekte in einer relationalen Datenbank ablegen kann. Dem Programm erscheint die Datenbank dann als objektorientierte Datenbank, was die Programmierung erleichtert. Implementiert wird diese Technik normalerweise mit Klassenbibliotheken, wie beispielsweise ADO.NET Entity Framework, Hibernate für die Programmiersprache Java oder SQLAlchemy für Python.

Im einfachsten Fall werden Klassen auf Tabellen abgebildet, jedes Objekt entspricht einer Tabellenzeile und für jedes Attribut wird eine Tabellenspalte reserviert. Die Identität eines Objekts entspricht dem Primärschlüssel der Tabelle. Hat ein Objekt eine Referenz auf ein anderes Objekt, so kann diese mit einer Fremdschlüssel-Primärschlüssel-Beziehung in der Datenbank dargestellt werden.

Die „Naukanu Sailing School Manager“-Anwendung setzt das Entity Framework in der Version 6.0.0 ein.

## Modern UI

Für die grafische Oberfläche haben wir uns für ein an Windows 8 angelehntes Aussehen entschieden. Um nicht alle optischen Features neu entwickeln zu müssen, wurde das UI-Framework „Modern UI“ eingesetzt. Damit lassen sich z.B. die aus Windows 8 bekannten „Circle-Buttons“ oder „Live-Tiles“ leichter einbinden. Das Framework steht kostenlos zur Verfügung.

## MVVM

Model View ViewModel (MVVM) ist eine Variante des Model View Controller-Musters (MVC) zur Trennung von Darstellung und Logik der Benutzerschnittstelle (UI). Es zielt auf moderne UI-Plattformen wie Windows Presentation Foundation (WPF), Silverlight und HTML5 ab. MVVM sieht eine Rollentrennung von UI-Designern und Entwicklern vor, wodurch Anwendungsschichten von verschiedenen Arbeitsgruppen entwickelt werden können. Designer können einen Fokus auf User Experience setzen und Entwickler die UI- und Geschäftslogik schreiben.

Das MVVM wurde 2005 von Microsoft MVP John Gossman veröffentlicht. Es ist eine Spezialisierung des Presentation Model von Martin Fowler und ist mit diesem insofern identisch, als beide Muster Zustand und Verhalten der View in ein separates UI-Model (Presentation bzw. View Model) verschieben. Das Presentation Model ermöglicht allerdings das Erzeugen einer View unabhängig von der UI-Plattform, wohingegen das MVVM ursprünglich auf UIs mittels WPF abzielt. Es findet allerdings inzwischen auch in anderen Bereichen Anwendung, ähnlich wie bei MVC.

Das MVVM nutzt die funktionale Trennung des MVC und Datenbindung, um eine lose Kopplung zu erreichen. Es beinhaltet drei Komponenten, wobei Model und View denen des klassischen MVC ähneln:

* **Model**: Datenzugriffsschicht für die Inhalte, die dem Benutzer angezeigt und von ihm manipuliert werden. Dazu benachrichtigt es über Datenänderungen und führt eine Validierung der vom Benutzer übergebenen Daten durch. Es beinhaltet die gesamte Geschäftslogik und ist für sich alleine durch Unit Tests überprüfbar.
* **View:** Alle durch die Grafische Benutzeroberfläche (GUI) angezeigten Elemente. Es bindet sich an Eigenschaften des ViewModel, um Inhalte darzustellen und zu manipulieren sowie Benutzereingaben weiterzuleiten. Durch die Datenbindung ist die View einfach austauschbar und ihr Code-Behind gering.
* **ViewModel**: beinhaltet die UI-Logik (Model der View) und dient als Bindeglied zwischen View und obigem Model. Einerseits tauscht es Information mit dem Model aus, ruft also Methoden oder Dienste auf. Andererseits stellt es der View öffentliche Eigenschaften und Befehle zur Verfügung. Diese werden von der View an Steuerelemente gebunden, um Inhalte auszugeben bzw. UI-Ereignisse weiterzuleiten. Insgesamt wird CRUD ermöglicht. Das ViewModel darf dabei keinerlei Kenntnis der View besitzen.

In Bezug auf WPF bedeutet MVVM, dass die View aus rein deklarativem XAML-Markup besteht. Sie kann separat von UI-Designern entworfen werden, wobei lediglich Datenbindungen zum ViewModel erzeugt werden müssen, aber kein Code-Behind. Die Logik wird in einer Programmiersprache wie C# implementiert. Die Abhängigkeiten zwischen Markup und Code werden minimiert.

Die Menge an Geschäftslogik im Code-Behind der View wird reduziert. Dadurch können UI-Designer Views rein gestalten während Entwickler unabhängig davon die Models und ViewModels implementieren. Des Weiteren sind – die Korrektheit der Datenbindung vorausgesetzt – keine (in der Regel manuellen) UI-Tests nötig. Stattdessen genügen codebasierte Modultests des ViewModel. Zuletzt „erbt“ MVVM von MVC die leichtere Austauschbarkeit der View.

## Microsoft SQL Server

Der SQL Server ist ein relationales Datenbankmanagementsystem, das sich am Standard der aktuellen SQL-Version orientiert. Der Microsoft SQL Server liegt in verschiedenen Editionen vor, die ein vielfältiges Angebot abdecken. Die Editionen unterscheiden sich vor allem im Preis, ihren Funktionen und Hardwareeinschränkungen. Der SQL Server besteht aus vielen Services und Tools, wie Analysis Services, Reporting Services, Integration Services und Sync Services.

Microsoft SQL Server verwendet für Datenbankabfragen die SQL-Variante T-SQL (Transact-SQL). T-SQL fügt hauptsächlich zusätzliche Syntax zum Gebrauch in Stored Procedures und Transaktionen hinzu. Weiterhin unterstützt er OLE DB und ODBC (Open Database Connectivity).

In diesem Projekt kommt der SQL Server in der Version 2008 R2 Express zum Einsatz. Diese Variante steht kostenlos zur Verfügung, hat aber folgende Einschränkungen:

* Es wird nur ein Prozessor bzw. ein Prozessorkern verwendet.
* Die Express-Edition nutzt maximal 1 GB Arbeitsspeicher.
* Eine Datenbank darf maximal 10 GB groß sein.

# Softwareentwicklung [Benjamin Böcherer]

Für die Umsetzung der Anwendung musste vorab eine generelle Struktur aufgebaut werden. Nachdem die ersten Entscheidungen im Bereich der zu verwendenden Technologien geklärt wurden, ging es um die konkrete Entwicklung der Software. Wir haben uns dazu entschieden, als Vorgehensmodell Scrum einzusetzen und zunächst einen ersten Prototyp zu entwickeln, um darauf aufbauend die Anwendung abzuschließen. Die folgenden Abschnitte erklären die Prozesse detailliert.

## Vorgehensmodell Scrum

Auf Basis des V-Modells haben wir uns konkret für das in der Softwareentwicklung typische Vorgehensmodell Scrum entschieden. Der Ansatz von Scrum ist empirisch, inkrementell und iterativ. Er beruht auf der Erfahrung, dass die meisten modernen Entwicklungsprojekte zu komplex sind, um durchgängig planvoll umgesetzt zu werden, und auf der Erkenntnis, dass allein ständig verfügbares Feedback den Erfolg sichert. Damit wird vermieden, die anfänglich gegebene Komplexität durch einen komplexeren Plan zu steigern.

Ziel ist die schnelle, kostengünstige und qualitativ hochwertige Fertigstellung eines Produktes, das einer zu Beginn formulierten Vision entsprechen soll. Scrum kennt drei Rollen für direkt am Prozess Beteiligte: Product Owner (stellt fachliche Anforderungen und priorisiert sie), ScrumMaster (managt den Prozess und beseitigt Hindernisse) und Team (entwickelt das Produkt). Daneben gibt es als Beobachter und Ratgeber noch die Stakeholders.

Die Anforderungen (Requirements) werden in einer Liste (Product Backlog) gepflegt, erweitert und priorisiert. Das Product Backlog ist ständig im Fluss. Um ein sinnvolles Arbeiten zu ermöglichen, wird monatlich vom Team in Kooperation mit dem Product Owner ein definiertes Arbeitspaket dem oberen, höher priorisierten Ende des Product Backlogs entnommen und komplett in Funktionalität umgesetzt (inkl. Test und notwendiger Dokumentation). Dieses Arbeitspaket, das Increment, wird während der laufenden Iteration, des sog. Sprints, nicht durch Zusatzanforderungen modifiziert, um seine Fertigstellung nicht zu gefährden. Alle anderen Teile des Product Backlogs können vom Product Owner in Vorbereitung für den nachfolgenden Sprint verändert bzw. neu priorisiert werden.

Das Arbeitspaket wird in kleinere Arbeitspakete (Tasks) heruntergebrochen und mit jeweils zuständigem Bearbeiter und täglich aktualisiertem Restaufwand in einer weiteren Liste, dem Sprint Backlog, festgehalten. Während des Sprints arbeitet das Team konzentriert und ohne Störungen von außen daran, die Tasks aus dem Sprint Backlog in ein Increment of Potentially Shippable Functionality, also einen vollständig fertigen und potentiell produktiv einsetzbaren Anwendungsteil, umzusetzen.



## Prototyping

Das Prototyping ist eine Methode der Softwareentwicklung, die vor dem Beginn der Entwicklungsphase die benötigten Risiken aufzeigt bzw. diese egalisiert und zudem schnell zu ersten Ergebnissen führt und frühzeitiges Feedback bezüglich der Eignung eines Lösungsansatzes ermöglicht. Dadurch ist es möglich, Probleme und Änderungswünsche frühzeitig zu erkennen und mit weniger Aufwand zu beheben, als es nach der kompletten Fertigstellung möglich gewesen wäre.

Die verschiedenen Arten des Prototyping werden im Folgenden näher erläutert

* **Exploratives Prototyping**

Erstes Ergebnis: Eine übersichtliche Anforderungsspezifikation

Ziel: Ziel ist es nachzuweisen, dass Spezifikationen oder Ideen tauglich sind.

Das explorative Prototyping wird zur Bestimmung der Anforderungen und zur Beurteilung bestimmter Problemlösungen verwendet und konzentriert sich dabei auf die Funktionalitäten des Systems.

* **Evolutionäres Prototyping**

Erstes Ergebnis: Ein Programm mit den Grundfunktionalitäten

Ziel: Anhand der Grundfunktionalitäten die Akzeptanz beim Nutzer und die Notwendigkeit ergänzender Funktionen zu überprüfen

Beim evolutionären Prototyping wird die Anwendung nach und nach erweitert. Dabei wird vor Allem das Feedback der zukünftigen Nutzer bzw. des Auftraggebers genutzt. Der Prototyp wird dabei stets lauffähig gehalten und bis zur Produktreife weiterentwickelt.

* **Experimentelles Prototyping**

Erstes Ergebnis: ein erster experimenteller Prototyp

Ziel: Sammeln von Erfahrungen mit dem Prototyp

Bei diesem Vorgehen wird zu Forschungszwecken bzw. zur Suche nach Möglichkeiten zur Realisierung ein experimenteller Prototyp entwickelt. An diesem wird anschließend eine sehr umfangreiche Problemanalyse und Systemspezifikation durchgeführt. Die gewonnenen Erkenntnisse können anschließend in einem richtigen Produkt verwertet werden.

* **Rapid Control Prototyping**

Rapid Control Prototyping bezeichnet die Softwareentwicklung von Regelungen und Steuerungen, mit Hilfe grafischer Tools.

* **Vertikales Prototyping (Durchstich)**

Erstes Ergebnis: Ein ausgewählter Teil des Systems wird durch alle Ebenen hindurch implementiert.

Ziel: Bestrebung explizit einen konkreten Teil eines Programms anzufertigen.

Hierbei wird ein ausgewählter Teil umgesetzt. Dies eignet sich besonders für Fälle, in denen noch Funktionalitäts- oder Implementierungsfragen ungeklärt sind. Abgeschlossene Teile können dann bereits umgesetzt werden, bevor die Anforderungen für den Rest komplett festgelegt wurden.

* **Horizontales Prototyping**

Erstes Ergebnis: Eine ausgewählte Ebene des Gesamtsystems wird fertiggestellt.

Ziel: Eine funktionierende Ebene, die vorgestellt werden kann, oder an der sich andere Ebenen orientieren können.

In diesem Fall wird nur eine spezifische Ebene des Gesamtsystems realisiert, welche jedoch möglichst vollständig abgebildet wird. (z. B. Realisierung der GUI (Oberfläche) (ohne tiefer liegende Funktionalitäten), zur Vorlage für den Auftraggeber.) Diese Methode hat den Vorteil, dass man dem Auftraggeber schon etwas zeigen kann, ohne das komplette System entwickelt zu haben. Dies setzt jedoch eine starke (sowieso sinnvolle) Trennung der einzelnen Komponenten voraus. Die Oberfläche muss dementsprechend unabhängig von der dahinter liegenden Logik funktionieren oder wenn die Logik-Ebene umgesetzt wird, muss sie unabhängig von der Oberfläche funktionieren.

Wir haben uns für einen evolutionären Prototyp entschieden. Somit konnte jeder Projektteilnehmer sein Feedback zu neuen Funktionen oder optischen Anpassungen abgeben.

## Qualitätsmanagement

Die technischen und projektorganisatorischen Projektteilnehmer verwendeten das zuvor vorgestellte SCRUM Vorgehensmodell in Verbindung mit dem „Continuous Integration“ (CI) Ansatz.

Dabei erstellt der Entwickler zunächst in seiner lokalen Arbeitsumgebung den Quellcode, der innerhalb der Entwicklungsumgebung (IDE) kompiliert und lokal installiert wird. Zuvor und sukzessive schreibt er für die fachlichen und nicht-fachlichen Anforderungen einen oder mehrere Unit Tests und führt diese aus (z. B. TestNG, JUnit, MSUnit etc.). Dies kann innerhalb der IDE oder als separater Build (Maven, Ant, MSBuild) geschehen. Das Implementieren von Komponententests dient der. Werden alle Unit Tests erfolgreich in der lokalen Umgebung ausgeführt, so „checkt“ der besagte Entwickler seine Änderungen in den zentralen Quellcodeserver ein (GitHub).

Zu vorgegebener Zeit greift nun ein sog. CI-Server (Team Foundation Server) dieses Repository ab und führt ein „Check-Out“ durch, so dass der gesamte Quellcode aller Entwickler nun innerhalb des Servers vorliegt. Danach werden die entsprechenden Quellcodedateien kompiliert und zusammen mit eventuell vorhandenen Ressourcendateien zu Artefakten gebunden (exe, dll etc.). Zudem werden nun die bestehenden Tests mittels der CI-Engine ausgeführt.

Durch geeignete Schwellenwerte kann hier der CI-Server entscheiden, ob es sinnvoll und lohnenswert erscheint, ein neues Release zu erstellen. Typische Kennzahlen hierfür sind:

* Sind alle fachlichen Tests positiv verlaufen?
* Haben alle Negativ-Tests das gewünschte Ergebnis erzielt?
* Konnten die Performance-Tests innerhalb des designierten Zeitrahmens ausgeführt werden?
* Wurden mindestens 75 % der funktionalen Anforderungen durch Tests abgedeckt?
* Wurden alle erforderlichen Formatierungsregeln eingehalten?
* Gibt es keine offensichtlichen Fehler im Quellcode?

Können alle oben angegebenen Fragen positiv beantwortet werden, so wird ein Release erstellt und automatisch in der Testumgebung installiert. Somit steht nun ein neues Release (Zwischenergebnis) für „Beta-Tester“ bzw. Kunden zur Verfügung. Entwickler und Kunden haben dadurch eine sehr konkrete Vorstellung vom aktuellen Entwicklungsstand und den letzten umgesetzten Änderungen (direkte Aktions-Reaktions-Analyse). Basierend auf diesem Release kann nun ein gezielter Dialog geführt werden, wenn es darum geht, Anforderungen anzupassen oder zu erweitern.

Zusammengefasst ergeben sich aus dem Konzept „Continuous Integration“ folgende Vorteile:

* Umsetzung der Release-Often-Paradigmen der agilen Softwareentwicklung
* Zentrale Quellcodeversionierung, Möglichkeit des „Zurückspringens“ auf ältere Versionen
* Zeitnahes Testen unter „Realbedingungen“. Wenn signifikante fachliche Fehler oder Performance Probleme auftreten, kann die Ursache schnell gefunden werden, da zwischen zwei Releases wenige Änderungen stattfinden.
* Fach- und Performancetests als integraler Bestandteil der ganzheitlichen Softwareentwicklung. Keine funktionale Anforderung wird eingecheckt ohne zugehörigen Test.
* Identifikation der Testabdeckung. Wenn beispielsweise 95% des Gesamt-Quellcodes getestet ist und keine Fehler auftraten, dann kann sehr sicher davon ausgegangen werden, dass die Software macht, was sie soll.
* Identifikation von Hot Spots und Bottlenecks bei jedem Release. Welche Methoden werden besonders oft aufgerufen, welche nehmen absolut und relativ am meisten Zeit ein? Wo ist somit Optimierungspotential?
* Prüfung, ob vorgegebene Quellcodemetriken und Dokumentationsregeln eingehalten wurden.
* Identifikation von offensichtlichen Programmierfehlern.
* Automatische Benachrichtigungsfunktion beim Erstellen des Release-Artefakts (E-Mail an Fachbereich, falls neues Release vorhanden, E-Mail an Entwicklerkreis im Falle eines Fehlers).



## Organisationswerkezeuge

Zur Erfassung der fachlichen Vorgaben (User Stories), Aufgaben, Releases und identifizierten Bugs wird ein so genanntes Application Lifycycle Management (ALM)-System eingesetzt (Microsoft Team Foundation Server). ALM ist eine Kombination aus der Entwicklung und Betreuung von Applikationen (Anwendungssoftware) über deren gesamten Lebenszyklus. Dies beinhaltet auch eine umfassende Anwenderbetreuung (Support) und die Weiterentwicklung der Software. Somit können zu jeder Zeit Informationen über den aktuellen Projektstand ermittelt werden, die in aller Regel über folgende Kennzahlen definiert werden:

* Welche Aufgaben / User Stories / Bugs sind noch offen, in Bearbeitung, fertig und geprüft in dem aktuellen Release?
* Welche Aufwände wurden bereits geleistet und welche Aufwände stehen für das aktuelle Release noch aus?
* Können die noch ausstehenden Aufwände in der noch zur Verfügung stehenden Zeit geleistet werden?
* Gibt es eine Diskrepanz zwischen der ursprünglich geschätzten und tatsächlich benötigten Zeit?



Durch die Visualisierung und Identifikation dieser Kennzahlen ist es kurzfristig möglich, einen objektiven Entwicklungsstand des Softwarevorhabens zu bekommen. Dadurch wird der Projektorganisation die Möglichkeit gegeben, aktiv Ressourcen im Projekt zielorientiert zu steuern. Ist z. B. anhand der noch ausstehenden Tätigkeiten klar, dass in der zur Verfügung stehenden Zeit die Aufgaben nicht abgeschlossen werden können (ausgehend von einem 8 Std. Werktag), kann die Projektleitung nun aktiv Gegenmaßnahmen einleiten (Features aus dem Release herausnehmen, weitere Ressourcen kurzfristig binden, Fertigstellungstermin korrigieren etc.). Somit können zu jeder Zeit authentische Aussagen zur aktuellen Projektlage und den erwarteten Ergebnissen getätigt werden.

Die Dokumentation während der Entwicklung erfolgt in einem Wiki-System (GitHub Wiki), wodurch die Entwickler kollaborativ an der Dokumentation arbeiten und diese kontinuierlich erweitern können.

## Datenbank

Durch den Einsatz des Entity Frameworks gibt es zwei Möglichkeiten, eine Datenbank zu erstellen:

* **Keine Datenbank vorhanden (Code First)**

Bestehende Klassen werden mit sogenannten Annotationen (Table, Column) ausgezeichnet, welche die Abbildung auf eine Datenbank steuern. Darauf aufbauend werden die Datenbank und die Datenbank-Tabellen modelliert und erstellt. Zudem besteht die Möglichkeit ein SQL-Skript zu erstellen, mit dem die Datenbank erstellt wird.

* **Verwendung einer bestehenden Datenbank (Model First)**

Die Entity-Klassen können entsprechend der vorgegebenen Datenbank manuell erstellt, modelliert und ausgezeichnet werden. Mit Hilfe eines Assistenten wird die Datenbank abgefragt und entsprechend der Datenbankstruktur ein passendes Modell erstellt.

Wir haben uns für den Code First-Ansatz entschieden. Es wurden zunächst die notwendigen Klassen (Entitäten) erstellt mit den entsprechenden Beziehungen. Danach wurde das Datenbankmodell mit Hilfe es Entity Framework automatisch erstellt. Bei Bedarf wurde die Datenbank aktualisiert.

## UML

Bei der Erstellung der Klassen wurde UML zur Unterstützung eingesetzt. Die Unified Modeling Language (UML) ist eine grafische Modellierungssprache zur Spezifikation, Konstruktion und Dokumentation von Software-Teilen und anderen Systemen. Sie wird von der Object Management Group (OMG) entwickelt und ist sowohl von ihr als auch von der ISO (ISO/IEC 19505 für Version 2.1.2) standardisiert. Folgende Grafik zeigt das UML-Diagramm der fertigen Anwendung:

Bild

## Quellcodeverwaltung

Da der Quellcode aufgrund der räumlichen Trennung der Projektteilnehmer an verschiedenen Orten entwickelt wurde, haben wir uns für die zentrale Quellcodeverwaltung GitHub entschieden. GitHub ist ein webbasierter Hosting-Dienst für Software-Entwicklungsprojekte. Namensgebend ist das Versionsverwaltungs-System Git. Eine Versionsverwaltung ist ein System, das zur Erfassung von Änderungen an Dokumenten oder Dateien verwendet wird. Alle Versionen werden in einem Archiv mit Zeitstempel und Benutzerkennung gesichert und können später wiederhergestellt werden. Somit kann jederzeit nachvollzogen werden, welcher Benutzer welchen Quellcode erstellt oder geändert hat. Ein weiterer Vorteil besteht darin, dass bei Fehlern in der Version zurückgesprungen werden kann.

# Die Prozesse der Software „Naukanu Sailing School Manager“

## Die Softwarearchitektur

## Die graphische Oberfläche (GUI)

## Die Stammdatenverwaltung

### Kunden



### Kursleiter



### Material





### Boote





### Qualifikationen



## Die Kursverwaltung



## Die Materialverwaltung

## Die Rechnungsverwaltung

## Die Terminverwaltung

# Die Dokumentation

## Definition

Unter dem Begriff „Dokumentation“ versteht man die gezielte Auffindung und Aufarbeitung von Informationen (Dokumente) um diese weiter verarbeiten zu können. Dokumente können Bilder, Filme, Audio, Zeitschriften, Fachbücher oder auch wissenschaftlich erhobene Daten sein.

## Merkmale einer Dokumentation

* Objektivität

Die Dokumentation soll sachlich neutral geschrieben sein. Eine subjektive Bewertung durch den Autor ist nicht erwünscht.

* Vollständigkeit

Eine Dokumentation soll alle nötigen Informationen über das Objekt bzw. eines Prozesses enthalten.

* Korrektheit

Die niedergelegten Informationen sollen inhaltlich fehlerfrei sein.

* Verständlichkeit und Nachvollziehbarkeit

Der Inhalt muss sich am jeweiligen Fachpublikum orientieren und so gestaltet sein, dass dieser klar verständlich und durch die jeweilige Zielgruppe leicht nachvollzogen werden kann.

* Authentizität und Integrität

Eine geführte Versionierung stellt eine eindeutige Nachvollziehbarkeit des geänderten bzw. neu hinzugefügten Inhaltes sicher

## Dokumentationstechniken

## Die technische Dokumentation

Die Bezeichnung „technische Dokumentation“ steht als allgemeiner Oberbegriff für die Dokumentation zu einem Produkt. Sie beinhaltet v.a. Dokumente, welche der Hersteller extern freigibt.

Dies können z.B. folgende Schriften sein:

* Bedienungs-/ Service- und Betriebsanleitungen
* Installations- und Softwarehandbücher

Des Weiteren enthält eine technische Dokumentation Informationen zu „Produktdefinition und Produktspezifikation, Konstruktion, Herstellungsverfahren, Qualitätssicherung, Produkthaftung, Produktdarstellung, Beschreibung von Funktionen und Schnittstellen, bestimmungsgemäße, sichere und korrekte Anwendung, Instandhaltung und Reparatur eines technischen Produkts sowie gefahrlose Entsorgung.“ ([www.tekom.de](http://www.tekom.de))

Dieser Projektendbericht kann als technische Dokumentation aufgefasst werden, da dieser umfangreiche Hintergrundinformationen zu der Software „Naukanu Sailing School Manager“ beinhaltet:

* Prozessabläufe
* Verwendete Architekturen (z.B. Entity Framework) und deren Aufbau
* UML - Diagramme

## Die Benutzerdokumentation

Eine Benutzerdokumentation beinhaltet sämtliche Informationen zur sicheren und fehlerfreien Bedienung eines Produktes für einen Endanwender. Bezogen auf ein Softwareprodukt enthält sie eine detaillierte Beschreibung über die Bedienung der graphischen Oberfläche, stellt die Funktionen der Anwendung in einem strukturierten und sinnvollen Kontext dar und gibt Hilfestellungen, welche die Arbeit des Benutzers vereinfachen und beschleunigen. Sie dient in erster Linie zur Einführung in die Applikation und Vertiefung des Anwenderwissens.

Das beiliegende Handbuch ist als Benutzerdokumentation anzusehen, da es neben der Installation der Datenbank, das GUI-Konzept erläutert, sowie alle nötigen Prozesse zur erfolgreichen Kursverwaltung der Segelschule umfassend beschreibt.

# Ausblick und zusätzliche Features

# Zusammenfassung

# Glossar

# Literaturverzeichnis

Abrahamson, N. (1986). Development of the ALOHANET. *IEEE TRANSACTIONS ON INFORMATION THEORY, VOL IT-31*, 10-20.

Arnold, A. (02. 05 2009). *WLAN - Verschlüsselung - WEP und WPA durchleuchtet*. Abgerufen am 01. 01 2010 von http://www.heise.de/netze/artikel/WEP-und-WEPplus-224028.html

Grechenig, T., & Bernhart, M. (2010). *Softwaretechnik.* München: Pearson Studium.

Grochla, E. (1982). *Grundlagen der organisatorischen Gestaltung.* Stuttgart: Schäffer-Poeschel.

Hein, M., & Maciejeski, B. (2003). *Wireless LAN - Funknetze in der Praxis.* Poing: Franzis' Verlag.

Kafka, G. (2005). *WLAN - Technik, Standards, Planung und Sicherheit für Wireless LAN.* München / Wien: Hanser.

Management, H. (2014). *PM-Handbuch.com*. Abgerufen am 27. 06 2014 von http://www.pm-handbuch.com/nutzen/

*PM-Handbuch.com*. (2014). (H. Management, Produzent) Abgerufen am 27. 06 2014 von http://www.pm-handbuch.com/begriffe/

# Abbildungsverzeichnis

[Abbildung 12‑2 18](#_Toc391475386)

[Abbildung 12‑3 18](#_Toc391475387)

# Tabellenverzeichnis

[Table 1 19](#_Toc391476773)

[Table 2 19](#_Toc391476774)

|  |  |
| --- | --- |
|  |  |
|  |  |

Table

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |

Table

# Abkürzungsverzeichnis

# Anhang

## 

1. Quelle: http://www.kraus-und-partner.de/projektmanagement/wiki/projektmanagement [↑](#footnote-ref-2)