HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Flight Logging System

Bachelorarbeit

Abteilung Informatik
Hochschule für Technik Rapperswil

Frühjahrssemester 2012

Autor(en):	Marion Frei, Lara Mühlemann, Marion Walser
Betreuer:	Prof. Hansjörg Huser
Projektpartner:	Flugsportgruppe Zürcher Oberland (FGZO), Patrick Schuler
Experte:	Stefan Zettel, Ascentiv Zürich
Gegenleser:	Ivan Bütler

Inhaltsverzeichnis

I

Aufgabenstellung

Datum: 15. Juni 2012

Flight Logging System

Aufgabenstellung für Marion Frei, Lara Mühlemann, Marion Walser

Aufgabe: gemäss Aufgabenstellung von Patrick Schuler
Resultate: gemäss Aufgabenstellung von Patrick Schuler

Betreuung HSR

Hansjörg Huser, hhuser@hsr.ch, Tel: 055 222 49 12 (HSR Raum 6.010)

Projektabwicklung

Termine:

- Beginn der Arbeit: M**o., 20. Feb. 2012**
- Abgabetermin Kurzfassung/Poster/Mgmt-Summary zum Review: 8.06.2011
- Abgabetermin (ink. Poster): 15.06.2011, 12.00 Uhr
- **HSR-Forum, Vorträge und Präsentation der Bachelor- und Diplomarbeiten, 15.6.2012 nachmittags**
- Mündliche BA-Prüfung : genaues Datum folgt (6.8.-25.8.)
- Zwischenbesprechung/Review mit Auftraggeber nach Projektplan

Arbeitsaufwand

Für die erfolgreich abgeschlossene Arbeit werden 12 ECTS angerechnet. Dies entspricht einer Arbeitsleistung von mind. 360 Stunden pro Student. (2 Wochen zu 45h=90h, 14 Wochen zu 20h=280h)

Hinweise für die Gliederung und Abwicklung des Projektes:

Gliedern Sie Ihre Arbeit in 4 bis 5 Teilschritte. Schliessen Sie jeden Teilschritt mit einem Meilenstein ab. Definieren Sie für jeden Meilenstein, welche Resultate dann vorliegen müssen!

Folgende Teilschritte bzw. Meilensteine sollten Sie in Ihrer Planung vorsehen:
Schritt 1: Projektauftrag inkl. Projektplan (mit Meilensteinen),

- Meilenstein 1: Review des Projektauftrages abgeschlossen. Projektauftrag von Auftraggeber und Dozent genehmigt
  Letzter Meilenstein: Systemtest abgeschlossen
  Termin: ca. eine Woche vor Abgabe
- Entwickeln Sie Ihre SW in einem iterativen, inkrementellen Prozess: Planen Sie möglichst früh einen ersten lauffähigen Prototypen mit den wichtigsten und kritischsten Kernfunktionen. In die folgenden Phasen können Sie dieses Kernsystem schrittweise ausbauen und testen.
- Falls Sie in Ihrer Arbeit neue oder Ihnen unbekannte Technologien einsetzen, sollten Sie parallel zum Erarbeiten des Projektauftrages mit dem Technologiestudium beginnen.
- Setzen Sie konsequent Unit-Tests ein! Verwalten Sie ihre Software und Dokumente auf einem SVN-Repository. Stellen Sie sicher, dass der/die Betreuer jederzeit Zugriff auf das Repository haben und dass das Projekt anhand des Repositories jederzeit wiederhergestellt werden kann.
- Achten Sie auf die Einhaltung guter Programmier- und Designprinzipien
  - DRY, high cohesion, loose coupling, etc.

- Clean Code!
- Halten Sie sich im Übrigen an die Vorgaben aus dem Modul SE-Projekt.

Projektadministration

- Führen Sie ein individuelles Projekttagebuch aus dem ersichtlich wird, welche Arbeiten Sie durchgeführt haben (inkl. Zeitaufwand). Diese Angaben sollten u.a. eine individuelle Beurteilung ermöglichen.
- Dokumentieren Sie Ihre Arbeiten laufend. Legen Sie Ihre Projektdokumentation mit der aktuellen Planung und den Beschreibungen der Arbeitsresultate elektronisch in einem Projektordner ab. Dieser Projektordner sollte jederzeit einsehbar sein (z.B svn-Server oder File-Share).

Inhalt der Dokumentation

Bei der Abgabe muss jede Arbeit folgende Inhalte haben:

- Dokumente gemäss Vorgabe: https://www.hsr.ch/Allgemeine-Infos-Diplom-Bach.4418.0.html
- Aufgabenstellung
- Technischer Bericht
- Projektdokumentation
- Die Abgabe ist so zu gliedern, dass die obigen Inhalte klar erkenntlich und auffindbar sind.
- Zitate sind zu kennzeichnen, die Quelle ist anzugeben.
- Verwendete Dokumente und Literatur sind in einem Literaturverzeichnis aufzuführen.
- Projekttagebuch, Dokumentation des Projektverlaufes, Planung etc.

Fortschrittsbesprechung:

Regelmässig findet zu einem fixen Zeitpunkt eine Fortschrittsbesprechung statt.
Teilnehmer: Dozent und Studenten, bei Bedarf auch Vertreter der Auftraggeber
Termin: jeweils tbd, Raum 6.010 (Abweichungen werden rechtzeitig kommuniziert)
Traktanden

- Was wurde erreicht, was ist geplant, welche Probleme stehen an
- Review von Code/Dokumentation (Abgabe jeweils einen Tag vor dem Meeting

Falls notwendig, können weitere Besprechungen / Diskussionen einberufen werden.
Sie erstellen zu jeder Besprechung ein Kurzprotokoll, welches Sie spätestens 2 Tage nach der Sitzung per e-mail an den Betreuer senden.

Rapperswil, 20. Feb. 2012
Hansjörg Huser

Flight Logging System

Aufgabenstellung für Bachelorarbeit

Projekt:	Flight Logging System (FLS) 2012
Auftraggeber:	Flugsportgruppe Zürcher Oberland (FGZO) in Zusammenarbeit mit clTius AG, Patrick Schuler Ifangstrasse 10 8302 Kloten
Betreuer:	Patrick Schuler
Dozent:	Hansjörg Huser, hhuser@hsr.ch
Studenten:	Frei Marion, m1frei@hsr.ch Mühlemann Lara, lmuehlem@hsr.ch Walser Marion, mwalser@hsr.ch
Erstellt:	12. Februar 2012
Letzte Änderung:	17. Februar 2012
Version:	1.0
Ablage:	Aufgabenstellung BA FLS 2012.docx

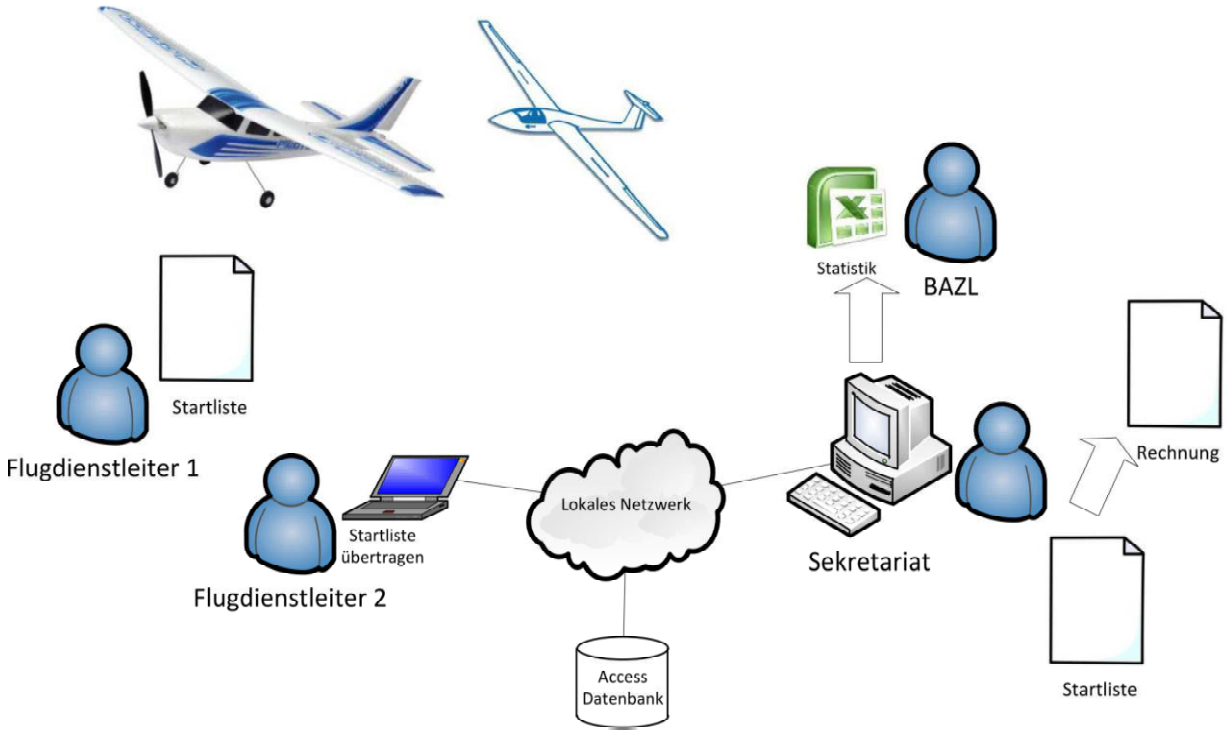
1 Einleitung

Das Bundesamt für Zivilluftfahrt (BAZL) schreibt vor, dass sämtliche Flugbewegungen auf Schweizer Flugplätzen auf Startlisten erfasst werden müssen. Dies gilt auch für Segelflieger. Diese Daten müssen monatlich im vordefinierten Excel-Formular zu statistischen Zwecken ans BAZL geliefert werden. Weiter werden die Flüge den Piloten periodisch durch die Fluggruppe in Rechnung gestellt, was eine detaillierte Aufstellung der Flüge erfordert.
Mangels Verfügbarkeit passender Software sind die Prozesse für beide Vorgänge sehr umständlich und verlangen an mehreren Stellen einiges an manueller Arbeit. Dadurch können sich einerseits leicht Fehler einschleichen und andererseits ist die anfallende Arbeit sehr unerfreulich.

Einzelne Insel- oder Individuallösungen sind zwar verfügbar, jedoch sind keine ausreichenden Schnittstellen definiert. Es fehlt eine modulare Gesamtlösung, die Schnittstellen zu weiteren Systemen unterstützt und leicht integriert und erweitert werden kann.

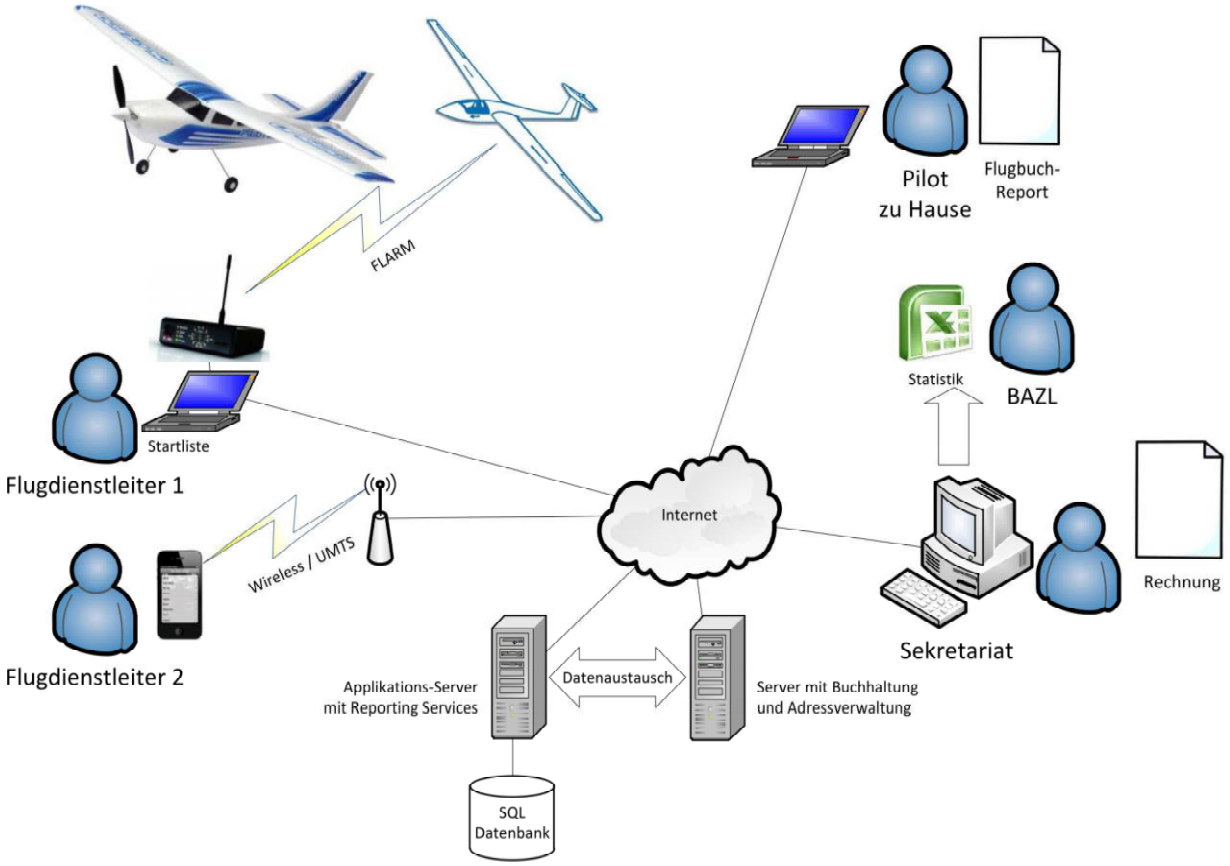
2 System

Die nachfolgende Abbildung zeigt die aktuelle Systemumgebung.



Der Flugdienstleiter erfasst sämtliche Flüge auf einem Flugplatz in Papierform auf dem Startlistenformular. Am Abend wird diese Liste dann jeweils in ein MS Access-basiertes System übertragen. Erste Fehler können sich hier bereits einschleichen, da keine Validierung der Daten durchgeführt wird. Die handgeschriebenen Startlisten werden im Sekretariat hinterlegt. Ende Monat werden durch das Sekretariat anhand der Startlisten jeweils die Rechnungen erstellt. In der heutigen Situation werden die Rechnungen manuell erfasst.
Statistische Daten für das BAZL werden in der entsprechenden Excel-Vorlage eingegeben und ans BAZL weitergeleitet.
Piloten tragen ihre Flüge jeweils am Abend vor dem Verlassen des Flugplatzes in ihr Flugbuch ein.

Folgende Abbildung zeigt die neue Ziel-System-Übersicht.



Der Flugdienstleiter erfasst sämtliche Flüge direkt elektronisch im System. Dies kann über ein PDA, Laptop oder anderes Device erfolgen. Mittels FLARM könnten Start- und Landevorgänge sogar automatisch erkannt und die entsprechende Start- und Landezeit erfasst werden. Ende Monat werden anhand der Startlisten jeweils die Rechnungen erstellt. Dies soll ebenfalls automatisiert werden. Dazu müssen gegebenenfalls auch Adressdaten und Buchhaltungsartikel zwischen mehreren Systemen ausgetauscht werden. Dies soll mittels definierbaren und konfigurierbaren Schnittstellen ermöglicht werden.
Statistische Daten für das BAZL in Excel-Format sollen via Sekretariat oder direkt vom System versendet werden.
Piloten sollen die Möglichkeit erhalten, sich auf dem Web-Interface einzuloggen, um ihre Flüge anzusehen und entsprechende Reports erzeugen zu lassen.
Workflows und automatische System-Prozesse sollen die Prozesse weiter unterstützen. Dies kann zum Beispiel sein, dass das Sekretariat jeweils Anfangs Monat die Rechnungen automatisch per Mail vom System erhält ohne einen Klick zu machen oder dass der Pilot den Flugrapport am Abend bereits in seiner Mailbox hat.
Damit nicht jede Fluggruppe sämtliche Flugzeuge, Flugplätze und Piloten erfassen muss, soll dies ein schweizweites, mandantenfähiges, zentrales System geben. Ein modularer Aufbau und eine ausbaufähige Architektur ist somit ein Muss.

Aus heutiger Sicht soll der Endausbau der Softwarelösung folgende Funktionen umfassen:

- Authentifizierung und Authorisierung der Benutzer
- User-Management mit Berechtigungsmöglichkeiten
- Erfassen der Stammdaten, wie:
  - Adressen der Mitglieder bzw. Besatzungsmitglieder und Passagieren
  - Flugzeuge (Schlepp- und Segelflugzeuge)
  - Flugplätze und Aussenlandefelder
- Erfassen der Startliste (mit Angaben über Datum und Startlistenführer)
- Erfassen der Flüge (inkl. Schleppflug) mit folgenden Details:
  - Angaben zu Besatzung, wie: Pilot, Schlepppilot, Copilot, Fluglehrer, Passagier
  - Immatrikulation
  - Start- und Landezeit, Schleppzeit, Anzahl Landungen
  - Start- und Landeort sowie Pistenrichtung
  - Flugart, wie:
    - Normaler Segelflug (Solo)
    - Doppelsitzerflug (Pilot/Pilot)
    - Ausbildungsflug mit Fluglehrer am Doppelsteuer oder Solo-Ausbildungsflug
    - Checkflug (mit Fluglehrer)
    - Passagierflug mit oder ohne Gutschein (Passagierflugdaten mit Gutschein oder Rechnungsadresse)
    - Segelflug mit Aussenlandung auf anderem Flugplatz oder auf Acker (Erfassung von neuen Standorten)
    - Erfassen von auswärtigen Segelflügen
  - Bemerkungen
  - Verrechnungsinformationen
- Reportings/Statistiken für Piloten und Flugzeugwarte
- Mandantenfähigkeit (Ausbaufähigkeit ermöglichen)
- Plugin-Architektur und Workflow-Integration pro Mandant
- Synchronisieren der Adressdaten mit dem Buchhaltungs- und anderen Systemen
- Startlisten-Export für BAZL
- Erfassen von Defekten/Schäden an Flugzeugen (Incident-Management für die Flugzeugwartung)
- Vollautomatische Erfassung der Flüge mittels FLARM (www.flarm.com)

3 Aufgabe

- Analyse des bestehenden Datenbank-Designs
- Analyse der Use Cases
- Analyse der gesamten Architektur und Anbindung an andere Systeme (CRM, ERP)
- Definieren des Projektumfanges (was soll implementiert werden)
- Design der System- und Software-Architektur und der Schnittstellen
- Implementation der Server-Applikation mit Entity Framework und Silverlight Client
- Implementation der Reports basierend auf MS SQL Reporting Services
- Ausarbeiten und Umsetzung eines Test-Konzepts mittels automatisierten Tests
- Dokumentation sämtlicher Definitionen und Entscheide

3.1 Ziel

- Lauffähige, webbasierte Client- / Server-Applikation für das Erfassen von Flügen und der Stammdaten, sowie das Erstellen von Reports
- Sauber durchdachte Software-Architektur und gute Test-Abdeckung mit automatisierten Tests
- Dokumentation der Arbeit und des Source Codes

3.2 Zur Verfügung gestelltes Material

Durch den Auftraggeber wird folgende Infrastruktur bereitsgestellt:

- Entwicklungs-Server, mit:
  - SVN
  - MediaWiki
  - MantisBT
  - Backup der Daten
- Test-Server
- SSL VPN Client für Server-Zugriff
- MS SQL Server 2008 R2 Express Edition auf Test-Server
- Infragistics GUI Library

Durch die Schule wird folgende Infrastruktur bereitsgestellt:

- Arbeitsplatz inkl. Visual Studio 2010
- PC und Labor
- Weitere Software

3.3 Vorgesehene Technologien:

- Silverlight 5 mit Infragistics GUI Library
- C# .NET 4.0
- Entity Framework 4.1
- Model-View-ViewModel Design Pattern Architektur
- Plugin-Architekur (für weitere Schnittstellen, etc.)
- MS SQL-Server 2008 R2 inkl. Reporting Services

4 Nächste Schritte

Wie unter Kapitel 3 erwähnt sollen die Funktionen priorisiert und entsprechend realisiert werden. In der ersten Phase der Arbeit sollen die Aufgaben analysiert und die entsprechenden Arbeitspakete definiert werden. Sobald bekannt ist, wie sich der Aufwand auf die einzelnen Funktionen und Teilbereiche verteilt, soll festgelegt werden was genau realisiert werden soll.
Die Arbeit soll nach den Richtlinien der HSR durchgeführt und dokumentiert werden.
Die Entwicklungs-Methoden und Werkzeuge müssen mit dem Kunden abgesprochen werden.

5 Termine

- Aufgabenstellung und Beginn der Arbeit: 20.02.2012
- Abgabe der Arbeit 15.06.2012 12:00 Uhr
- Experten Gespräch Im August 2012, Termin offen
- HSR Forum 15.06.2012 13:00 – 18:00 Uhr

6 Betreuung

Betreuung: Patrick Schuler
Dipl. Inf. Ing. FH, Segelfluglehrer bei der FGZO, Geschäftsleitungsmitglieder der cITius AG

II

Eigenständigkeitserklärung

Datum: 15. Juni 2012

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Erklärung zu Bachelorarbeit Flight Logging System auf Eigenständigkeit der Arbeit

Erklärung

Wir erklären hiermit,

- dass wir die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt haben, 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 haben.
- dass wir keine durch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt haben.

Ort, Datum:

Rapperswil, 14.06.2012

Name, Unterschrift:

1. Lara Mühlemann, L. Mü
2. Marion Walser, M. Walser
3. Marion Frei, M. Frei

III

Abstract

Datum: 15. Juni 2012

Abstract

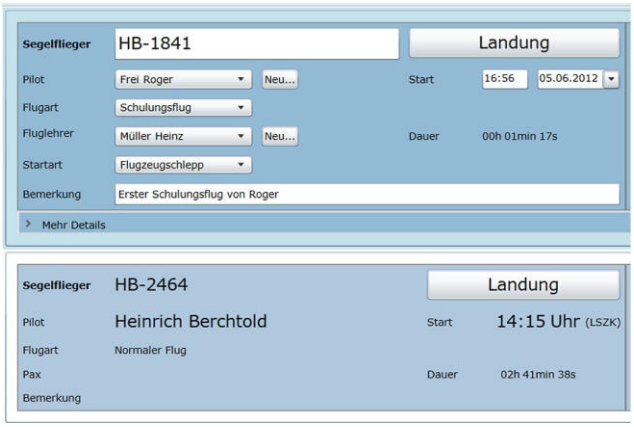


Abbildung 0.1: Ausschnitt Startliste

Abbildung 0.2: Ausschnitt Stammdaten

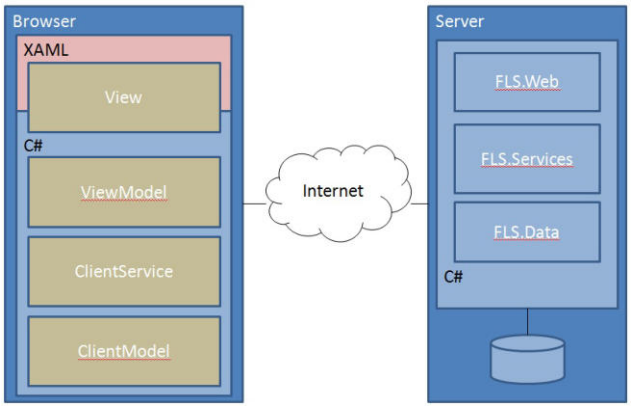


Abbildung 0.3: Architekturübersicht

Ausgangslage: Der Segelflugverein Flugsportgruppe Zürcher Oberland erfasst die Starts und Landungen der Segelflieger von Hand auf Startlisten. Dazu müssen diverse Informationen erfasst werden, unter anderem die Start-/Landezeiten. Ende Monat verrechnet das Sekretariat anhand dieser Startlisten die Flüge. Die Anzahl Flüge und die Flugdauer müssen hierfür zusammengerechnet werden. Das Ziel dieser Bachelorarbeit ist die Erstellung einer Applikation, in welcher Flüge erfasst, die dazu benötigten Stammdaten verwaltet und Auswertungen erstellt werden können. Die Applikation muss mandantenfähig sein, da sie zentral mehreren Segelflugvereinen zur Verfügung gestellt wird.

Vorgehen/Technologien: Aufgrund der bisherigen Erfahrungen der Teammitglieder wurde das Vorgehen an Rational Unified Process (RUP) angelehnt. Da das Entwicklerteam mit den verwendeten Technologien wenig Erfahrung hatte, wurde in der ersten Phase viel Zeit für die Einarbeitung in die neuen Technologien verwendet. Bereits in der zweiten Woche wurde mit der Implementation begonnen. Dadurch konnte das Team die Theorie früh anwenden und sich ein besseres Bild über die Aufgabe machen. Durch die Aufgabenstellung und die klaren Vorstellungen des Kunden waren die meisten Technologien gegeben. Es wurde einer Silverlight RIA Applikation aufgebaut, welche im Browser gestartet wird und über WCF-Services und das Entity Framework auf eine Microsoft SQL Datenbank zugreift. Um Reports zu erstellen wurde der Microsoft SQL Reporting Service verwendet.

Ergebnis: In der Abbildung 0.1 ist die Flugerfassung ersichtlich. Flüge können damit einfach erfasst und mittels Buttons gestartet und gelandet werden. In der Abbildung 0.2 ist eine der Stammdatenmasken zu sehen. Über das System können Vereine, Benutzer, Personen, Flugzeuge und Flugplätze verwaltet werden. Täglich werden an die Piloten per E-Mail Flugreports über die an diesem Tag ausgeführten Flüge verschickt. Jeden Monat erhält das Sekretariat für die Abrechnung einen Report über die Flüge des Monats. Beide Reporte sind auch über den Browser abrufbar. Weiter können unter Import/Export die Adressen und Standorte exportiert und importiert werden. Umgesetzt wurde eine Client-Server Architektur. Auf dem Server befinden sich Services, über die der Client Daten laden und bearbeiten kann. Der Client ist aufgeteilt in ClientModels, welche die Daten laden und buffern, ClientServices welche für die Manipulationen verantwortlich sind und Views, deren Elemente an ViewModels gebunden sind. (Siehe Abbildung 0.3)

IV

Management Summary

Datum: 15. Juni 2012

Management Summary

Ausgangslage

Der Auftraggeber dieser Bachelorarbeit ist der Segelflugverein Flugsportgruppe Zürcher Oberland. Segelflüge müssen mit verschiedenen Informationen, wie zum Beispiel den Start- und Landezeiten der Segelflieger und Schleppflieger, protokolliert werden. Bisher geschah dies mit Papier und Bleistift (siehe Abbildung 0.4: „Beispiel einer Startliste [FGZ11]“. Die Zeit musste von einer Uhr abgelesen werden. Oftmals sind diese Startlisten nicht gut lesbar.

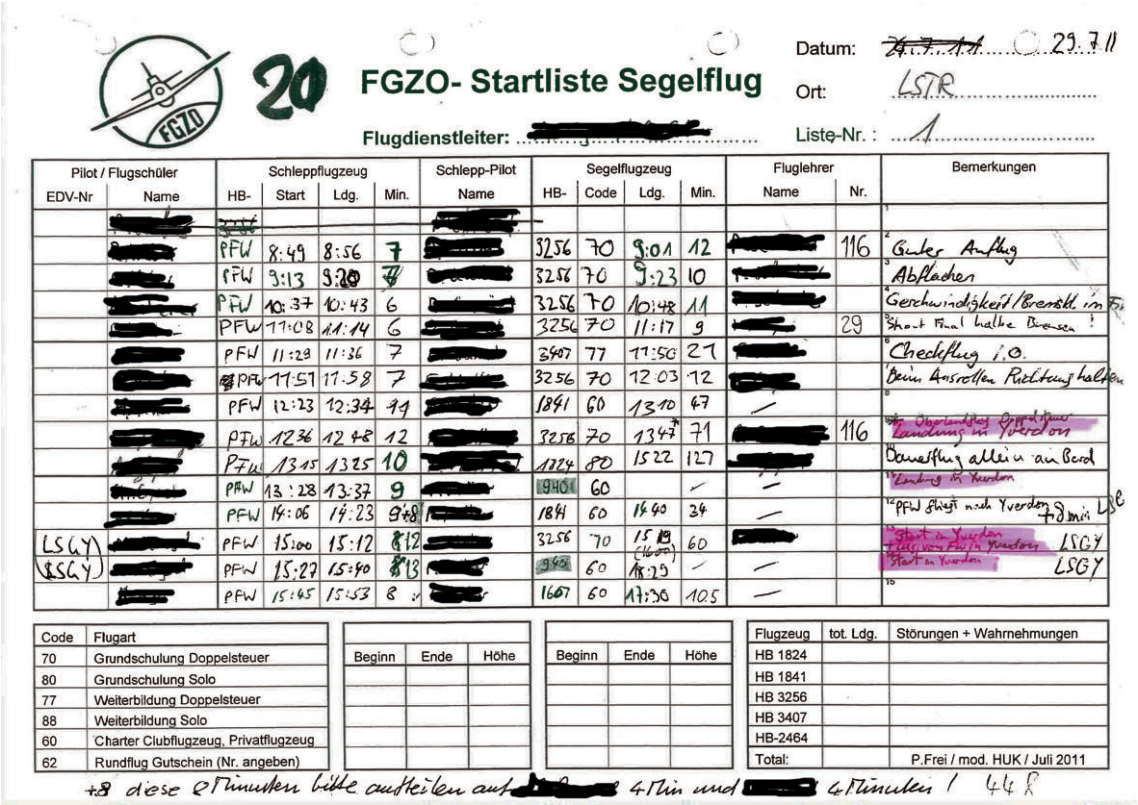


Abbildung 0.4: Beispiel einer Startliste [FGZ11]

Ende Monat verrechnet das Sekretariat anhand dieser Startlisten die Flüge an die Piloten. Die Anzahl Flüge und die Flugdauer müssen hierfür zusammengerechnet werden. Da dies bisher ebenfalls von Hand geschah, konnten dabei schnell Fehler entstehen.

Das Ziel dieser Bachelorarbeit ist die Erstellung einer Applikation, in der Flüge erfasst, die

dazu benötigten Stammdaten verwaltet und Auswertungen erstellt werden können. Die Applikation muss mandantenfähig sein, da sie zentral mehreren Segelflugvereinen zur Verfügung gestellt wird. Um die Adressen mit einem Buchhaltungssystem abzugleichen, müssen sie mittels Import und Export synchronisiert werden können.

Vorgehen, Technologien

Das Entwicklerteam hatte in den verwendeten Technologien bisher nur sehr wenig bis keine Erfahrungen. Aus diesem Grund wurde zu Beginn des Projektes neben der Projektplanung viel Zeit für die Einarbeitung in neue Technologien verwendet. Damit die Theorie verfestigt werden und sich das Team ein besseres Bild über die Aufgabenstellung machen konnte, wurde bereits in der zweiten Projektwoche mit der Implementation begonnen.

Durch die Aufgabenstellung und die klaren Vorstellungen des Kunden waren die meisten Technologien gegeben. Die Applikation wurde auf einer Silverlight RIA Struktur aufgebaut, welche im Browser gestartet wird, über WCF-Services und das Entity Framework auf die Datenbank zugreift und auf einem Microsoft SQL Server 2008 R2 liegt. Um Reports zu erstellen wurden der Microsoft SQL Reporting Server verwendet.

Ergebnisse

In der Abbildung 0.5: „Neue Flugerfassung“ ist die Eingabemaske für die Flugerfassung ersichtlich. Flüge können darin einfach erfasst und mittels Buttons gestartet und gelandet werden. Die Flugdauer wird automatisch berechnet. Der Schleppflieger und der Schlepppilot werden zwischengespeichert und bei neu erfassten Flügen vorgeschlagen. Ein veränderbarer Filter über dem Datum verhindert das unnötige Laden von älteren Flügen (Standard-Einstellung: Flüge älter als 10 Tage werden nicht geladen).

In der Abbildung 0.6: „Stammdaten“ ist eine der Stammdatenmasken zu sehen. Über das System verwaltbar sind die Vereine, Benutzer, Personen, Flugzeuge und die Flugplätze.

Täglich werden aus dem Reporting Server Flugreports an die Piloten verschickt. Diese enthalten die ausgeführten Flüge dieses Tages. Jeden Monat erhält das Sekretariat ebenfalls per E-Mail einen Report über die Flüge des letzten Monats für die Abrechnung. Alle Reports sind auch vom Benutzer aus der Applikation generierbar.

Unter Import/Export können Adressen und Standorte exportiert und importiert werden.

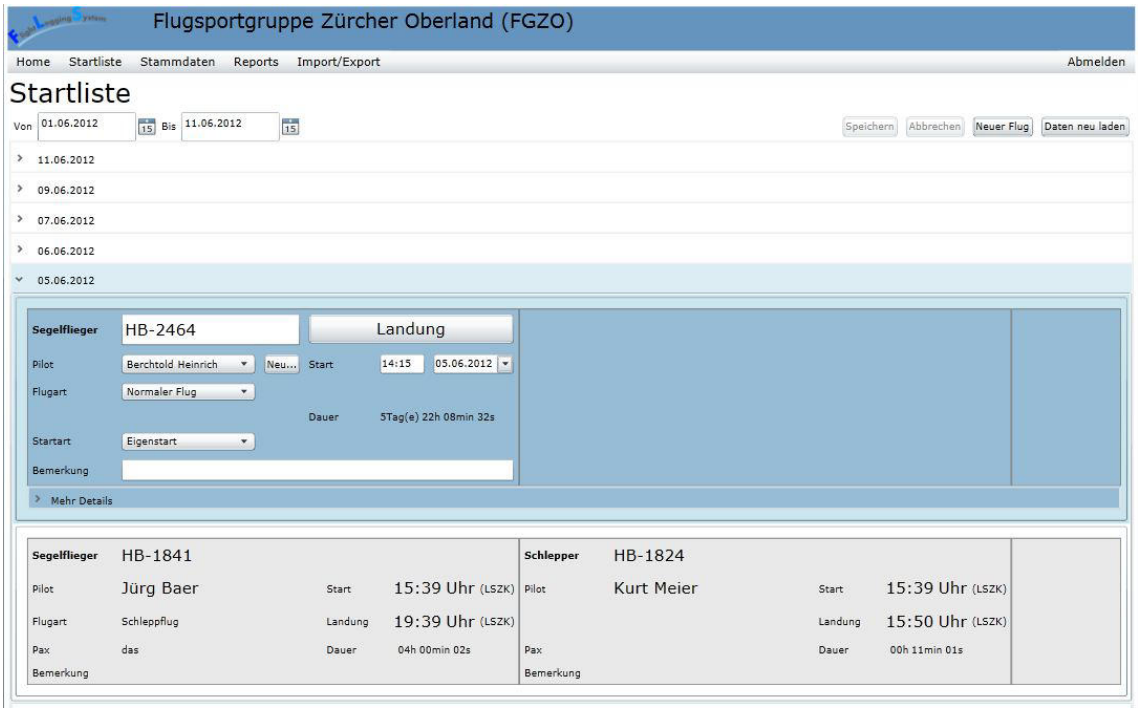


Abbildung 0.5: Neue Flugerfassung

Abbildung 0.6: Stammdaten

Ausblick

Die Applikation kann in Zukunft durch folgende Funktionalitäten erweitert werden:

- Es können weitere Reports umgesetzt werden.
- Eine Anbindung an FLARM kann implementiert werden.
- Ein Client für Handys kann entwickelt werden.
- Momentan kann der Vereinsadministrator neue Mitglieder zum Verein hinzufügen. Eine mögliche Weiterentwicklung wäre, dass eine Person eine Anfrage für eine Clubmitgliedschaft an einen Verein schicken kann.
- Die Erfassung einer Zwischenlandung ist noch nicht optimal. Es gibt zwei mögliche Verbesserungen:
  1. Durch eine Zwischenlandung entstehen im UI nicht mehr zwei Flüge, sondern die Zwischenlandung wird als Bestandteil des Fluges in der Detailansicht verwaltet (In der Datenbank werden immer noch zwei Flüge gespeichert). Für diese Lösungsmöglichkeit sind Datenbankanpassungen nötig (Verknüpfung als Zwischenlandungsflug).
  2. Es wird ein Button „Zwischenlandung hinzufügen“ eingeführt. Beim Klicken auf diesen Button wird ein neuer Flug erstellt. Die Landezeit und der Landeort des alten Fluges werden in den neuen Flug verschoben. Beide Flüge sind nun im Bearbeitungsmodus. Der Flugdienstleiter muss nun nur noch den Ort der Zwischenlandung und die Start und Landezeiten entsprechend in die Flüge eingeben. (Das Kopieren von Hand der Landezeit/Landeort wird so eingespart.)
- Piloten könnten für eine schnellere Eingabe nach folgenden Kriterien sortiert werden:
  - Piloten, die am aktuellen Tag bereits geflogen sind, erscheinen zuoberst.
  - Piloten aus dem eigenen Verein erscheinen danach.
  - Es folgen die restlichen Personen.
- Wenn eine Person eine Lizenz als Pilot besitzt, könnten weitere Felder in den Stammdaten als Muss-Felder definiert werden.

V

Technischer Bericht

Datum: 15. Juni 2012

1.1 Einleitung und Übersicht

Bisher erfasst der Segelflugverein Flugsportgruppe Zürcher Oberland die Start und Landungen der Segelflieger von Hand auf Startlisten. Hierfür müssen diverse Angaben aufgeschrieben werden, wie zum Beispiel die Start- und Landezeiten. Die Startlisten werden jeweils Ende Monat vom Sekretariat für das BAZL zusammengefasst. In dieser Bachelorarbeit wurde eine Applikation entwickelt, um Segelflüge zu verwalten. Das Programm beinhaltet eine effiziente Erfassungsmöglichkeit für Flüge, welche unter anderem die Erfassung von Start- und Landezeiten beinhaltet. Die benötigten Daten können über eine Stammdatenverwaltung erfasst und bearbeitet werden. Diese enthält Vereine, Benutzer, Personen, Flugzeuge und Flugplätze. Das Flight Loggin System ist mandantenfähig und verfügt über eine Authentifizierung und Autorisierung. Weiter beinhaltet die Applikation die Möglichkeit Reports zu generieren und eine Export- und Import-Möglichkeit von Daten.

Die Applikation ist auf einer Silverlight RIA Struktur aufgebaut, welche im Browser gestartet wird, über WCF-Services und das Entity Framework auf eine Datenbank zugreift und die auf einem Microsoft SQL Server 2008 R2 liegt. Um Reports zu erstellen wird der Microsoft SQL Reporting Server verwendet.

1.2 Ergebnisse

1.2.1 Stammdatenverwaltung

Es wurde eine umfangreiche Stammdatenverwaltung erstellt. In der Abbildung 1.1: „Menü Stammdaten“ ist das Menü mit allen verfügbaren Stammdatenmasken zu sehen. In jeder Ansicht ist die Funktionalität für die CRUD-Operationen vorhanden.

Abbildung 1.1: Menü Stammdaten

Alle Stammdatenmasken sind ähnlich aufgebaut. In der Abbildung 1.2: „Stammdaten Vereine“ ist als Beispiel die Verwaltung für die Vereine ersichtlich. Pfeil 1 zeigt auf den Filter. Es gibt eine Textsuche über alle Attribute und gelöschte Daten können ein- oder

ausgeblendet werden. Der Pfeil 2 zeigt auf die Buttons. Buttons werden je nachdem ob Änderungen vorhanden sind, aktiviert oder deaktiviert. Im Bild sind Änderungen vorhanden und die Buttons „Speichern“ und „Abbrechen“ sind aus diesem Grund aktiviert, während die Buttons „Verein löschen“ und „Neuer Verein“ deaktiviert sind. Falls der Benutzer einen anderen Verein auswählen oder auf eine andere Ansicht wechseln möchte, erscheint eine Meldung, die ihn daran erinnert, dass er vorher speichern oder abbrechen muss.

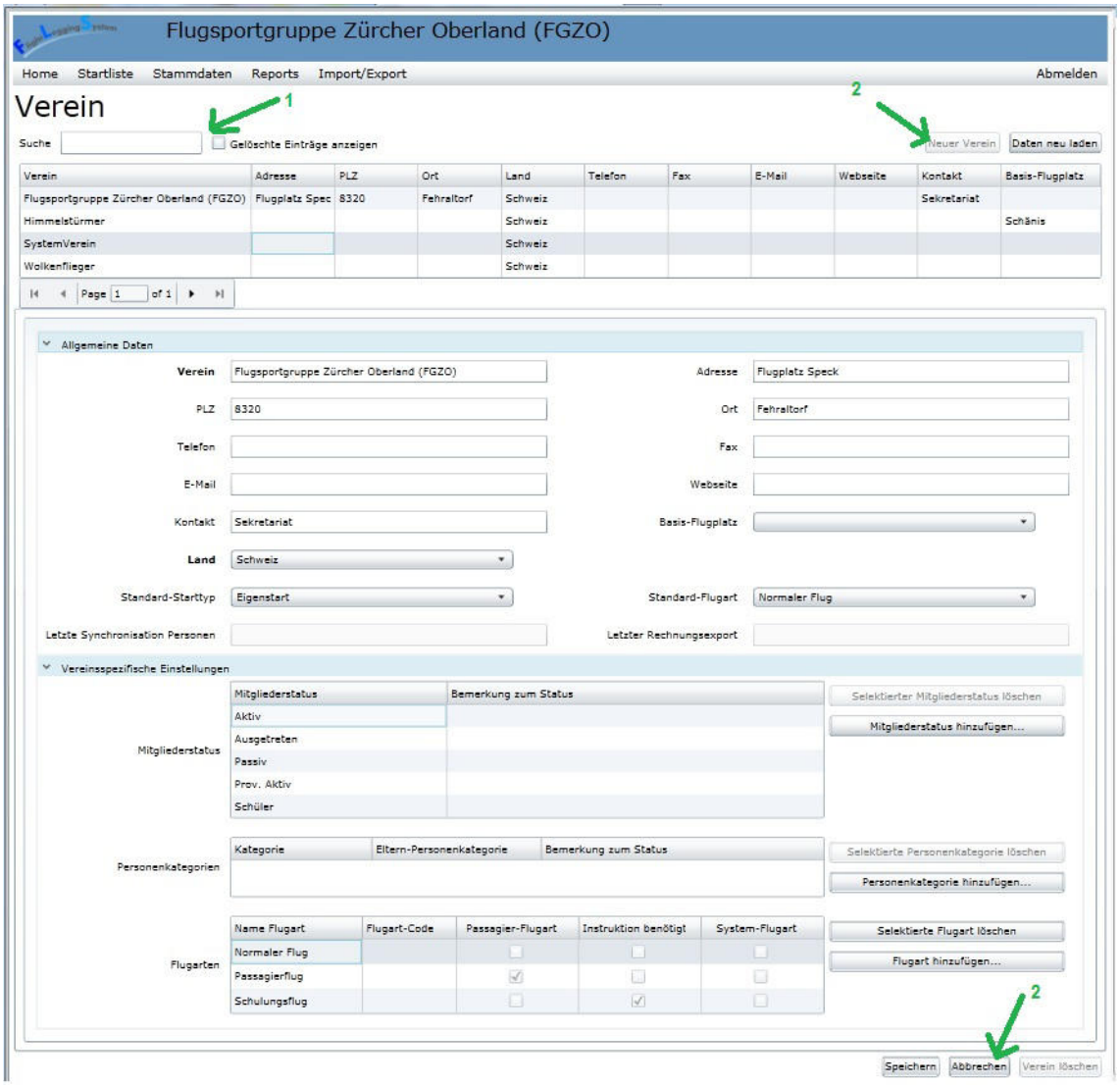


Abbildung 1.2: Stammdaten Vereine

1.2.2 Startliste

In der Startliste (Abbildung 1.3: „Startliste“ können die Flüge verwaltet werden. Die Flüge werden gruppiert nach Datum in einem Accorion angezeigt. Ein Flieger kann mit Buttons gestartet oder gelandet werden (siehe Pfeil 1). Die Dauer des Fluges wird automatisch jede Sekunde neu berechnet (siehe Pfeil 2). Die Start- und Landezeit kann im nachhinein angepasst werden (siehe Pfeil 3). Die Startzeit des Schleppfliegers ist an die des Segelfliegers gebunden (siehe Pfeil 4). Die Landezeiten sind unabhängig. Die Piloten des Fluges können über Comboboxen ausgewählt werden. Da oftmals Probeflüge mit noch nicht erfassten Piloten durchgeführt werden, können neue Personen einfach über ein Popup erstellt werden. Damit nicht zu viele Daten geladen werden müssen, ist standardmässig in der Datenbank eingestellt, dass nur die Flüge der letzten 10 Tage geladen werden. Der Benutzer kann diesen Zeitraum über einen Filter ändern (siehe Pfeil 5).

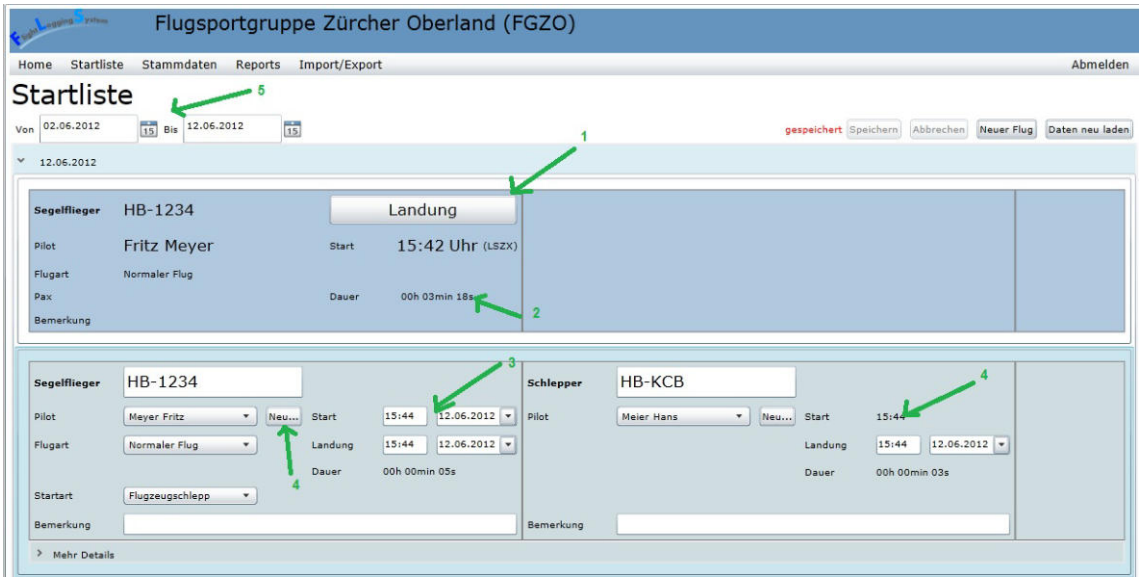


Abbildung 1.3: Startliste

1.2.3 Reports

In der Applikation können Reports erstellt werden. Es gibt drei verschiedene Arten von Reports:

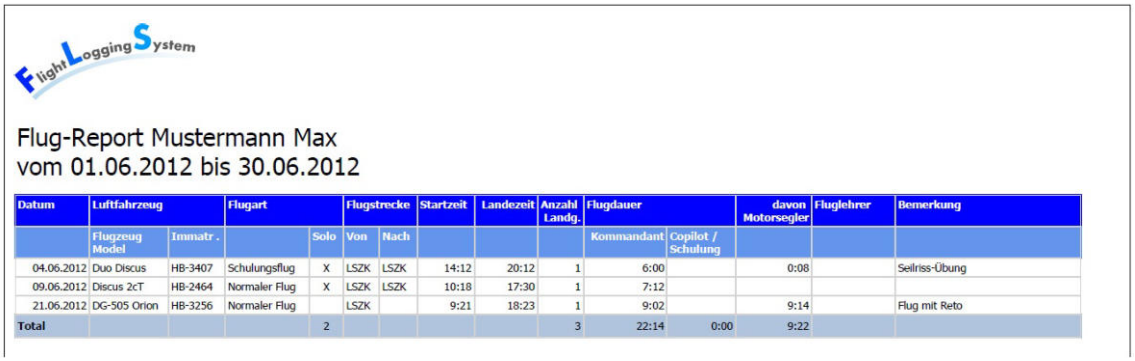
1. Monatsreport für die Rechnungsstellung: Dieser Report enthält die Informationen aller Flüge eines Vereins über den ausgewählten Monat. Jeden Monat erhält das Sekretariat ein E-Mail mit diesem Report und den Daten aus dem letzten Monat.
2. Monatsreport für BAZL (Abbildung 1.4: „Monatsreport für BAZL“): Dieser Report enthält eine Zusammenfassung mit der Anzahl der Flüge pro Standort.

3. Flugreport für Piloten (Abbildung 1.5: „Flugreport“): Dieser Report enthält alle Flüge eines Piloten über den entsprechenden Zeitraum. Jeden Tag erhalten die Piloten in einem E-Mail einen solchen Report dieses Tages.

Flugsportgruppe Zürcher Oberland (FGZO)
Zusammenfassung für BAZL vom Juni

Anzahl Starts				Anzahl Landungen
Flugzeugschlepp	Windenstart	Eigenstart	Total von LSZK	in LSZK
3	0	3	6	5

Abbildung 1.4: Monatsreport für BAZL



Flug-Report Mustermann Max
vom 01.06.2012 bis 30.06.2012

Datum	Luftfahrzeug		Flugart	Flugstrecke			Startzeit	Landezeit	Anzahl Landg.	Flugdauer		davon Motorsegler	Fluglehrer	Bemerkung
	Flugzeug Model	Immatr.		Solo	Von	Nach				Kommandant	Copilot / Schulung			
04.06.2012	Duo Discus	HB-3407	Schulungsflug	X	LSZK	LSZK	14:12	20:12	1	6:00		0:08		Seilriss-Übung
09.06.2012	Discus 2cT	HB-2464	Normaler Flug	X	LSZK	LSZK	10:18	17:30	1	7:12				
21.06.2012	DG-505 Orion	HB-3256	Normaler Flug		LSZK		9:21	18:23	1	9:02		9:14		Flug mit Reto
Total				2					3	22:14	0:00	9:22		

Abbildung 1.5: Flugreport

1.2.4 Architektur

Die Architektur vom FLS besteht aus einer Silverlight RIA Applikation, welche im Webbrowser gestartet wird und über WCF-Services und das Entity Framework auf die Datenbank zugreift.

Im Client wird das MVVM-Pattern möglichst konsequent angewendet. Damit soll eine hohe Testbarkeit erreicht werden.

WCF RIA Services

Es wurde der WCF RIA Service verwendet, welcher die Netzwerkfunktionen mit einer stabilen Verbindung zum Server anbietet.In Silverlight können Domain-Service-Klassen erstellt werden, wodurch unter anderem einfach CRUD-Operationen für die Entities generiert und weiter bearbeitet werden können.

WCF RIA Service generiert automatisch Code für die Entities und die Services im Client-Projekt. Dieser wird beim Starten der Applikation im Browser über das Xap-File geladen. Für die DomainServices im Server werden im Client DomainContexte generiert, die die Schnittstelle zum Server bereitstellen. Die Entity-Klassen, welche den Datenbank-Tabellen entsprechen, werden ebenso bereitgestellt. Die geladenen Daten können damit von den DomainContexten verwaltet werden.

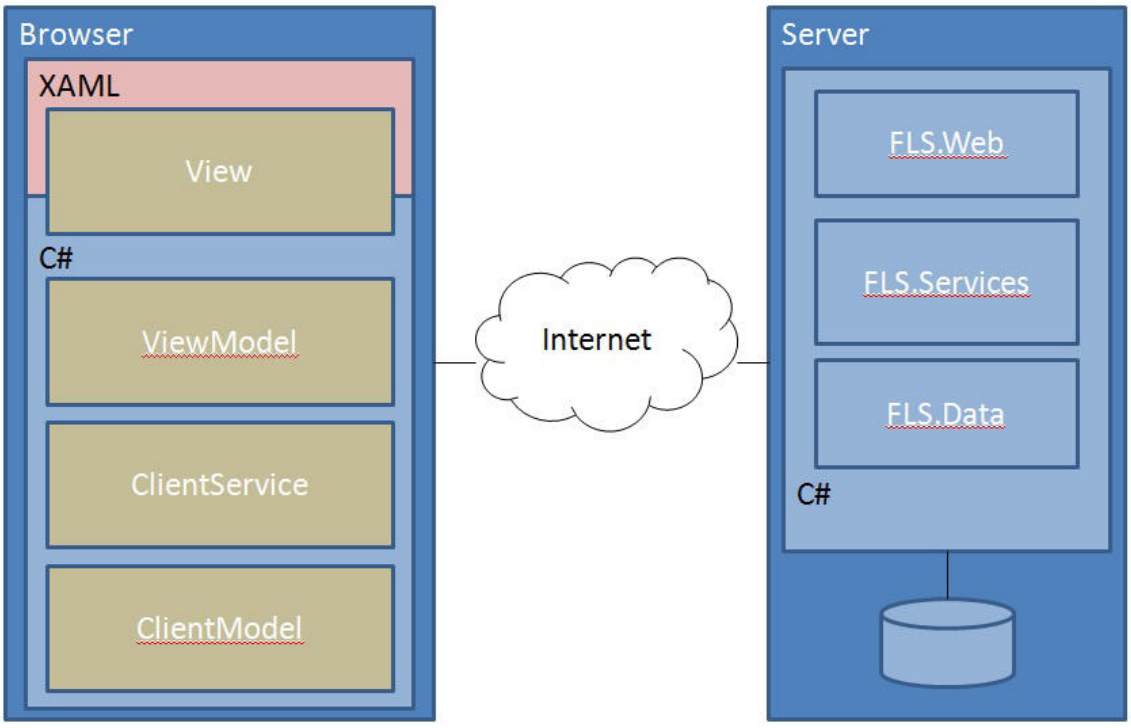


Abbildung 1.6: Architektur

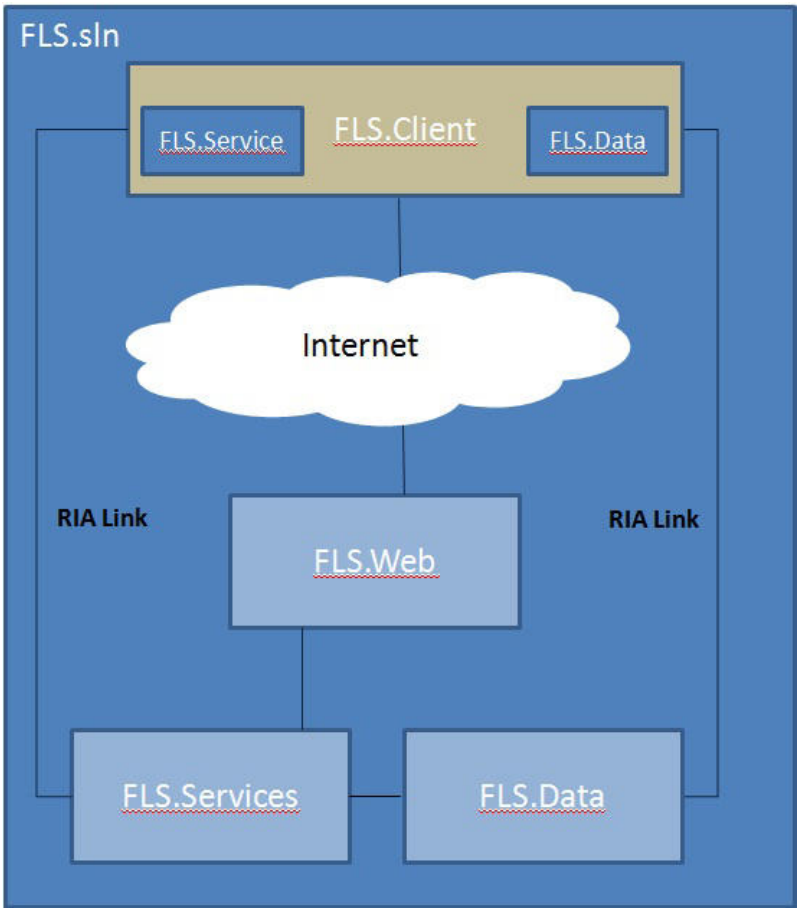


Abbildung 1.7: Architektur

Authentifizierung

Für die Authentifizierung wurde ein Domain Service erstellt und der MembershipProvider aus System.Web.Security erweitert.

Der Authentifizierungsservice von .NET erstellt eine eigene Datenbank, in welcher die Benutzer erstellt und verwaltet werden können. Diese Datenbank wurde jedoch in diesem Projekt nicht verwendet. Die Überprüfung, ob ein Login existiert und das Passwort übereinstimmt, wurde auf die eigene Datenbank umgeleitet. Damit liess sich vermeiden, dass zwei Datenbanken verwaltet werden müssen.

Autorisierung

Für die Autorisierung wurde in der Datenbank eine Tabelle Permissions erstellt. In dieser Tabelle können die kleinstmöglichen Einheiten der Berechtigung definiert werden (z.B createUser, readUser usw.). Den Permissions sind Rollen zugeordnet, die an Benutzer vergeben werden können. Die voreingestellten Rollen sind „Verein Administrator“, „System Administrator“ und „Flugdienstleiter“.

Prism

Da für die Applikation noch einige Erweiterungen geplant sind (siehe Management Summary), wurde die Prism-Library verwendet, um eine zukünftige Modularisierung der Applikation vorzubereiten. Dafür wurde die Klasse Bootstrapper vom MefBootstrapper abgeleitet. Um die Modularisierung jedoch definitiv umzusetzen, müsste das aktuelle Client-Projekt vom Bootstrapper separiert werden.

Die Dependency Injection von Prism wurde verwendet, um sicherzustellen, dass immer nur eine Instanz von den Views, ViewModels und ClientModels erstellt wird. Damit wird das mehrmalige Laden der Daten vom Server verhindert.

Datenaustausch zwischen Server und Client

Grundsätzlich werden alle Daten vom Server geladen, ausgenommen die Flugdaten, da deren Anzahl innerhalb kürzester Zeit stark wachsen wird. Ausserdem werden diese nicht für andere Arbeiten mit der Applikation benötigt. In der Datenbank wurde dazu ein Feld definiert, über das bestimmt werden kann, wie viele Tage zurück die Flugdaten geladen werden.

Die Daten werden automatisch nur einmal pro Client geladen, wenn dieser die Applikation im Browser startet. Änderungen werden in den DomainContext im Client vermerkt und beim Speichern an den Server übermittelt.

Implementation Reporting

Für das Erstellen der Reports wurde der SQL Server Reporting Server von Microsoft benutzt. Dazu wurde eine eigene Solution erstellt, welche über das Business Intelligence Development Studio von Visual Studio 2008 bearbeitet wird. Als Datenquelle dient die FLS-Datenbank, auf die mit Hilfe eines Connection-Strings zugegriffen wird. Die Datenbank wurde im Report-Projekt als geteilte Datenquelle konfiguriert, um allen Reports Zugriff zu geben. Für jeden Report wurde ein entsprechendes Query geschrieben, welches aus mitgegebenen Parametern die gewünschten Daten zurück gibt. Die Daten werden im Report für die Darstellung je nach Bedarf gruppiert und teilweise noch über eine Visual Basic Expression aufbereitet oder über eine TextBox-Property speziell formatiert.

Implementation Import/Export

Bei der Implementierung des Imports und Exports wurde darauf geachtet, dass ein Grossteil der Verarbeitung im Server erfolgt. Trotzdem dauert die Verarbeitung bei einer grossen Menge von Daten schnell einige Sekunden, da die Daten über das Netzwerk verschoben werden müssen.

Für die Implementation der Verarbeitung im Server wurde mit Interfaces und abstrakten Klassen gearbeitet. So kann bei Erweiterungen um weitere Entitäten ein Grossteil der Funktionalität verwendet werden und nur einzelne Methoden müssen zusätzlich implementiert werden.

Implementation E-Mail Versand

E-Mails werden sowohl für das Versenden der Reports, wie auch für das Zuschicken von Passwörter an Benutzer verwendet.

Design Namespaces

Die einzelnen Schichten von FLS werden in Namespaces aufgeteilt. Dabei stellt jede Schicht ein eigenes Visual Studio-Projekt dar. In der Abbildung 1.8: „Übersicht Namespaces“ ist eine Übersicht über die Namespaces abgebildet. Der Namespace FLS.Client repräsentiert die Applikation, die in den Client geladen wird. FLS.Web, FLS.ExportImportService, FLS.ExportImport, FLS.Service und FLS.Data wird auf dem Server benötigt.

Im Namespace FLS.Client ist mit den Sub-Namespaces Views und ViewModels Umsetzung des MVVM Patterns ersichtlich. Weiter befinden sich dort ClientServices, welche für die Manipulationen verantwortlich sind und die ClientModels, die für das Laden und Zwischenspeichern der Daten zuständing sind. Die Converters werden für das Umwandeln von Werten zwischen View und ViewModel benötigt. Unter Resources liegen Ressourcen mit String-Konstanten und Texte die dem Benutzer angezeigt werden.

Der Namespace FLS.Web ist der Einstiegspunkt für den Client. Er enthält das Xap-

File, welches der Browser, bei einem Zugriff vom Client auf den Server, lädt. Der Namespace enthält auch die Authentifizierung und verifiziert den Benutzer beim Einloggen.

Der Namespace FLS.Service beinhaltet die verschiedenen Services, die dem Client zur Verfügung gestellt werden.

Der Namespace FLS.ExportImportService bietet die Import- und Export-Funktionalitäten dem Client an, während im FLS.ExportImport die Verarbeitung erfolgt.

In FLS.Data befindet sich das EntityModel und die Entities.

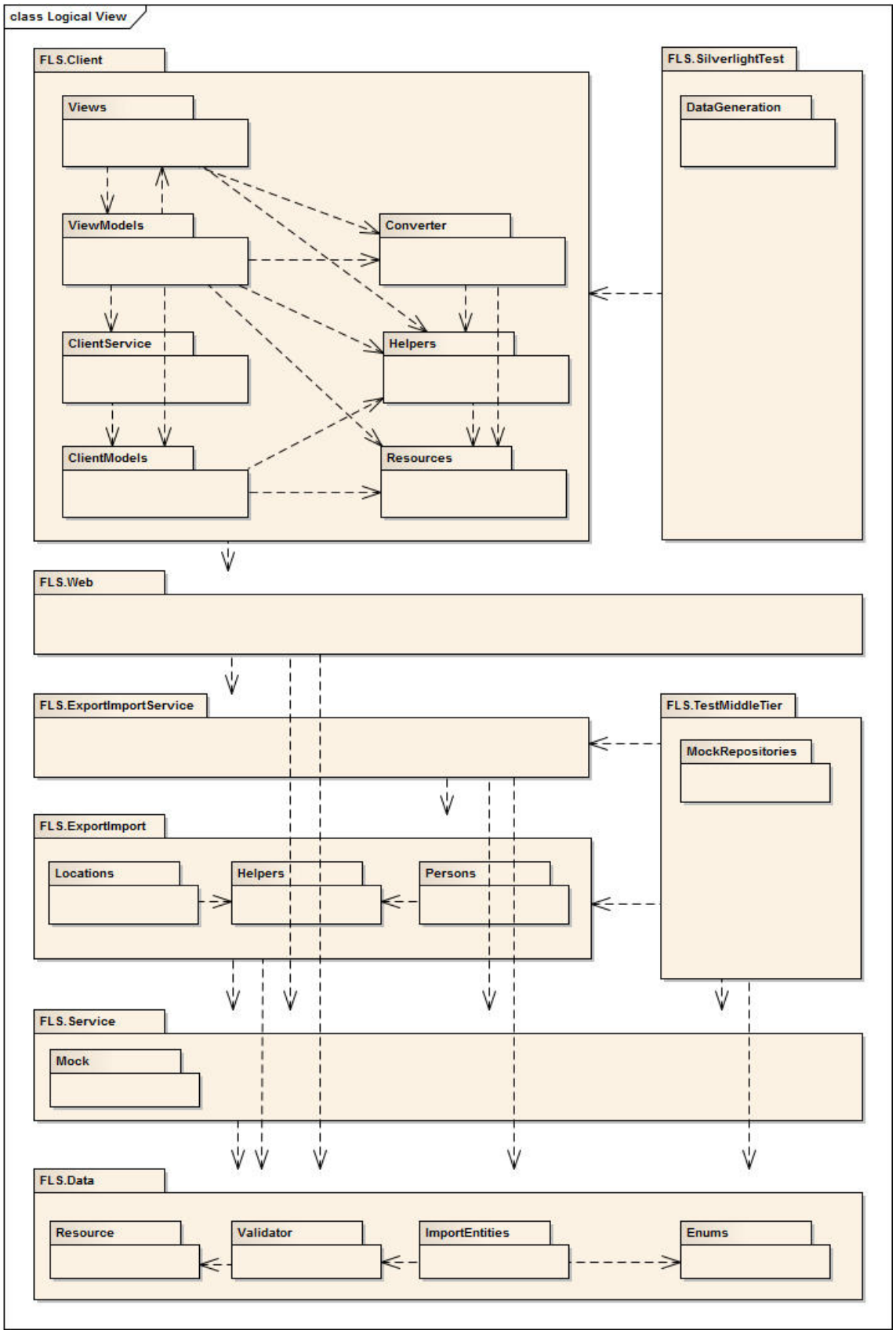


Abbildung 1.8: Übersicht Namespaces

Prozesse und Threads

Alle Aufrufe des WCF RIA Services sind asynchron. Deshalb benutzt WCF RIA Service für die Netzwerkkommunikation einen eigenen Thread, welcher durch das Silverlight-Framework verwaltet wird. Dieser Thread wartet nach Anfragen an den Server auf dessen Antwort und aktualisiert allenfalls die DomainContexte mit den geladenen Daten. Der GUI-Thread kann in dieser Zeit weiterlaufen und auf Eingaben vom Benutzer reagieren.

Ein zusätzlicher Thread aktualisiert im Sekundentakt die Flugdauer im StartlistViewModel.

Für den Report-Versand werden zwei Threads für die Scheduler über die Library Quarz erstellt.Für jede Ausführung des Schedulers nimmt diese einen neuen Thread aus dem Thread-Pool. Der eine Scheduler verschickt die Flug-Reports täglich an die Piloten. Ein weiterer Scheduler sendet monatlich Reports an das Sekretariat mit den Daten fürs BAZL und für die Rechnungsstellung.

Mandantenfähigkeit

Um die Mandantenfähigkeit zu gewährleisten, wurde unterschieden, welche Tabellen für alle Mandanten zur Verfügung stehen und welche für jeden Mandanten individuell sind. Tabellen, deren Daten pro Mandant separat gehalten und gepflegt werden, verfügen über eine Verbindung zu der Tabelle Club.

Tabellen - Mandanten

Prozess	Bemerkung	Tabelle benötigt Mandant
AircraftAircraftStatus		Nein
Aircrafts	zwei Spalten für Owner: Club oder Person	Nein
AircraftStatus		Nein
Countries		Nein
Flight	Mandantenzuordnung wird über Flugtyp bestimmt	Nein
FlightCrew		Nein
FlightTypes	bestimmt, welchem Mandanten der Flug zugeordnet wird	Ja
Club		Nein
Locations		Nein
LocationTypes	statisch, unveränderbar	Nein
MemberStatus	werden pro Mandant verwaltet	Ja
PersonCategories	werden pro Mandant verwaltet	Ja
PersonClub	Zuordnung der Person einem Mandanten, enthält Rating	Nein
PersonMemberStatus	Zuordnung der Person einem Mandanten als Vereinsmitglied	Nein
PersonPersonCategories	Zuordnung einer Person zu einer PersonCategory	Nein
Persons	kann mehreren oder keinem Mandanten zugeordnet sein, über PersonCategories / PersonMemberStatus / PersonClub bestimmt	Nein
Roles		Nein
StartTypes	statisch, unveränderbar	Nein
SystemData		Nein
SystemLogs		Nein
SystemVersion		Nein
User	User kann nur einem Mandanten zugeteilt sein	Ja
UserRoles		Nein

Berechtigungskonzept

Um den Zugriff auf die Daten zu kontrollieren und zu beschränken, wurde ein Berechtigungskonzept erstellt. Dieses zeigt die Berechtigungen auf den CRUD-Operationen (Create, Read, Update, Delete) für die einzelnen Rollen. Dabei werden drei Berechtigungstypen unterschieden: FlightOperator, ClubAdministrator und SystemAdministrator. Diese Typen werden weiter unterteilt nach Berechtigung für den eigenen Mandanten oder für alle Mandanten.

Berechtigungskonzept

Prozess	Bemerkung	CRUD-Operation	FlightOperator eigener Mandant	FlightOperator fremder Mandant	ClubAdministrator eigener Mandant	ClubAdministrator fremder Mandant	Systemadministrator eigener + fremder Mandant
Aircraft		Create	X		X		X
		Read	X	X	X	X	X
		Update	X		X		X
		Delete	X		X		X
Country		Create					X
		Read	X	X	X	X	X
		Update					X
		Delete					X
Flight		Create	X		X		X
		Read	X		X		X
		Update	X		X		X
		Delete	X		X		X
Location		Create			X	X	X
		Read	X	X	X	X	X
		Update			X	X	X
		Delete			X	X	X
LocationType	**muss nicht in Applikation bearbeitet werden**	Create					
		Read	X	X	X	X	X
		Update					
		Delete					
Mandant + FlightType		Create					X
		Read	X	X	X	X	X
		Update			X		X
		Delete					X
Person	**- * nur Adresse + Rating sollen mandanten-übergreifend sichtbar sein, jedoch keine TelNr. etc -** Ein Flugoperator kann eine Person erstellen, diese als Mitglied dem eigenen Mandaten hinzufügen kann aber nur der ClubAdministrator + Systemadministrator - Rating von Person kann pro Club unterschiedlich sein - Jeder Benutzer kann die eigene Person lesen und bearbeiten**	Create	X**		X		X
		Read	X	(X)*	X	(X)*	X
		Update	X		X		X
		Delete	X		X		X
StartType	**muss nicht in Applikation bearbeitet werden**	Create					
		Read	X	X	X	X	X
		Update					
		Delete					
User		Create			X		X
		Read	X	X	X	X	X
		Update			X		X
		Delete			X		X

Datenspeicherung

FLS benutzt eine Datenbank, welche auf einem Microsoft SQL-Server 2008 R2 Express läuft. Diese Datenbank wurde von Auftraggeber Patrick Schuler erstellt. Während des Projekts wurden gewisse Anpassungen vorgenommen, um weitere Anforderungen erfüllen zu können.

1.2.5 Offene Punkte

Die Umsetzung gestaltete sich in vielen Punkten einiges umständlicher als zu Beginn erwartet. Die fehlende Erfahrung des Entwicklungsteams wirkte sich ebenso stark aus wie die Komplexität des Frameworks .NET. Trotzdem wurden die gemäss Anforderungsspezifikation geplanten Use Cases umgesetzt, soweit sie nicht als optional eingestuft waren.

Einzig die Verwaltung von Wartungen und Störungen wurde innerhalb des Projekt aus zeitlichen Gründen nicht umgesetzt. Diese Funktionalität wurde auch in der Planung mit letzter Priorität eingestuft.

In der Aufgabenstellung waren einige Aufgaben enthalten, bei denen schon zu Beginn des Projekt entschieden wurde, dass diese erst in einer weiteren Arbeit umgesetzt werden können und den Rahmen dieser Bachelorarbeit sprengen würden.

Während des Projekt ergaben sich Probleme mit der Validierung. Diese wurde in der Startlisten-Erfassung nicht zufriedenstellend umgesetzt, da das Verhalten des Framework nicht nachvollziehbar war. Ein Work-Around wurde bei der Personenverwaltung umgesetzt, aus zeitlichen Gründen jedoch nicht mehr in der Startlisten-Erfassung.

1.3 Schlussfolgerungen

Im Verlaufe dieser Arbeit entstand ein nützliches Programm, welches bereits produktiv eingesetzt werden kann. Die grundlegenden Funktionalitäten für eine effiziente Arbeitsweise bestehen.

Die Menge an kleinen Funktionalitäten in dieser Applikation, wie zum Beispiel die Verwaltung von Flugzeugstatus und Mitgliederstatus sowie vereinsspezifische Voreinstellungen, verbrauchte mehr Zeit als geplant. In den letzten Wochen des Projekts wurde deshalb mehr gearbeitet, um die letzten Implementationen und das Bugfixing umzusetzen. Hätte man die Funktionalität jedoch gekürzt, wäre nur eine Applikation mit reduzierter Funktionalität entstanden. So entschied das Team, zusätzliche Zeit zu investieren, um die Anforderungen

aus den Use Cases umzusetzen. Gekürzt wurde vor allem bei den Teilen der Implementierung, welche für eine schnelle Erweiterbarkeit der Applikation notwendig gewesen wäre, wie zum Beispiel die Plug-In-Architektur oder die Modularisierung.

Die Anwendung wird voraussichtlich in einer nächsten Studien- bzw. Bachelorarbeit weiterentwickelt. Trotzdem ist die Applikation einsatzfähig und kann die Arbeit auf dem Flugplatz vereinfachen.

VI

Persönliche Berichte

Datum: 15. Juni 2012

2 Erfahrungsbericht Marion Frei

2.1 Projektverlauf

Nach dem Besuch des Moduls Microsoft-Technologien, welches Lust auf mehr .NET-Technologie machte, freute ich mich darauf, die dort kennengelernten und angeschnittenen Technologien anwenden zu können und mein Wissen darüber zu vertiefen.

Da dies ausser dem Miniprojekt des Microsoft-Technologie Moduls mein erstes .NET-Projekt war, waren mir viele der angewandten und vom Auftraggeber gewünschten Technologien noch nicht geläufig. So konnte ich bisher nur wenig Erfahrung mit Silverlight sammeln. Mit RIA-Services hatte ich zuvor noch nie gearbeitet und Prism war mir völlig unbekannt. Dieser Umstand führte von Anfang an zu einer hohen Einarbeitungszeit um die Technologien besser kennen zu lernen. Dies wurde dadurch etwas abgefangen, dass wir bereits ab der zweiten Woche mit der Implementation anfingen und so früh erste Konzepte ausprobieren und umsetzen konnten.

Das ganze Projekt war aufwändiger als angenommen. So wurde knapp doppelt so viel Zeit als ursprünglich geplant in die Implementation der Stammdatenverwaltung und Flugerfassung investiert. Auch ich hatte den Aufwand, eine Stammdatenverwaltung zu realisieren komplett unterschätzt und dort sehr viel Zeit investiert. Vor allem die kleinen aber feinen Details waren sehr zeitaufwändig und konnten trotz des grossen Aufwands nicht immer ganz wie gewünscht umgesetzt werden. Darunter litt leider das Reviewing und Refactoring des Codes wofür mir leider fast keine Zeit blieb. Um trotzdem eine benutzbare und nützliche Applikation abzuliefern haben wir die Wochen-Soll-Zeit angehoben.

2.2 Zusammenarbeit im Team

Nachdem ich meine Studienarbeit alleine bearbeiten musste, da mein Teampartner die Anforderungen dafür nicht erfüllte, habe ich mich umso mehr darauf gefreut, meine Bachelorarbeit wieder im Team bearbeiten zu können. Damals fehlten mir vor allem der Austausch und auch die Unterstützung bei Problemen.

Mit meinen zwei Teampartnerinnen habe ich bereits ein kleines Projekt realisiert, welches innerhalb des Software-Engineering Moduls 2 stattfand. Deshalb habe ich mich natürlich sehr gefreut, die Bachelorarbeit wieder mit diesen zwei Freundinnen bearbeiten zu dürfen. Die Zusammenarbeit war immer sehr angenehm und von gegenseitigem Respekt

gekennzeichnet. Wir haben die meiste Zeit gemeinsam in dem von der Schule zur Verfügung gestelltem Raum gearbeitet. Dies ermöglichte einen ständigen und fliessenden Austausch von Informationen, Ideen und Hilfestellungen wobei auch der Spass nicht zu kurz kam.

2.3 Fazit

Obwohl wir den Zeitaufwand für das Projekt deutlich unterschätzt haben und einige Funktionen streichen mussten, hat die Umsetzung des FLS und das Kennenlernen der Technologien auch sehr viel Spass gemacht. Ein Highlight des Projekts war sicher der Besuch des Flugplatzes Speck, wo ich die Gelegenheit erhalten haben, mit Patrick Schuler einen kurzen Segelflug zu unternehmen.

3 Erfahrungsbericht Lara Mühlemann

3.1 Projektverlauf

Ich habe mich für dieses Projekt entschieden, da ich das .NET-Framework und C# näher kennen lernen wollte. Bisher kannte ich diese Technologien nur aus der Vorlesung Microsoft Technologien und Silverlight habe ich kurz in der Vorlesung Internettechnologien kennen gelernt. Dementsprechend schwierig war es für mich den Umfang der Aufgabenstellung abzuschätzen. Beim Erhalt der Aufgabenstellung dachte ich, dass der Aufwand für diese Arbeit nicht sehr gross ist, da es vor allem um die Umsetzung von CRUD-Operationen ging. Um so ernüchternder war es, als bei der Einarbeitung und der Umsetzung so viele Probleme auftraten. Vor allem die Einarbeitung in Prism empfand ich als sehr schwierig.

Im Verlaufe des Projektes musste ich merken, dass die Umsetzung von kleinen Details ebenfalls ein sehr grosser Aufwand war. Insbesondere die kleinen Probleme bei der Verwendung der xaml-Elemente brauchten Zeit.

Durch die vorher erwähnten Zeitverluste befanden wir uns gegen Ende des Projektes unter starkem Zeitdruck. Dieses Problem konnten wir abschwächen, in dem wir mit Absprache des Auftraggebers gewisse Features und einen niedrig priorisierten UseCase weglassen konnten. Zusätzlich haben wir unsere Soll-Planung erhöht um trotzdem eine einsatzfähige Applikation entwickeln zu können.

Ich hätte mir gewünscht, dass wir mehr Zeit in Refactoring und Review hätten investieren können. Insbesondere die dafür geplante Zeit in der Transition musste für andere Arbeiten verwendet werden.

3.2 Zusammenarbeit im Team

Ich habe bereits früher Projekte mit den beiden Teammitglieder durchgeführt und darum waren wir bereits eingespielt. Dadurch dass wir alle die meiste Arbeit direkt an der HSR im selben Raum erledigt haben, konnte einander bei Problemen schnell geholfen werden, was sehr nützlich war. Allerdings störten zu viele Unterbrechungen zum Teil auch den eigenen Gedankenfluss. Aber grundsätzlich haben wir in diesem Bereich ein gutes Mittelmass gefunden.

Über die wöchentlichen Sitzungen konnten die anstehenden Arbeiten gut koordiniert

werden. Zum Teil empfanden wir zusätzlich innerhalb der Woche einen kurzen Austausch des Arbeitsstandes als nützlich.

3.3 Fazit

Trotz der vielen Probleme konnten wir eine nützliche und einsatzfähige Applikation entwickeln und ich habe im Verlaufe dieser Arbeit sehr viel Neues gelernt. Es ist sehr schön zu sehen, dass auch wenn in einem Projekt Probleme auftauchen, ein gutes Resultat erreicht werden kann. Insbesondere gefällt mir, dass wir uns in die vielen neuen Technologien einarbeiten und diese anwenden konnten.

4 Erfahrungsbericht Marion Walser

4.1 Projektverlauf

Das Projekt war ein sich Einlassen auf viel Unbekanntes. Wir mussten uns zuerst in den vielen neuen Technologien zurechtfinden und viele neue Konzepte verstehen. Eine gute Einschätzung über den Aufwand, den die Umsetzung benötigen würde, war praktisch unmöglich, da es zu viele unbekannte Faktoren gab. Als Firma hätte man einen solchen Auftrag niemals angenommen. Die Kenntnisse des Entwicklerteam über .NET und Silverlight waren nicht ausreichend, um mit einem konkurrenzfähigen Angebot sicherstellen zu können, dass das Projekt kostendeckend werden würde. So war mir bewusst, dass wir viel Zeit benötigen werden, uns in die Materie einzuarbeiten und ein grosser Anteil unter Lernen fallen würde.

Ich hoffte eigentlich, dass nach ein paar Wochen Einarbeitung und Implementierung das Verständnis für die Konzepte da sei. Jedoch stellte sich heraus, dass bei jeder neuen Anforderung an das User Interface oder an die Datenverwaltung wieder ein neues Konzept erarbeitet werden musste.

Deshalb lief es nur zwischendrin mal richtig rund, bis ich wieder an ein nächstes Problem stiess, das mit viel Ausprobieren und Nachlesen im Internet gelöst werden musste.

4.2 Zusammenarbeit im Team

Wie bei den letzten zwei Arbeiten, SE2-Projekt und der Studienarbeit, war die Zusammenarbeit einfach toll. Jeder setzte sich ein, übernahm Verantwortung und war fähig, Probleme selbstständig zu lösen. Trotzdem haben wir uns immer gegenseitig geholfen. Zu zweit war es öfters effizienter, ein Konzept zu erarbeiten, da jeder ein unterschiedliches Verständnis der Sache hatte.

Auch die internen Sitzungen waren zielgerichtet. Jeder leistete seinen Beitrag. Bei der Verteilung der Arbeit gab es nie ernsthafte Diskussionen. Alle brachten Themen und Lösungsvorschläge ein, die zu diskutieren waren. Die Diskussionen verliefen immer sehr sachlich und lösungsorientiert.

Natürlich gab es zwischendurch mal klare Worte oder etwas Kritik. Dies wurde aber von allen nicht persönlich genommen, sondern als Feedback und Verbesserungsvorschlag

entgegengenommen.

4.3 Fazit

Auch wenn das Projekt sehr schwierig war, eine grosse Herausforderung bedeutete und einen sehr grossen Einsatz verlangt hat, bereitete es mir viel Freude. Die Probleme zu lösen und am Schluss gelöst zu haben, gab mir immer wieder die Motivation, auch die nächste Aufgabe anzugehen.

Am Schluss hatten wir leider etwas wenig Zeit, alles zu bereinigen und alle Bugs zu beheben. Trotzdem konnten wir eine brauchbare Anwendung abliefern, die bereits im Echt-Betrieb verwendet werden kann und einige manuelle Arbeit ersetzt.

VII

Projektplan

Datum: 15. Juni 2012

5 Projekt Übersicht

In einer Fluggruppe muss der Flugdienstleiter sämtliche Flüge erfassen. Mit diesen Daten werden periodisch Rechnungen an die Piloten gestellt. Zu statistischen Zwecken müssen diese Daten monatlich an das BAZL (Bundesamt für Zivilluftfahrt) gesendet werden. Auch müssen die Piloten ein Flugbuch führen, in welchem sie alle Flüge täglich eintragen müssen.

Viele dieser Arbeiten werden heute manuell erledigt. Dabei werden Daten von einem Formular in ein anderes übertragen, wobei sich schnell Fehler einschleichen können.

Das Flight Logging System soll den manuellen Aufwand auf ein Minimum reduzieren. So soll der Flugdienstleiter die Flüge direkt im System erfassen. Am Monatsende sollen die Daten für die Rechnungen automatisch generiert werden. Piloten erhalten die Möglichkeit, sich über ein Web-Interface einzuloggen, um ihre Flüge zu überprüfen und Reports zu erzeugen.

5.1 Zweck und Ziel

Das Ziel dieser Bachelorarbeit ist es, eine mandantenfähige Client- / Serverapplikation zu entwickeln, welche über einen modularen Aufbau und eine ausbaufähige Architektur verfügt. Die Applikation soll Flüge und Stammdaten erfassen und Reports erstellen können.

5.2 Projektdauer und Abgabetermin

Das Projekt beginnt am 20.02.2012 und endet am 15.06.2012, um 12:00 Uhr, womit 17 Wochen zur Verfügung stehen. Ca. eine Woche vor Projektende müssen die Systemtests abgeschlossen sein.

6 Projektorganisation

Das Team besteht aus 3 Personen, die gleichberechtigt sind. Probleme werden im Team diskutiert und Entscheidungen gemeinsam gefällt.

Jedes Team-Mitglied ist neben der Entwicklung noch für eine Iteration des Projekts verantwortlich. Diese Verantwortlichkeit besteht aus der Organisation und Koordination.

6.1 Organisationsstruktur

Abbildung 6.1: Projekt-Team: v.l.n.r Marion Frei, Lara Mühlemann, Marion Walser

Die folgende Tabelle zeigt, welches Teammitglied in welcher Phase die Projektleitung übernimmt und damit die Verantwortlichkeit über die Phase hat.

Projektmitglied	Verantwortlichkeit
Marion Frei	Inception, Construction 1
Lara Mühlemann	Elaboration 1, Construction 2
Marion Walser	Elaboration 2, Transition

Informationen, Fragen oder Aufträge sollen immer an alle Projektmitglieder gerichtet werden. Allfällige Arbeiten oder Abklärungen werden an der nächsten Besprechung zugeteilt, beziehungsweise besprochen.

6.2 Externe Schnittstellen

Name	Mail-Adresse	Rolle
Patrick Schuler	-	Auftraggeber, Betreuer Extern
Hansjörg Huser	-	Betreuer HSR
Marion Frei	m1fei@hsr.ch	Projektmitglied
Lara Mühlemann	lmuehlem@hsr.ch	Projektmitglied
Marion Walser	mwalser@hsr.ch	Projektmitglied

7 Management Abläufe

7.1 Projekt Kostenvoranschlag

Für die Bachelorarbeit stehen 17 Wochen zur Verfügung. Dazwischen liegen einzelne Feiertage. Das Projekt dauert vom 20.02.2012 bis zum 15.06.2012. Innerhalb dieser Zeitspanne arbeitet jedes Teammitglied etwa 360 Stunden am Flight Logging System.

7.2 Projektplan

7.2.1 Artefaktplan

Phase	Inception	Elaboration 1			Elaboration 2			Construction 1			Construction 2			Transition
Zeitabschnitt beginnend am	20.02.12	27.02.12	05.03.12	12.03.12	19.03.12	26.03.12	02.04.12	09.04.12	16.04.12	23.04.12	30.04.12	07.05.12	14.05.12	21.05.12	28.05.12	04.06.12	11.06.12
Projektwoche	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Projekt Management																	
Projektplan		v1.0			v2.0			v3.0				v4.0				v2.0	v3.0
Zeitplan		v1.0														v5.0	v6.0
Iterationspläne inkl. Risikomanagement		1			2			3				4				5	
Sitzungsprotokolle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Glossar					v1.0												v2.0
ToDo und Bug-Liste																	v1.0
Requirements																	
Anforderungsspezifikation			v1.0		v2.0			v3.0				v4.0				v5.0	v6.0
Paper Prototypen					v1.0			v2.0									
Business Modeling																	
Domainanalyse						v1.0		v2.0				v3.0					v4.0
Architektur + Design																	
System Architecture Document												v1.0				v2.0	v3.0
Implementation																	
Prototyp					v1.0			v2.0				v3.0				v4.0	
FLS																	v1.0
Test + Qualität																	
Systemtests								v1.0				v2.0				v3.0	v4.0
Code-Reviews								v1.0				v2.0					v3.0
Abschlussbericht/-dokumente																	
Aufgabenstellung																v1.0	v2.0
Abstract																v1.0	v2.0
Management Summary																v1.0	v2.0
Technischer Bericht																v1.0	v2.0
Persönliche Berichte																v1.0	v2.0
Literaturverzeichnis																v1.0	v2.0
Kurzfassung Studienarbeit																v1.0	v2.0
Poster																v1.0	v2.0

7.2.2 Beschreibung Artefakte

Artefakt	Beschreibung
Abstract	Abriss des Projekts, richtet sich an den Spezialisten. *
Anforderungsspezifikation	Detaillierte Auflistung und Beschreibung der funktionalen und nicht funktionalen Anforderungen.
Aufgabenstellung	Detaillierte Aufgabenstellung zur Bachelorarbeit.
Code-Reviews	Dokumentation der Ergebnisse aus den Code-Reviews.
Domainanalyse	Beschreibung der Domäne mittels Darstellung der Konzepte in einem Domain Model. Use Cases werden mittels Sequenzdiagrammen dargestellt, Kontakte der wichtigsten Funktionen definiert.
FLS	Version 1.0 der FLS-Software
Glossar	Auflistung der projektspezifischen Begriffe mit Erklärungen.
Iterationsplan	Definition und Einteilung der Aufgaben einer Iteration inkl. Neubewertung der Risiken zu Beginn der Iteration.
Literaturverzeichnis	Auflistung der verwendeten Quellen. *
Management Summary	Kompakte Darstellung des Projekt auf abstrakter Ebene. *
Paper Prototypen	Skizzen der Benutzeroberflächen.
Persönliche Berichte	Bericht jedes Projektmitglied über die Erfahrungen bei der Arbeit.
Poster	Zusammenfassung der Arbeit auf einem Poster. *
Projektplan	Beschreibung über den geplanten Ablauf des Projekts.
Prototyp	Lauffähige FLS-Software
Risikomanagement	Aufstellung über die Risiken des Projekts, wird in den Iterationsplänen erstellt bzw. aktualisiert.
Sitzungsprotokoll	Dokumentation der Sitzung mit Entscheidungen und Beschlüssen.
System Architecture Document	Beschreibung und Darstellung der Architektur, Namespaces und des Klassendesign.
Systemtests	Beschreibung und Auswertung der Systemtests.
Technischer Bericht	Ergebnisse und Schlussfolgerungen aus dem Projekt. *
ToDo-Liste und Bug-Liste	Liste der ToDos und Bugs mit Erstelldatum und Erledigt-Datum.
Zeitplan	Zeitplanung des Projektes mit Arbeitspaketen, Soll- und Ist-Stunden. Der Zeitplan wird jeweils anfangs Iteration mit den Soll-Zeiten für die Iteration ergänzt.

* Diese Berichte enthalten Redundanz zu anderen Dokumenten.

7.2.3 Zeitplan

Detailplan — **Flight Logging System**

Ferien — F

Phasen — vor Projektstart | Inception | Elaboration 1 | Elabo

Meilensteine — MS | MS I | MS II

Woche (von - bis)				SW 0						SW 1						SW 2						SW 3						SW 4						SW 5						SW
				13.02				17.02		20.02				24.02		27.02				02.03		05.03				09.03		12.03				16.03		19.03				23.03		26.03		
				wals		frei		mueh		wals		frei		mueh		wals		frei		mueh		wals		frei		mueh		wals		frei		mueh		wals		frei		mueh		wals		fr
Arbeitspaket	**Geschätzt**	**SOLL**	**IST**	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S
Projekt Management	**203.0**	**201.5**	**197.5**	0.0	0.0	0.0	0.0	0.0	0.0	7.5	9.0	14.0	13.5	14.0	12.5	4.0	6.5	6.0	10.5	7.0	9.0	3.0	5.0	3.0	4.0	2.5	4.5	2.5	5.0	3.0	4.0	3.0	4.5	6.5	5.5	5.0	5.0	5.0	3.5	4.0	5.5	4.0
Projektplan / Zeitplan	40	37.5	37.0							6.0	7.5	12.0	11.5	10.0	9.5	1.0		3.0	3.5	3.0							0.5							1.0	1.5							
Iterationspläne / Risikomanagement	10	6.0	12.0											2.0	1.0	0.5	2.0		0.5	1.0	3.0													0.5	0.5							
Besprechungen im Team	55	39.0	18.0													1.0		1.0	1.0	1.0	0.5	1.0		1.0		1.0		1.0	1.0	1.0	0.5	1.0	0.5	1.0		1.0	1.0	1.0	0.5	1.0	1.0	1.0
Besprechungen mit Betreuer/Auftraggeber	60	117.5	130.0							1.5	1.5	1.5	2.0	2.0	2.0	1.5	4.5	2.0	5.5	1.5	5.0	2.0	5.0	1.5	4.0	1.5	4.0	1.5	4.0	1.5	4.0	2.0	4.0	4.0	3.5	4.0	4.0	4.0	3.0	3.0	4.5	3.0
Reserven aus Riskmanagement	35	0.0	0.0																																							
Glossar	3	1.5	0.5									0.5								0.5	0.5								0.5												
Qualitätsmassnahmen	**135.0**	**168.5**	**164.5**	0.0	0.0	0.0	0.0	0.0	0.0	5.0	4.0	6.0	9.0	6.0	6.5	5.0	6.5	5.0	4.0	3.0	5.0	2.5	4.5	2.5	0.0	2.5	3.5	4.5	4.0	2.5	5.0	2.5	12.0	3.0	2.5	4.0	3.0	4.0	2.5	3.0	3.0	3.0
Review Code	28	38.0	16.0																									1.0		1.0		1.0					1.0				1.5	
Refactoring	30	23.5	18.0																									2.0														
Einarbeitung in neue Technologien	45	72.5	91.5							5.0	4.0	6.0	9.0	6.0	6.5	4.5	5.5	4.5	2.0	2.5	2.5	2.0	4.0	2.0		2.0	3.0	1.0	4.0	1.0	5.0	1.0	12.0	3.0	2.5	3.0	2.0	3.0		3.0	1.5	3.0
Dokumente überprüfen	32	34.5	39.0													0.5	1.0	0.5	2.0	0.5	2.5	0.5	0.5	0.5		0.5	0.5	0.5		0.5	0.5	0.5					1.0	1.0	2.5			
Requirements	**25.0**	**30.5**	**26.0**	0.0	0.0	0.0	0.0	0.0	0.0	8.0	6.5	0.0	0.0	0.0	0.0	3.0	1.0	1.0	0.0	2.0	2.5	2.0	0.0	0.0	0.0	2.0	7.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Anforderungsspezifikation	10	10.0	7.0							6.0	5.5					1.0		1.0									0.5														
Use Cases	15	20.5	19.0							2.0	1.0					2.0	1.0			2.0	2.5	2.0				2.0	7.0															
Business Modeling	**70.0**	**42.5**	**19.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.5	2.0	3.0	4.0	0.0	4.0	0.0	7.0	5.5	2.0	0.0	1.0	0.0	4.0	1.5	0.0	0.0	0.0	0.0	0.0	0.0	4.0	0.0	0.0
Domain Analyse Dokument	10	8.0	5.5																							2.0	4.5	2.0		1.0		2.0	1.0									
Domain Model	10	8.0	5.5																0.5	2.0	3.0					4.0	1.0					2.0	0.5									
System Sequenz Diagramm	30	18.0	7.0																			4.0		2.0																4.0		
Operation Contracts	20	8.5	1.0																					2.0		1.0																
Analyse und Design	**95.0**	**87.0**	**49.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	1.5	1.0	0.0	3.0	6.0	3.0	3.5	2.0	0.0	3.0
Sequenzdiagramm	15	9.0	2.0																																							
Klassendiagramme	15	11.0	4.5																																							
System Architecture Document	50	52.0	31.0																													2.0	1.5									
Paper Prototype	15	15.0	11.5																															1.0		3.0	6.0	3.0	3.5	2.0		3.0
Implementation	**373.0**	**416.0**	**709.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	6.0	7.0	8.0	6.0	6.0	1.5	8.5	11.0	10.5	21.0	6.0	0.5	9.0	13.0	11.0	12.0	7.0	3.0	11.0	17.0	9.0	8.0	9.0	11.5	8.0	13.5	11.0
Aufbau Architektur	18	37.0	63.0													4.0	5.0	4.0	6.0	4.0	1.0		4.0		2.5	1.0		7.0	10.0	7.0	6.5	6.0	3.0		3.5							
Summary für Rechnungserstellung	30	30.0	19.0																																							
Flugreport für Piloten	30	50.0	45.0																																							
Mandatenfähigkeit	20	6.0	5.0																															4.0	5.0							
Authentifizierung und Authorisierung von Benutzern	40	43.0	44.5														2.0						2.0		10.5	3.0															
Stammdatenverwaltung	70	133.5	255.0													2.0		4.0				8.5	5.0	10.5	8.0		0.5	2.0	3.0	3.0	5.5			7.0	8.5	9.0	8.0	9.0	11.5	8.0	9.0	7.0
Flugerfassung	45	43.5	111.5																																							
Schadenerfassung inkl. Report	45	0.0	0.0																																							
Schnittstelle für Synchronisation von Adressdaten	25	0.0	15.0																																							
Startlisten-Export für BAZL	20	38.0	4.5																																							
Datenbank-Anpassungen	10	11.0	23.5																	2.0	0.5					2.0				1.0		1.0								4.5	4.0
Allgemeine Implementationen	20	24.0	123.0																																							
Konfigurationsmanagement	**5.0**	**2.0**	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Projektautomation	5	2.0	0.0													2.0																										
Tests	**135.0**	**136.5**	**171.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	2.0	0.0	3.0	0.0	3.0	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	1.0	0.0
Systemtests	20	34.5	23.5																													3.0										
Unit Tests	80	61.0	70.0																								2.0			3.0										1.0	
Bugfixing	25	38.0	71.0																									2.0									1.0				
Usability Tests	10	3.0	6.5																																							
Dokumentation und Deployment	**44.0**	**46.5**	**88.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Schlusspräsentation	10	18.0	13.0																																							
Abgabe vorbereiten	10	12.0	10.5																																							
Schlussberichte	12	12.5	61.0																																							
Bedienungsanleitung	12	4.0	3.5																																							
Wochen Subtotal				0.0	0.0	0.0	0.0	0.0	0.0	20.5	19.5	20.0	22.5	20.0	19.0	20.0	21.0	20.0	21.0	20.0	21.0	20.0	20.5	20.0	25.0	20.0	23.5	20.0	22.0	20.5	21.0	21.5	22.5	21.5	25.0	21.0	23.0	21.0	21.0	21.0	23.0	21.0
Total / Wochentotal	**1085.0**	**1131.0**	**1424.0**					0.0	0.0					60.5	61.0					60.0	63.0					60.0	69.0					62.0	65.5					63.5	69.0			
Kontrolle			**0.0**																																							

Auswertung:

	Gesamt		wals		frei		mueh		Diffenenz (Ist-Soll)				Diffenenz (Akkumuliert)			
	SOLL	IST	SOLL	IST	SOLL	IST	SOLL	IST	wals	frei	mueh	Total	wals	frei	mueh	Total
vor Projektstart	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Inception	60.5	61.0	20.5	19.5	20.0	22.5	20.0	19.0	-1.0	2.5	-1.0	0.5	-1.0	2.5	-1.0	0.5
Elaboration 1	182.0	197.5	60.0	63.5	60.5	67.0	61.5	67.0	3.5	6.5	5.5	15.5	2.5	9.0	4.5	16.0
Elaboration 2	189.5	204.5	63.5	75.0	63.0	63.0	63.0	66.5	11.5	0.0	3.5	15.0	14.0	9.0	8.0	31.0
Construction 1	239.0	281.5	83.0	102.5	78.0	93.0	78.0	86.0	19.5	15.0	8.0	42.5	33.5	24.0	16.0	73.5
Construction 2	263.5	417.0	85.5	143.0	85.5	134.5	92.5	139.5	57.5	49.0	47.0	153.5	91.0	73.0	63.0	227.0
Transition	196.5	262.5	65.5	88.0	65.5	82.5	65.5	92.0	22.5	17.0	26.5	66.0	113.5	90.0	89.5	293.0
Total	**1131.0**	**1424.0**	**378.0**	**491.5**	**372.5**	**462.5**	**380.5**	**470.0**	**113.5**	**90.0**	**89.5**	**293.0**	**113.5**	**90.0**	**89.5**	**293.0**

Kontrolle

			SOLL	IST	Plan
Total SOLL	**1131.0**	**Total** wals	378.0	491.5	360
Total IST	**1424.0**	**Total** frei	372.5	462.5	360
		Total mueh	380.5	470.0	360

Detailplan **Flight Logging System**

Ferien F Ostern 6.4 - 10.4

Phasen ration 2 Construction 1

Meilensteine MS MS III MS IV

Woche (von - bis)				W 6			SW 7						SW 8						SW 9						SW 10						SW 11						SW
		Total		ei		30.03	02.04				06.04		09.04				13.04		16.04				20.04		23.04				27.04		30.04				04.05		07.05			fre
						mueh	wals		frei		mueh		wals		frei		mueh		wals		frei		mueh		wals		frei		mueh		wals		frei		mueh		wals		
Arbeitspaket	**Geschätzt**	**SOLL**	**IST**	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S
Projekt Management	**203.0**	**201.5**	**197.5**	5.5	4.0	5.5	4.0	1.5	4.0	1.5	4.0	0.5	4.0	3.0	4.0	4.5	4.0	1.5	4.0	1.0	4.0	1.0	4.0	1.0	4.0	2.0	4.0	2.5	4.0	3.0	4.0	1.0	4.0	2.0	4.0	1.0	3.5	3.0	3.5
Projektplan / Zeitplan	40	37.5	37.0													1.0																						
Iterationspläne / Risikomanagement	10	6.0	12.0											1.0		1.0																				0.5		
Besprechungen im Team	55	39.0	18.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0		1.0		1.0		1.0		1.0	0.5	1.0	0.5	1.0	0.5	1.0		1.0	0.5	1.0	0.5	1.0	0.5	1.0	1.0	1.0	0.5	0.5	0.5	0.5
Besprechungen mit Betreuer/Auftraggeber	60	117.5	130.0	4.5	3.0	4.5	3.0	0.5	3.0	0.5	3.0	0.5	3.0	2.0	3.0	2.5	3.0	1.5	3.0	0.5	3.0	0.5	3.0	0.5	3.0	2.0	3.0	2.0	3.0	2.5	3.0	0.5	3.0	1.0	3.0	0.5	3.0	2.0	3.0
Reserven aus Riskmanagement	35	0.0	0.0																																			
Glossar	3	1.5	0.5																																			
Qualitätsmassnahmen	**135.0**	**168.5**	**164.5**	7.5	3.0	1.5	4.0	2.0	4.0	1.5	4.0	4.0	1.0	2.0	1.0	0.5	1.0	1.0	0.0	0.0	0.0	1.5	0.0	0.0	2.0	1.5	3.5	0.0	2.0	3.0	0.0	0.0	0.0	0.0	0.0	7.5	8.0	2.0	0.0
Review Code	28	38.0	16.0			1.5	1.0	2.0	1.0		1.0	2.0													2.0	1.5	2.0		2.0	1.5									
Refactoring	30	23.5	18.0																								1.5			1.5								
Einarbeitung in neue Technologien	45	72.5	91.5	7.5	3.0		3.0		3.0	1.5	3.0	2.0										1.0													7.5	8.0	2.0		
Dokumente überprüfen	32	34.5	39.0										1.0	2.0	1.0	0.5	1.0	1.0				0.5																
Requirements	**25.0**	**30.5**	**26.0**	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0	1.0	2.0	0.5	1.0	1.0	1.0	0.0	0.0	2.5	1.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Anforderungsspezifikation	10	10.0	7.0											1.0	1.0																							
Use Cases	15	20.5	19.0		2.0								1.0		1.0	0.5	1.0	1.0	1.0			2.5	1.5															
Business Modeling	**70.0**	**42.5**	**19.0**	0.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	0.0	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.5	0.0	1.0	2.0	1.0	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Domain Analyse Dokument	10	8.0	5.5																																			
Domain Model	10	8.0	5.5																																			
System Sequenz Diagramm	30	18.0	7.0			2.0							3.0													2.0	1.0		1.0									
Operation Contracts	20	8.5	1.0																				1.5		1.0													
Analyse und Design	**95.0**	**87.0**	**49.0**	2.0	3.0	0.0	4.0	1.5	4.0	5.0	4.0	0.0	1.0	0.5	0.0	0.0	0.0	0.0	1.0	0.0	4.0	0.0	1.0	0.0	3.0	0.0	2.0	0.0	2.0	0.0	1.0	1.0	3.0	1.0	2.0	0.0	0.0	0.0	0.0
Sequenzdiagramm	15	9.0	2.0																1.0		2.0												2.0		2.0				
Klassendiagramme	15	11.0	4.5																																			
System Architecture Document	50	52.0	31.0				4.0	1.5	4.0	5.0	4.0		1.0	0.5							2.0		1.0		3.0		2.0		2.0		1.0	1.0	1.0	1.0					
Paper Prototype	15	15.0	11.5	2.0	3.0																																	
Implementation	**373.0**	**416.0**	**709.0**	2.5	9.0	13.5	6.0	22.0	6.0	8.0	6.0	16.0	7.0	20.0	5.0	10.0	7.5	12.5	13.0	25.0	11.0	14.0	13.0	22.5	11.0	18.5	9.5	21.5	10.0	12.5	12.0	24.0	10.0	20.5	10.0	19.5	6.0	13.5	11.0
Aufbau Architektur	18	37.0	63.0	2.5		3.0		4.0				3.0		3.0					2.0				2.0			3.5												
Summary für Rechnungserstellung	30	30.0	19.0																																			
Flugreport für Piloten	30	50.0	45.0																						9.0						6.0		2.0		3.0			3.0
Mandatenfähigkeit	20	6.0	5.0										2.0																									
Authentifizierung und Authorisierung von Benutzern	40	43.0	44.5																	2.0		5.0																
Stammdatenverwaltung	70	133.5	255.0		9.0	1.5	6.0	16.0	6.0	3.5	6.0		5.0	14.0	4.0	10.0			7.0	20.5	7.0	9.0			2.0	12.0	1.5	20.5	2.0			19.5		16.5				3.0	8.0
Flugerfassung	45	43.5	111.5									6.0		1.5			7.5	12.5	2.0		2.0		9.0	20.0		2.0	2.0		8.0	12.5	1.0	4.5	1.0	2.0	3.0	16.0		3.0	
Schadenerfassung inkl. Report	45	0.0	0.0																																			
Schnittstelle für Synchronisation von Adressdaten	25	0.0	15.0																																			
Startlisten-Export für BAZL	20	38.0	4.5																							6.0					2.0		6.0		2.0			
Datenbank-Anpassungen	10	11.0	23.5					2.0				0.5		1.5	1.0					2.5																3.5		2.5	
Allgemeine Implementationen	20	24.0	123.0			9.0				4.5		6.5							2.0		2.0		2.0	2.5		1.0		1.0			3.0		1.0	2.0	2.0		6.0	5.0	
Konfigurationsmanagement	**5.0**	**2.0**	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Projektautomation	5	2.0	0.0																																			
Tests	**135.0**	**136.5**	**171.0**	6.5	0.0	0.0	3.0	0.0	3.0	0.0	3.0	2.5	3.0	0.0	3.0	0.0	1.5	0.0	2.0	0.0	2.0	7.0	0.0	0.0	0.0	0.0	1.0	1.5	2.0	0.0	4.0	0.0	4.0	2.5	5.0	0.0	3.0	4.0	5.0
Systemtests	20	34.5	23.5								3.0	2.5					1.5				2.0										2.0		2.0		2.0				4.0
Unit Tests	80	61.0	70.0	6.5			3.0		3.0				3.0		3.0				2.0			7.0					1.0	1.5	2.0		2.0		2.0	2.5	3.0		2.0	3.0	
Bugfixing	25	38.0	71.0																																			1.0	
Usability Tests	10	3.0	6.5																																		1.0		1.0
Dokumentation und Deployment	**44.0**	**46.5**	**88.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	5.0	2.0
Schlusspräsentation	10	18.0	13.0																																		2.0	5.0	2.0
Abgabe vorbereiten	10	12.0	10.5																																				
Schlussberichte	12	12.5	61.0																																				
Bedienungsanleitung	12	4.0	3.5																																				
Wochen Subtotal				24.0	21.0	22.5	21.0	27.0	21.0	16.0	21.0	23.0	20.0	26.5	15.0	15.5	15.0	16.0	21.0	26.0	21.0	26.0	21.0	23.5	21.0	24.0	21.0	25.5	21.0	18.5	21.0	26.0	21.0	26.0	21.0	28.0	22.5	27.5	21.5
Total / Wochentotal	**1085.0**	**1131.0**	**1424.0**		63.0	69.5					63.0	66.0					50.0	58.0					63.0	75.5					63.0	68.0					63.0	80.0			
Kontrolle			**0.0**																																			

Auswertung:

	Gesamt SOLL	IST
vor Projektstart	0.0	0.0
Inception	60.5	61.0
Elaboration 1	182.0	197.5
Elaboration 2	189.5	204.5
Construction 1	239.0	281.5
Construction 2	263.5	417.0
Transition	196.5	262.5
Total	**1131.0**	**1424.0**

Kontrolle

Total SOLL	
Total IST	

Detailplan **Flight Logging System**

Ferien F — Auffahrt 17.5 — Pfingsten 28.5.

Phasen Construction 2 — Transition

Meilensteine MS — MS V — MS VI — ABGABE: BIS 15.6. 12:00 Uhr

Woche (von - bis)				12		11.05	SW 13				18.05	SW 14				25.05	SW 15				01.06	SW 16				08.06	SW 17				15.06
		Total				mueh	14.05				mueh	21.05				mueh	28.05				mueh	04.06				mueh	11.06				mueh
							wals		frei			wals		frei			wals		frei			wals		frei			wals		frei		
Arbeitspaket	**Geschätzt**	**SOLL**	**IST**	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I	S	I
Projekt Management	**203.0**	**201.5**	**197.5**	1.5	5.5	4.0	2.0	3.5	2.0	4.0	2.0	3.5	2.0	3.0	2.0	6.0	2.0	3.5	2.0	2.0	2.0	2.0	2.0	2.0	3.0	4.0	2.0	4.0	2.0	4.0	2.0	1.0	2.0	1.0	2.0	0.0
Projektplan / Zeitplan	40	37.5	37.0		1.0	1.0																										1.0				
Iterationspläne / Risikomanagement	10	6.0	12.0		1.0	1.0																			1.0	1.5										
Besprechungen im Team	55	39.0	18.0		0.5	0.5	0.5	0.5	0.5		0.5		0.5	0.5	0.5	0.5	0.5		0.5		0.5		0.5	0.5	0.5	0.5	0.5	0.5	0.5		0.5		0.5		
Besprechungen mit Betreuer/Auftraggeber	60	117.5	130.0	1.5	3.0	1.5	1.5	3.0	1.5	4.0	1.5	3.5	1.5	3.0	1.5	5.5	1.5	3.0	1.5	2.0	1.5	2.0	1.5	2.0	1.5	2.0	1.5	3.5	1.5	3.5	1.5		1.5	1.0	1.5	
Reserven aus Riskmanagement	35	0.0	0.0																																	
Glossar	3	1.5	0.5																																	
Qualitätsmassnahmen	**135.0**	**168.5**	**164.5**	0.0	0.0	3.0	0.0	0.0	0.0	3.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	11.0	8.5	3.0	0.0	11.0	7.5	11.0	6.0	12.0	1.0	11.0	0.0	4.0	6.0	4.0	9.5	4.0	9.0
Review Code	28	38.0	16.0																4.0	2.5			4.0	2.5	6.0		6.0		6.0							
Refactoring	30	23.5	18.0																4.0	6.0			4.0	3.5	4.0	6.0	4.0	1.0	4.0							
Einarbeitung in neue Technologien	45	72.5	91.5			3.0				3.5																										
Dokumente überprüfen	32	34.5	39.0																3.0		3.0		3.0	1.5	1.0		2.0		1.0		4.0	6.0	4.0	9.5	4.0	9.0
Requirements	**25.0**	**30.5**	**26.0**	0.0	1.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	0.0	1.0	1.5	0.0	0.0	0.0	2.0	0.0	0.0	0.0	0.0
Anforderungsspezifikation	10	10.0	7.0		1.0																															
Use Cases	15	20.5	19.0																						2.0		1.0	1.5				2.0				
Business Modeling	**70.0**	**42.5**	**19.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	7.0	4.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Domain Analyse Dokument	10	8.0	5.5																				1.0													
Domain Model	10	8.0	5.5																					0.5												
System Sequenz Diagramm	30	18.0	7.0																				3.0	3.0												
Operation Contracts	20	8.5	1.0																				3.0	1.0												
Analyse und Design	**95.0**	**87.0**	**49.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	5.0	0.0	6.0	0.0	0.0	0.0	3.0	2.0	3.0	0.0	3.0	0.0	6.0	15.0	6.0	6.0	6.0	4.0
Sequenzdiagramm	15	9.0	2.0																1.0		1.0											2.0				
Klassendiagramme	15	11.0	4.5																		5.0												6.0	4.5		
System Architecture Document	50	52.0	31.0																4.0						3.0	2.0	3.0		3.0		6.0	13.0		1.5	6.0	4.0
Paper Prototype	15	15.0	11.5																																	
Implementation	**373.0**	**416.0**	**709.0**	18.0	12.0	6.0	19.0	27.5	20.0	26.5	23.0	23.5	19.0	33.5	20.0	25.0	20.0	25.5	0.0	33.0	0.0	23.0	0.0	26.0	0.0	17.0	0.0	3.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Aufbau Architektur	18	37.0	63.0									2.5																								
Summary für Rechnungserstellung	30	30.0	19.0										10.0		20.0	14.0						4.5					0.5									
Flugreport für Piloten	30	50.0	45.0	2.5	4.0				20.0	21.5	3.0					10.0						10.5					0.5									
Mandatenfähigkeit	20	6.0	5.0																																	
Authentifizierung und Authorisierung von Benutzern	40	43.0	44.5								20.0	16.0					20.0	7.0																		
Stammdatenverwaltung	70	133.5	255.0	13.5				13.5		3.5				4.5				7.0								8.0										
Flugerfassung	45	43.5	111.5		8.0	6.0		1.0				5.0								6.0				13.5												
Schadenerfassung inkl. Report	45	0.0	0.0																																	
Schnittstelle für Synchronisation von Adressdaten	25	0.0	15.0																	15.0																
Startlisten-Export für BAZL	20	38.0	4.5				13.0						9.0			0.5						3.5					0.5									
Datenbank-Anpassungen	10	11.0	23.5							1.5						0.5		4.0																		
Allgemeine Implementationen	20	24.0	123.0	2.0			6.0	13.0						29.0				7.5		12.0		4.5		12.5		9.0		2.0								
Konfigurationsmanagement	**5.0**	**2.0**	**0.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Projektautomation	5	2.0	0.0																																	
Tests	**135.0**	**136.5**	**171.0**	2.5	5.0	10.5	0.0	1.0	0.0	0.0	0.0	9.0	0.0	1.0	0.0	0.0	0.0	2.0	3.0	2.5	9.0	19.0	0.0	4.5	21.0	20.0	22.0	31.0	24.0	40.0	0.0	0.0	0.0	0.0	0.0	0.0
Systemtests	20	34.5	23.5	2.5															3.0						3.0	7.0	6.0	5.5	3.0	6.0						
Unit Tests	80	61.0	70.0		4.0	4.5								1.0						2.0	4.0	19.0			8.0	8.5	8.0	8.5	8.0	3.0						
Bugfixing	25	38.0	71.0			6.0		1.0				3.0						2.0		0.5	5.0			4.0	10.0	4.5	8.0	17.0	13.0	31.0						
Usability Tests	10	3.0	6.5		1.0							6.0												0.5												
Dokumentation und Deployment	**44.0**	**46.5**	**88.0**	3.5	2.0	4.5	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	1.5	2.0	1.5	7.5	1.5	8.5	12.0	13.0	12.0	17.5	12.0	26.5
Schlusspräsentation	10	18.0	13.0	3.5	2.0	4.5																									4.0		4.0		4.0	
Abgabe vorbereiten	10	12.0	10.5																												4.0	3.0	4.0	4.5	4.0	3.0
Schlussberichte	12	12.5	61.0																						1.5	2.0	1.5	7.5	1.5	8.5	4.0	10.0		9.5	4.0	23.5
Bedienungsanleitung	12	4.0	3.5																														4.0	3.5		
Wochen Subtotal				25.5	25.5	28.0	21.0	32.0	22.0	34.0	25.0	36.0	21.0	37.5	22.0	31.0	22.0	31.0	21.0	46.0	20.0	44.0	20.0	44.5	41.5	51.0	41.5	48.5	41.5	52.5	24.0	37.0	24.0	34.0	24.0	39.5
Total / Wochentotal	**1085.0**	**1131.0**	**1424.0**		69.5	61.0					68.0	102.0					65.0	99.5					61.0	134.5				124.5	152.0					72.0	110.5
Kontrolle			**0.0**																																	

Auswertung:

	Gesamt SOLL	IST
vor Projektstart	0.0	0.0
Inception	60.5	61.0
Elaboration 1	182.0	197.5
Elaboration 2	189.5	204.5
Construction 1	239.0	281.5
Construction 2	263.5	417.0
Transition	196.5	262.5
Total	**1131.0**	**1424.0**

Kontrolle

Total SOLL	
Total IST	

7.2.4 Meilensteine

MS	Beschreibung	Anforderungen an Release	Termin
MS I	Ende Inception	Projektauftrag/Projektplan vom Auftraggeber und Dozent genehmigt.	29.02.2012
MS II	Ende Elaboration 1	Anforderungsspezifikation von Patrick Schuler genehmigt und 1. Prototyp erstellt.	21.03.2012
MS III	Ende Elaboration 2	Domainanalyse Version 1 fertiggestellt und 2. Prototypen erstellt.	01.04.2012
MS IV	Ende Construction 1	Review Architekturdesign und Dokumentation.	02.05.2012
MS V	Ende Construction 2	Alle Systemtests abgeschlossen.	06.06.2012
MS VI	Ende Transition	Präsentation der Arbeit im HSR-Forum & Projektende.	15.06.2012

Die Termine repräsentieren die Projektsitzungen am entsprechenden Mittwoch. Die Arbeiten an den Iterationen sind jedoch am vorhergehenden Freitag beendet. Ergänzung aufgrund der Sitzung werden in der folgenden Iteration ausgeführt.

7.2.5 Besprechungen

Am Mittwoch um 15 Uhr findet jeweils eine Sitzung mit dem Betreuer statt. Das Protokoll dazu ist innerhalb von 2 Tagen nach Sitzungsende an den Betreuer abzuliefern.

Das Team trifft sich jeden Montagmorgen für etwa eine Stunde, um aktuelle Themen zu besprechen und die Aufgaben für die Woche zu definieren.

Das Team trifft sich nach Bedarf mit dem Auftraggeber, um die Umsetzungen zu diskutieren und die Ergebnisse überprüfen zu lassen.

8 Risikomanagement

Das Risikomanagement befindet sich in den Iterationsplänen. Es wird jeweils am Ende einer Iteration neu beurteilt. Die erste Version des Risikomanagements befindet sich im Iterationsplan Elaboration 1.

9 Infrastruktur

9.1 Räumlichkeiten

Die Besprechung mit den Betreuern finden im Sitzungszimmer 6.010 statt. Besprechungen mit dem Auftraggeber finden in der Cafeteria statt. Die Projektarbeit findet hauptsächlich in dem zugewiesenen Labor 1.258 statt.

9.2 Technische Hilfsmittel

Jedes Teammitglied arbeitet auf dem eigenen Notebook oder an der Arbeitsstation im Labor. Drucker stehen an der HSR zur Verfügung.

Der Server für das Versionsmanagement, sowie die Infrastruktur für die Projektautomation wird vom Auftraggeber zur Verfügung gestellt.

Ergänzung: Projektautomation wurde vom Auftraggeber auf dem SVN-Server nicht umgesetzt und deshalb nicht verwendet.

9.3 Verwendete Programme

- Betriebssystem
  - Windows 7
  - Windows Vista
- Entwicklungstools
  - Microsoft Visual Studio 2010 mit SP1
  - Microsoft .NET Framework 4.0
  - Microsoft SQL Server 2008 R2
  - Silverlight 5 & Silverlight 5 Toolkit Dez. 2011 & Silverlight 5 Tools
  - ADO.NET Entity Framework 4.1
  - ReSharper 6
  - Doxygen
- Versionsmanagement
  - Tortoise SVN Client

- Dokumentation
  - LaTeX
  - Microsoft Excel

10 Qualitätsmassnahmen

10.1 Arbeiten im Team und Dokumentation

Die Arbeit wird mittels Dokumentation festgehalten. Dabei orientieren sich die Teammitglieder an RUP. Am Ende jeder Iteration werden die Dokumente auf ihre Aktualität und Korrektheit geprüft.

Die wöchentlichen Sitzungen werden protokolliert, damit bei Unstimmigkeiten oder Unsicherheiten auf diese Informationen zurückgegriffen werden kann.

Jedes Projektmitglied führt seine eigene Zeiterfassung, die täglich aktuell gehalten und mindestens wöchentlich versioniert wird. Zusätzlich wird jeweils vor den Besprechungen mit dem Betreuer das Dokument „Zeitplan.xls“ aktualisiert.

10.2 Konfigurationsmanagement

Für sämtliche Projektinhalte wird das Versionsmanagement-Tool TortoiseSVN benutzt.

Damit bei Ausfall des Tools kein zu grosser Mehraufwand entsteht, wird von allen Teammitgliedern regelmässig in das SVN Repository eingechecked und aktualisiert. Dadurch sind die 4 Kopien der Projektdaten immer aktuell.

10.3 Code-Qualität

Um eine gute Code-Qualität zu erreichen, werden sich die Projektmitglieder an die Code-Richtlinien halten. Es werden Teile Code-Richtlinien von Patrick Schuler[Sch12] verwendet. Allerdings kann das Kapitel ’Loggen von Debug-Informationen’ gemäss Sitzungsprotokoll Woche 2 - 28.02.2012 ignoriert werden. Ebenso wurde während des Projekts entschieden, das StandardClass Template in Kapitel 7.4 nicht zu verwenden. Dies wurde im Sitzungsprotokoll Woche 8 - 11.04.2012 (Besprechungen mit externem Betreuer) festgehalten.

Interessante Methoden und Klassen und Schnittstellen werden dokumentiert.

Der bereits geschriebene Code wird am Ende jeder Iteration überprüft und wenn nötig umgeschrieben. Insbesondere wird darauf geachtet, dass die Code-Richtlinien eingehalten werden. Diese Reviews werden im Dokument Code-Review festgehalten.

Um Code-Smells aufzudecken wird das Tool ReShaper benutzt.

10.4 Tests

Zum Testen werden Unit-Tests eingesetzt. Diese werden über die Projektautomation regelmässig ausgeführt. Zusätzlich werden die im nächsten Unterkapitel beschriebenen Systemtests durchgeführt.

Zu den Tests werden Protokolle mit den Findings erstellt. In der darauf folgenden Team-Besprechung werden die daraus entstehenden Arbeiten den Projektmitgliedern zugeteilt.

10.5 Systemtests

Die Systemtests werden am Ende folgender Iteration durchgeführt:

- Ende Construction 2

VIII

Iterationspläne

Datum: 15. Juni 2012

11 Iterationsplan Elaboration 1

In dieser Iteration wird der grösste Teil der Anforderungsanalyse und der Domainanalyse erstellt. Ebenfalls wird bereits mit der Implementation begonnen und der Prototyp 1 erstellt.

Um eine genauere Planung für die Iteration zu erhalten, wird in der folgenden Tabelle Redundanz zum Zeitplan in Kauf genommen.

Regelmässig wiederkehrende Arbeitspakete wie z.B. Risikomanagement oder Besprechungen werden in der folgenden Tabelle nicht extra aufgeführt.

Arbeitspaket	Kommentar	Soll-Stunden	Fertigungsgrad nach Iteration
Projektplan / Zeitplan	Werden auf den aktuellen Stand aktualisiert.	7.5	89 %
Glossar		1	50 %
Review Code	Am Ende der Iteration wird der darin entstandene Code reviewed.	3	10 %
Refactoring	Findings des Reviews werden verbessert.	2	6 %
Einarbeiten in neue Technologien	Das Team hat sich zu Ende dieser Iteration gut in die neuen Technologien eingearbeitet.	20.5	83 %
Anforderungsspezifikation	Die Anforderungsspezifikation wird erstellt.	2	80 %
Use Cases	Alle Use Cases werden in Brief-Format verfasst und wichtige Use Cases ebenfalls fully-dressed ausgeschrieben.	8	60 %
Domain Analyse Dokument	Die Domain Analyse wird erstellt.	7	70 %
Domain Model	Das Domain Model wird erstellt .	8	80 %

System Sequenz Diagramm	Die meisten System Sequenzdiagramme werden erstellt.	20	66 %
Operation Contracts	Die wichtigsten Operation Contracts werden erstellt.	12	60 %
System Architecture Document	Das SAD wird vorbereitet und wichtige Erkenntnisse festgehalten.	2	4 %
Implementation Aufbau Architektur	Das Grundgerüst der Applikation wird erstellt.	13	72 %
Implementierung Authentifizierung und Authorisierung von Benutzern	Es wird mit der Implementierung begonnen.	8	7 %
Implementierung Stammdatenverwaltung	Es wird mit der Implementierung begonnen Als erstes wird die Benutzerverwaltung umgesetzt.	27	38 %
Datenbank-Anpassungen	Falls nötig wird die Datenbank angepasst.	6	60 %
Projektautomation	Aufsetzen der Projektautomation	2	40 %
Systemtests	Dokument wird erstellt und erste Tests geschrieben.	3	15 %
Unit Tests	Für bereits erstellten Code werden Unit Tests geschrieben.	3	3 %
Bugfixing	Beim Testen gefundene Bugs werden behoben.	2	2 %

11.1 Risikomanagement

ID	Risiko	Massnahme	Vorgehen bei Eintreffen	Wahrscheinlichkeit [%]	Zusatzaufwand [h]	GewichteterZusatzaufwand [h]
R01	Zeitplan kann generell nicht eingehalten werden	Termine für Besprechungen und gemeinsames Arbeiten werden eingehalten und jeweils pünktlich begonnen.	gemeinsames Arbeiten am Wochenende	10	100	10
R02	Ausfall eines Teammitglieds durch Krankheit/Unfall	Gesunde und vernünftige Lebensweise	Reorganisation Team	5	180	9
R03	Datenverlust auf einem der Laptops oder auf SVN-Server	Daten werden auf Laptops und auf SVN-Server aktuell gehalten. Es sollten bei Datenverlust nicht mehr als 2 Stunden Arbeit verloren gehen Der SVN-Server wird mittels Backup gesichert.	Kopie einer aktuellen Version wird erstellt.	10	2	0.2
R04	Aufbau der Architektur bereitet Schwierigkeiten	Gute Einarbeitung in Technologien, intensiver Austausch im Team	Hilfe von Patrick Schuler beanspruchen	10	20	2
R05	Mandantenfähigkeit komplexer als erwartet oder nicht alle Einflüsse auf die Logik/Datenbank berücksichtigt	Auswirkungen auf das System erarbeiten / Ausführung der Use Cases unter Berücksichtigung der Mandantenfähigkeit sowie Erweiterung Dokumentation um die speziellen Abläufe	Arbeiten am Wochenende, um Mängel zu beheben und Hilfe von Patrick Schuler beanspruchen	15	40	6
R06	Missverständnisse und Streitigkeiten im Team	Anständiger Umgang miteinander, bei Fehler konstruktive Kritik üben	Aussprache und Entschuldigungen	5	20	1

R07	Schwierigkeiten bei Terminfindung für Besprechungen mit Betreuer	Termine frühzeitig planen und festlegen	Telefonisch und über E-Mail kommunizieren, Teammitglied reist zum Betreuer	10	30	3
R08	Änderungen der Anforderungen durch den Auftraggeber	Use Cases werden mit dem Auftraggeber besprochen um Missverständnisse zu vermeiden.	gewünschte Änderungen werden falls möglich in einer späteren Iteration umgesetzt.	10	40	4
Total						35.2

12 Iterationsplan Elaboration 2

In dieser Iteration wird zum grössten Teil an der Stammdatenverwaltung, Konzept der Mandantenfähigkeit geschrieben und mit den Sequenzdiagrammen begonnen. Als Ergebnis der Iteration wird der Prototyp 2 erstellt.

Um eine genauere Planung für die Iteration zu erhalten, wird in der folgenden Tabelle Redundanz zum Zeitplan in Kauf genommen.

Regelmässig wiederkehrende Arbeitspakete wie z.B. Risikomanagement oder Besprechungen werden in der folgenden Tabelle nicht extra aufgeführt.

Arbeitspaket	Kommentar	Soll-Stunden	Fertigungsgrad nach Iteration
Einarbeiten in neue Technologien	Das Team hat sich zu Ende dieser Iteration gut in die neuen Technologien eingearbeitet.	27	100 %
Use Cases	Es werden noch wenige Use Cases für diese Iteration fully-dressed geschrieben.	2	80 %
System Sequenz Diagramm	Die ersten Sequenzdiagramme werden erstellt.	4	33 %
System Architecture Document	Die Architektur und Schichten werden beschrieben.	12	28 %
Paper Prototyping	Es wird ein Paper Prototyp erstellt.	15	100 %
Mandantenfähigkeit	Konzept für die Mandantenfähigkeit wird erstellt	4	20 %
Implementierung Stammdatenverwaltung	Es wird weiter an der Benutzerverwaltung und an der Verwaltung weiterer Stammdaten gearbeitet (Person, Flugzeug, Mandant).	67	80 %
Datenbank-Anpassungen	Die Datenbank soll auf die Mandantenfähigkeit ausgerichtet werden.	4	100 %

Systemtests	Dokument wird erstellt und erste Tests geschrieben.	3	30 %
Unit Tests	Für bereits erstellten Code werden Unit Tests geschrieben.	6	11 %
Review Code	Am Ende der Iteration wird der in der Iteration entstandene Code überprüft.	3	21 %

12.1 Risikomanagement

ID	Risiko	Massnahme	Vorgehen bei Eintreffen	Wahrscheinlichkeit [%]	Zusatzaufwand [h]	GewichteterZusatzaufwand [h]
R01	Zeitplan kann generell nicht eingehalten werden	Termine für Besprechungen und gemeinsames Arbeiten werden eingehalten und jeweils pünktlich begonnen.	gemeinsames Arbeiten am Wochenende	9	90	8.1
R02	Ausfall eines Teammitglieds durch Krankheit/Unfall	Gesunde und vernünftige Lebensweise	Reorganisation Team	5	144	7.2
R03	Datenverlust auf einem der Laptops oder auf SVN-Server	Daten werden auf Laptops und auf SVN-Server aktuell gehalten. Es sollten bei Datenverlust nicht mehr als 2 Stunden Arbeit verloren gehen Der SVN-Server wird mittels Backup gesichert.	Kopie einer aktuellen Version wird erstellt.	9	2	0.2
R04	Aufbau der Architektur bereitet Schwierigkeiten	Gute Einarbeitung in Technologien, intensiver Austausch im Team	Hilfe von Patrick Schuler beanspruchen	4	15	1
R05	Mandantenfähigkeit komplexer als erwartet oder nicht alle Einflüsse auf die Logik/Datenbank berücksichtigt	Auswirkungen auf das System erarbeiten / Ausführung der Use Cases unter Berücksichtigung der Mandantenfähigkeit sowie Erweiterung Dokumentation um die speziellen Abläufe	Arbeiten am Wochenende, um Mängel zu beheben und Hilfe von Patrick Schuler beanspruchen	15	40	6
R06	Missverständnisse und Streitigkeiten im Team	Anständiger Umgang miteinander, bei Fehler konstruktive Kritik üben	Aussprache und Entschuldigungen	4	20	0.8

R07	Schwierigkeiten bei Terminfindung für Besprechungen mit Betreuer	Termine frühzeitig planen und festlegen	Telefonisch und über E-Mail kommunizieren, Teammitglied reist zum Betreuer	7	30	2.1
R08	Änderungen der Anforderungen durch den Auftraggeber	Use Cases werden mit dem Auftraggeber besprochen um Missverständnisse zu vermeiden.	gewünschte Änderungen werden falls möglich in einer späteren Iteration umgesetzt.	10	40	4
Total						29.4

13 Iterationsplan Construction 1

In dieser Iteration wird die Stammdatenverwaltung fertiggestellt und die Flugerfassung beendet. Ausserdem wird mit dem Flugreport für den Piloten sowie dem Startlisten-Export für das BAZL begonnen. Als Ergebnis der Iteration wird der Prototyp 3 erstellt.

Um eine genauere Planung für die Iteration zu erhalten, wird in der folgenden Tabelle Redundanz zum Zeitplan in Kauf genommen.

Regelmässig wiederkehrende Arbeitspakete wie z.B. Risikomanagement oder Besprechungen werden in der folgenden Tabelle nicht extra aufgeführt.

Arbeitspaket	Kommentar	Soll-Stunden	Fertigungsgrad nach Iteration
Anforderungsspezifikation	Spezielle Abläufe werden ergänzt.	1	100%
Use Cases	Die restlichen Use Cases werden fertiggestellt.	5.5	100 %
System Sequenz Diagramm	Weitere System-Sequenzdiagramme werden erstellt.	5	60 %
Operation Contracts	Die ersten Operation Contracts werden erstellt.	2.5	30 %
Sequenzdiagramme	Die ersten Sequenzdiagramme werden erstellt.	7	50 %
System Architecture Document	Die Architektur und Schichten werden beschrieben.	13	60 %
Aufbau Architektur	Die endgültige Architektur wird aufgebaut.	4	100 %
Flugreport für Piloten	Es wird mit dem Flugreport begonnen.	20	65 %

Mandantenfähigkeit	Konzept für die Mandantenfähigkeit wird fertiggestellt.	2	100 %
Implementierung Stammdatenverwaltung	Es wird weiter an der Benutzerverwaltung und an der Verwaltung weiterer Stammdaten gearbeitet (Person, Flugzeug, Mandant).	28.5	100 %
Flugerfassung	Flugerfassung wird fertiggestellt.	33.5	100 %
Startlistenexport BAZL	Es wird mit dem Startlistenexport begonnen.	16	70 %
Datenbank-Anpassungen	Letzte Änderungen für Mandantenfähigkeit	1	100 %
Systemtests	Weitere Tests werden geschrieben.	9.5	60 %
Unit Tests	Für bereits erstellten Code werden Unit Tests geschrieben.	18	20 %
Review Code	Am Ende der Iteration wird der in der Iteration entstandene Code überprüft.	6	50 %
Refactoring	Erkenntnisse aus Review werden umgesetzt.	1.5	12 %

13.1 Risikomanagement

ID	Risiko	Massnahme	Vorgehen bei Eintreffen	Wahrscheinlichkeit [%]	Zusatzaufwand [h]	GewichteterZusatzaufwand [h]
R01	Zeitplan kann generell nicht eingehalten werden	Termine für Besprechungen und gemeinsames Arbeiten werden eingehalten und jeweils pünktlich begonnen.	gemeinsames Arbeiten am Wochenende. Soll-Zeit erhöhen.	7	80	5.6
R02	Ausfall eines Teammitglieds durch Krankheit/Unfall	Gesunde und vernünftige Lebensweise	Reorganisation Team	5	100	5
R03	Datenverlust auf einem der Laptops oder auf SVN-Server	Daten werden auf Laptops und auf SVN-Server aktuell gehalten. Es sollten bei Datenverlust nicht mehr als 2 Stunden Arbeit verloren gehen Der SVN-Server wird mittels Backup gesichert.	Kopie einer aktuellen Version wird erstellt.	7	2	0.1
R04	Aufbau der Architektur bereitet Schwierigkeiten	Gute Einarbeitung in Technologien, intensiver Austausch im Team	Hilfe von Patrick Schuler beanspruchen	3	12	1
R05	Mandantenfähigkeit komplexer als erwartet oder nicht alle Einflüsse auf die Logik/Datenbank berücksichtigt	Auswirkungen auf das System erarbeiten / Ausführung der Use Cases unter Berücksichtigung der Mandantenfähigkeit sowie Erweiterung Dokumentation um die speziellen Abläufe	Arbeiten am Wochenende, um Mängel zu beheben und Hilfe von Patrick Schuler beanspruchen	15	40	6
R06	Missverständnisse und Streitigkeiten im Team	Anständiger Umgang miteinander, bei Fehler konstruktive Kritik üben	Aussprache und Entschuldigungen	3	16	0.5

R07	Schwierigkeiten bei Terminfindung für Besprechungen mit Betreuer	Termine frühzeitig planen und festlegen	Telefonisch und über E-Mail kommunizieren, Teammitglied reist zum Betreuer	6	25	1.5
R08	Änderungen der Anforderungen durch den Auftraggeber	Use Cases werden mit dem Auftraggeber besprochen um Missverständnisse zu vermeiden.	gewünschte Änderungen werden falls möglich in einer späteren Iteration umgesetzt.	8	30	2.4
Total						22.1

14 Iterationsplan Construction 2

In dieser Iteration werden die Reports, die Exports/Imports und die Berechtigungen implementiert. Als Ergebnis der Iteration wird der Prototyp 4 erstellt.

Um eine genauere Planung für die Iteration zu erhalten, wird in der folgenden Tabelle Redundanz zum Zeitplan in Kauf genommen.

Regelmässig wiederkehrende Arbeitspakete wie z.B. Risikomanagement oder Besprechungen werden in der folgenden Tabelle nicht extra aufgeführt.

Da sich die Implementation komplexer gestaltet als angenommen, konnte in den vorherigen Iterationen nicht alle geplanten Arbeiten durchgeführt werden. Aufgrund des dadurch entstandenen Zeitdrucks wurden tief-priorisierte Anforderungen weggelassen und es wurden die Stunden der Soll-Planung erhöht.

Arbeitspaket	Kommentar	Soll-Stunden	Fertigungsgrad nach Iteration
Anforderungsspezifikation	Wird auf Aktualität überprüft	1	100%
System Sequenz Diagramm	Weitere System-Sequenzdiagramme werden erstellt.	3	60 %
Operation Contracts	Es wird überprüft ob wichtige Operation Contracts nachgeführt werden müssen.	3	45 %
Sequenzdiagramme	Sequenzdiagramme werden erstellt.	2	60 %
Klassendiagramme	Wird erstellt.	5	35 %
System Architecture Document	Die Architektur und Schichten werden beschrieben.	4	62 %
Summery für Rechnungsstellung	Wird implementiert.	30	100 %

Flugreport für Piloten	Wird implementiert.	30	100 %
Authentifizierung und Authorisierung von Benutzern	Wird implementiert.	40	100 %
Stammdatenverwaltung	Korrekturen nach Usability Tests werden umgesetzt.	8	100 %
Flugerfassung	Korrekturen nach Usability Tests werden umgesetzt.	8	100 %
Schadenerfassung inkl.Report	Wird aufgrund von Zeitmangel nicht eingeplant.	0	0 %
Schnittstelle für Synchronisation von Addressdaten	Wird aufgrund von Zeitmangel nicht eingeplant.	0	0 %
Startlistenexport BAZL	Der Startlistenexport wird umgesetzt.	22	100 %
Allgemeine Implementationen	Letzte Implementationen	12	100 %
Systemtests	Weitere Tests werden geschrieben.	7	80 %
Unit Tests	Für bereits erstellten Code werden Unit Tests geschrieben.	10	46.5 %
Review Code	Am Ende der Iteration wird der in der Iteration entstandene Code überprüft.	8	71.5 %
Refactoring	Erkenntnisse aus Review werden umgesetzt.	8	38.5 %

14.1 Risikomanagement

ID	Risiko	Massnahme	Vorgehen bei Eintreffen	Wahrscheinlichkeit [%]	Zusatzaufwand [h]	GewichteterZusatzaufwand [h]
R01	Zeitplan kann generell nicht eingehalten werden	Termine für Besprechungen und gemeinsames Arbeiten werden eingehalten und jeweils pünktlich begonnen.	gemeinsames Arbeiten am Wochenende	7	70	4.9
R02	Ausfall eines Teammitglieds durch Krankheit/Unfall	Gesunde und vernünftige Lebensweise	Reorganisation Team	3	60	1.8
R03	Datenverlust auf einem der Laptops oder auf SVN-Server	Daten werden auf Laptops und auf SVN-Server aktuell gehalten. Es sollten bei Datenverlust nicht mehr als 2 Stunden Arbeit verloren gehen Der SVN-Server wird mittels Backup gesichert.	Kopie einer aktuellen Version wird erstellt.	5	2	0.1
R04	Aufbau der Architektur bereitet Schwierigkeiten	Gute Einarbeitung in Technologien, intensiver Austausch im Team	Hilfe von Patrick Schuler beanspruchen	1	6	0.1
R05	Mandantenfähigkeit komplexer als erwartet oder nicht alle Einflüsse auf die Logik/Datenbank berücksichtigt	Auswirkungen auf das System erarbeiten / Ausführung der Use Cases unter Berücksichtigung der Mandantenfähigkeit sowie Erweiterung Dokumentation um die speziellen Abläufe	Arbeiten am Wochenende, um Mängel zu beheben und Hilfe von Patrick Schuler beanspruchen	5	20	1
R06	Missverständnisse und Streitigkeiten im Team	Anständiger Umgang miteinander, bei Fehler konstruktive Kritik üben	Aussprache und Entschuldigungen	2	8	0.1

R07	Schwierigkeiten bei Terminfindung für Besprechungen mit Betreuer	Termine frühzeitig planen und festlegen	Telefonisch und über E-Mail kommunizieren, Teammitglied reist zum Betreuer	4	17	0.7
R08	Änderungen der Anforderungen durch den Auftraggeber	Use Cases werden mit dem Auftraggeber besprochen um Missverständnisse zu vermeiden.	gewünschte Änderungen werden falls möglich in einer späteren Iteration umgesetzt.	8	30	2.4
Total						11.1

15 Iterationsplan Transition

In dieser Iteration werden die letzten Probleme in der Applikation behoben sowie alle Dokumente fertiggestellt. Das Ergebnis der Iteration ist die vollständige Abgabe der Arbeit.

Um eine genauere Planung für die Iteration zu erhalten, wird in der folgenden Tabelle Redundanz zum Zeitplan in Kauf genommen.

Regelmässig wiederkehrende Arbeitspakete wie z.B. Risikomanagement oder Besprechungen werden in der folgenden Tabelle nicht extra aufgeführt.

Arbeitspaket	Kommentar	Soll-Stunden	Fertigungsgrad nach Iteration
Use Cases	Fully-dressed Use Cases für Reporting und Import/Export	3	100%
Klassendiagramme	Definitive Version der Klassendiagramme	6	100 %
System Architecture Document	Fertigstellung	21	100 %
Systemtests	Durchführung und Bereinigung	12	100 %
Unit Tests	Erstellung von Unit Tests aller komplexen Methoden	24	100 %
Bugfixing	Letzte Bugfixing-Aktivitäten	31	100 %
Review Code	Letzte Überprüfungen des Codes	18	100 %
Refactoring	Überarbeitung des Codes aufgrund Review	12	100 %
Bedienungsanleitung		4	100 %
Schlusspräsentation	Vorbereitung für die Schlusspräsentation	12	100 %
Schlussberichte	Alle Dokumente für die Abgabe erstellen	12.5	100 %

Abgabe vorbereiten	Dokumente ausdrucken und kontrollieren und binden	12	100 %

15.1 Risikomanagement

ID	Risiko	Massnahme	Vorgehen bei Eintreffen	Wahrscheinlichkeit [%]	Zusatzaufwand [h]	GewichteterZusatzaufwand [h]
R01	Zeitplan kann generell nicht eingehalten werden	Termine für Besprechungen und gemeinsames Arbeiten werden eingehalten und jeweils pünktlich begonnen.	gemeinsames Arbeiten am Wochenende	1	50	0.5
R02	Ausfall eines Teammitglieds durch Krankheit/Unfall	Gesunde und vernünftige Lebensweise	Reorganisation Team	1	20	0.2
R03	Datenverlust auf einem der Laptops oder auf SVN-Server	Daten werden auf Laptops und auf SVN-Server aktuell gehalten. Es sollten bei Datenverlust nicht mehr als 2 Stunden Arbeit verloren gehen Der SVN-Server wird mittels Backup gesichert.	Kopie einer aktuellen Version wird erstellt.	0	2	0.0
R04	Aufbau der Architektur bereitet Schwierigkeiten	Gute Einarbeitung in Technologien, intensiver Austausch im Team	Hilfe von Patrick Schuler beanspruchen	0	0	0.0
R05	Mandantenfähigkeit komplexer als erwartet oder nicht alle Einflüsse auf die Logik/Datenbank berücksichtigt	Auswirkungen auf das System erarbeiten / Ausführung der Use Cases unter Berücksichtigung der Mandantenfähigkeit sowie Erweiterung Dokumentation um die speziellen Abläufe	Arbeiten am Wochenende, um Mängel zu beheben und Hilfe von Patrick Schuler beanspruchen	0	0	0.0
R06	Missverständnisse und Streitigkeiten im Team	Anständiger Umgang miteinander, bei Fehler konstruktive Kritik üben	Aussprache und Entschuldigungen	0	0	0.0

R07	Schwierigkeiten bei Terminfindung für Besprechungen mit Betreuer	Termine frühzeitig planen und festlegen	Telefonisch und über E-Mail kommunizieren, Teammitglied reist zum Betreuer	0	0	0.0
R08	Änderungen der Anforderungen durch den Auftraggeber	Use Cases werden mit dem Auftraggeber besprochen um Missverständnisse zu vermeiden.	gewünschte Änderungen werden falls möglich in einer späteren Iteration umgesetzt.	0	0	0.0
Total						0.7

IX

Anforderungsspezifikation

Datum: 15. Juni 2012

16 Allgemeine Beschreibung

16.1 Produkt Perspektive

Die Applikation 'Flight Logging System' wird für die Flugsportgruppe Zürcher Oberland entwickelt und als Bachelorarbeit an der HSR erstellt. Sie soll die Erfassung der Startlisten von Segelflug-Bewegungen erleichtern und ein Reporting über die Daten ermöglichen.

16.2 Produkt Funktionen

Die Startlisten sollen vom Flugdienstleiter elektronisch erfasst werden können. Dies geschieht auf dem jeweiligen Flugplatz. Dafür sollen die Stammdaten vorher erfasst und ständig aktuell gehalten werden können. Auswertungen für Buchhaltung, Piloten und BAZL sollen darauf folgend auch an einem anderen Ort vorgenommen werden können. Die Architektur muss deshalb eine örtliche Flexibilität bieten.

Folgend sind die Hauptaufgaben zusammengefasst:

- Verwalten der Stammdaten
- Erfassen der Start- und Landebewegungen von Segelflügen
- Implementierung von diversen Schnittstellen

Der Anwender wird übers Internet Zugriff auf die Anwendung erhalten, indem er einen Login erhält und sich damit authentifizieren und autorisieren kann.

16.3 Benutzer Charakteristik

Die Benutzer dieser Applikation werden keine Informatik-Experten sein weshalb die Applikation möglichst benutzerfreundlich sein soll. Hohe Priorität hat ein selbsterklärendes User Interface, damit ein neuer Benutzer sich möglichst schnell alleine zurechtfinden kann.

Eine möglichst schnelle Erfassung der Daten ist wichtig, wird aber dem selbsterklärenden Charakter der Applikation untergeordnet. Um Fehler zu vermeiden, werden die

Daten validiert und es wird auf eine möglichst verständliche Fehlerbehandlung geachtet.

16.4 Einschränkungen

Da die Projektdauer sich auf die 17 Wochen entsprechend der Dauer der Bachelorarbeit beschränkt, werden nicht sämtliche möglichen Funktionalitäten gemäss Aufgabenstellung umgesetzt werden können. Darunter fallen mit Sicherheit die Anbindung an FLARM sowie Mappings zu Rechnungssystemen.

Die Anwendung wird nicht auf Parallelbetrieb in Bezug auf Datenbank-Aktualisierungen ausgerichtet. Bei Überschneidung von Transaktionen werden die zuletzt gespeicherten Daten in der Datenbank erfasst sein. Die Wahrscheinlichkeit, dass solche Fälle eintreten, ist aus dem Ablauf der Prozesse sehr klein. Sollte die Anwendung im Grossen eingesetzt werden und dadurch sehr viele Mandanten enthalten, sollte diese Funktionalität überarbeitet werden.

16.5 Annahmen

- Es steht immer eine zuverlässige Internetverbindung zur Verfügung.

16.6 Abhängigkeiten

Die örtliche Unabhängigkeit, wodurch die Applikation an verschiedenen Orten verwendet werden kann, setzt voraus, dass auf dem benutzten Gerät eine Verbindung zum Internet erstellt werden kann.

Weiter muss der Browser des Anwenders Silverlight-Anwendungen ausführen können.

17 Spezifische Anforderungen

17.1 Qualitätsmerkmale

17.1.1 Funktionalität

17.1.1.1 Richtigkeit

Die Datenbank für die Verwaltung der Daten soll zu jeder Zeit einen konsistenten Zustand haben und entsprechend der Eingabe von den Benutzern richtig sein. Die Anzeige der Daten im User Interface soll nach einem Neuladen ebenfalls dem Inhalt der Datenbank entsprechen.

17.1.2 Benutzbarkeit

17.1.2.1 Verständlichkeit

Die Benutzeroberfläche soll für eine mit der Segelflug-Materie vertrauten Person leicht zu verstehen sein. Die Fehlermeldungen sollen ebenfalls leicht verständlich sein.

17.1.2.2 Erlernbarkeit

Das Erfassen der Stammdaten und die Aktualisierung der Startliste sollten grundsätzlich ohne weitere Schulung oder Unterlagen für jemanden, der weiss, welche Daten zu erfassen sind, benutzbar sein. Spezielle Aufgaben wie Reports erstellen oder Import/Export von Daten sollen mittels einer Anleitung ausgeführt werden können.

17.1.2.3 Bedienbarkeit

Die Anwendung soll intuitiv bedienbar sein. Sie soll in erster Linie mit der Maus bedient werden können und das Springen zwischen den Eingabefeldern mit Tabulatoren unterstützen. Die komplette Bedienbarkeit nur mit der Tastatur wird nicht priorisiert.

17.1.3 Zuverlässigkeit

17.1.3.1 Fehlertoleranz

Das System soll bei falscher Eingabe verständliche Rückmeldung an den Benutzer liefern und danach wieder bereit für eine weitere Eingabe sein.

Die Fehlertoleranz gegenüber Netzwerkproblemen kann nicht beeinflusst werden und entspricht deshalb der Fehlertoleranz des darunter liegenden Transport-Protokolls.

17.1.3.2 Netzwerk

Die Zuverlässigkeit des Netzwerks muss der Zuverlässigkeit der Netzwerkverbindungen und der TCP/IP-Kommunikation entsprechen. Die Kommunikation zwischen Client und Server wird nicht verschlüsselt erfolgen.

17.1.4 Wartbarkeit & Erweiterbarkeit

17.1.4.1 Schichten-Architektur

Eine klare Schichten-Architektur soll die Wartbarkeit und die Erweiterbarkeit einfach gestalten. Für eine mögliche Erweiterung um ein GUI für PDA oder Smartphones, sowie für serverseitige REST-Services wird dadurch die Grundlage geboten.

17.1.4.2 Modularität

Mittels einem modularen Aufbau der Software soll die Erweiterung um Schnittstellen oder weiteren Funktionalitäten sichergestellt werden.

17.2 Leistung

Die Erfassung eines Startes soll innerhalb einer Minute ausgeführt werden können, unter der Voraussetzung, dass der Server aktiv ist und die Netzwerkverbindung keinen Engpass darstellt.

17.3 Schnittstellen

In diesem Kapitel werden die Schnittstellen aufgeführt, die zur Aussenwelt bestehen.

17.3.1 Benutzer-Schnittstelle

Diese Schnittstelle erlaubt die Erfassung und Verwaltung von Daten wie Stammdaten und Startlisten. Sie wird mittels einem grafischen User Interface implementiert, welches in einem Webbrowser dargestellt wird. Dafür wird Silverlight 5 und die GUI-Library Infragistics verwendet.

17.3.2 Hardware-Schnittstelle

Die Applikation wird auf den Frameworks von .NET aufbauen und keinen direkten Zugriff auf die Hardware benötigen.

17.3.3 Software-Schnittstelle

Der Client besitzt eine Schnittstelle zum Webbrowser, in dem er gestartet wird. Der Webbrowser muss das Silverlight-Plugin installiert haben.

Es wird eine Schnittstelle zum Buchhaltungssystem benötigt, damit Veränderungen der Mitglieder-Daten vom FLS an die Buchhaltung weitergeleitet werden können. Änderungen von Daten im Buchhaltungssystem, die im FLS ebenfalls erfasst sind, müssen voraussichtlich manuell aktualisiert werden, da solche Änderungen weniger häufig sind.

17.3.4 Datenbank-Schnittstelle

Eine Schnittstelle zu einem MS SQL Server 2008 R2 muss erstellt werden, welcher die Daten der Applikation speichert.

17.3.5 Kommunikations-Schnittstelle

Die Kommunikation zwischen Clients und Server muss über eine zuverlässige Kommunikations-Schnittstelle erfolgen. Das .NET-WCF-Framework bietet eine Webservice-Schnittstelle,

die für solche Anwendungen gedacht ist.

17.4 Lizenzanforderungen

Für die Entwicklung von FLS werden folgende Lizenzen benötigt:

- Infragistics GUI Library (es wurde eine Trial-Version verwendet)
- Visual Studio 2010 (von der HSR bereitgestellt)
- Entreprise Architect (von der HSR bereitgestellt)
- Resharper (von der HSR bereitgestellt)

17.5 Verwendete Standards

- Silverlight 5 mit Infragistics GUI Library
- .NET Framework 4.0
- MS SQL-Server 2008 R2
- Entity Framework 4.1

18 Use Cases

18.1 Use Case Diagram

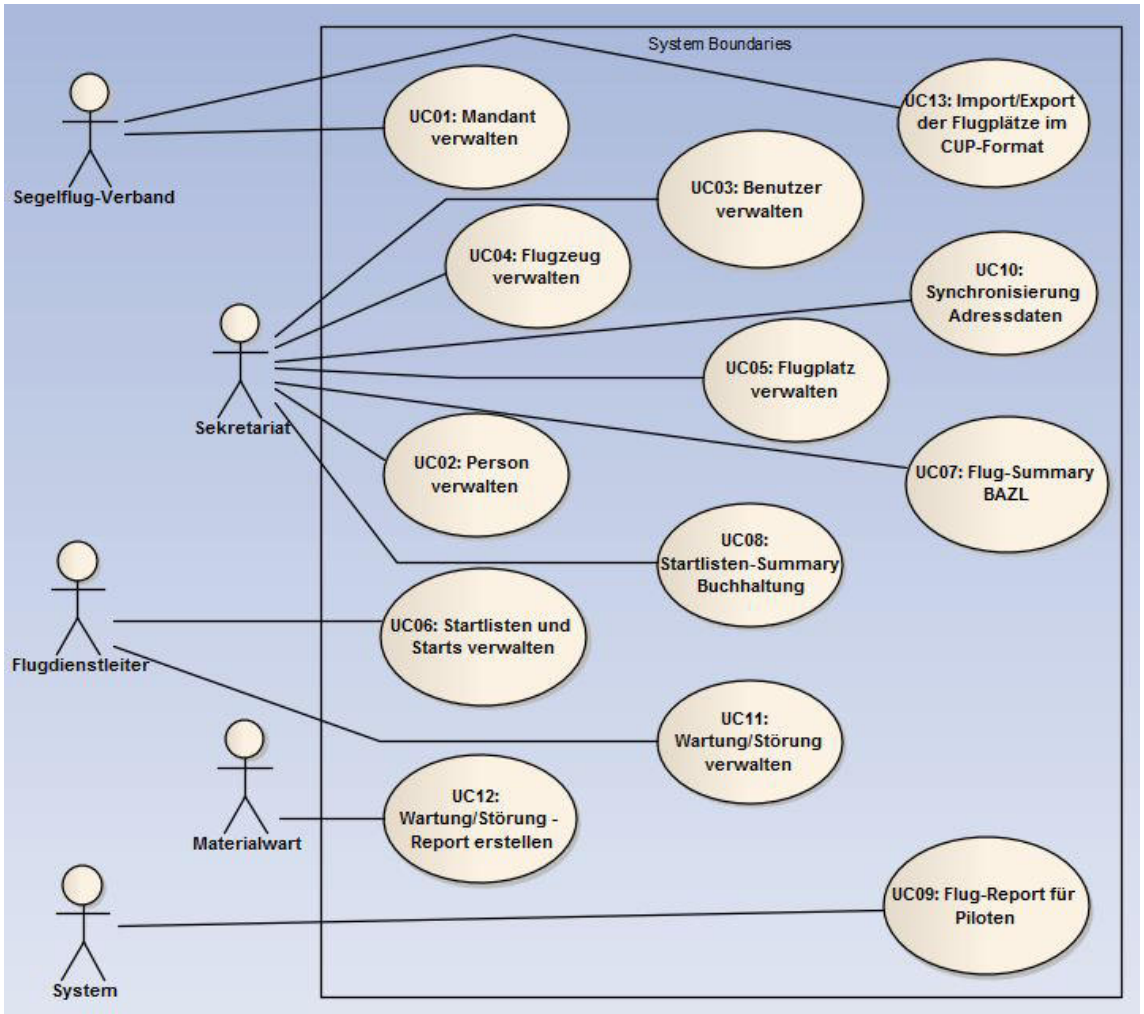


Abbildung 18.1: Use Case Diagram

Aktor	Beschreibung
Flugdienstleiter	Ist auf dem Flugplatz und erfasst die Starts, meistens während eines ganzen Tages.
Sekretariat	Verwaltet die Stammdaten und macht die Buchhaltung eines Vereins.
Pilot	Führt einen Flug mit einem Segelflugzeug aus.
Segelflug-Verband	Erstellt und verwaltet die Mandanten.
Materialwart	Wartet die Flugzeuge.

Stakeholder	Beschreibung
BAZL	Will eine monatliche Statistik über die Flugbewegungen in einem vordefinierten Excel-Formular.
Pilot	Möchte eine Liste seiner ausgeführten Flüge während des Tages, falls solche stattgefunden haben.
Buchhaltung	Möchte eine monatliche Aufstellung für die Verrechnung der Flüge an die Piloten.

18.2 Aktoren & Stakeholders

18.3 Use Case brief

Die Use Cases sind priorisiert, wobei Priorität 4 als optionales Feature eingestuft wird und nur bei gutem Projektverlauf umgesetzt wird.

Voraussetzung: Bei jedem Use Case loggt sich der Aktor zuerst im System ein und hat die notwendigen Berechtigungen.

Priorität	Use Case	Beschreibung
1	UC01: Mandant verwalten	Der Segelflug-Verband erstellt einen neuen Mandanten, ändert die Daten eines Mandanten oder löscht einen.
1	UC02: Person verwalten	Das Sekretariat erfasst eine neue Person. Weiter kann das Sekretariat Daten der Person ändern oder löschen.
1	UC03: Benutzer verwalten	Das Sekretariat oder der Segelflug-Verband erfasst einen neuen Benutzer und definiert dessen Berechtigungen im System. Weiter können die Daten des Benutzers geändert oder gelöscht werden.
1	UC04: Flugzeug verwalten	Das Sekretariat erfasst ein neues Flugzeug, ändert die Daten eines Flugzeuges oder löscht eines.
1	UC05: Flugplatz verwalten	Das Sekretariat erfasst einen neuen Flugplatz, ändert die Daten eines Flugplatzes oder löscht einen.
1	UC06: Startliste und Flüge verwalten	Der Flugdienstleiter erfasst Flüge, startet und landet diese.
2	UC07: Flug-Summary BAZL	Das System erstellt automatisch den monatlichen Report für das BAZL und sendet diesen an den Club. Das Sekretariat erstellt jederzeit einen Report für das BAZL.
2	UC08: Startlisten-Summary Buchhaltung	Das System erstellt automatisch das Startlisten-Summary und sendet dieses an den Club. Das Sekretariat erstellt jederzeit ein Startlisten-Summary über einen frei gewählten Zeitraum.
2	UC09: Flug-Report für Piloten	Täglich werden automatisch Reports für die Piloten generiert und per E-Mail verschickt. Der Pilot kann jederzeit einen Report über einen frei gewählen Zeitraum erstellen.
3	UC10: Synchronisierung Adressdaten	Das Sekretariat erstellt eine Datei mittels eines Reports, welcher in das Buchhaltungssystem importiert werden kann, damit die Daten des Buchhaltungssystems aktualisiert werden.
4	UC11: Wartung/Störung verwalten	Der Flugdienstleiter erfasst eine neue Wartung/Störung, ändert eine oder löscht eine.
4	UC12: Wartung/Störung-Report erstellen	Der Materialwart erstellt einen Report der erfassten Wartungen/Störungen über einen gewissen Zeitraum oder für ein Segelflugzeug.
2	UC13: Import/Export der Flugplätze im CUP-Format	Der Segelflug-Verband importiert das CUP-File von See You, um die Flugplätze im FLS zu aktualisieren. Export der aktuell im FLS erfassten Flugplätze ist für jeden Benutzer möglich.

18.4 Use Case fully dressed

18.4.1 UC01: Mandant verwalten

Primärakteur	Segelflug-Verband
Stakeholder und Interessenten	Segelflug-Verband: • Will eine übersichtliche und schnelle Eingabe und Bearbeitungsmöglichkeit des Mandanten. Sekretariat: • Will seinem Verein einen eigenen Mandanten zugeteilt bekommen.
Vorbedingungen	• Der Segelflug-Verband ist identifiziert und authentifiziert und hat Verwaltungsrechte auf den zu bearbeitenden Mandanten.
Nachbedingungen	Der Verein hat einen eigenen Mandanten und das Sekretariat dieses Vereins hat die Verwaltungsrechte für diesen Mandanten.
Standardablauf	1. Ein neuer Verein möchte die Applikation benutzen. 2. Der Segelflug-Verband startet die Mandantenerfassung. 3. Der Segelflug-Verband gibt den Namen, die Adresse und die Kontaktdaten des Vereins ins System ein. 4. Der Segelflug-Verband speichert den Mandanten. 5. Der Segelflug-Verband erfasst einen neuen Benutzer für diesen Mandant: UC03 Benutzer verwalten.

Erweiterte oder alternative Abläufe	3a. Es ist bereits ein Verein mit diesem Namen im System erfasst. 1. Das System signalisiert einen Fehler. 2. Der Segelflug-Verband klärt ab, ob dieser Verein bereits einen Mandanten besitzt. 2a. Der Verein besitzt bereits einen Mandanten. 1. Der Segelflug-Verband bricht die Transaktion ab. 2b. Der Segelflug-Verband hat einen falschen Namen eingegeben. 1. Der Segelflug-Verband gibt den korrekten Namen ein.
Spezielle Anforderungen	Keine
Liste der Technik und Datenvariationen	Sämtliche Angaben werden über Tastatur und Maus eingegeben.
Häufigkeit des Auftretens	selten
Offene Fragen	Keine

18.4.2 UC02 Person verwalten

Primärakteur	Sekretariat
Stakeholder und Interessenten	Sekretariat: • Will eine übersichtliche und schnelle Eingabe und Bearbeitungsmöglichkeit. Pilot: • Will vom Sekretariat im System erfasst werden.
Vorbedingungen	Das Sekretariat ist identifiziert und authentifiziert und hat Verwaltungsrechte auf den zu bearbeitenden Mandanten.
Nachbedingungen	Pilot ist im System erfasst.

Standardablauf

1. Ein Pilot tritt dem Verein bei. Aus diesem Grund möchte ihn das Sekretariat im System erfassen.
2. Das Sekretariat startet die Personenverwaltung.
3. Das Sekretariat gibt Vorname und Nachname ein und wählt das Land aus.
4. Das Sekretariat wählt aus, ob der Pilot ein Motorfliegerpilot, ein Schlepppilot und/oder ein Segelfluglehrer ist.
5. Das Sekretariat speichert die bisher eingegebenen Daten.

Erweiterte oder alternative Abläufe

3a. Das Sekretariat möchte zusätzlich eine oder mehrere der folgenden Informationen erfassen: Mitgliedernummer, zweiter Vorname, Arbeitsort, Adresse, Zip, Ort, Region, Telefonnummern, Faxnummer, E-Mail-Adresse, Geburtsdatum.
   1. Das Sekretariat gibt die gewünschten Informationen ein.
      1a. Es wurde ein ungültiges Geburtsdatum eingegeben
         1. Das System signalisiert einen Fehler.
         2. Das Sekretariat gibt das korrekte Geburtsdatum ein.

4a. Der Pilot ist ein Segelfluglehrer.
   1. Das Sekretariat gibt die Segelflieger-Instruktor-Nummer des Piloten ein.

5a. Das Sekretariat hat nicht alle im Standardablauf erwähnten Informationen eingegeben.
   1. Das System signalisiert einen Fehler.
   2. Das Sekretariat erfasst die fehlenden Daten.

5b. Es soll für den Piloten ein Benutzer angelegt werden.
   1. Das Sekretariat führt den Use Case UC03 Benutzer verwalten durch, um für den Piloten einen Benutzer zu erstellen.

Spezielle Anforderungen	Keine
Liste der Technik und Datenvariationen	Sämtliche Angaben werden über Tastatur und Maus eingegeben.
Häufigkeit des Auftretens	regelmässig
Offene Fragen	Keine

18.4.3 UC03 Benutzer verwalten

Primärakteur	Sekretariat
Stakeholder und Interessenten	Sekretariat: • Will eine übersichtliche und schnelle Eingabe und Bearbeitungsmöglichkeit. Pilot: • Will einen Benutzer erhalten, um sich mit diesem ins System einloggen zu können.
Vorbedingungen	Das Sekretariat ist identifiziert und authentifiziert und hat Verwaltungsrechte auf dem zu bearbeitenden Mandanten.
Nachbedingungen	Der Benutzer ist im System erfasst.
Standardablauf	1. Das Sekretariat startet die Benutzerverwaltung. 2. Das Sekretariat erfasst den gewünschten Benutzernamen. 3. Das Sekretariat erfasst eine E-Mail-Adresse, an die Benachrichtigungen für den Benutzer geschickt werden können. 4. Das Sekretariat speichert den Benutzer. 5. Das System schickt ein generiertes Passwort und den Username an die vorher eingegebene E-Mail Adresse. 6. Der Pilot meldet sich mit dem mit den erhaltenen Logindaten am System an um diese zu überprüfen.

Erweiterte oder alternative Abläufe	2a. Der Benutzername wird bereits im System verwendet. 1. Das System signalisiert einen Fehler. 2. Das Sekretariat gibt einen neuen Benutzernamen ein. 3a. Der Benutzer wird für eine im System angelegte Person erfasst und es soll eine bereits für diese Person erfasste E-Mail Adresse verwendet werden. 1. Das System präsentiert dem Sekretariat die erfassten E-Mail Adressen 2. Das Sekretariat wählt die E-Mail Adresse aus. 3b. Der Benutzer wird für keine bereits im System angelegte Person erfasst, oder es soll keine der bisher erfassten E-Mail Adressen dieser Person verwendet werden. 1. Das Sekretariat gibt die gewünschte E-Mail Adresse ins System ein. 6a. Der Pilot hat falsche Logindaten eingegeben. 1. Das System signalisiert einen Fehler. 2. Der Pilot gibt die richtigen Logindaten ein.
Spezielle Anforderungen	Keine
Liste der Technik und Datenvariationen	Sämtliche Angaben werden über Tastatur und Maus eingegeben.
Häufigkeit des Auftretens	Einmal pro gewünschten Benutzer
Offene Fragen	Keine

18.4.4 UC04 Flugzeug verwalten

Primärakteur	Sekretariat
Stakeholder und Interessenten	Sekretariat: • Will eine übersichtliche und schnelle Eingabe und Bearbeitungsmöglichkeit. • Will dass ein neues Flugzeug im System zur Verfügung steht.
Vorbedingungen	Das Sekretariat ist identifiziert und authentifiziert und hat Verwaltungsrechte auf den zu bearbeitenden Mandanten.
Nachbedingungen	Ein neues Flugzeug ist im System erfasst.

Standardablauf	1. Es wurde ein neues Flugzeug gekauft. Das Sekretariat möchte dieses im System erfassen. 2. Das Sekretariat startet die Flugzeugerfassung. 3. Das Sekretariat gibt den Namen das Flugzeugmodels, die Immatrikulation, die Anzahl Plätze und die Flarm-Id ein. 4. Das Sekretariat teilt dem Flugzeug die Zugehörigkeit zu. 5. Das Sekretariat speichert die erfassten Daten ab.
Erweiterte oder alternative Abläufe	3a. Das Sekretariat möchte zusätzlich ein Wettbewerbs-Zeichen, den Daec-Index oder einen Kommentar eingeben. 1. Das Sekretariat gibt die entsprechende Information ein. 3b. Das Flugzeug kann nur geschleppt werden. 1. Das Sekretariat markiert das Flugzeug als 'muss geschleppt werden'. 3c. Das Flugzeug ist ein Schleppflugzeug. 1. Das Sekretariat markiert das Flugzeug als 'Schleppflugzeug'. 4a. Das Flugzeug ist ein eigenes Flugzeug. 1. Das Sekretariat definiert den Club, dem das Flugzeug gehört. 4b. Das Flugzeug ist kein eigenes Flugzeug. 1. Das Sekretariat definiert die Person, der das Flugzeug gehört. 5a. Das Sekretariat hat nicht alle im Standardablauf erwähnten Informationen eingegeben. 1. Das System signalisiert einen Fehler. 2. Das Sekretariat erfasst die fehlenden Daten.
Spezielle Anforderungen	Keine
Liste der Technik und Datenvariationen	Sämtliche Angaben werden über Tastatur und Maus eingegeben.
Häufigkeit des Auftretens	selten
Offene Fragen	Keine

18.4.5 UC05 Flugplatz verwalten

Primärakteur	Sekretariat
Stakeholder und Interessenten	Sekretariat: • Will eine übersichtliche und schnelle Eingabe und Bearbeitungsmöglichkeit. Pilot: • Will dass der Flugplatz von dem er startet im System erfasst ist.
Vorbedingungen	Das Sekretariat ist identifiziert und authentifiziert und besitzt die Verwaltungsrechte für den eigenen Verein.
Nachbedingungen	Der Flugplatz ist im System erfasst.
Standardablauf	1. Das Sekretariat startet die Ortserfassung 2. Das Sekretariat gibt den Namen des Ortes ein, wählt das Land aus in dem er sich befindet und wählt den Typ des Ortes aus. 3. Das Sekretariat speichert den Ort.
Erweiterte oder alternative Abläufe	2a. Es sind Informationen über Kurzname, Icao-Code, Längengrad, Breitengrad, Höhe, Pistenrichtung, Pistenlänge, Flughafenfrequenz und/oder Beschreibung vorhanden. 1. Das Sekretariat gibt die Informationen ins System ein.
Spezielle Anforderungen	Keine
Liste der Technik und Datenvariationen	Sämtliche Angaben werden über Tastatur und Maus eingegeben.
Häufigkeit des Auftretens	Einmal pro benutzten Lande/Start-Ort
Offene Fragen	Keine

18.4.6 UC06 Startliste und Flüge verwalten

Primärakteur	Flugdienstleiter
Stakeholder und Interessenten	Flugdienstleiter: • Will eine übersichtliche und schnelle Eingabe und Bearbeitungsmöglichkeit. • Will die Starts und Landungen der Segelflüge erfassen. Pilot: • Will dass seine ausgeführten Flüge im System erfasst werden, damit sie im täglichen Report erscheinen. Buchhaltung: • Will dass alle Flüge im System erfasst sind, damit sie im monatlichen Auszug erscheinen. BAZL: • Will dass alle Flüge im System erfasst sind, damit sie in der monatlichen Statistik erscheinen.
Vorbedingungen	Der Flugdienstleiter ist identifiziert und authentifiziert und hat Verwaltungsrechte auf den zu bearbeitenden Mandanten.
Nachbedingungen	Ein neuer Flug ist im System erfasst.
Standardablauf	1. Ein Pilot möchte einen Segelflug starten und positioniert einen Segelflieger auf der Startbahn. 2. Der Flugdienstleiter wählt das verwendete Flugzeug, den Pilot, die Startart und die Flugart aus. 3. Der Segelflieger startet. 4. Der Flugdienstleiter erfasst die Startzeit. 5. Der Segelflieger landet. 6. Der Flugdienstleiter erfasst die Landezeit.

Erweiterte oder alternative Abläufe	1a. Der Segelflieger muss geschleppt werden. 1. Ein weiterer Pilot positioniert einen Schlepper auf der Startbahn. 2. Es wird der Use Case 'UC06 Startliste und Flüge verwalten' ausgeführt um den Schleppflieger zu erfassen. 1b. Der Segelflieger startet mittels Windenstart. 1. Der Pilot hackt das Windenseil im Flugzeug ein. 2a. Es ist ein Passagier- oder Lehrflug. 1. Der Flugdienstleiter erfasst den Namen des Passagiers oder des Instruktors. 3a. Der Segelflieger wurde geschleppt. 1. Das Schleppflugzeug landet wieder. 2. Der Flugdienstleiter erfasst die Motorlaufzeit und die Landezeit des Schleppers. 3-5a. Der Flugdienstleiter muss vor der Landung des Segelfliegers einen anderen Flug erfassen. 1. Der Flugdienstleiter speichert die bisherigen Eingaben und startet eine neue Flugerfassung. 6b. Der Flugdienstleiter hat diesen Flug nicht mehr offen. 1. Der Flugdienstleiter öffnet den gespeicherten Flug. 6c. Der Flieger muss vorher zwischenlanden. 1. Der Pilot notiert sich die Start- und Landezeiten bei der Zwischenlandung. 2. Der Pilot lässt sich von einem Schleppflugzeug eines anderen Vereins wieder in die Luft ziehen. 3. Der Pilot landet auf dem eigenen Flugplatz. 4. Der Pilot teilt dem Flugdienstleiter mit, dass er zwischengelandet ist. 5. Der Flugdienstleiter erfasst die Landezeit und den Landeort der Zwischenlandung und schliesst den Flug ab. 6. Der Flugdienstleiter erfasst einen neuen Flug mit der Startzeit und dem Ort der Zwischenlandung und der Landezeit und dem Landeort am eigenen Flugplatz.
Spezielle Anforderungen	Keine
Liste der Technik und Datenvariationen	Sämtliche Angaben werden über Tastatur und Maus eingegeben.
Häufigkeit des Auftretens	Häufig: Für jedem Flug
Offene Fragen	Keine

18.4.7 UC07 Flug-Summary BAZL

Primärakteur	System
Stakeholder und Interessenten	Sekretariat: • Will monatlich eine Zusammenfassung über alle Flüge erhalten.
Vorbedingungen	Das Sekretariat hat einen Benutzer und dieser hat die Verwaltungsrechte über den Mandaten.
Nachbedingungen	Das Sekretariat hat einen Report erhalten.
Standardablauf	1. Der erste Tag eines Monats wurde erreicht. 2. Das System generiert einen Zusammenfassung über alle Flüge des letzen Monats. Dabei wird angegeben, wie viele Flugzeugschleppstarts, Windenstarts, Eigenstarts und Total Starts an der Homebase des Vereins gemacht wurden und wie viele Landungen es gegeben hat. 3. Das System schickt den Report an das Sekretariat
Erweiterte oder alternative Abläufe	1a. Jederzeit 1. Das Sekretariat möchte manuell einen Flug-Summary BAZL erstellen 1. Das Sekretariat loggt sich im FLS ein. 2. Das Sekretariat startet die Report-Generierung. 3. Das Sekretariat wählt den gewünschten Monat und den Report Flug-Summary aus. 4. Das Sekretariat öffnet den Report.
Spezielle Anforderungen	-
Liste der Technik und Datenvariationen	Der Report wird per E-Mail verschickt.
Häufigkeit des Auftretens	-
Offene Fragen	Keine

18.4.8 UC08 Startlisten-Summary Buchhaltung

Primärakteur	System
Stakeholder und Interessenten	Sekretariat: • Will monatlich eine Zusammenfassung über alle Flüge erhalten.
Vorbedingungen	Das Sekretariat hat einen Benutzer und dieser hat die Verwaltungsrechte über den Mandaten.
Nachbedingungen	Das Sekretariat hat den Report erhalten.
Standardablauf	1. Der erste Tag eines Monats wurde erreicht. 2. Das System generiert einen Report über alle Flüge des letzten Monats. Der Report ist unterteilt nach Personen. Für jeden Flug wird der Flugtyp mit Anzahl Flugminuten und allenfalls der Schleppflug mit den Schleppminuten aufgeführt. Ebenfalls aufgeführt ist, ob die Landung/Start am Heimflughafen oder Auswärts erfolgte. Zusätzlich wird am Ende des Raports eine Zusammenfassung über alle Landungen/Starts von Auswärtigen Vereinen aufgeführt. 3. Das System schickt den Report an das Sekretariat.
Erweiterte oder alternative Abläufe	Keine 1a. Jederzeit 1. Das Sekretariat möchte manuell ein Startlisten-Summary erstellen 1. Das Sekretariat loggt sich im FLS ein. 2. Das Sekretariat startet die Report-Generierung. 3. Das Sekretariat wählt den gewünschten Monat und den Report Startlisten-Summary aus. 4. Das Sekretariat öffnet den Report.
Spezielle Anforderungen	-
Liste der Technik und Datenvariationen	Der Report wird per E-Mail verschickt.
Häufigkeit des Auftretens	Einmal pro Monat
Offene Fragen	Keine

18.4.9 UC09 Flug-Report für Piloten

Primärakteur	System
Stakeholder und Interessenten	Pilot: • Will am Abend eines Tages, an dem er einen Flug durchgeführt hat, einen Report erhalten um diese Daten in sein Flugbuch einzutragen.
Vorbedingungen	Der Pilot ist als Person im System erfasst.
Nachbedingungen	Der Pilot hat einen Report erhalten.
Standardablauf	1. Es ist 23 Uhr. 2. Das System erstellt für einen Piloten, der an diesem Tag geflogen ist, einen Flugreport. Dieser enthält den Flugtyp, die Anzahl Flugminuten, allenfalls die Schleppminuten und ob die Landung/Start am Heimflughafen oder Auswärts erfolgte. 3. Der Flugreport wird an den Piloten geschickt. *Der Schritt 2 und 3 wird für jeden Piloten wiederholt, der an diesem Tag geflogen ist.*

Erweiterte oder alternative Abläufe	1a. 1. Jederzeit 1. Der Pilot möchte manuell einen Flug-Report erstellen. 2. Der Pilot loggt sich im FLS ein. 3. Der Pilot startet die Report-Generierung. 4. Der Pilot wählt eine Person aus. 5. Der Pilot wählt das „Report von“ und das „bis“-Datum aus. 6. Der Pilot öffnet den Report. 3a. Im System ist als bevorzugte E-Mail Adresse die private E-Mail Adresse erfasst. 1. Das Mail wird an die private E-Mail Adresse geschickt. 1a. Es ist keine private E-Mail Adresse erfasst 1. Es wird keine E-Mail verschickt. 3b. Im System ist als bevorzugte E-Mail Adresse die geschäftliche E-Mail Adresse erfasst. 1. Das Mail wird an die Geschäfts- E.Mail Adresse geschickt. 1a. Es ist keine private E-Mail Adresse erfasst 1. Es wird keine E-Mail verschickt.
Spezielle Anforderungen	-
Liste der Technik und Datenvariationen	Der Report wird per E-Mail verschickt.
Häufigkeit des Auftretens	Einmal pro Flugtag
Offene Fragen	

18.4.10 UC10 Synchronisierung Adressdaten

Primärakteur	Sekretariat
Stakeholder und Interessenten	Sekretariat: • Will die Adressdaten vom eigenen Verein im Buchhaltungssystem aktualisiert haben.
Vorbedingungen	Das Sekretariat ist identifiziert und authentifiziert und hat Verwaltungsrechte auf den zu bearbeitenden Mandanten.
Nachbedingungen	Das Sekretariat besitzt eine Auflistung der im FLS geänderten Adressdaten seit der letzten Adresssynchronisation.

Standardablauf	1. Das Sekretariat öffnet die Ansicht für den Adress-Export. 2. Das Sekretariat wählt aus, wohin die Daten lokal auf seinem Rechner gespeichert werden soll. 3. Das Sekretariat bestimmt, dass nur Adressdaten mit Änderungen seit der letzten Adress-Synchronisation geliefert werden sollen. 4. Das System generiert die Daten gemäss Vorgaben und speichert sie auf dem Rechner des Benutzers ab. 5. Das Sekretariat holt die Daten an der vorher bestimmten Stelle ab und verwendet diese für die Aktualisierung des Buchhaltungssytems.
Erweiterte oder alternative Abläufe	3a. Das Sekretariat setzt das Datum der letzten Adress-Synchronisation um einen Monat zurück, da die zu jenem Zeitpunkt exportierten Daten nicht verarbeitet wurden. 3b. Das Sekretariat bestimmt, dass alle Adressen exportiert werden, die für den eigenen Verein existieren.
Spezielle Anforderungen	Keine
Liste der Technik und Datenvariationen	Sämtliche Angaben werden über Tastatur und Maus eingegeben.
Häufigkeit des Auftretens	monatlich
Offene Fragen	Keine

18.4.11 UC13 Import/Export der Flugplätze im CUP-Format

18.4.11.1 Import der Flugplätze im CUP-Format

Primärakteur	Sekretariat
Stakeholder und Interessenten	Sekretariat: • Will die Flugplätze / Standorte im FLS gepflegt bzw. aktualisiert haben.
Vorbedingungen	Das Sekretariat ist identifiziert und authentifiziert und hat Verwaltungsrechte auf den zu bearbeitenden Mandanten.

Nachbedingungen	Die Flugplätze / Standorte im FLS sind aktualisiert.
Standardablauf	1. Das Sekretariat öffnet die Ansicht für Import Flugplätze / Standorte. 2. Das Sekretariat wählt aus, woher die Daten importiert werden sollen. 3. Das System verarbeitet die ausgewählten Daten und zeigt an, welche Daten zum Import ausgewählt werden können. 4. Das Sekretariat wählt aus, welche Flugplätze / Standorte importiert werden sollen. 5. Das System verarbeitet die ausgewählten Flugplätze / Standorte und zeigt diese in den existierenden Flugplätze / Standorte zusätzlich an.
Erweiterte oder alternative Abläufe	Keine
Spezielle Anforderungen	Keine
Liste der Technik und Datenvariationen	Keine
Häufigkeit des Auftretens	Bei Neuinstallation, Aktualisierung der Flugplätze / Standorte
Offene Fragen	Keine

18.4.11.2 Export der Flugplätze im CUP-Format

Primärakteur	Sekretariat
Stakeholder und Interessenten	Sekretariat: • Will eine Datei erhalten, die die selektierten Flugplätze im Cup-Format enthält.
Vorbedingungen	Das Sekretariat ist identifiziert und authentifiziert und hat Verwaltungsrechte auf den zu bearbeitenden Mandanten.
Nachbedingungen	Es existiert eine Datei auf dem Rechner des Sekretariat, die die selektierten Daten im Cup-Format enthalten.

Standardablauf

1. Das Sekretariat öffnet die Ansicht für Export Flugplätze / Standorte.
2. Das Sekretariat wählt aus, wohin die Daten exportiert werden sollen und definiert eine Selektion.
3. Das System verarbeitet die ausgewählten Daten und speichert sie auf dem Rechner des Sekretariats ab.

Erweiterte oder alternative Abläufe	Keine
Spezielle Anforderungen	Keine
Liste der Technik und Datenvariationen	Keine
Häufigkeit des Auftretens	Bei Wettbewerben, um Standorte ins Flarm zu importieren
Offene Fragen	Keine

18.5 Spezielle Abläufe und Sub-Use Case

18.5.1 Einloggen

1. Der Benutzer möchte sich einloggen.
   - 1a. Der Benutzer verfügt bereits über einen Login.
     1. Der Benutzer gibt seinen Benutzernamen und sein Passwort ein.
        - 1a. Der Benutzername und das Passwort sind korrekt.
          - 1a. Der Account ist aktiv.
            1. Der Benutzer wird eingeloggt und auf die Willkommensseite weitergeleitet.
            2. Der FailedLoginCount des Accounts wird auf 0 gesetzt.
          - 1b. Der Account ist gesperrt.
            1. Es erscheint die Meldung: 'Ihr Benutzerkonto wurde gesperrt, da zu viele Login-Versuche fehlgeschlagen sind. Bitte wenden Sie sich an den Systemadministrator ihres Vereins.'
          - 1c. Der Account ist deaktiviert.
            1. Es erscheint die Meldung: 'Ihr Benutzerkonto wurde deaktiviert. Bitte wenden Sie sich an den Systemadministrator ihres Vereins.'
          - 1d. Der Account wurde gelöscht.
            1. Es erscheint die Meldung: 'Ihr Benutzerkonto existiert nicht mehr.'

1e. Der Account ist noch nicht aktiv.
1. Des erscheint die Meldung: 'Ihr Benutzerkonto muss aktiviert werden. Bitte wenden Sie sich an den Systemadministrator ihres Vereins.'

1b. Der Benutzername ist korrekt, das Passwort aber falsch.
1a. Der FailedLoginCount des Accounts ist kleiner als 10.
1. Es erscheint die Meldung: 'Login fehlgeschlagen! Bitte überprüfen Sie Ihren Benutzername und Ihr Passwort.'
2. Der FailedLoginCount wird um 1 erhöht.
3. Der Benutzer gibt nochmals seine Logindaten ein.

1b. Der FailedLoginCount des Accounts ist grösser als 10.
1. Es erscheint die Meldung: 'Ihr Benutzerkonto wurde gesperrt, da zu viele Login-Versuche fehlgeschlagen sind. Bitte wenden Sie sich an den Systemadministrator ihres Vereins.'

1c. Der Benutzername ist nicht korrekt.
1. Es erscheint die Meldung: 'Login fehlgeschlagen! Bitte überprüfen Sie Ihren Benutzername und Ihr Passwort.'

2a. Der Benutzer verfügt über keinen Login.
1. Das Sekretariat führt den UC03 Benutzer verwalten aus.

18.5.2 Flugart erfassen

1. Der Benutzer gibt den Flugartnamen ins System ein und wählt aus, ob für diese Art ein Fluglehrer benötigt wird und ob es ein Passagierflug ist.
   2a. Es ist ein Flugcode vorhanden
      1. Der Benutzer erfasst den Flugcode.
2. Der Benutzer speichert den neuen Typ.

X

Domainanalyse

Datum: 15. Juni 2012

19 Domain-Modell

19.1 Strukturdiagramm

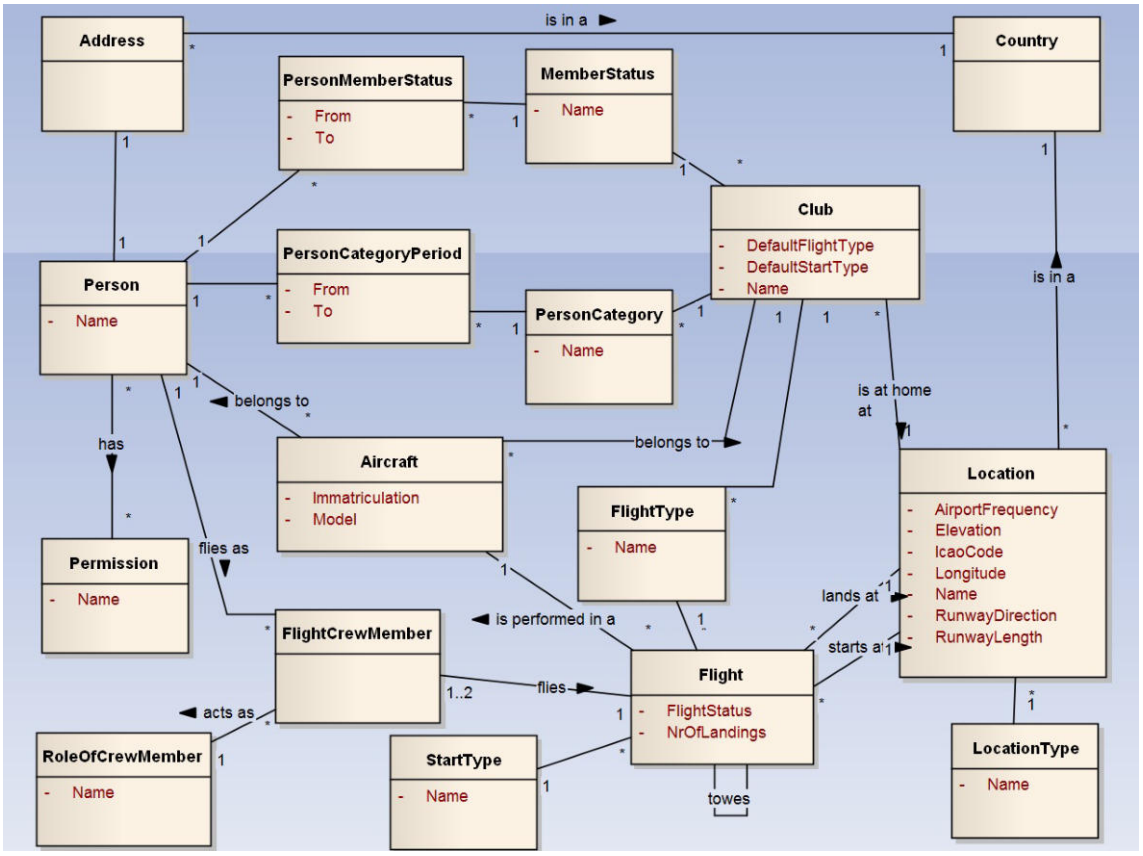


Abbildung 19.1: Domain Model

19.2 Konzeptbeschreibung

Address	
Beschreibung	Adressen der Personen. Diese können zum Beispiel Wohnadressen, oder Lieferadressen sein.
Attribute	
Beziehungen	• Address hat eine Referenz auf Person, welche definiert zu welcher Person die Adresse gehört. • Address hat eine Referenz auf Country, welche definiert in welchem Land diese Adresse ist.

Aircraft	
Beschreibung	Flugzeuge die für die Flüge benutzt werden.
Attribute	• Immatrikulation: Eindeutige Identifizierung eines Flugzeuges weltweit. • Model: Das Model des Fliegers.
Beziehungen	• Jedes Flugzeug hat einen Eigentümer (Club oder Person). • Aircraft hat eine Beziehung zu Flight, da der Flug mit einem Flugzeug geflogen wird.

Club	
Beschreibung	Ein Club widerspiegelt einen Segelflugverein.
Attribute	• Name: Der Name des Vereins. • DefaultFlightType: Die Voreinstellung für den Flugtyp, wenn ein neuer Flug erstellt wird. • DefaultStartType: Die Voreinstellung für den Starttyp, wenn ein neuer Flug erstellt wird.
Beziehungen	• Club hat eine Beziehung zu PersonCatergoryPeriod, da ein Club mit mehreren Personen, die nicht Clubmitglieder sein müssen, interagiert. • Club hat eine Beziehung zu PersonMemberStatus, da ein Club mehrere Mitglieder oder ehemalige Mitglieder mit unterschiedlichen Status hat. Der Status eines Mitgliedes kann sich über die Zeit ändern. • Ein Flug(Flight) gehört zu einem Club, da der Flug über den Club verrechnet wird. • Flugzeuge können einem Club gehören. • Ein Club hat einen Heimatstandort (Location).

Country	
Beschreibung	Ein Land
Attribute	
Beziehungen	• Jede Adresse (Address) befindet sich in einem Land (Country). • Jede Lande/Startmöglichkeit (Location) befindet sich in einem Land (Country).

Flight	
Beschreibung	Ein Flug beinhaltet Start, Flug und Landung.
Attribute	• NrOfLandings: Wie oft der Segelflieger am Zielflughafen gelandet ist. (Dieser Wert ist grösser als 1, wenn das Landen geübt wird) • FlightStatus: Der Status des Fluges: New, Started, Landed, Locked
Beziehungen	• Jeder Flug wird im Rahmen eines Clubs gemacht. • Jeder Flug wird mit einem Flugzeug (Aircraft) durchgeführt. • Jeder Flug startet an einem Ort (Location). • Jeder Flug landet an einem Ort (Location). • Jeder Flug wird von einem oder mehreren FlightCrewMember durchgeführt. • Jeder Flug wird mit einem gewissen Starttyp (StartType) gestartet. • Jeder Flug hat einen Typ (FlightType). • Ein Flug hat eine Beziehung zu einem anderen Flug (der Schleppflug), wenn der Segelflieger geschleppt wird.

FlightCrewMember	
Beschreibung	Jede Person, die einen Flieger fliegt, ist ein FlightCrewMember mit einer bestimmten Rolle in diesem Flug.
Attribute	
Beziehungen	• Ein oder zwei FlightCrewMember fliegen einen Flug (Flight). • Eine Person übernimmt für jeden Flug eine Rolle als FlightCrewMember. • Der FlightCrewMember hat in einem Flug eine bestimmte Rolle. Diese wird mit einer Beziehung zu RoleOfCrewMember spezifiziert.

FlightType	
Beschreibung	Der Flugart definiert in welchem Rahmen (z.B Passagierflug oder Weiterbildungsflug) der Flug durchgeführt wird.
Attribute	
Beziehungen	• FlightType gibt dem Flug(Flight) einen Typ.

Location	
Beschreibung	Eine Location ist ein Ort, an dem gelandet oder gestartet wird. Dies ist meistens ein Flugplatz, kann aber auch ein Acker oder Aussenlandefeld sein.
Attribute	• Name: Die Bezeichnung der Location. • RunwayDirection: Die Himmelsrichtung der Piste. • RunwayLength: Die Länge der Start/Landebahn. • Elevation: Wie viele Meter oder Fuss über Meer sich die Piste befindet. • IcaoCode: Eine internationale Identifizierung von Flugplätze. • Longitude: Der Längengrad der Location. • Latitude: Der Breitengrad der Location. • AirportFrequency: Die Funkfrequenz, die der Flugplatz benutzt.
Beziehungen	• Ein Flug(Flight) startet auf einer Location. • Ein Flug(Flight) landet auf einer Location. • Eine Location kann der Heimatstandort von einem Club sein. • Eine Location hat einen Typ (LocationType). • Eine Location befindet sich in einem Land (Country).

LocationType	
Beschreibung	Beschreibt den Typ einer Location. Beispiele dazu sind: Aussenlandefeld oder Flugplatz mit Graspiste.
Attribute	
Beziehungen	• Ein LocationType gibt mehreren Locations einen Typ.

MemberStatus	
Beschreibung	Ein Status, den eine Person in einem Club haben kann.
Attribute	• Name: Der Name des Status.
Beziehungen	• MemberStatus hat eine Beziehung zu PersonMemberStatus. Damit kann einer Person ein Status in einem Club zugewiesen werden.

Permission	
Beschreibung	Berechtigung für eine Person, gewisse Rollen, wie zum Beispiel Motorflieger oder Segelfluglehrer, zu übernehmen.
Attribute	• Name: Der Name der Person.
Beziehungen	• Eine Person kann mehrere Berechtigungen haben.

Person	
Beschreibung	Eine Person.
Attribute	• Name: Der Name der Person.
Beziehungen	• Eine Person hat eine Adresse. • Person hat eine Beziehung zu PersonCategoryPeriod, da eine Person mit Clubs interagieren kann. • Person hat eine Beziehung zu PersonMemberStatus, da eine Person in mehreren Clubs Mitglied sein kann. Diese Mitgliedschaft kann sich über die Zeit ändern. • Die Person fliegt einen Flug in der Rolle eines FlightCrewMembers. • Eine Person kann mehrere Berechtigungen (Permission) haben. (Motorflieger fliegen usw.)

PersonCategory	
Beschreibung	Ein Vereinsmitglied(Person) nimmt eine bestimmte Rolle im Club ein.
Attribute	• Name: Der Name der Kategorie.
Beziehungen	• PersonCategory hat eine Beziehung zu PersonCategoryPeriod, da eine Person über einen gewissen Zeitraum mit dieser Rolle mit einem Verein interagieren kann.

PersonCategory-Period	
Beschreibung	Definiert einen Zeitraum, über den eine Person mit einem Verein in einer gewisse Rolle (PersonCategory) interagiert.
Attribute	• From: Das Startdatum der Interaktion. • To: Das Enddatum der Interaktion, welches offen sein kann.
Beziehungen	• PersonCategoryPeriod hat eine Beziehung zu Person. • PersonCategoryPeriod hat eine Beziehung zu Club. • PersonCategoryPeriod hat eine Beziehung zu PersonCategory.

RoleOfCrewMember	
Beschreibung	Beschreibt, welche Rolle eine Person auf einem Flug einnimmt. Beispiele dazu sind Pilot, Copilot, Fluglehrer, Passagier und Instruktor.
Attribute	
Beziehungen	• Definiert die Rolle eines FlightCrewMembers.

StartType	
Beschreibung	Definiert, wie ein Flug gestartet wird. Beispiele sind Flugzeugschlepp, Windenstart oder Eigenstart.
Attribute	
Beziehungen	• Ein StartType kann mehrere Flüge typisieren.

20 System Sequenzdiagramme

20.1 Benutzer erfassen

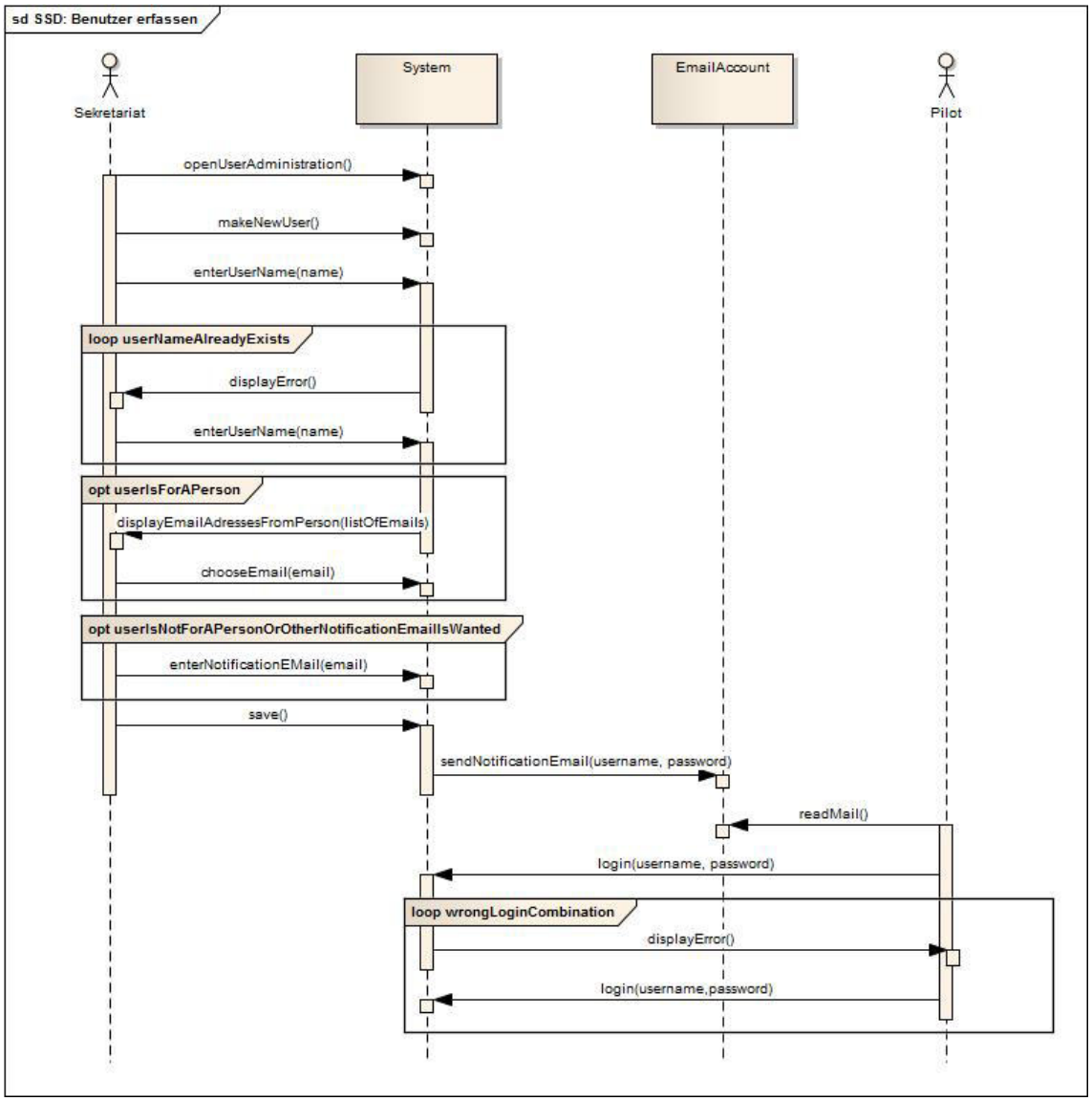


Abbildung 20.1: SSD: Benutzer erfassen

20.2 Flugzeug erfassen

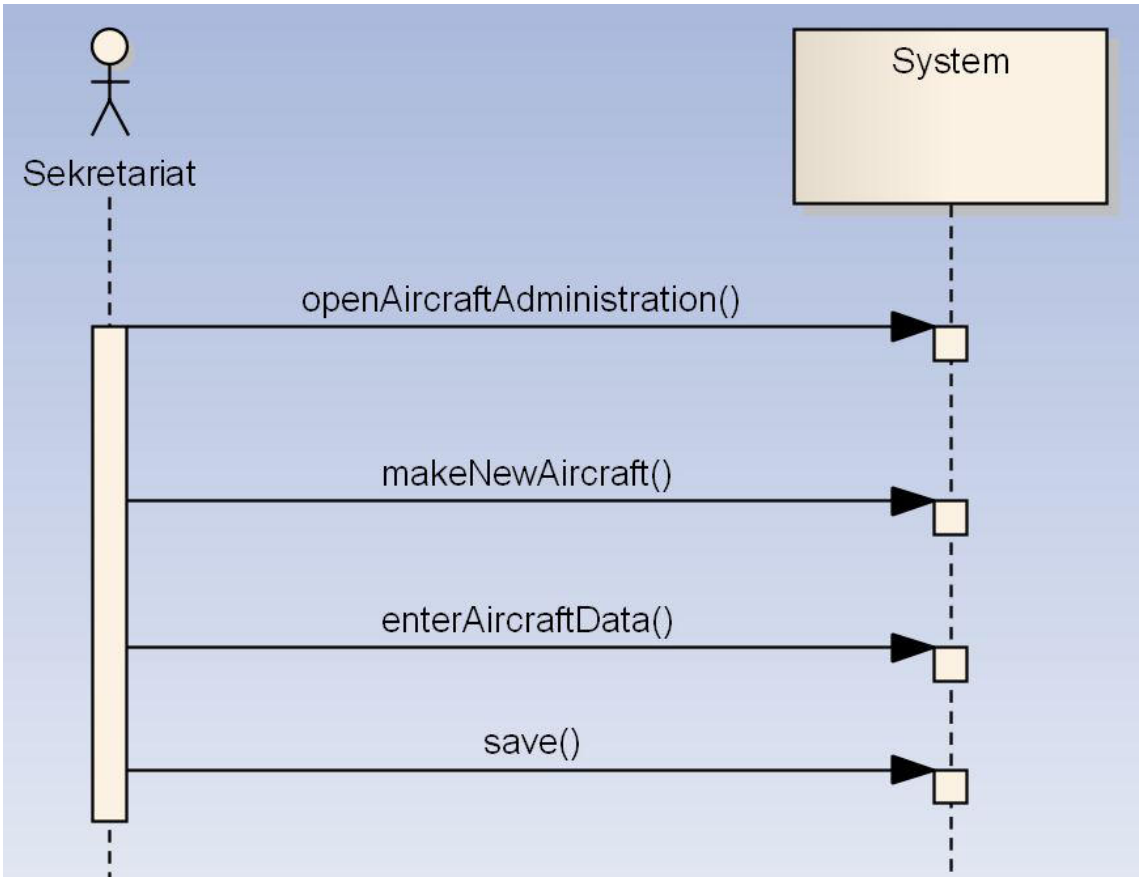


Abbildung 20.2: SSD Flugplatz erfassen

20.3 Flugplatz erfassen

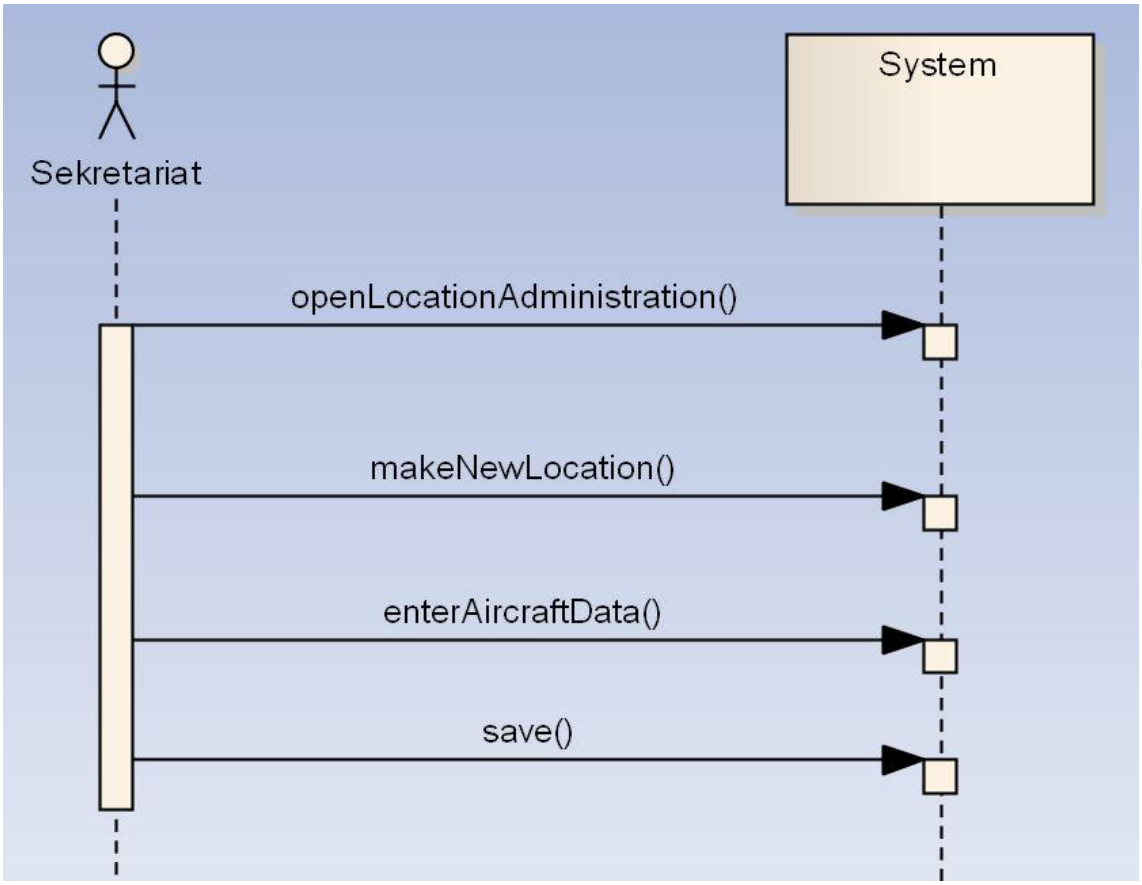


Abbildung 20.3: SSD Flugzeug erfassen

20.4 Mandant erfassen

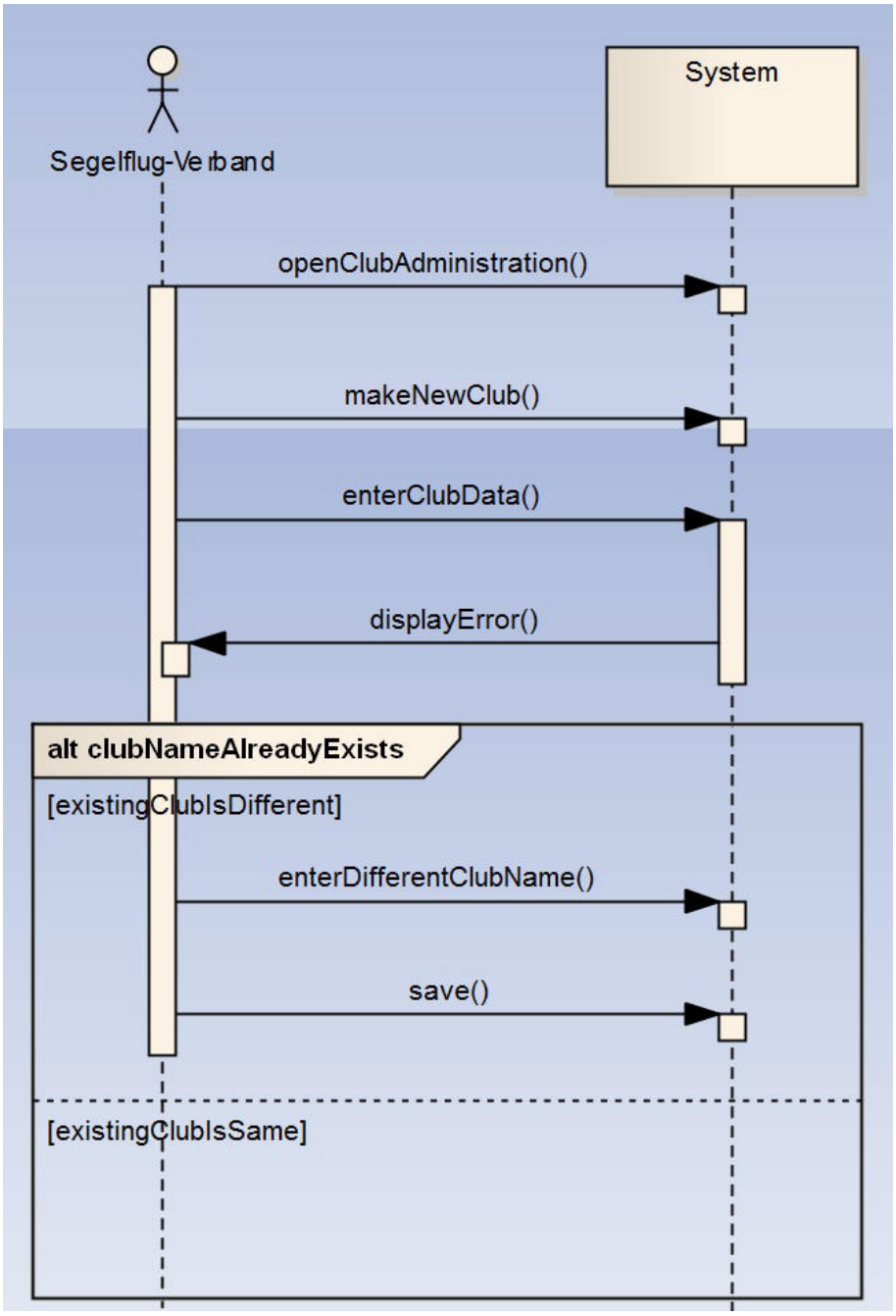


Abbildung 20.4: SSD Mandant erfassen

20.5 Person erfassen

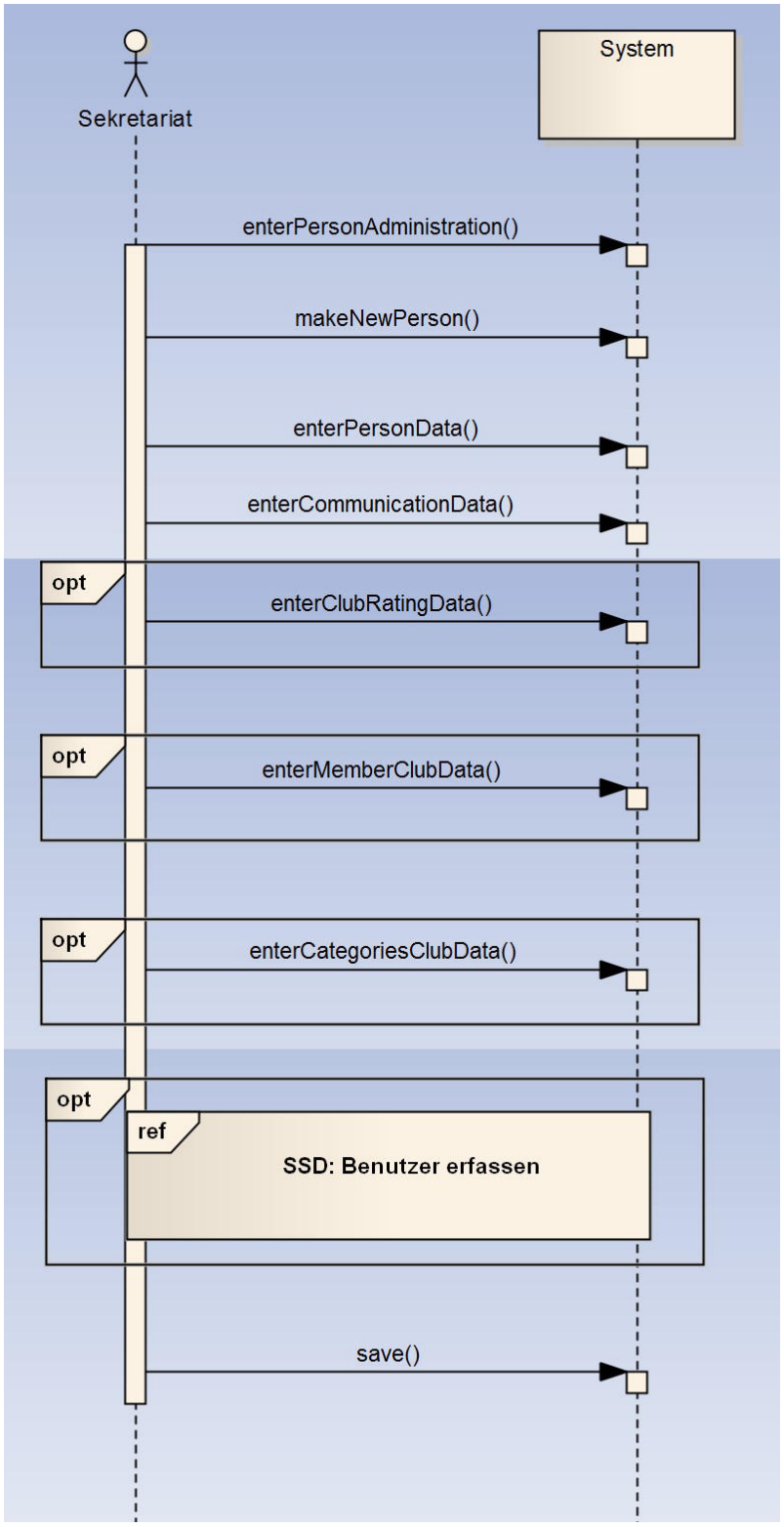


Abbildung 20.5: SSD Person erfassen

20.6 Startliste und Flüge erfassen

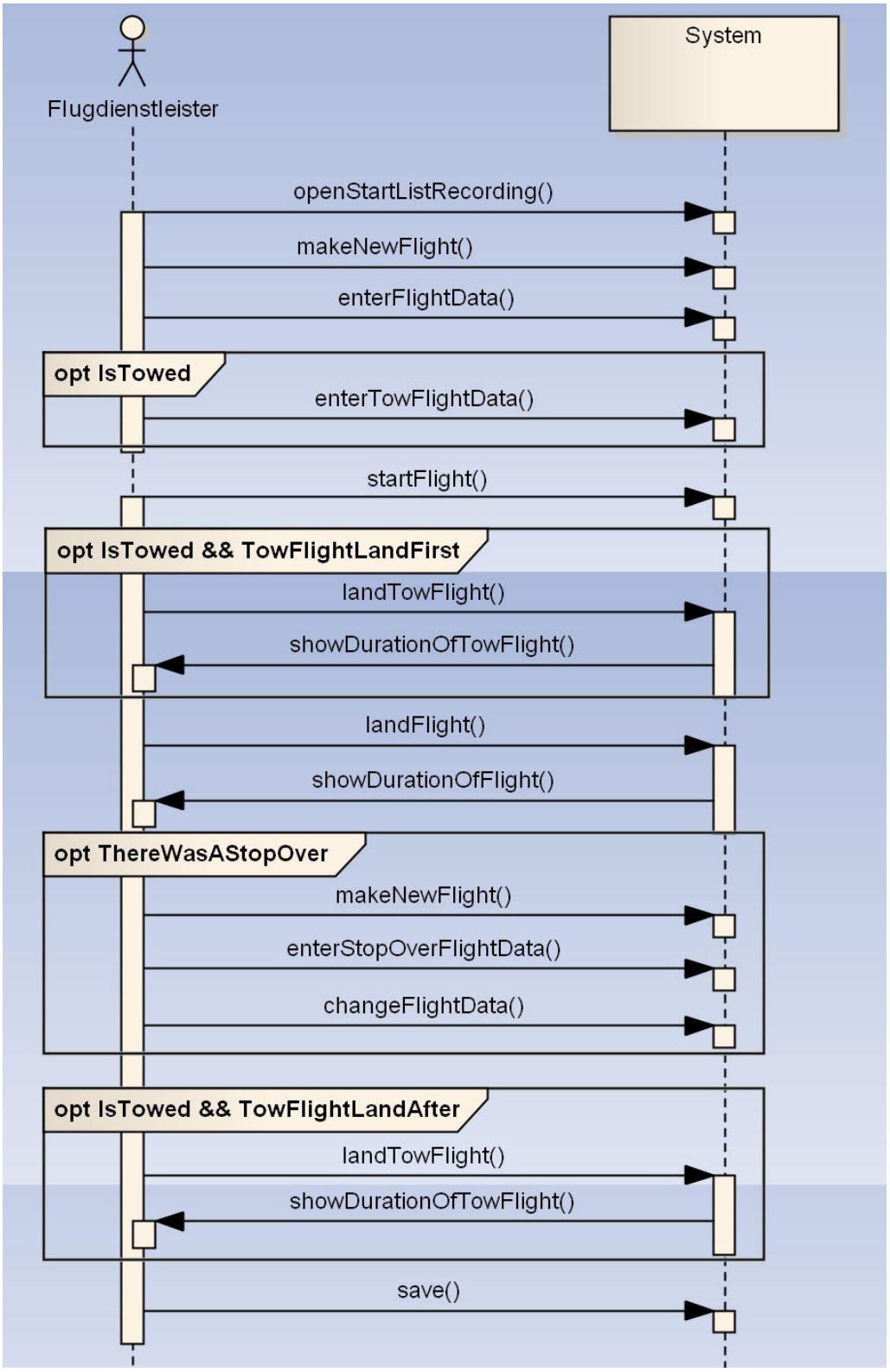


Abbildung 20.6: SSD Startliste und Flüge erfassen

20.7 Startlisten Summary Buchhaltung

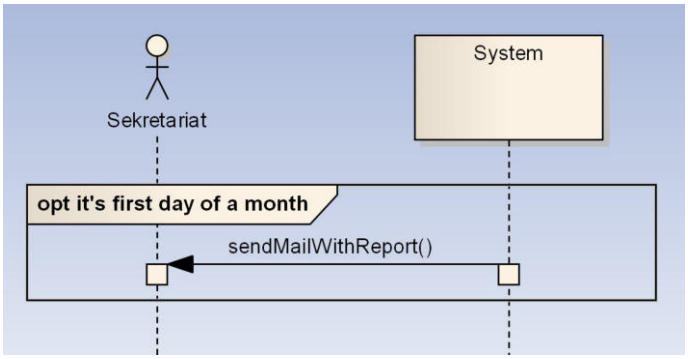


Abbildung 20.7: SSD Startliste und Flüge erfassen

20.8 Flugreport für Piloten

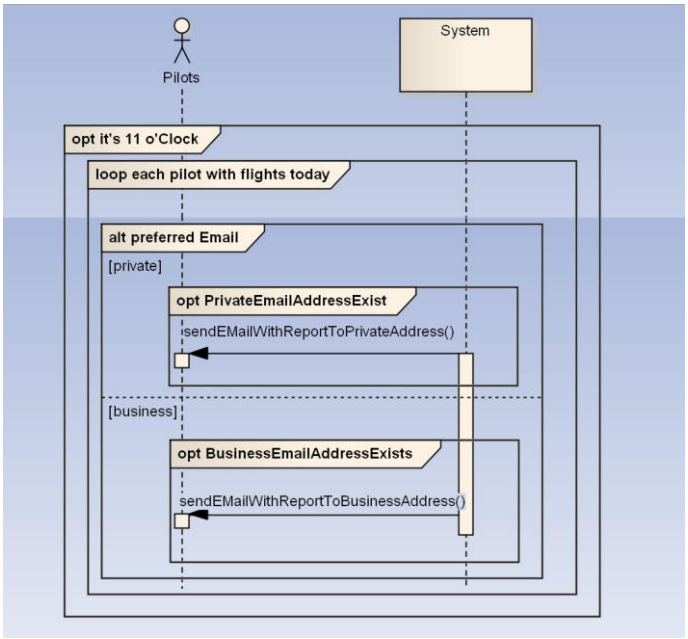


Abbildung 20.8: SSD Flugreport für Piloten

21 Systemoperationen

21.1 Contract CO1: MakeNewFlight

Operation	**MakeNewFlight**
Querverweise	Use Cases: UC06 Startliste und Flüge verwalten
Vorbedingung	• Es ist ein Benutzer eingeloggt, welcher einem Verein zugewiesen ist. • Der Verein hat einen Standard Flugtyp und einen Standard Starttyp gespeichert.
Nachbedingungen	• Eine Instanz flight vom Typ Flight wurde erstellt. • flight.StartType wurde mit Verein.DefaultStartType des eingeloggten Benutzers assoziiert. • flight.FlightType wurde mit Verein.DefaultFlightType des eingeloggten Benutzers assoziiert. • flight.FlightStatus wurde auf New gesetzt. • flight.LdgLocation wurde mit Club.Homebase des eingeloggten Users assoziiert, falls Club.Homebase gesetzt. • flight.StartLocation wurde mit Club.Homebase des eingeloggten Users assoziiert, falls Club.Homebase gesetzt. • Falls der Standard-Starttyp 'TowingByAircraft' ist: – Es wurde eine Instanz towFlight vom Typ Flight erstellt. – towFlight wurde mit flight assoziiert. – towFlight.FlightStatus wurde auf New gesetzt. – towFlight.LdgLocation wurde mit Club.Homebase des eingeloggten Users assoziiert, falls Club.Homebase gesetzt. – towFlight.StartLocation wurde mit Club.Homebase des eingeloggten Users assoziiert, falls Club.Homebase gesetzt.

21.2 Contract CO2: StartFlight

Operation	StartFlight
Querverweise	Use Cases: UC06 Startliste und Flüge verwalten
Vorbedingung	• Ein Flug flight existiert und ist im FlightStatus New
Nachbedingungen	• flight.StartTime wurde auf die aktuelle Zeit gesetzt. • flight.FlightStatus wurde auf Started gesetzt. • Falls flight.Starttyp 'TowingByAircraft' ist: – flight.towFlight.StartTime wurde auf die aktuelle Zeit gesetzt. – flight.towFlight.FlightStatus wurde auf Started gesetzt.

21.3 Contract CO3: LandFlight

Operation	LandFlight
Querverweise	Use Cases: UC06 Startliste und Flüge verwalten
Vorbedingung	• Ein Flug flight existiert und ist im FlightStatus Started
Nachbedingungen	• flight.LandingTime wurde auf die aktuelle Zeit gesetzt. • flight.FlightStatus wurde auf Landed gesetzt.

21.4 Contract CO4: LandTowFlight

Operation	LandFlight
Querverweise	Use Cases: UC06 Startliste und Flüge verwalten
Vorbedingung	• Ein Flug flight existiert und ist im FlightStatus Started und hat StartType IsTowedByAircraft.
Nachbedingungen	• flight.towFlight.LandingTime wurde auf die aktuelle Zeit gesetzt. • flight.towFlight.FlightStatus wurde auf Landed gesetzt.

XI

Paper Prototype

Datum: 15. Juni 2012

HOME | STARTLISTE | BERICHTE | STAMMDATEN

Startliste – Speck (LSZK)

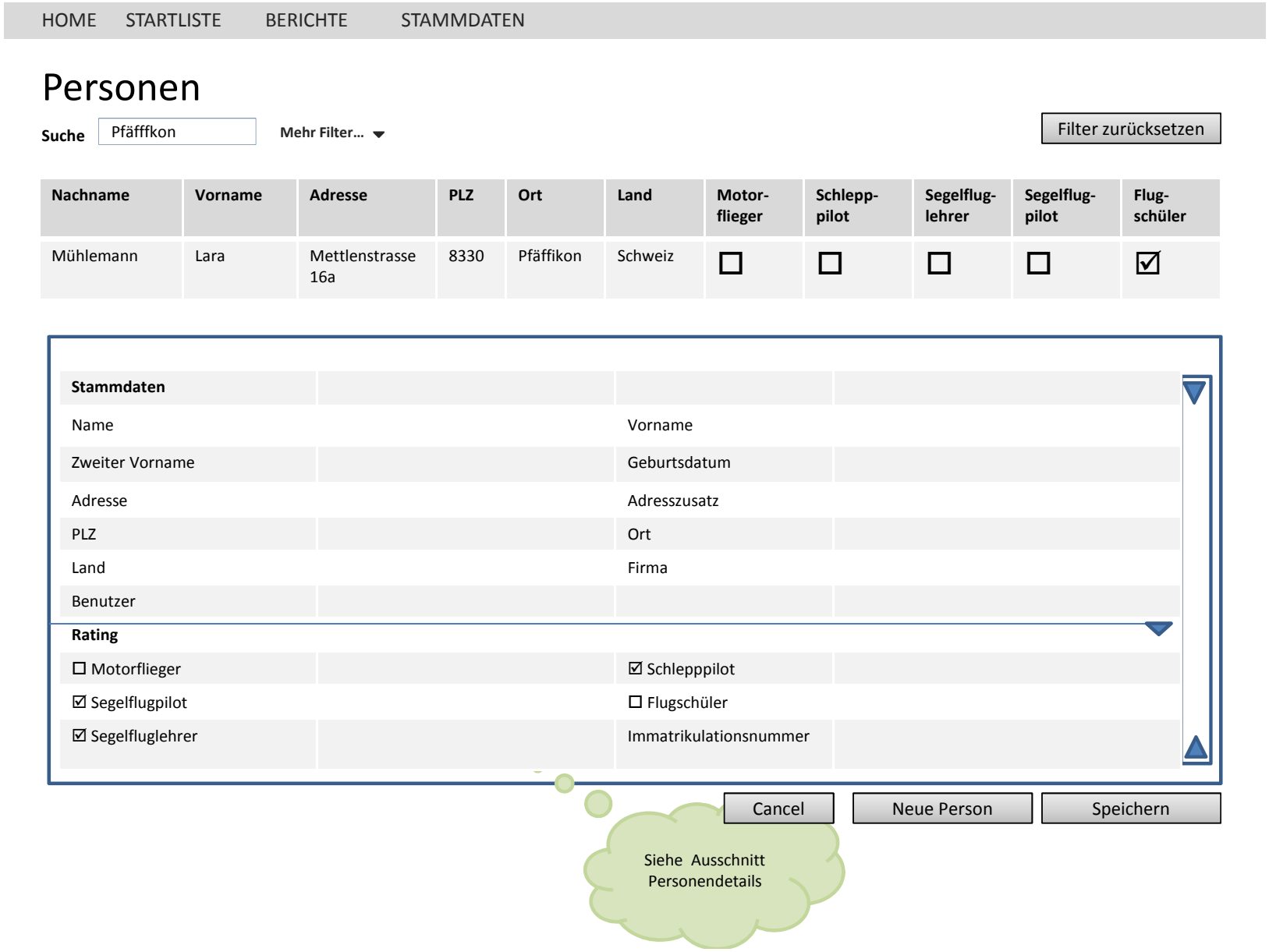
Flug hinzufügen

▼ 27.03.2012

SEGELFLIEGER		SCHLEPPER		
HB-1824 **Moritz Pfister**		**HB-KCB** **Michael Müller**		**Bereit zum Start…** Start
HB-1841 **Patrick Schuler**	Start: 10:31 Uhr (LSZK) Flugdauer: **2h 30min** Landung	**HB-KCB** **Michael Müller**	Start: 10:31 Uhr (LSZK) Landung: 10:40 Uhr (LSZK) Dauer: **0h 09min**	**In der Luft…**
HB-3407 **Roger Frei** / Silvan Huber	Start: 10:51 Uhr (LSZK) Flugdauer: **2h 10min** Landung	**HB-KCB** **Michael Müller**	Start: 10:51 Uhr (LSZK) Flugdauer: **0h 14min** Landung	**In der Luft…**
HB-3256		**HB-KCB**		**Gelandet…**
Pilot: **Patrick Schuler** Flugart: Passagierflug PAX: Christian Moser Bemerk.: Schöner Passagierflug, etc.	Start: 10:11 Uhr (LSZK) Landung: 11:25 Uhr (LSZK) Dauer: **1h 14min**	Pilot: **Michael Mülller**	Start: 10:11 Uhr (LSZK) Landung: 10:20 Uhr (LSZK) Dauer: **0h 09min**	
Mehr Details…				
HB-1824 **Gino Styger**	Start: 09:11 Uhr (LSZK) Landung: 10:25 Uhr (LSZK) Dauer: **1h 14min**	**HB-KCB** **Michael Müller**	Start: 09:11 Uhr (LSZK) Landung: 09:20 Uhr (LSZK) Dauer: **0h 09min**	**Gelandet…**

Quelle: Patrick Schuler

HOME
STARTLISTE
BERICHTE
STAMMDATEN
IMPORT/EXPORT
Mandaten
Benutzer
Personen
Flugzeuge
Flugplätze


HOME STARTLISTE BERICHTE STAMMDATEN
Personen
Suche
Pfäfffkon
Mehr Filter...
Filter zurücksetzen
Nachname
Vorname
Adresse
PLZ
Ort
Land
Motor-flieger
Schlepp-pilot
Segelflug-lehrer
Segelflug-pilot
Flug-schüler
Mühlemann
Lara
Mettlenstrasse 16a
8330
Pfäffikon
Schweiz
Stammdaten
Name
Vorname
Zweiter Vorname
Geburtsdatum
Adresse
Adresszusatz
PLZ
Ort
Land
Firma
Benutzer
Rating
☐ Motorflieger
☑ Schlepppilot
☑ Segelflugpilot
☐ Flugschüler
☑ Segelfluglehrer
Immatrikulationsnummer
Cancel
Neue Person
Speichern
Siehe Ausschnitt Personendetails

Ausgeklappter Filter in Personen (Standard)

Suche Pfäfffkon

Filter zurücksetzen

☑ **Nur Clubmitglieder**

☑ **Motorflieger** ☑ **Segelfluglehrer** ☑ **Segelflugpilot** ☑ **Schlepppilot** ☑ **Flugschüler**

Weniger Filter... ▲

Vorname	Zweiter Vorname	Nachname	Adresse	PLZ	Ort	Land	Motor-flieger	Schlepp-pilot	Segelflug-lehrer	Segelflug-pilot	Flug-schüler
Lara	Christina	Mühlemann	Zürcherstrasse 16	8330	Pfäffikon	Schweiz	☐	☐	☐	☐	☑

Ausgeklappter Filter in Personen (Alle Personen)

Suche Pfäfffkon

Filter zurücksetzen

☑ **Motorflieger** ☑ **Segelfluglehrer** ☑ **Segelflugpilot** ☑ **Schlepppilot** ☑ **Flugschüler**

☑ **Alle erfassten Personen (auch Personen, die keine Verbindung zum Verein haben)**

☑ Ist/War Clubmitglied mit Status Alle von bis 20.03.2012

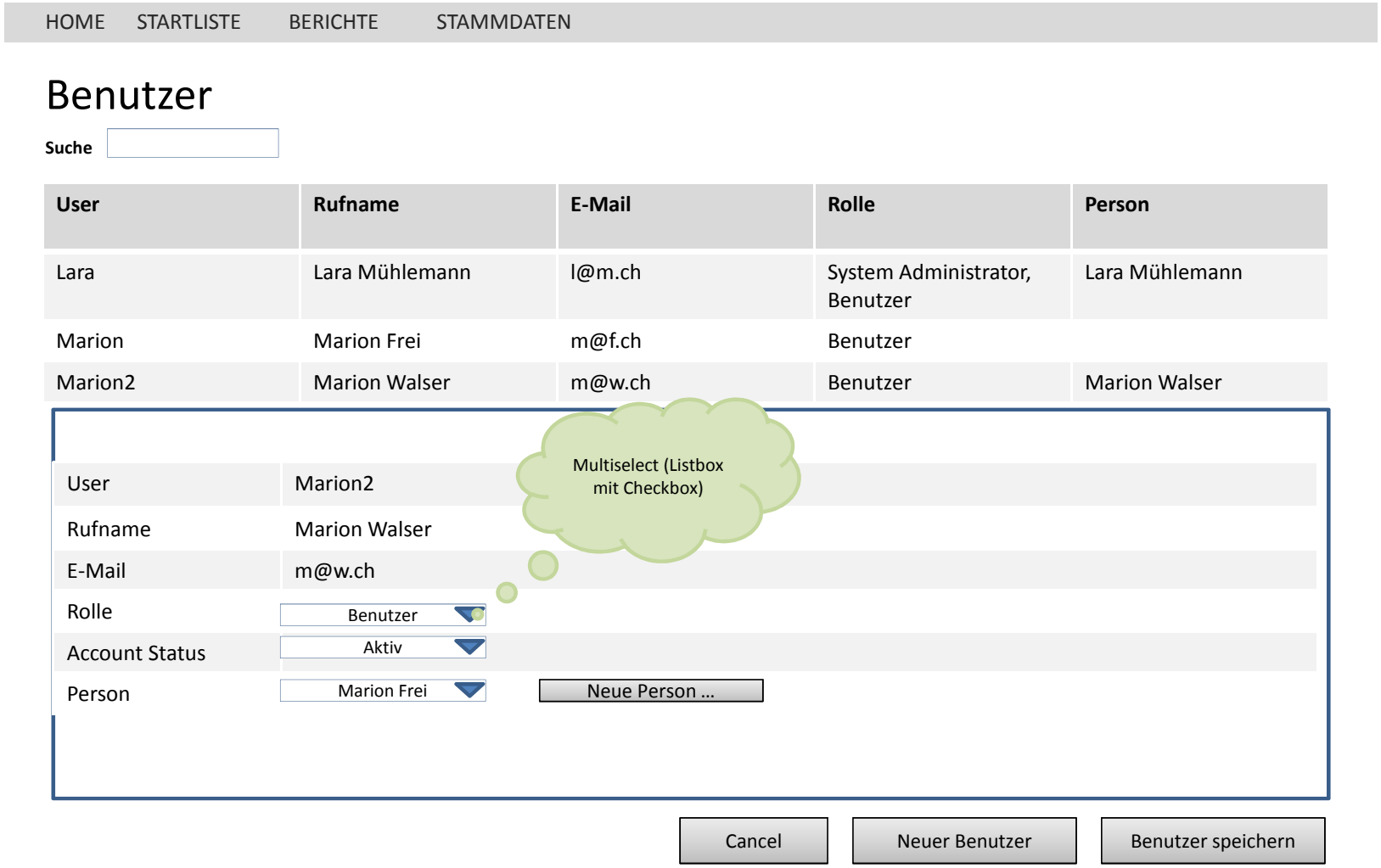
☑ Ist/War in Kategorie des Clubs Alle von bis 20.03.2012

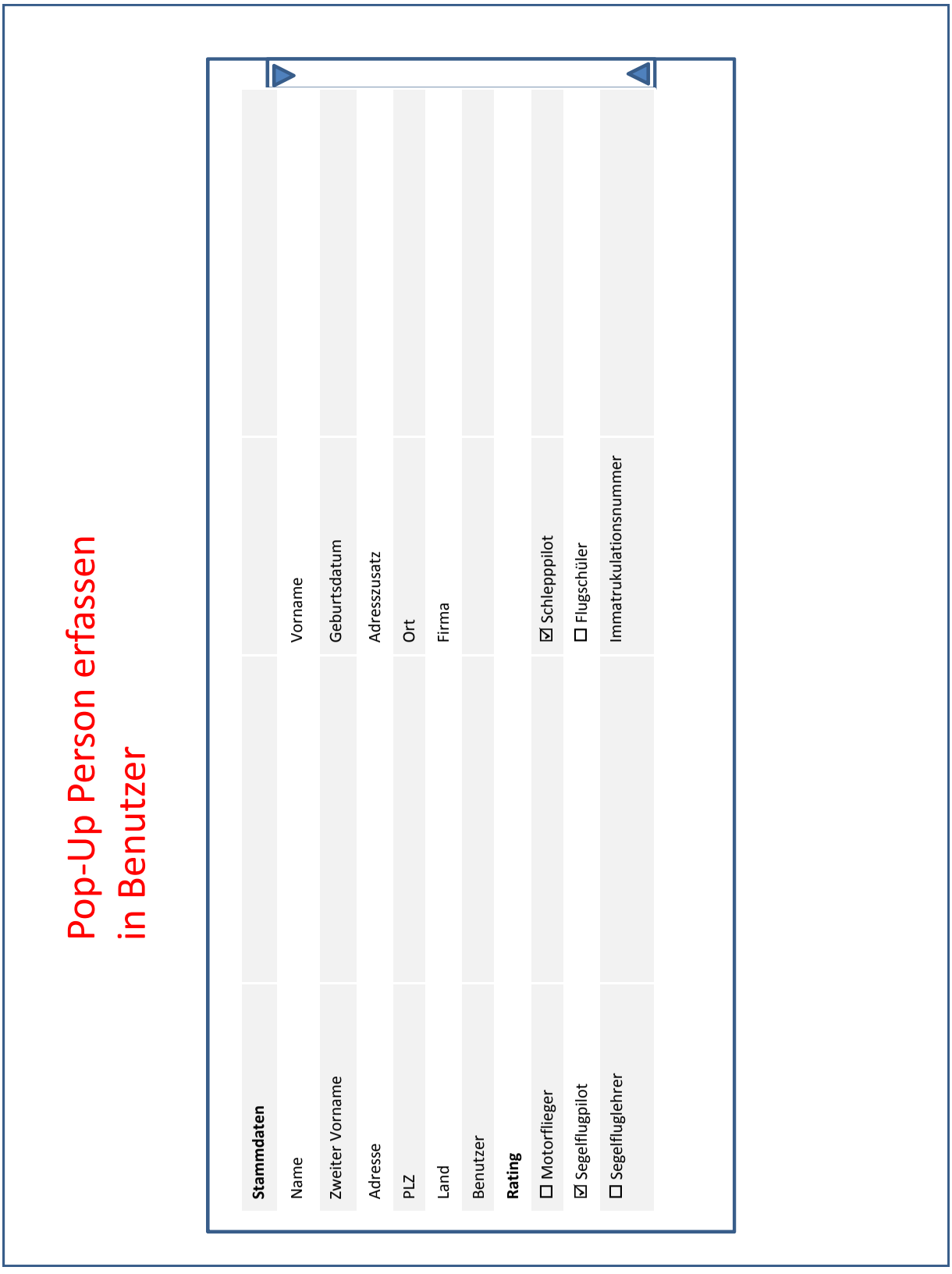
Weniger Filter...

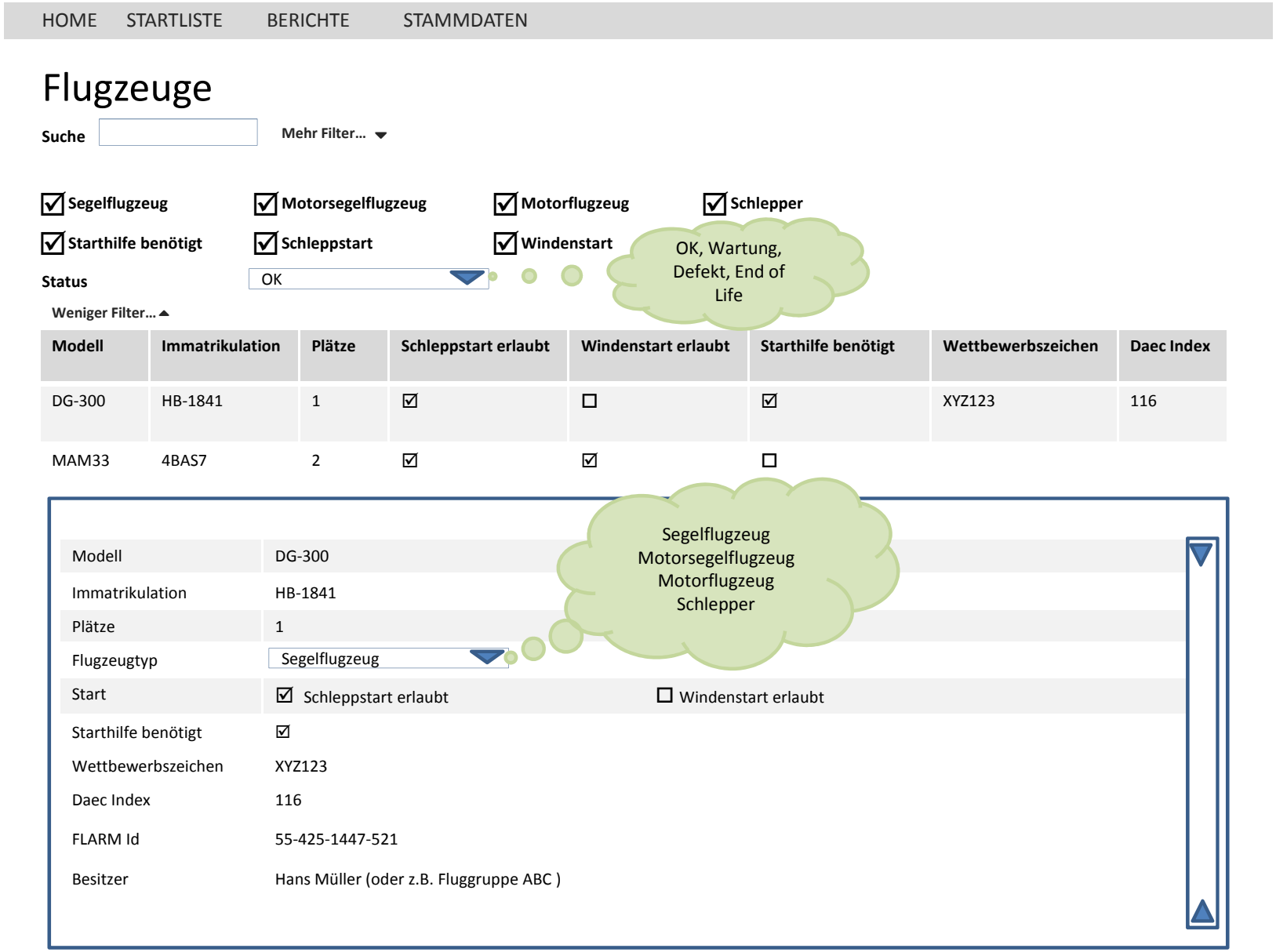
Vorname	Zweiter Vorname	Nachname	Adresse	PLZ	Ort	Land	Motor-flieger	Schlepp-pilot	Segelflug-lehrer	Segelflug-pilot	Flug-schüler
Lara	Christina	Mühlemann	Zürcherstrasse 16	8330	Pfäffikon	Schweiz	☐	☐	☐	☐	☑

Ausschnitt Personendetail:

Stammdaten			
Name		Vorname	
Zweiter Vorname		Geburtsdatum	
Adresse		Adresszusatz	
PLZ		Ort	
Land		Firma	
Benutzer	Lara Mühlemann / Lara Administrator		Neuer Benutzer erstellen...
Rating			
☐ Motorflieger		☑ Schlepppilot	
☑ Segelflugpilot		☐ Flugschüler	
☐ Segelfluglehrer		Immatrikulationsnummer	
Kommunikation			
Telefon Privat		Telefon Business	
Email Privat		Email Business	
Fax			
Metadaten			
Created		Updated	
Owner	ClubXY		Übertragen


HOME STARTLISTE BERICHTE STAMMDATEN
Benutzer
Suche
User
Rufname
E-Mail
Rolle
Person
Lara
Lara Mühlemann
l@m.ch
System Administrator, Benutzer
Lara Mühlemann
Marion
Marion Frei
m@f.ch
Benutzer
Marion2
Marion Walser
m@w.ch
Benutzer
Marion Walser
Multiselect (Listbox mit Checkbox)
User
Marion2
Rufname
Marion Walser
E-Mail
m@w.ch
Rolle
Benutzer
Account Status
Aktiv
Person
Marion Frei
Neue Person ...
Cancel
Neuer Benutzer
Benutzer speichern


Pop-Up Person erfassen
in Benutzer
Stammdaten
Name
Vorname
Zweiter Vorname
Geburtsdatum
Adresse
Adresszusatz
PLZ
Ort
Land
Firma
Benutzer
Rating
☐ Motorflieger
☑ Segelflugpilot
☐ Segelfluglehrer
☑ Schlepppilot
☐ Flugschüler
Immatrukulationsnummer


HOME STARTLISTE BERICHTE STAMMDATEN
Flugzeuge
Suche
Mehr Filter...
Segelflugzeug
Motorsegelflugzeug
Motorflugzeug
Schlepper
Starthilfe benötigt
Schleppstart
Windenstart
Status
OK
OK, Wartung, Defekt, End of Life
Weniger Filter...
Modell
Immatrikulation
Plätze
Schleppstart erlaubt
Windenstart erlaubt
Starthilfe benötigt
Wettbewerbszeichen
Daec Index
DG-300
HB-1841
1
XYZ123
116
MAM33
4BAS7
2
Segelflugzeug
Motorsegelflugzeug
Motorflugzeug
Schlepper
Modell
DG-300
Immatrikulation
HB-1841
Plätze
1
Flugzeugtyp
Segelflugzeug
Start
Schleppstart erlaubt
Windenstart erlaubt
Starthilfe benötigt
Wettbewerbszeichen
XYZ123
Daec Index
116
FLARM Id
55-425-1447-521
Besitzer
Hans Müller (oder z.B. Fluggruppe ABC)

HOME STARTLISTE BERICHTE STAMMDATEN

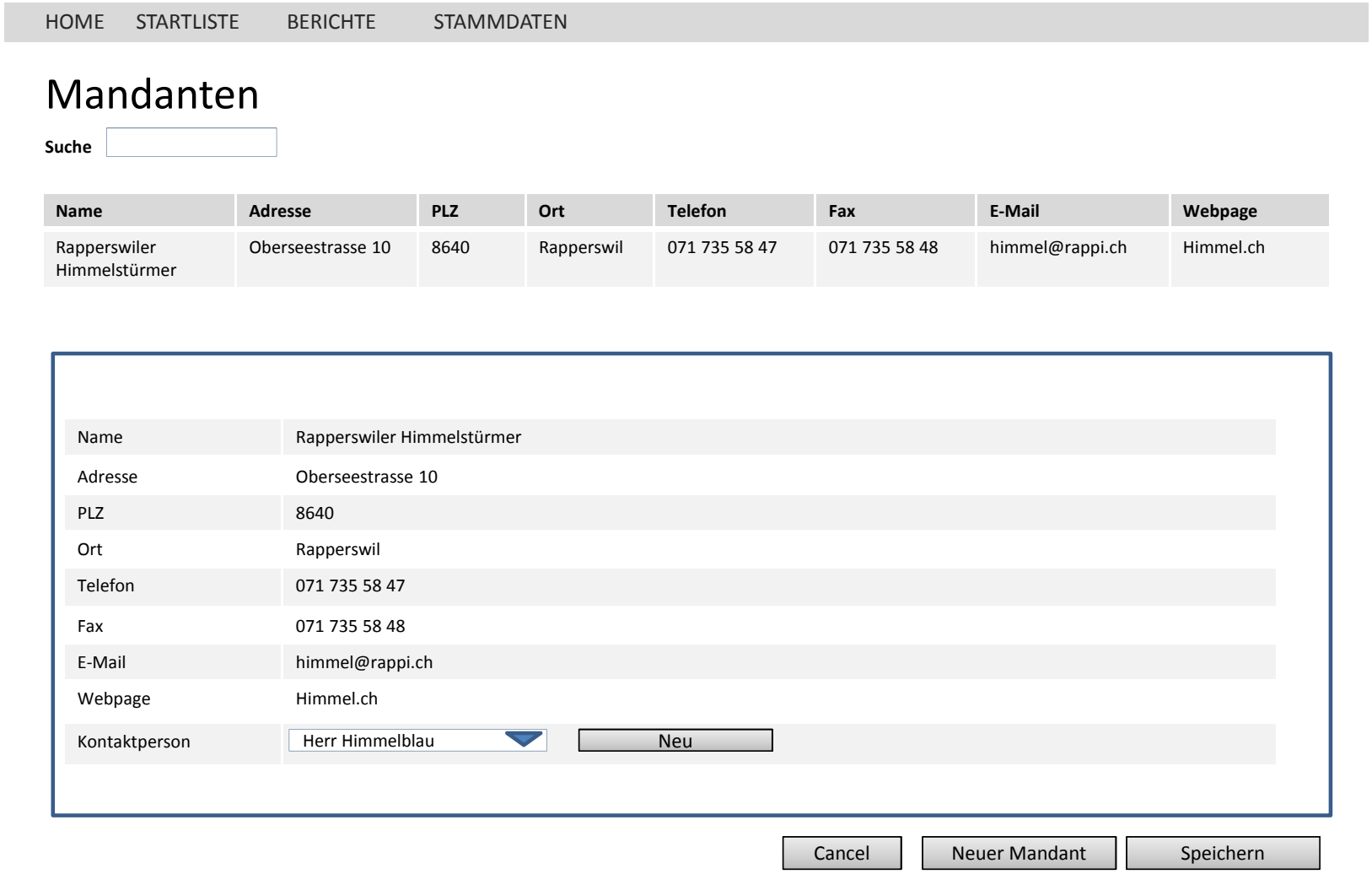
Flugplätze

Suche

Name	Kurzname	Flugplatztyp	Icao-Code	Länge Landebahn	Richtung Landebahn	Frequenz
Feld1	Fe1	Gras	555.555	150m	30/12	153.32
Im Grünen	Grü	Gras	746.024	100m		257.58

Name	Im Grünen	Kurzname	Grü
Flugplatztyp	Flugplatz Gras	Icao-Code	LSZX
Land-ID	CH		
Latitude	458624	Longitude	5622145
Elevation	812 m.ü.M	Frequenz	130.25
Länge Landebahn	100m		
Richtung Landebahn	30/12	Richtung Landebahn	30/12

Cancel Neuer Flugplatz Speichern


HOME STARTLISTE BERICHTE STAMMDATEN
Mandanten
Suche
Name Adresse PLZ Ort Telefon Fax E-Mail Webpage
Rapperswiler Himmelstürmer Oberseestrasse 10 8640 Rapperswil 071 735 58 47 071 735 58 48 himmel@rappi.ch Himmel.ch
Name Rapperswiler Himmelstürmer
Adresse Oberseestrasse 10
PLZ 8640
Ort Rapperswil
Telefon 071 735 58 47
Fax 071 735 58 48
E-Mail himmel@rappi.ch
Webpage Himmel.ch
Kontaktperson Herr Himmelblau
Neu
Cancel
Neuer Mandant
Speichern

XII

Software Architecture Document

Datum: 15. Juni 2012

22 Architektonische Ziele & Einschränkungen

22.1 Ziele

- FLS soll auf Clients bedienbar sein und über das Internet mit dem Server kommunizieren.
- FLS soll mehrbenutzerfähig sein. Änderungen anderer Nutzer sollen spätestens nach einem Neuladen sichtbar sein.
- FLS soll mandantenfähig sein.
- Die Architektur soll modular und ausbaufähig aufgebaut werden.
- Die Architektur wird in Schichten aufgeteilt. Es wird das Model-View-ViewModel Design Pattern angewendet.
- Die Applikation soll stabil laufen. Sie soll vorhersehbare Fehler durch eine Fehlerbehandlung abfangen.
- Das GUI soll übersichtlich gestaltet und intuitiv bedienbar sein.

23 Logische Architektur

23.1 Übersicht

Die Architektur vom FLS besteht aus einer Silverlight RIA Applikation, welche im Webbrowser gestartet wird und über WCF-Services und das Entity Framework auf die Datenbank zugreift.

Im Client wird das MVVM-Pattern möglichst konsequent angewendet. Damit soll eine hohe Testbarkeit erreicht werden.

23.2 Silverlight & WCF RIA Services

Für diese Applikation bot sich die Verwendung der WCF RIA Services an. Die Netzwerkfunktionen sind damit bereitgestellt und bieten eine von Seite der Entwicklung stabile Verbindung zum Server an. Durch die Asynchronität der Kommunikation stellen diese Services eine benutzerfreundliche Arbeitsumgebung zur Verfügung.

Eine Silverlight-Applikation wurde vom Auftraggeber als attraktiv angesehen, da damit keine Installation auf einem Rechner notwendig ist. Die Anwendung läuft auf dem Browser, der nur mit dem Silverlight-Plugin ausgestattet sein muss.

Visual Studio bietet für die Verwendung von Silverlight mit WCF RIA Services ein Projekt-Template an. Weiter können Domain-Service-Klassen hinzugefügt werden, womit über einen Wizard der Zugriff auf die Datenbank eingestellt werden kann. Dabei wird ein Connection-String im Web.config-File des Web-Projekts generiert.

WCF RIA Service generiert beim Kompilieren automatisch Code für die Entities und die Services im Client-Projekt. Dieses wird beim Starten der Applikation im Browser über das Xap-File, welches weitere benötigte DLLs enthält, geladen. Für die DomainServices im Server werden dabei im Client DomainContexte generiert, die die Schnittstelle zum Server bereitstellen. Die Entity-Klassen, welche den Datenbank-Tabellen entsprechen, werden ebenso bereitgestellt und die geladenen Daten von den DomainContexten verwaltet.

23.3 Authentifizierung

Diese Struktur, bestehend aus Silverlight und WCF RIA Service, wurde folgend mit einem Authentication Domain Service und dem CustomMembershipProvider erweitert. Damit wird ein sicheres Login gewährleistet. Der CustomMembershipProvider ist eine Erweiterung von System.Web.Security.MembershipProvider.

Der Authentifizierungsservice von .NET erstellt eine eigene Datenbank, in welcher die Benutzer erstellt und verwaltet werden können. Diese wurde jedoch in diesem Projekt nicht verwendet, da der Auftraggeber die Benutzer über die eigene Datenbank verwalten möchte. Die Überprüfung, ob ein Login existiert und das Passwort übereinstimmt, wurde umgeleitet. Damit liess sich auch vermeiden, dass zwei Datenbanken verwaltet werden müssen.

Der Authentifizierungsservice wurde ausschliesslich für das Login benutzt. Logout wird über ein Refresh des DOM's ausgeführt. Erstellung eines Benutzers, Ändern und Löschen wird den UserDomainService ausgeführt, entsprechend dem Ablauf für alle anderen Tabellen.

23.4 Autorisierung

Auch die Autorisierung liesse sich über den Authentifizierungsservice und den CustomRoleProvider abwickeln. Es wurde jedoch aus Gründen der Komplexität dagegen entschieden.

Es wurde eine Tabelle mit Berechtigungen erstellt (Permissions), worin die kleinstmöglichen Einheiten der Berechtigungen definiert werden können. Den einzelnen Einträgen der Berechtigungen werden Rollen zugeordnet, die damit ein einzelnes Berechtigungskonzept repräsentieren. Über die UserRoles können die Rollen den Benutzern (Users) zugeordnet werden. Damit können unterschiedliche Berechtigungen vergeben werden.

Die voreingestellten Rollen sind „Verein Administrator", „System Administrator" und „Flugdienstleiter". Möglich sind weitere Konzepte. Jedoch ist zu berücksichtigen, dass gewisse Berechtigungen andere voraussetzen und damit keine grosse Flexibilität besteht. Als Beispiel dient hier die Flugerfassung, die voraussetzt, dass Leserechte auf fast allen Stammdaten bestehen. Die Abhängigkeiten sind in der Datenbank mit Constraints sichergestellt.

Die Eingrenzung der Sichtbarkeit der Daten und die Berechtigung auf deren Veränderbarkeit basiert auf dem eingeloggten User (UserLoggedIn im UserClientModel). Dieser besitzt Rollen, aufgrund dessen die einzelnen Berechtigungen bestimmt sind. Entsprechend

wird im UserClientModel ein Property „Permissions“ mitgeführt, das im View verwendet wird, um die Sichtbarkeit und Aktivierung der Elemente zu steuern.

In der aktuellen Version der Applikation wird nur beim Laden der Flugdaten die Berechtigung berücksichtigt und damit die Menge der geladenen Daten reduziert. Alle anderen Tabellen werden komplett geladen und erst beim Anzeigen der Daten nach Berechtigung des Benutzers gefiltert oder gewisse GUI-Elemente inaktiv gesetzt.

23.5 Sicherheit

Die sicherheitstechnischen Umsetzungen wurden in diesem Projekt auf das Minimum beschränkt. Die Daten sind weder geheim noch sehr vertraulich. Der Anreiz, diese Anwendung zu hacken, besteht kaum.

Trotzdem wurden die grundlegenden Konzepte für die Benutzung der Applikation und Datenzugriffe umgesetzt. Dazu wurde ein Login implementiert, wie bereits unter Authentifizierung erwähnt, sowie ein Berechtigungskonzept umgesetzt (siehe Kapitel Mandantenfähigkeit).

Dadurch, dass die Autorisierung nur im Client erfolgt, also alle Daten an den Client geschickt werden, ist diese Umsetzung nicht sicher genug für wirklich geheime Daten. Für diese Applikation und in diesem Projekt wurde es aber nach Absprache mit dem Auftraggeber trotzdem als genügend erachtet.

Das Passwort der Benutzer wird mittels einer SHA1-Hash-Funktion codiert und in dieser Form in der Datenbank abgelegt. Auf ein vom Benutzer eingegebenes Passwort wird noch auf der Client-Seite eine SHA1-Hash-Funktion angewendet und dieses erst danach an den Server übermittelt.

23.6 Validierung

Die Validierung wurde mehrheitlich auf den Metadaten der Entities im FLS.Data mittels Data Annotation umgesetzt. Zusätzlich waren Validator-Klassen notwendig, um die Validierung von ComboBoxen zu forcieren.

Trotzdem haben einige ComboBoxen nicht das zu erwartenden Verhalten gezeigt. Es wurde deshalb in den Popups des PersonViews beim Hinzufügen von PersonPersonCategory, PersonMemberStatus und PersonClub (Rating) im PersonView eine manuelle Validierung mittels zusätzlichen Textfeldes umgesetzt. Im StartListView wurde dies jedoch aus zeitlichen Gründen nicht mehr geändert. Die Validierung im StartListView ist

deshalb nicht zuverlässig und sollte bei einer Weiterentwicklung unbedingt noch gelöst werden.

Für die Popups der gerade oben erwähnten PersonView und der Popups im ClubView wurde im ViewModelBase eine Methode für eine sofortige Validierung implementiert, damit vor dem Schliessen der Popups überprüft wird, ob die eingegeben Daten fehlerfrei sind. Damit liess sich umsetzen, dass nicht der ganze Kontext gespeichert und damit validiert werden muss, wenn ein Popup geschlossen wird. Nach dem Schliessen des Popups ist eine Änderung der darin gepflegten Daten nicht mehr möglich.

23.7 Prism

Da für die Applikation noch einige Erweitungen geplant sind (siehe Management Summary), wurde die Prism-Library verwendet, um eine zukünftige Modularisierung der Applikation vorzubereiten. Dafür wurde die Klasse Bootstrapper vom MefBootstrapper abgeleitet. Um die Modularisierung jedoch definitiv umzusetzen, müsste das aktuelle Client-Projekt vom Bootstrapper separiert werden.

In der Klasse Bootstrapper werden auch die Views der zum jetzigen Stand einzig existierten Region, „MainRegion“ dem RegionManager angemeldet. Dazu wurde der Regions-Teil der Prism-Library verwendet.

Die Navigation erfolgt über den RegionManager. Um die Überprüfung zu ermöglichen, ob zu einem anderen View gewechselt werden kann, implementiert das ViewModelBase das Interface IConfirmNavigationRequest. Dieses wurde verwendet um sicherzustellen, dass nicht in den nächsten View gewechselt werden kann, wenn Änderungen im aktuellen View vorhanden sind.

Die Dependency Injection von Prism wurde verwendet, um sicherzustellen, dass immer nur eine Instanz von den Views, ViewModels und ClientModels erstellt wird. Damit wird das mehrmalige Laden der Daten vom Server verhindert, da das ClientModel bei jeder Instanzierung die Daten vom Server holt.

Von Prism wurden weiter die DelegateCommands verwendet. So lässt sich die richtige Aktivierung und Deaktivierung der Buttons in den Views einfach steuern.

23.8 Datenaustausch zwischen Server und Client

Grundsätzlich werden alle Daten vom Server geladen. Davon ausgenommen sind die bereits unter Berechtigung erwähnten Flugdaten, da deren Anzahl innerhalb kürzester Zeit

stark wachsen wird. Ausserdem werden diese nicht für andere Arbeiten mit der Applikation benötigt. In der Datenbank wurde für diese Selektion ein Feld definiert, über das bestimmt werden kann, wie weit zurück die Flugdaten geladen werden. Die Voreinstellung aus der Auslieferung ist auf 10 Tage festgelegt, ist aber einfach über eine Änderung des Datenbank-Eintrags „CountOfDaysOfLoadedFlights“ in der SystemData-Tabelle änderbar.

Obwohl das DataGrid in den Stammdatenverwaltungen über eine PagedCollectionView erstellt wird und damit ein Laden pro Seite möglich wäre, wurde dies nicht verwendet. Die meisten Stammdaten müssen beim Arbeiten mit der Anwendung zur Verfügung stehen, um über die ComboBoxen angezeigt zu werden.

Die Daten werden automatisch nur einmal pro Client geladen, wenn dieser die Applikation im Browser startet. Änderungen werden in den DomainContext im Client vermerkt und beim Speichern an den Server übermittelt. Pro Client wird von jedem ClientModel nur eine Instanz erstellt, um mehrmaliges Laden zu verhindern. Dies wurde mittels Prism und DependencyInjection umgesetzt und wird genauer unter Absatz „Prism“ beschrieben.

23.9 Datenfluss und vertikale Unterteilung

Die Entities wurden anhand der Hauptaufgaben der Views in Services gruppiert. Die Services im Client verwalten die geladenen Daten. Die ClientModels sind für das Laden der Daten vom Server verantwortlich. Dadurch werden die Services und die Kontexte in einer übersichtlichen Grösse gehalten. Damit die Views die Daten trotzdem jederzeit aktualisieren können, werden die Lade-Operation auch den Views direkt zur Verfügung gestellt.

Zwischen den ViewModels und den ClientModels wurden ClientServices hinzugefügt. Deren Zuständigkeit ist das Erstellen von neuen Entitäten sowie die Aktualisierung derselben. Services besitzen keine Verbindung zum ClientModel, der FlightClientService ausgenommen, sondern unterstützen nur die Aufgaben der ViewModels. Dies wurde so umsetzt, weil sich das Hinzufügen einer Entität im ClientService und die Aktualisierung der selektierten Entität nicht optimal umsetzen liess, ohne dass eine Meldung über vorhandene Änderungen erschien, und deshalb die Selektion nicht geändert wurde. Im FlightClientService besteht diese Einschränkung nicht. Es wurde deshalb dem ClientService erlaubt, neu erstellte Entitäten dem ClientModel hinzuzufügen.

Es soll möglichst vermieden werden, dass Daten von den verschiedenen Views einzeln vom Server angefordert werden. Die verschiedenen Kontexte werden deshalb mit einem ExternalReference-Attribut verbunden. Include-Attribute in den Get-Methoden der Services werden nur innerhalb desselben Kontextes verwendet.

In der unterstehenden Grafik ist diese Struktur visuell dargestellt.

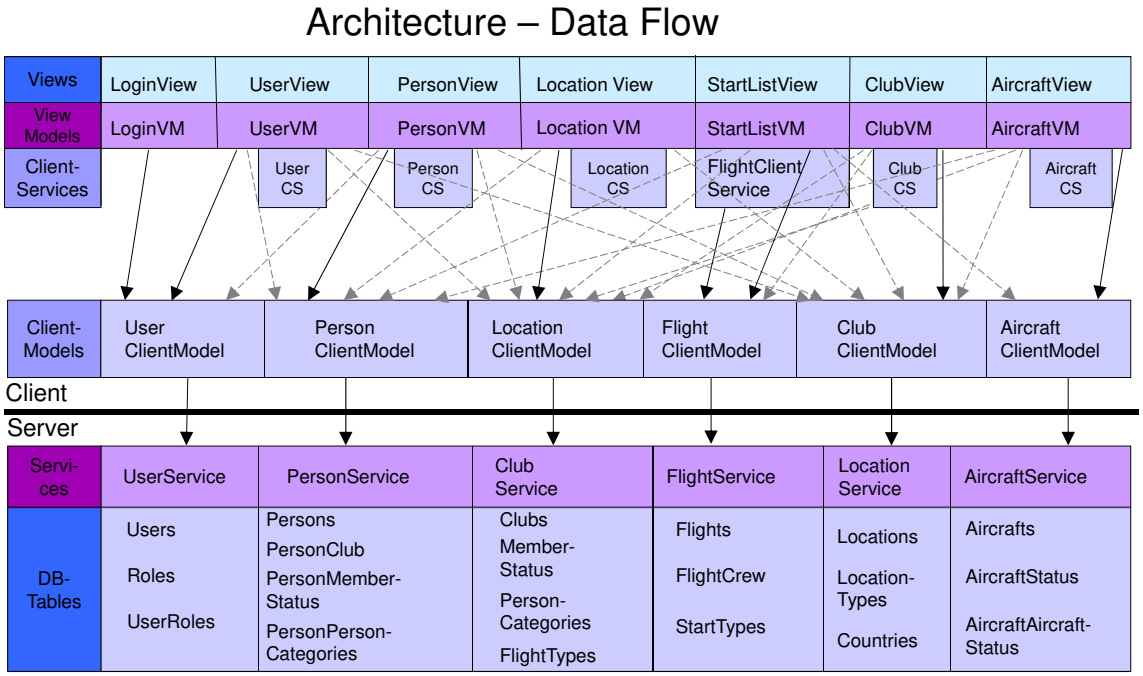


Abbildung 23.1: Data Flow

23.10 Implementierung Reporting

Der Auftraggeber wünschte einige Reports, die die Nutzung der Applikation erst richtig sinnvoll machen. Dazu gehörten ein Flugreport für die Piloten, eine monatliche Zusammenstellung aller Flüge eines Vereins für die Rechnungsstellung sowie eine kurze Übersicht über die geflogenen Flüge für das BAZL. Zum einen sollten die Reports automatisch täglich / monatlich per E-Mail versendet werden (siehe Kapitel Implementierung E-Mail-Versand), zum anderen sollten diese Reports auch jederzeit über das User Interface abgerufen werden können.

Für das Erstellen der Reports wurde der SQL Server Reporting Service von Microsoft benutzt. Dazu wurde eine eigene Solution erstellt, welche über das Business Intelligence Development Studio von Visual Studio 2008 bearbeitet wird. Als Datenquelle diente die FLS-Datenbank, auf die mit Hilfe eines Connection-Strings zugegriffen wird. Die Datenbank wurde im Report-Projekt als geteilte Datenquelle konfiguriert, um allen Reports auf dieselbe Datenbank Zugriff zu geben. Für jeden Report wurde ein entsprechendes Query geschrieben, welches aus mitgegebenen Parametern die gewünschten Daten zurück gibt.

Die Daten werden im Report für die Darstellung je nach Bedarf gruppiert und teilweise noch über eine Visual Basic Expression aufbereitet oder über eine TextBox-Property speziell formatiert.

Im User Interface kann jeder Benutzer Flugreports von allen Personen, die Mitglieder im selben Verein sind wie der Benutzer, über eine frei wählbare Zeitspanne generieren. Dafür muss die Person, von der ein Report generiert werden soll, ein PersonClub, ein PersonMemberStatus oder eine PersonPersonCategory in dem Verein haben, welcher dem eingeloggten Benutzer zugeordnet ist.

Für die Zusammenstellung aller Flüge eines Monats sowie die Zusammenfassung fürs BAZL kann im User Interface ein Monat ausgewählt werden. Der Report wird dann vom ersten bis letzten Tag dieses Monats erstellt. Jeder Benutzer kann diese monatlichen Reports über den dem Benutzer zugeordneten Verein erstellen. Der Report unterscheidet zwischen Piloten, welche dem Verein angehören und somit einen PersonClub haben, und anderen Piloten, die Flüge von der Homebase des Vereins des Benutzers ausgeführt haben.

Alle Reports im User Interface werden über einen Hyperlink-Button geöffnet. Dieser holt sich aus den ausgewählten Parametern (z.B. Person) alle Parameter für den Report und generiert darauf die URL, die zum als PDF exportierten Report führt.

23.11 Implementierung E-Mail-Versand

Jeder Pilot, welcher während des Tages einen Flug gemacht hat und eine E-Mail-Adresse eingetragen hat, erhält am Abend automatisch einen Flugreport per Mail zugesendet. Auch hierfür wird wie im Kapitel „Implementierung Reporting“ zuerst ein PDF generiert, welches temporär auf dem lokalen Filesystem des Servers heruntergeladen wird und nachher als Anhang eines E-Mails verschickt wird.

Die monatlichen Reports werden einmal im Monat an die E-Mail-Adresse des Vereins geschickt. Auch diese zwei Reports werden zuerst lokal heruntergeladen und dann als E-Mail-Anhang verschickt.

Neben den Reports wird beim Erstellen eines Benutzers sowie bei der individuellen Anforderung eines neuen Passworts ein generiertes Passwort an den Benutzer gesendet.

Die Implementation des E-Mail-Versands sowie der File-Download wurde durch die vom .Net-Framework zur Verfügung gestellten Klassen im Namespace System.Net realisiert (MailMessage, SmtpClient, NetworkCredential, WebClient) welche eine einfache Implementation erlauben. Einzige Vorbedingung um E-Mails zu versenden, ist dabei das Vorhandensein eines SMTP-Servers, welcher vom Auftraggeber eingerichtet

wird.

23.12 Implementierung Import/Export

Bei der Implementierung des Imports und Exports wurde darauf geachtet, dass möglichst viel der Verabeitung im Server erfolgen, um eine gute Performance zu erhalten. Trotzdem dauert die Verarbeitung bei einer grossen Menge von Daten schnell einige Sekunden, da eine grössere Datenmenge über das Netzwerk verschoben werden muss.

Bei der Implementation der Verarbeitung im Server wurde mit Interfaces und abstrakten Klassen gearbeitet. So kann bei Erweiterungen auf weitere Entitäten ein Grossteil der Funktionalität verwendet werden und nur einzelne Methoden müssen zusätzlich implementiert werden.

Es wurden zwei Projekte erstellt: FLS.ImportExportService und FLS.ImportExport. Das Service-Projekt bietet die Funktionalität dem Client an, während im zweiten Projekt die Verarbeitung erfolgt. Diese Funktionalität des Imports/Exports wurde explizit in eigenen Projekten (d.h. eigene DLLs) erstellt. Damit sollen die Voraussetzungen gegeben sein, um eine Plug-In-Architektur aufzubauen und für eine flexible Erweiterung der Export- bzw. Import-Funktionalität auf weitere Entitäten.

23.12.1 Darstellung Import-Ablauf mittels Sequenzdiagrammen

Folgend wird der Ablauf des Imports mittels eines Sequenzdiagramms dargestellt. Dabei wurde der Fokus auf die klassenübergreifenden Aufrufe gelegt und die Namensgebung abstrakt gehalten, damit ein allgemeiner Überblick über den Ablauf und die wichtigsten Methodenaufrufe gegeben werden kann.

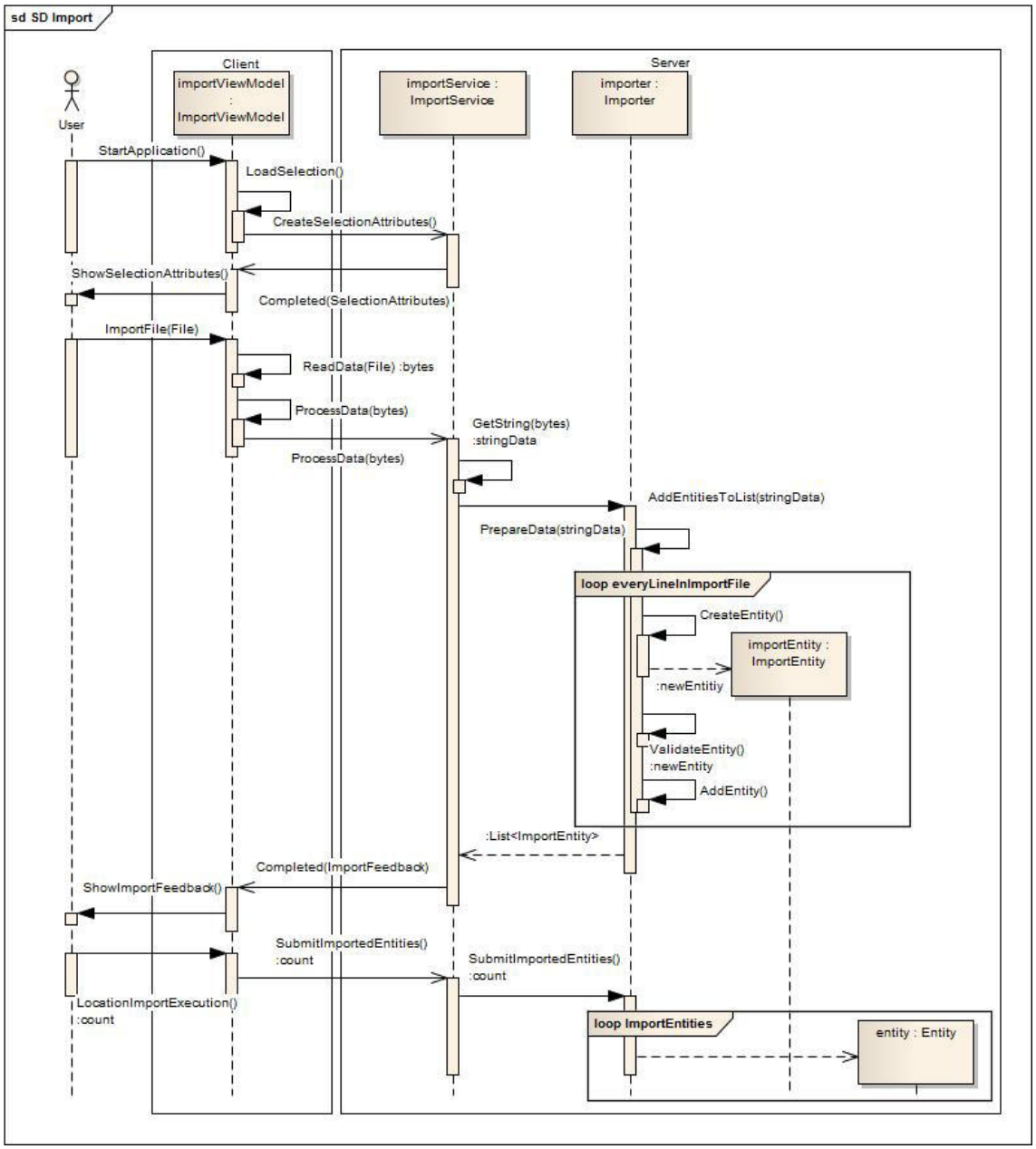


Abbildung 23.2: Sequenzdiagram Import

23.13 Design Namespaces

Die einzelnen Schichten vom FLS werden jeweils in Namespaces aufgeteilt. Dabei stellt jede Schicht ein eigenes Visual Studio-Projekt dar.

Die Abhängigkeit des Namespaces FLS.Client.ViewModels vom Namespace FLS.Client.Views ergibt sich aus einigen Popups, die in den ViewModels kreiert werden. Diese etwas unschöne Variante wurde wegen ihrer Einfachheit gewählt.

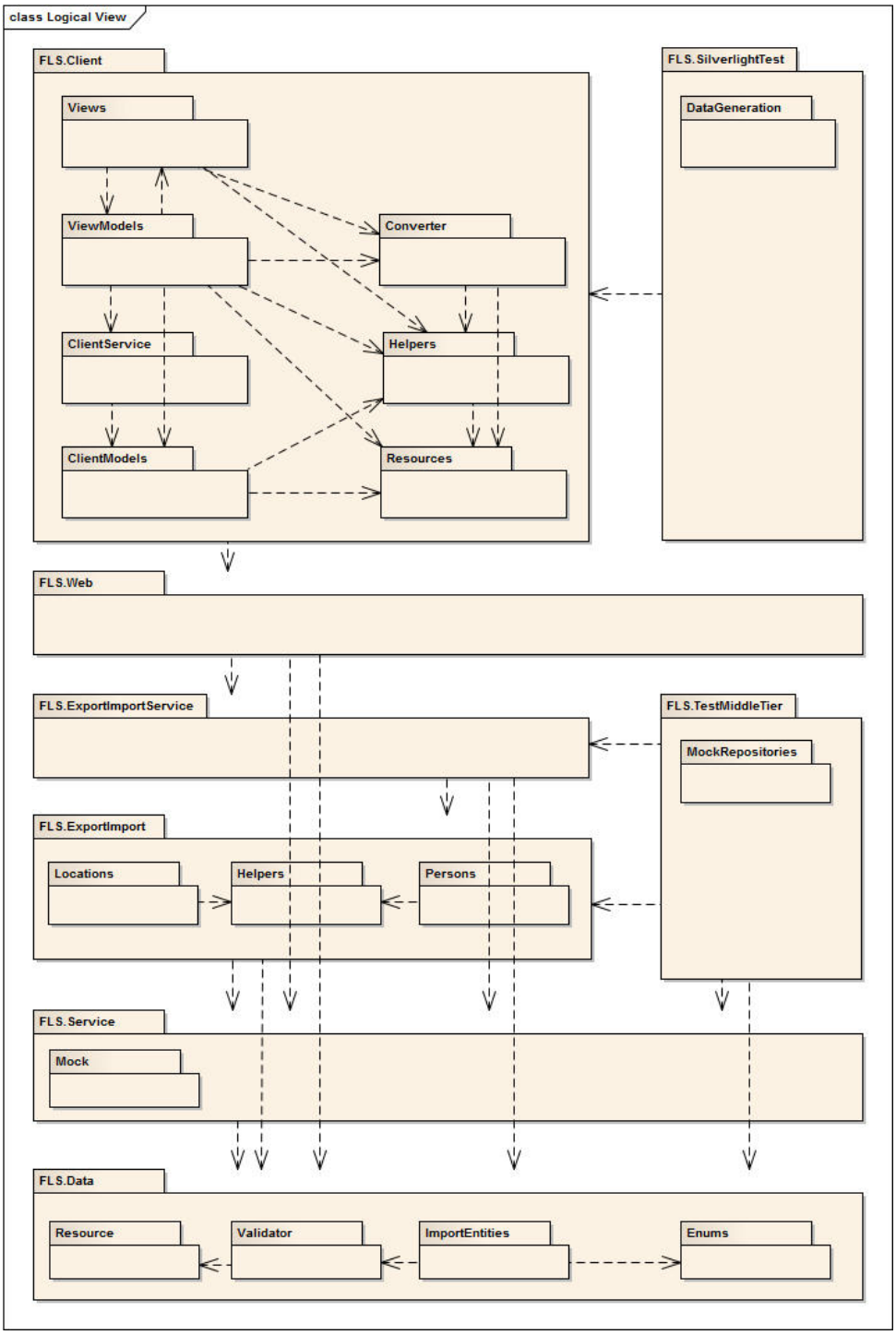


Abbildung 23.3: Übersicht Namespaces

23.13.1 FLS.Client

Der Namespace FLS.Client enthält alle Client-relevanten Klassen. Dabei wird er selbst nochmals in acht Unter-Namespaces aufgeteilt, welche teilweise wieder Unter-Namespaces enthalten.

- Views: Enthält die Xaml-Dateien, in denen die Views beschrieben werden.
- ViewModels: Pro View existiert ein ViewModel, das die View-Logik enthält.
- ClientModels: Stellen die Verbindung zwischen ViewModels und Services her.
- ClientService: Erweitert einige Entities um Methoden, um die Entities zu erstellen, verändern und zu löschen.
- EntityExtensions: Enthält Methoden, um die einige der Entities erweitert wurden.
- Helpers: Hilfsklassen, wie z.B. spezielle EventArgs-Klassen oder ein Enum-Description-Mapper.
- Converter: Die verschiedenen Converter werden für das Umwandeln von Werten zwischen View und ViewModel benötigt.
- Resources: In den Ressourcen werden die String-Konstanten sowie die Texte für das GUI abgelegt.

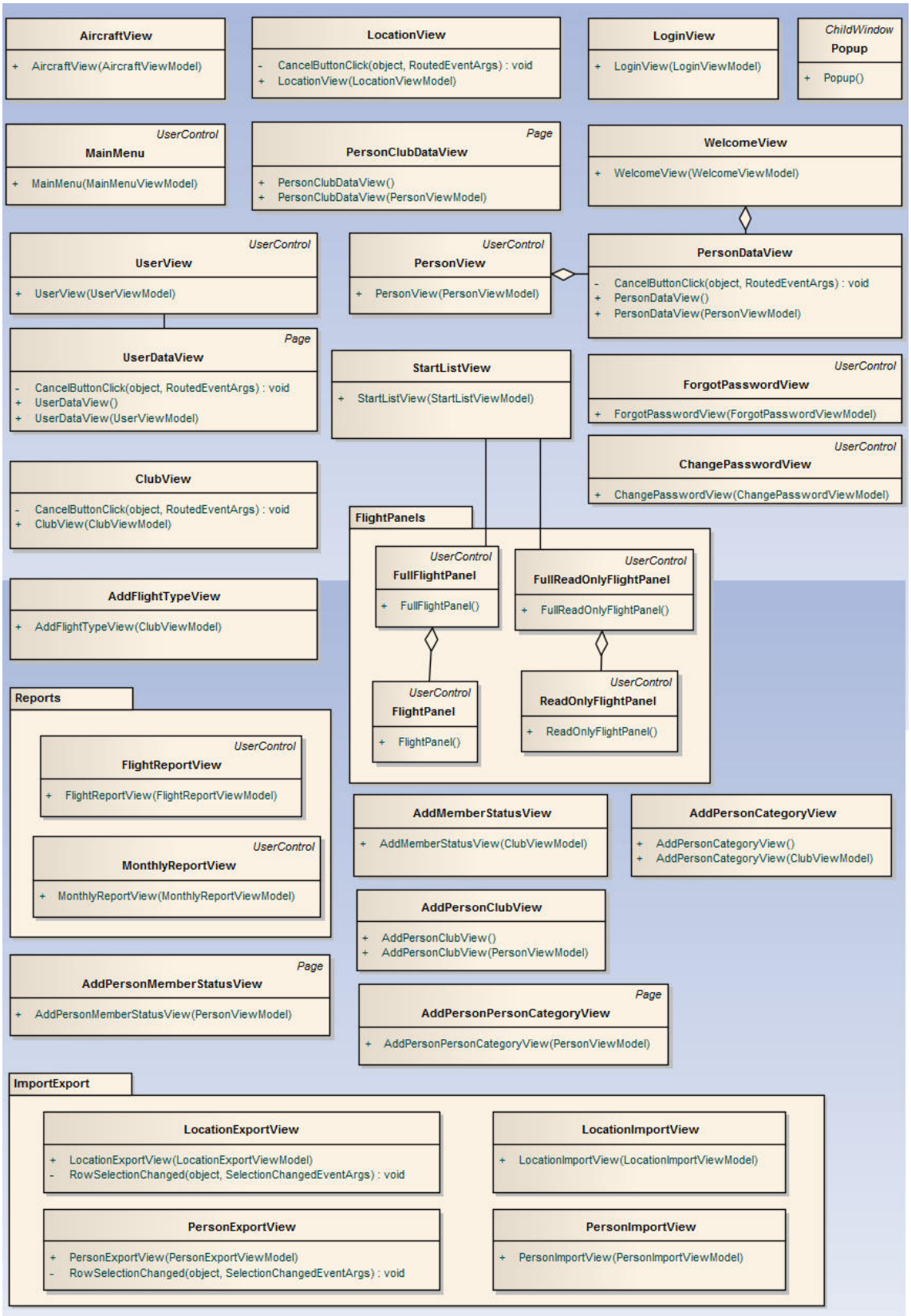


Abbildung 23.4: Namespace FLS.Client.Views

Die ViewModels leiten von ViewModelBase ab (MainMenuViewModel ist die einzige Ausnahme). Das ViewModelBase stellt dabei den abgeleiteten ViewModels vor allem die Interfaces INotifyPropertyChanged und IConfirmNavigationRequest aus Prism zur Verfügung. Es implementiert weitere Methoden wie Save und HandleValidationError, die aus den spezifischen Klassen aufgerufen werden. Ein Property für die Permission ist im ViewModelBase enthalten. Über dieses Property werden viele Elemente in den Views gesteuert bezüglich ihrer Sichtbarkeit und ihrem Aktivierungsstatus.

BaseClientModel ist die Oberklasse der verschiedenen ClientModels. Diese Klasse definiert einige abstrakte Methoden (Revert, HasChanges, Save), die von den Unterklassen implementiert werden müssen. Weiter bietet sie einen EventHandler an, an dem sich die ViewModels anmelden können, wenn Sie bei Änderungen des DomainContext informiert werden möchten.

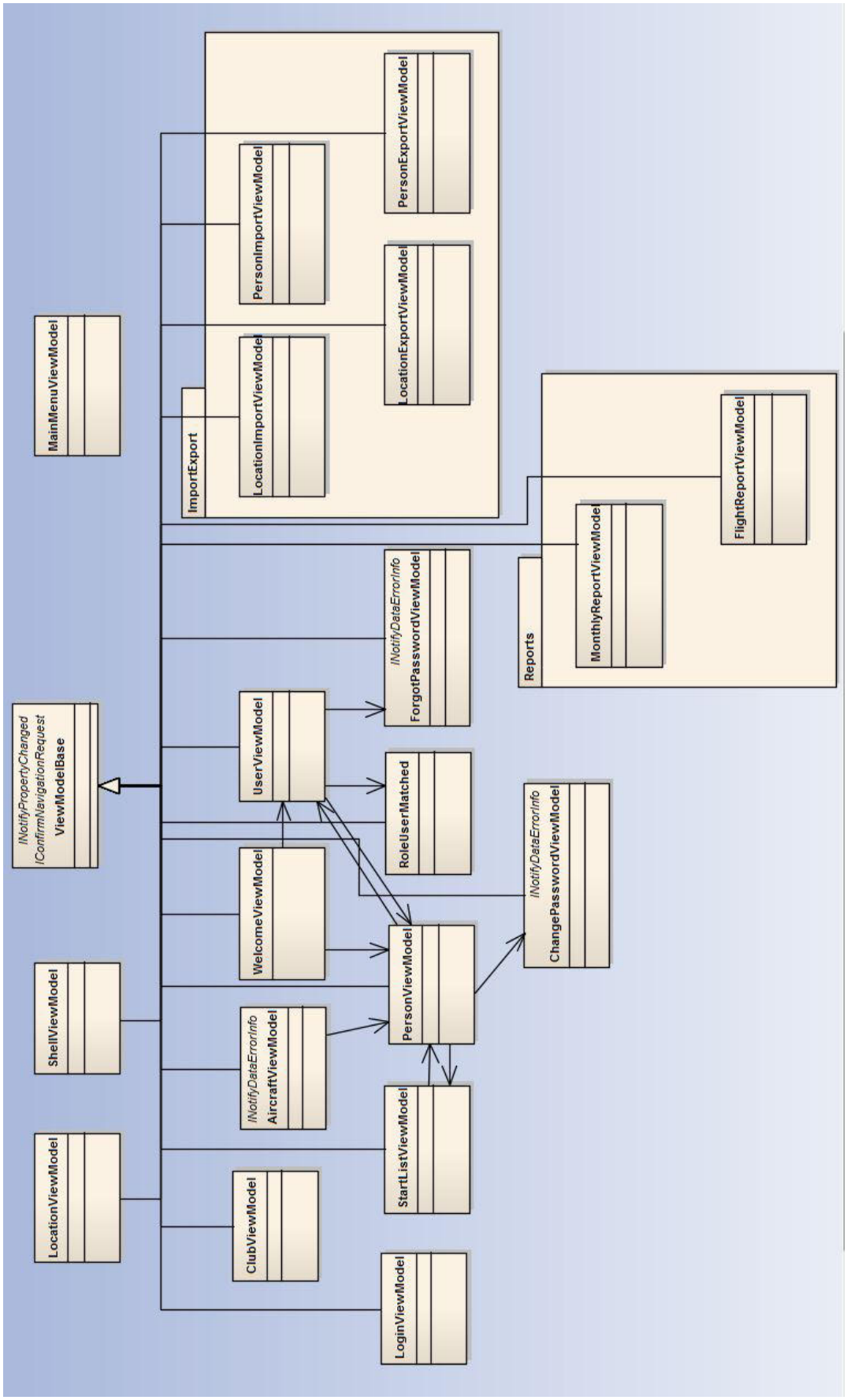


Abbildung 23.5: Namespace FLS.Client.ViewModels

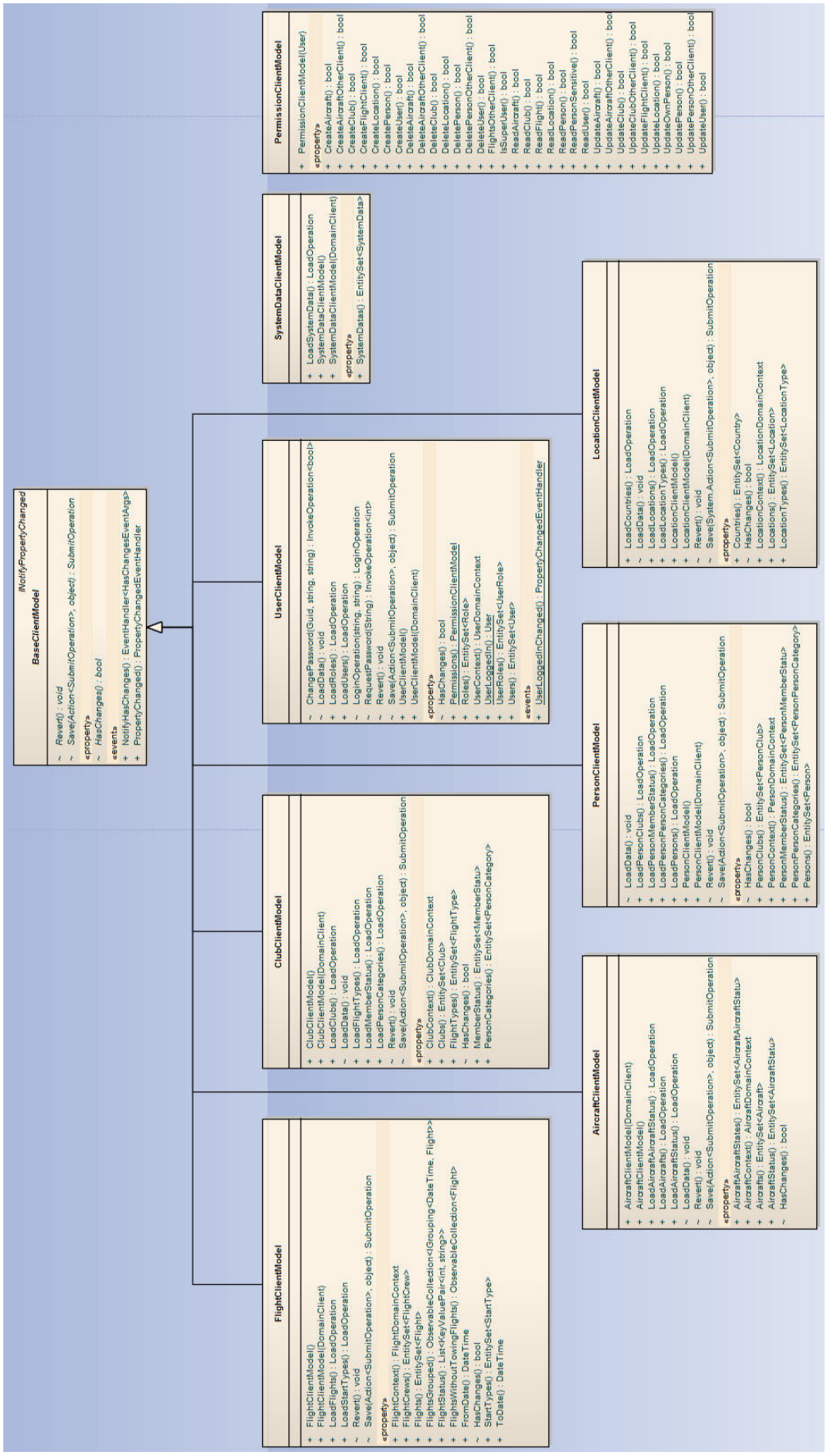


Abbildung 23.6: Namespace FLS.Client.ClientModels

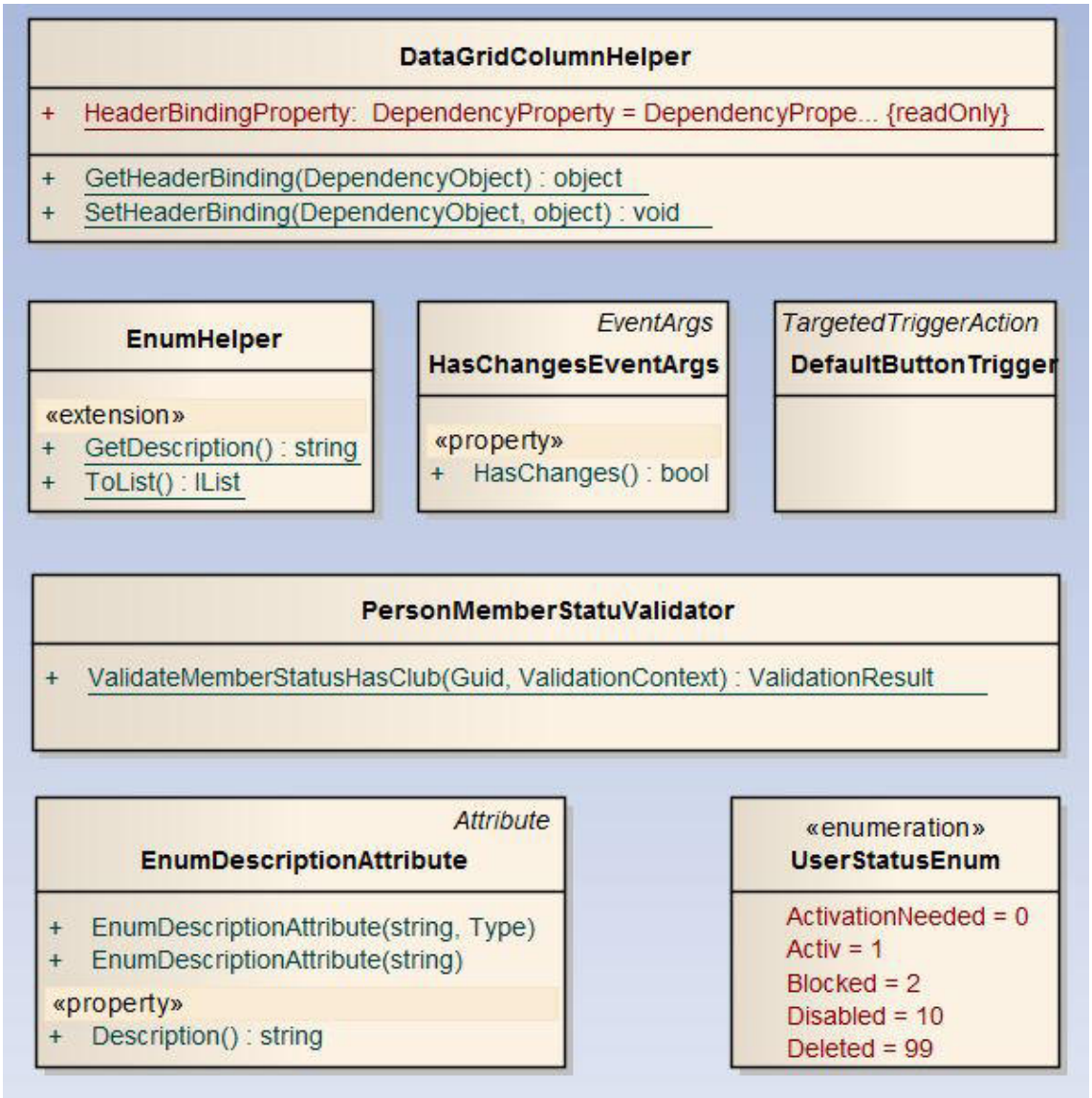


Abbildung 23.7: Namespace FLS.Client.Helpers

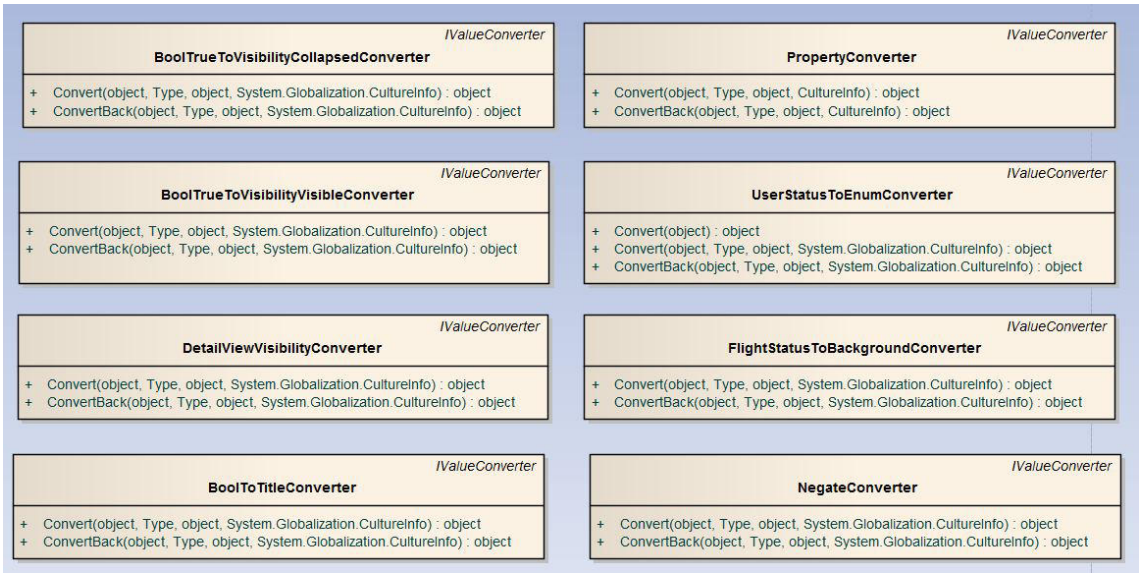


Abbildung 23.8: Namespace FLS.Client.Converter

23.13.2 FLS.Web

Der Namespace FLS.Web ist der Einstiegspunkt für den Client. Er enthält das XAP-File, welches vom Browser geladen wird, wenn eine Verbindung zum Server hergestellt wird. In der Global.asax-Datei des Web-Projektes werden die Scheduler für die Reporterstellung gestartet, einer für den monatlichen und einen für den täglichen Report.

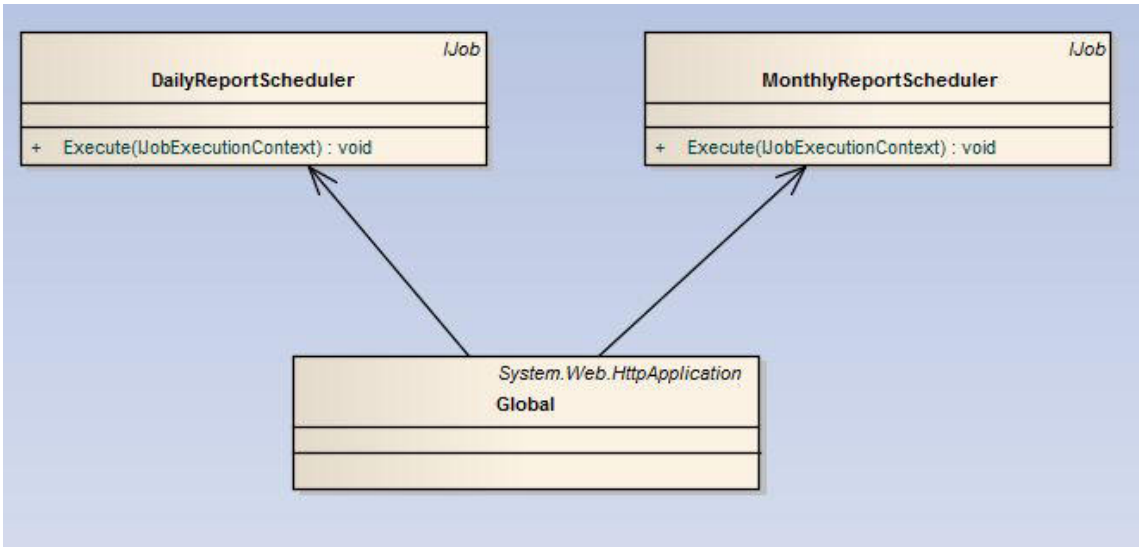


Abbildung 23.9: Namespace FLS.Web

23.13.3 FLS.Service

Der Namespace FLS.Service beinhaltet die verschiedenen Services, welche die CRUD-Operationen auf den Entitäten zur Verfügung stellen. Der Namespace enthält auch die Authentifizierung und überprüft das Einloggen. Für die Domain Services, wofür eine Mock-Klasse erstellt wurde, existiert ein Interface. Damit kann der echte Service mittels Dependency Injection durch den Mock-Service ersetzt werden. Die Definition, welche Entitäten über welchen Service verwaltet werden, orientiert sich an den Views. Diejenigen Entitäten, die in einem View verwaltet werden, wurden gruppiert. Für jede dieser Gruppen wurde einen Service erstellt.

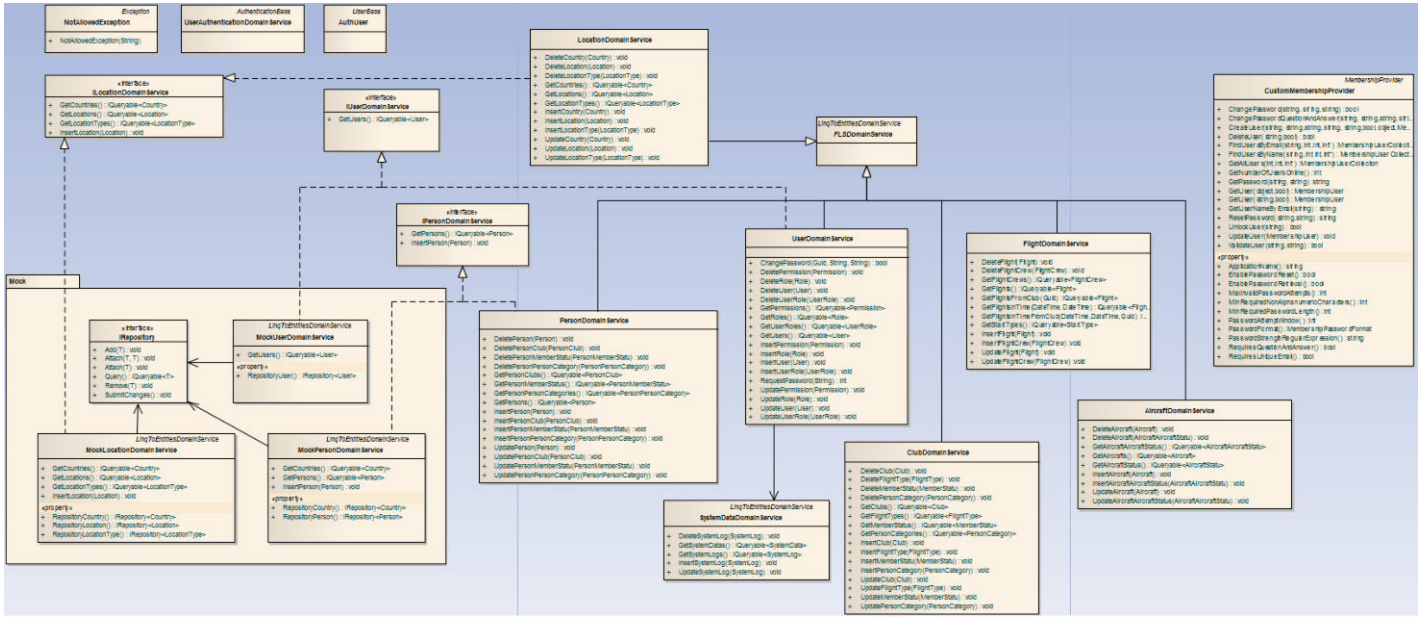


Abbildung 23.10: Namespace FLS.Service

23.13.4 FLS.Data

Der Namespace FLS.Data enthält die Entities sowie das Mapping auf die Datenbank.

- Resource: Beinhaltet konstante Strings.
- Validator: Beinhaltet verschiedene Validierungen.

23.13.5 FLS.SilverlightTest

In diesem Namespace befinden sich die Silverlight Unit Tests, welche die ViewModels und die ClientModels testen.

23.13.6 FLS.Testing

Der Namespace FLS.Testing enthält Unit Tests, die die Erweiterungen der Entitäten und den ExportImport-Service testen.

24 Prozesse und Threads

24.1 Silverlight & WCF RIA Services

Alle Aufrufe des WCF RIA Services sind asynchron. Deshalb benutzt WCF RIA Service für die Netzwerkkommunikation einen eigenen Thread, welcher durch das Silverlight-Framework verwaltet wird. Dieser Thread wartet nach Anfragen an den Server auf dessen Antwort und aktualisiert allenfalls die DomainContexte mit den geladenen Daten. Der GUI-Thread kann in dieser Zeit weiterlaufen und auf Eingaben vom Benutzer reagieren.

24.2 Anzeige Flugdauer

In der Startlisten-View wird im sekundentakt angezeigt, wie lange die Flugzeuge in der Luft sind. Dafür wurde ein DispatcherTimer verwendet, der im Konstruktor des StartList-ViewModels gestartet wird und die View dazu auffordert, die Dauer neu zu ermitteln. Die Entität Flight wurde deshalb im FLS.Data-Projekt um ein Property Duration erweitert, das die Differenz zwischen Startzeit und aktueller Zeit berechnet.

24.3 Reporting

Für den Report-Versand werden zwei Threads für die Scheduler über die Library Quarz erstellt. Für jede Ausführung des Schedulers nimmt diese einen neuen Thread aus dem Thread-Pool. Der eine Scheduler verschickt die Flug-Reports täglich an die Piloten. Ein weiterer Scheduler sendet monatlich Reports an das Sekretariat mit den Daten fürs BAZL und für die Rechnungsstellung.

Die Implementierung verwendet die Quartz-Library und befindet sich im FLS.Web-Projekt. In der Global.asax-Datei wird das Scheduling aufgesetzt. Die Klassen MonthlyReportScheduler und DailyReportScheduler sind für das Generieren der Reports und das Erstellen und Versenden der E-Mails verantwortlich.

25 Mandantenfähigkeit

Um die Mandantenfähigkeit zu gewährleisten, muss unterschieden werden, welche Tabellen global für alle Mandanten zur Verfügung stehen und welche für jeden Mandant individuell sind. Tabellen, deren Daten pro Mandant einzeln gehalten und gepflegt werden, verfügen über eine Verbindung zu der Tabelle Club.

Tabellen - Mandanten

Prozess	Bemerkung	Tabelle benötigt Mandant
AircraftAircraftStatus		Nein
Aircrafts	zwei Spalten für Owner: Club oder Person	Nein
AircraftStatus		Nein
Countries		Nein
Flight	Mandantenzuordnung wird über Flugtyp bestimmt	Nein
FlightCrew		Nein
FlightTypes	bestimmt, welchem Mandanten der Flug zugeordnet wird	Ja
Club		Nein
Locations		Nein
LocationTypes	statisch, unveränderbar	Nein
MemberStatus	werden pro Mandant verwaltet	Ja
PersonCategories	werden pro Mandant verwaltet	Ja
PersonClub	Zuordnung der Person einem Mandanten, enthält Rating	Nein
PersonMemberStatus	Zuordnung der Person einem Mandanten als Vereinsmitglied	Nein
PersonPersonCategories	Zuordnung einer Person zu einer PersonCategory	Nein
Persons	kann mehreren oder keinem Mandanten zugeordnet sein, über PersonCategories / PersonMemberStatus / PersonClub bestimmt	Nein
Roles		Nein
StartTypes	statisch, unveränderbar	Nein
SystemData		Nein
SystemLogs		Nein
SystemVersion		Nein
User	User kann nur einem Mandanten zugeteilt sein	Ja
UserRoles		Nein

25.1 Berechtigungskonzept

Um den Zugriff auf die Daten zu kontrollieren und zu beschränken, wurde ein Berechtigungskonzept erstellt. Dieses zeigt die Berechtigungen auf den CRUD-Operationen (Create, Read, Update, Delete) für die einzelnen Rollen. Es werden drei Berechtigungstypen unterschieden: FlightOperator, ClubAdministrator und SystemAdministrator. Diese Typen werden weiter unterteilt nach Berechtigung für den eigenen Mandant oder für alle Mandanten.

Berechtigungskonzept

Prozess	Bemerkung	CRUD-Operation	FlightOperator eigener Mandant	FlightOperator fremder Mandant	ClubAdministrator eigener Mandant	ClubAdministrator fremder Mandant	Systemadministrator eigener + fremder Mandant
Aircraft		Create	X		X		X
		Read	X	X	X	X	X
		Update	X		X		X
		Delete	X		X		X
Country		Create					X
		Read	X	X	X	X	X
		Update					X
		Delete					X
Flight		Create	X		X		X
		Read	X		X		X
		Update	X		X		X
		Delete	X		X		X
Location		Create			X	X	X
		Read	X	X	X	X	X
		Update			X	X	X
		Delete			X	X	X
LocationType	**muss nicht in Applikation bearbeitet werden**	Create					
		Read	X	X	X	X	X
		Update					
		Delete					
Mandant + FlightType		Create					X
		Read	X	X	X	X	X
		Update			X		X
		Delete					X
Person	**- * nur Adresse + Rating sollen mandanten-übergreifend sichtbar sein, jedoch keine TelNr. etc -** Ein Flugoperator kann eine Person erstellen, diese als Mitglied dem eigenen Mandaten hinzufügen kann aber nur der ClubAdministrator + Systemadministrator - Rating von Person kann pro Club unterschiedlich sein - Jeder Benutzer kann die eigene Person lesen und bearbeiten**	Create	X**		X		X
		Read	X	(X)*	X	(X)*	X
		Update	X		X		X
		Delete	X		X		X
StartType	**muss nicht in Applikation bearbeitet werden**	Create					
		Read	X	X	X	X	X
		Update					
		Delete					
User		Create			X		X
		Read	X	X	X	X	X
		Update			X		X
		Delete			X		X

25.1.1 Umsetzung des Berechtigungskonzepts

Die Tabelle auf der nächsten Seite zeigt folgende Punkte auf:

- In der Spalte „Recht“ ist die Bezeichnung.
- Einige Berechtigungen haben andere Berechtigungen als Voraussetzung. Umgesetzt ist dies durch Constraints in der Datenbank. In der Tabelle sind diese Berechtigungen in der Spalte „Benötigt Recht“ zu finden.
- In der Spalte „PermissionClientModel“ ist aufgeführt, mit welchem Property das betreffende Recht in der Klasse PermissionClientModel umgesetzt ist.
- In der Spalte „File“ ist aufgeführt, in welchem File das Property aus dem PermissionClientModel verwendet wird.
- In der Spalte „Element/Methode/Property“ ist das Element, die Methode oder das Property aufgelistet, welches dieses Recht umsetzt.

Recht	Benötigt Recht	PermissionClientModel	File	Element / Methode / Property
CreateAircraftClient	UpdateAircraftClient	CreateAircraft	AircraftView.xaml	NewAircraft Button
CreateAircraftOtherClient	CreateAircraftClient, UpdateAircraftOtherClient	CreateAircraftOtherClient	AircraftView.xaml	ClubDataField
ReadAircraftClient	ReadClient, ReadPerson	ReadAircraft	MainMenu.xaml	AircraftMenuItem
UpdateAircraftClient	ReadAircraftClient	UpdateAircraft	AircraftView.xaml	Save Button, Cancel Button, ContentControl im DataForm
UpdateAircraftOtherClient	UpdateAircraftClient	UpdateAircraftOtherClient	AircraftView.xaml	ContentControl im DataForm
DeleteAircraftClient	UpdateAircraftClient	DeleteAircraft	AircraftView.xaml	DeleteAircraft Button
DeleteAircraftOtherClient	UpdateAircraftOtherClient DeleteAircraftClient	DeleteAircraftOtherClient	AircraftView.xaml	DeleteAircraft Button
FlightsOtherClient	ReadFlightClient, ReadAircraftClient	FlightOtherClient	FlightClientModel.cs	LoadFlights
			StartListViewModel	FlightTypes
CreateFlightClient	UpdateFlightClient	CreateFlightClient	StartListView.xaml	AddFlight Button
ReadFlightClient	ReadLocation, ReadPerson, ReadAircraftClient	ReadFlight	MainMenu.xaml	StartListMenuItem
UpdateFlightClient	ReadFlightClient	UpdateFlightClient	StartListView.xaml	DataGrid mit GroupedFlights, StartButton, SaveButton
			ReadOnlyFlightPanel.xaml	LandungsButton
ReadLocation	-	ReadLocation	MainMenu.xaml	LocationMenuItem
DeleteLocation	UpdateLocation	DeleteLocation	LocationView.xaml	DeleteLocationButton
CreateLocation	UpdateLocation	CreateLocation	LocationView.xaml	AddLocationButton
UpdateLocation	ReadLocation	UpdateLocation	LocationView.xaml	Save Button, Cancel Button, ContentControl im DataForm
CreateClient	UpdateClient	CreateClub	ClubView.xaml	AddClubButton
UpdateClient	ReadClient	UpdateClub	ClubView.xaml	Save Button, Cancel Button, ContentControl im DataForm
UpdateOtherClient	UpdateClient	UpdateClubOtherClient		ContentControl im DataForm
DeleteClient	UpdateClient	DeleteClub		DeleteClubButton
ReadClient	ReadLocation	ReadClub	MainMenu.xaml	ClubMenuItem
CreatePerson	UpdatePersonClient	CreatePerson	PersonView.xaml	AddPersonButton

ReadPerson	-	ReadPerson	MainMenu.xaml	PersonMenuItem
ReadPersonSensitive	ReadPerson	ReadPersonReadPersonSensitive	MainMenu.xaml PersonDataView.xaml	PersonMenuItem Communication-AccordionItem
UpdateOwnPerson	-	UpdateOwnPerson	PersonDataView.xaml	Save Button, Cancel Button, ContentControl im DataForm
UpdatePersonClient	ReadPerson	UpdatePerson	PersonDataView.xaml	Save Button, Cancel Button, ContentControl im DataForm
UpdatePersonOtherClient	UpdatePersonClient	UpdatePersonOtherClient	PersonDataView.xaml	ContentControl im DataForm
DeletePersonClient	UpdatePersonClient	DeletePerson	PersonDataView.xaml	Delete Button
DeletePersonOtherClient	DeletePersonClient, UpdatePersonOtherClient	DeletePersonOtherClient	PersonDataView.xaml	Delete Button
CreateUser	UpdateUser			AddButton
ReadUser	ReadPerson	ReadClub	MainMenu.xaml	UserMenuItem
DeleteUser	UpdateUser			Delete Button
UpdateUser	ReadUser			Save Button, Cancel Button, DataGrid
IsSuperUser	All other Rights	IsSuperUser	User.cs RoleUserMatched.cs UserViewModel.cs	CanCreateUserInOtherClient CanUserChangeRole FilterRecords

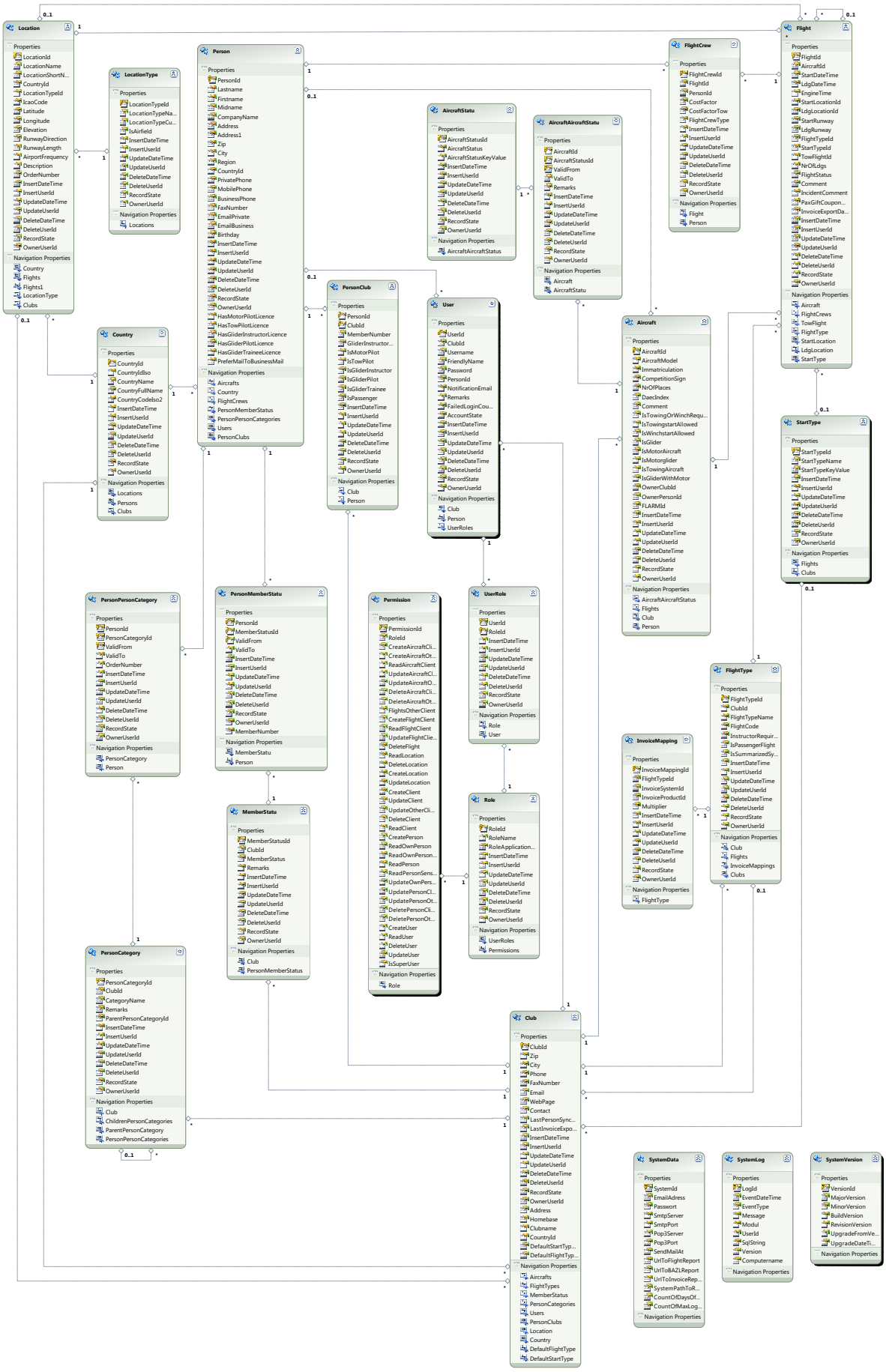
26 Datenspeicherung

FLS benutzt eine Datenbank, welche auf einem Microsoft SQL-Server 2008 R2 Express läuft. Diese Datenbank erstellte Patrick Schuler und für dieses Projekt verwendet. Während des Projekts wurden einige Anpassungen vorgenommen und neue Tabellen hinzugefügt, um weitere Anforderungen erfüllen zu können.

26.1 Datenbank

Nachfolgendes Diagramm zeigt die einzelnen Tabellen der Datenbank sowie deren Verbindungen untereinander. Darauf folgend befindet sich eine Beschreibung der einzelnen Tabellen, wobei nur die wichtigsten Attribute aufgeführt sind.

Zu Beginn des Projekts wurde vorgeschlagen, die Verwaltungsdaten, die in jeder Tabelle vorhanden sind (UpdateDateTime, UpdateUserId etc.), in eine Obertabelle auszulagern. Davon wurde jedoch im Verlauf des Projekts abgesehen. Die Struktur mit einer abstrakten Obertabelle wird im Entity Data Model Designer von ADO.NET mit „Tabelle pro Typ“ nicht unterstützt. Es hätten die generierten Dateien für jede Tabelle einzeln angepasst werden müssen. Aus Zeitgründen und aufgrund minimen Nutzens aus dieser Anpassung wurde die Struktur belassen.


Location
Properties
LocationId
LocationName
LocationShortN...
CountryId
LocationTypeId
IcaoCode
Latitude
Longitude
Elevation
RunwayDirection
RunwayLength
AirportFrequency
Description
OrderNumber
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Country
Flights
Flights1
LocationType
Clubs
LocationType
Properties
LocationTypeId
LocationTypeNa...
LocationTypeCu...
IsAirfield
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Locations
Person
Properties
PersonId
Lastname
Firstname
Midname
CompanyName
Address
Address1
Zip
City
Region
CountryId
PrivatePhone
MobilePhone
BusinessPhone
FaxNumber
EmailPrivate
EmailBusiness
Birthday
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
HasMotorPilotLicence
HasTowPilotLicence
HasGliderInstructorLicence
HasGliderPilotLicence
HasGliderTraineeLicence
PreferMailToBusinessMail
Navigation Properties
Aircrafts
Country
FlightCrews
PersonMemberStatus
PersonPersonCategories
Users
PersonClubs
AircraftStatu
Properties
AircraftStatusId
AircraftStatus
AircraftStatusKeyValue
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
AircraftAircraftStatus
AircraftAircraftStatu
Properties
AircraftId
AircraftStatusId
ValidFrom
ValidTo
Remarks
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Aircraft
AircraftStatu
FlightCrew
Properties
FlightCrewId
FlightId
PersonId
CostFactor
CostFactorTow
FlightCrewType
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Flight
Person
Flight
Properties
FlightId
AircraftId
StartDateTime
LdgDateTime
EngineTime
StartLocationId
LdgLocationId
StartRunway
LdgRunway
FlightTypeId
StartTypeId
TowFlightId
NrOfLdgs
FlightStatus
Comment
IncidentComment
PaxGiftCoupon...
InvoiceExportDa...
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Aircraft
FlightCrews
TowFlight
FlightType
StartLocation
LdgLocation
StartType
PersonClub
Properties
PersonId
ClubId
MemberNumber
GliderInstructor...
IsMotorPilot
IsTowPilot
IsGliderInstructor
IsGliderPilot
IsGliderTrainee
IsPassenger
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Club
Person
User
Properties
UserId
ClubId
Username
FriendlyName
Password
PersonId
NotificationEmail
Remarks
FailedLoginCou...
AccountState
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Club
Person
UserRoles
Aircraft
Properties
AircraftId
AircraftModel
Immatriculation
CompetitionSign
NrOfPlaces
DaecIndex
Comment
IsTowingOrWinchRequ...
IsTowingstartAllowed
IsWinchstartAllowed
IsGlider
IsMotorAircraft
IsMotorglider
IsTowingAircraft
IsGliderWithMotor
OwnerClubId
OwnerPersonId
FLARMId
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
AircraftAircraftStatus
Flights
Club
Person
StartType
Properties
StartTypeId
StartTypeName
StartTypeKeyValue
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Flights
Clubs
Country
Properties
CountryId
CountryIdIso
CountryName
CountryFullName
CountryCodeIso2
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Locations
Persons
Clubs
PersonPersonCategory
Properties
PersonId
PersonCategoryId
ValidFrom
ValidTo
OrderNumber
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
PersonCategory
Person
PersonMemberStatu
Properties
PersonId
MemberStatusId
ValidFrom
ValidTo
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
MemberNumber
Navigation Properties
MemberStatu
Person
Permission
Properties
PermissionId
RoleId
CreateAircraftCli...
CreateAircraftOt...
ReadAircraftClient
UpdateAircraftCl...
UpdateAircraftO...
DeleteAircraftCli...
DeleteAircraftOt...
FlightsOtherClient
CreateFlightClient
ReadFlightClient
UpdateFlightClie...
DeleteFlight
ReadLocation
DeleteLocation
CreateLocation
UpdateLocation
CreateClient
UpdateClient
UpdateOtherCli...
DeleteClient
ReadClient
CreatePerson
ReadOwnPerson
ReadOwnPerson...
ReadPerson
ReadPersonSens...
UpdateOwnPers...
UpdatePersonCl...
UpdatePersonOt...
DeletePersonCli...
DeletePersonOt...
CreateUser
ReadUser
DeleteUser
UpdateUser
IsSuperUser
Navigation Properties
Role
UserRole
Properties
UserId
RoleId
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Role
User
Role
Properties
RoleId
RoleName
RoleApplication...
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
UserRoles
Permissions
MemberStatu
Properties
MemberStatusId
ClubId
MemberStatus
Remarks
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Club
PersonMemberStatus
PersonCategory
Properties
PersonCategoryId
ClubId
CategoryName
Remarks
ParentPersonCategoryId
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Club
ChildrenPersonCategories
ParentPersonCategory
PersonPersonCategories
FlightType
Properties
FlightTypeId
ClubId
FlightTypeName
FlightCode
InstructorRequir...
IsPassengerFlight
IsSummarizedSy...
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
Club
Flights
InvoiceMappings
Clubs
InvoiceMapping
Properties
InvoiceMappingId
FlightTypeId
InvoiceSystemId
InvoiceProductId
Multiplier
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Navigation Properties
FlightType
Club
Properties
ClubId
Zip
City
Phone
FaxNumber
Email
WebPage
Contact
LastPersonSync...
LastInvoiceExpo...
InsertDateTime
InsertUserId
UpdateDateTime
UpdateUserId
DeleteDateTime
DeleteUserId
RecordState
OwnerUserId
Address
Homebase
Clubname
CountryId
DefaultStartTyp...
DefaultFlightTyp...
Navigation Properties
Aircrafts
FlightTypes
MemberStatus
PersonCategories
Users
PersonClubs
Location
Country
DefaultFlightType
DefaultStartType
SystemData
Properties
SystemId
EmailAdress
Passwort
SmtpServer
SmtpPort
Pop3Server
Pop3Port
SendMailAt
UrlToFlightReport
UrlToBAZLReport
UrlToInvoiceRep...
SystemPathToR...
CountOfDaysOf...
CountOfMaxLog...
Navigation Properties
SystemLog
Properties
LogId
EventDateTime
EventType
Message
Modul
UserId
SqlString
Version
Computername
Navigation Properties
SystemVersion
Properties
VersionId
MajorVersion
MinorVersion
BuildVersion
RevisionVersion
UpgradeFromVe...
UpgradeDateTi...
Navigation Properties

AircraftAircraft-Status	
Beschreibung	Zwischentabelle, welche einer Aircraft für eine Zeitspanne einen AircraftStatus zuordnet.
Attribute	
Beziehungen	• Ein AircraftAircraftStatus gehört zu einer Aircraft. • Ein AircraftAircraftStatus gehört zu einem AircraftStatus.

Aircrafts	
Beschreibung	Tabelle der Flugzeuge.
Attribute	• AircraftModel: Modell des Flugzeuges. • Immatriculation: Code, der ein Flugzeug eindeutig identifiziert. • DaecIndex: Ermöglicht den Vergleich der Streckenleistung für Wettbewerbe. • OwnerClubId/OwnerPersonId: Flugzeug kann einem Verein oder einer Person gehören. • FLARMid: Id des Kollisionswarngeräts.
Beziehungen	• Ein Aircraft hat mehrere AircraftAircraftStatus. • Ein Aircraft kann einer Person gehören. • Ein Aircraft kann einem Club gehören. • Ein Aircraft gehört zu mehreren Flügen.

AircraftStatus	
Beschreibung	Beschreibt den Status eines Flugzeuges.
Attribute	
Beziehungen	• Ein AircraftStatus hat mehrere AircraftAircraftStatus.

Clubs	
Beschreibung	Tabelle der Segelfluggruppen.
Attribute	• ClubName: Name der Segelfluggruppe. • Contact: Kontaktperson innerhalb der Segelfluggruppe.
Beziehungen	• Ein Club hat mehrere Aircrafts. • Ein Club hat mehrere FlugTypes. • Ein Club hat mehrere Users. • Ein Club hat mehrere MemberStatus. • Ein Club hat mehrere PersonCategories.

Country	
Beschreibung	Die Country-Tabelle beinhaltet sämtliche Länder. Sie ändert sich nur äusserst selten, weshalb Aktualisierungen direkt in die Datenbank eingelesen werden.
Attribute	• CountryName
Beziehungen	• Ein Country hat mehrere Locations. • Ein Country hat mehrere Personen.

FlightCrew	
Beschreibung	Ist für die Verlinkung zwischen Personen und Flights zuständig.
Attribute	• CostFactor: Prozentanteil an zu übernehmenden Kosten des Fluges. • CostFactorTow: Prozentanteil an zu übernehmenden Kosten des Schleppfluges. • FlightCrewType: z.B. Pilot, Copilot, Fluglehrer.
Beziehungen	• Eine FlightCrew gehört zu einem Flight. • Eine FlightCrew gehört zu einer Person.

Flight	
Beschreibung	In dieser Tabelle werden die einzelnen Flüge festgehalten.
Attribute	• StartDateTime: Startzeit des Flugs. • LdgDateTime: Landezeit des Flugs. • EngineTime: Laufdauer des Motors (falls vorhanden). • NrOfLdgs: Anzahl Landungen während eines Flugs.
Beziehungen	• Ein Flight kann einen weiteren Flight haben (z.B. Segelflug und dazugehöriger Schleppflug). • Ein Flight kann einen StartType haben. • Ein Flight hat mehrere FlightCrews. • Ein Flight findet mit einem Aircraft statt. • Ein Flight hat einen FlightType. • Ein Flight hat eine Start-Location. • Ein Flight hat eine Lande-Location.

FlightTypes	
Beschreibung	Teilt die Flüge in Kategorien ein. Jeder Mandant hat seine eigenen FlightTypes.
Attribute	• FlightTypeName: Name des Flugtyps. • InstructorRequired: Ausbildungsflug, es wird ein Fluglehrer benötigt. • IsPassengerFlight: Flug mit einer Person, welche nicht Mitglied im Verein ist. • IsSummarizedSystemFlight: Zusammengefasste Stunden über bereits geflogene Stunden vor Einführung des FLS.
Beziehungen	• Ein FlightType hat mehrere InvoiceMappings. • Ein FlightType hat mehrere Flights. • Ein FlightTyp gehört zu einem Club.

InvoiceMapping	
Beschreibung	Tabelle mit zusätzlichen Informationen für Rechnungsstellung. Diese Tabelle wird nicht verwendet.
Attribute	
Beziehungen	• Ein InvoiceMapping gehört zu einem FlightType.

Location	
Beschreibung	Beinhaltet die Flugplätze.
Attribute	• LocationName: Name der Location. • IcaoCode: Einzigartiger Code der Internationalen Zivilluftfahrt-organisation. • Latitude: Breitengrad des Flugplatzes. • Longitude: Längengrad des Flugplatzes. • RunwayDirection: Richtung der Landebahn. • RunwayLength: Länge der Landebahn.
Beziehungen	• Eine Location gehört zu einem Country. • Eine Location gehört zu einem LocationType. • Eine Location kann von Flights verwendet werden.

LocationTypes	
Beschreibung	Kategorisiert die Locations in verschiedene Typen, wobei hauptsächlich zwischen Flugplatz und Nicht-Flugplatz unterschieden wird.
Attribute	• LocationTypeName: Typ der Location, z.B. Flugplatz mit Grasspiste, Aussenlandefeld, Segelflugplatz, ... • isAirfield: Unterscheidung ob Flugplatz oder nicht.
Beziehungen	• Ein LocationType gehört zu mehreren Locations.

MemberStatus	
Beschreibung	Gliedert die Personen innerhalb eine Fluggruppe. Jeder Mandant erfasst seine eigenen MemberStatus.
Attribute	
Beziehungen	• Ein MemberStatus gehört zu einem Club. • Ein MemberStatus hat mehrere PersonMemberStatus.

PersonCategory	
Beschreibung	Ordnet Personen (über PersonPersonCategory) Mandanten zu.
Attribute	
Beziehungen	• Eine PersonCategory kann eine weitere PersonCategory haben. • Eine PersonCategory hat einen Club. • Eine PersonCategory hat mehrere PersonPersonCategories.

PersonMember-Status	
Beschreibung	Beschreibt, was für ein MemberStatus eine Person während eines Zeitraums besessen hat.
Attribute	• ValidFrom: Beginn des MemberStatus einer Person. • ValidTo:Ende des MemberStatus einer Person.
Beziehungen	• Ein PersonMemberStatus hat einen MemberStatus. • Ein PersonMemberStatus gehört zu Person.

PersonPersonCategory	
Beschreibung	Zwischentabelle, welche festhällt, wann und wie lange eine Person einer PersonCategory zugeordnet ist/war.
Attribute	• ValidFrom: Datum, wann die Person der PersonCategory zugeordnet wurde. • ValidTo: Enddatum der Zuordnung.
Beziehungen	• Eine PersonPersonCategory ist einer PersonCategory zugeordnet. • Eine PersonPersonCategory gehört zu einer Person.

Person	
Beschreibung	Erfasst die Daten von Personen und was für Flugzeuge diese fliegen dürfen.
Attribute	• Lastname / Firstname / Address / City / ...: Adressdaten und Kontaktdaten einer Person. • IsMotorPilot / IsGliderInstructor / ...: Welche Flugzeuge eine Person fliegen darf.
Beziehungen	• Eine Person lebt in einem Country. • Eine Person kann mehrere Aircrafts haben. • Eine Person kann mehrere User haben. • Eine Person gehört zu mehreren FlightCrews. • Eine Person hat mehrere PersonMemberStatus. • Eine Person hat mehrere PersonPersonCategories.

Role	
Beschreibung	Tabelle mit den möglichen Berechtigungsrollen für das FLS.
Attribute	
Beziehungen	• Eine Role hat mehrere UserRoles.

StartType	
Beschreibung	Tabelle mit statischen Werten. Hilft bei der Dateneingabe indem Einschränkungen gemacht werden.
Attribute	
Beziehungen	• Ein StartType hat mehrere Flights.

SystemData	
Beschreibung	Tabelle enthält Einträge für die Verbindung zum SMTP-Server, zum Reporing Server sowie Default-Einstellungen für die Applikation.
Attribute	
Beziehungen	

SystemLogs	
Beschreibung	Logt fehlgeschlagene Operationen.
Attribute	• UserId: User, bei dem der Fehler aufgetreten ist. • SqlString: Query, welche den Fehler verursacht hat. • Computername: Name des Computers, bei dem der Fehler aufgetreten ist.
Beziehungen	

SystemVersion	
Beschreibung	Enthält die aktuellen Versiondaten der Datenbank.
Attribute	
Beziehungen	

UserRole	
Beschreibung	Mapping zwischen User und Role.
Attribute	
Beziehungen	• Eine UserRole gehört zu einer Role. • Eine UserRole gehört zu einem User.

User	
Beschreibung	Enthält sämtliche Benutzer des Systems. Jeder Mandant hat seine eigene User-Tabelle.
Attribute	• Username: Eindeutiger Name, max. 50 Zeichen. • Password: Wird als SHA1-Hashwert abgelegt. • FailedLoginCount: Zählt fehlgeschlagene Logins. Nach 10 fehlgeschlagenen Versuchen wird der User gesperrt und kann nur von einem Sysadmin wieder aktiviert werden.
Beziehungen	• Ein User hat mehrere UserRoles. • Ein User gehört zu einem Club. • Ein User kann zu einer Person gehören.

PersonClub	
Beschreibung	Verwaltet die Beziehung zwischen Person und Club. Beinhaltet das Rating.
Attribute	
Beziehungen	• Ein PersonClub hat eine Person. • Ein PersonClub hat einen Club.

Permissions	
Beschreibung	Enthält die kleinstmögliche Einheit für eine Berechtigung.
Attribute	
Beziehungen	• Eine Permission hat eine Rolle.

27 Angaben zur Installation

Das Deployment bzw. die Installation dieser Web-Applikation auf einem Web-Server war nicht Teil dieses Projekt. Deshalb enthält diese Dokumentation keine Installationsanleitung.

Trotzdem sind während des Projekts einige Punkte aufgekommen, die applikationsspezifisch sind und deshalb folgend beschrieben werden. Die aufgeführten Programme wurden bei der Entwicklung benutzt, können aber durch ein anderes passendes Programm ersetzt werden.

FLS Haupt-Applikation

- In der Datei Web.config unter Applikation/FLS/FLS.Web muss die Datenbankverbindung angepasst werden.
- Die Solution FLS.sln unter Applikation/FLS müssen deployed werden.

Datenbank

- Es wird ein Microsoft SQL Server 2008 R2 benötigt.
- Unter Datenbank/scripts befinden sich die SQL-Scripts. Diese müssen im Microsoft SQL Server 2008 R2 ausgeführt werden.

Reporting

- Es wird ein Microsoft SQL Server Reporting Service benötigt, worüber die Reports erstellt werden.
- Die Einstellungen des Reporting Servers müssen berücksichtigen, dass für den Zugriff auf die Reports keine Administratoren-Rechte benötigt werden.
- Die Reports in der Solution Applikation/FLS/ReportsVS2008_SSRS/FlightReport.sln müssen deployed werden.
- In der Datei FLS.rds unter Applikation/FLS/ReportsVS2008_SSRS/FlightReport muss die Datenbankverbindung angepasst werden.
- In der Tabelle SystemData müssen die URL zum Reporting Server und zu den einzelnen Reports gepflegt werden.
- Im Feld SendMailAt wird bestimmt, wann die regelmässigen Reports erstellt und versendet werden. Die Zeit ist so einzutragen, dass der Tag des DateTime den Versand des monatlichen Reports (mit Vormonatsdaten) und die Zeit des Eintrages den Zeitpunkt des täglichen Reports bestimmt .

Email-Versand

- Auf dem Server sollte ein SMTP-Dienst eingerichtet sein, der von FLS für das Versenden der Reports und der Passwörter benutzt wird.
- Die Verbindungsangaben zum SMTP-Dienst können in der Datenbank in Tabelle SystemData abgelegt werden.

XIII

Code-Review

Datum: 15. Juni 2012

28 Code-Review

28.1 Abkürzungen

- lm: Lara Mühlemann
- mf: Marion Frei
- mw: Marion Walser

28.2 Allgemein

Datum/ Reviewer	Kommentar	Refactoring
19.03.2012 lm, mf	Template für Klassen muss definiert werden.	Template wurde definiert und in allen Klassen eingepflegt.
19.03.2012 lm, mf	Smell MagicNumbers muss in allen Klassen behoben werden.	MagicNumbers mit Ressoucen eliminiert.

28.3 FLS.Service

28.3.1 CustomMembershipProvider.cs

Datum/ Reviewer	Kommentar	Refactoring
05.04.2012 lm, mw	Viele Methoden sind nicht implementiert. Es muss überprüft werden, ob diese Methoden gebraucht werden oder nicht.	Die meisten Methoden werden nicht benötigt und deshalb nicht implementiert.
06.01.2012 lm, mw	Klasse gehört ins Projekt Service, da AuthenticationBase diese aufruft und ebenso im Projekt Service ist.	Klasse von Projekt Web in Projekt Service verschoben.

28.4 FLS.Client

28.5 FLS.Client.ClientModels

28.5.1 AircraftClientModel.cs

Datum/ Reviewer	Kommentar	Refactoring
27.04.2012 lm, mf	Es ist nicht nötig das Laden der Daten über Callbacks zu realisieren.	Callbacks entfernt

28.5.2 FlightClientModel.cs

Datum/ Reviewer	Kommentar	Refactoring
27.04.2012 lm, mf	Die Klasse FlightClientModel darf nur einen Context referenzieren. Die Methoden für die Locations müssen in ein eigenes ClientModel verschoben werden.	verschoben
01.06.2012 lm, mf	Im Property FlightStatus wird bei jedem Get eine neue Liste erstellt, obwohl diese sich nicht verändert.	private Variabel mit Liste erstellt. Property gibt mit Get immer die selbe Liste zurück.
01.06.2012 lm, mf	Methode LoadFlights ist etwas gross.	Extract Method für Laden von allen Flügen und Laden von nur Club-Flügen.
01.06.2012 lm, mf	ReloadFlights ruft nur LoadFlights auf und ist darum überflüssig.	Methode wurde inlined
01.06.2012 lm, mf	Methode StartTypeCallback ist überflüssig.	Methode wurde entfernt.

28.5.3 FLS.Client.ClientService

28.5.4 FlightClientService.cs

Datum/ Reviewer	Kommentar	Refactoring
01.06.2012 lm, mf	AddHomeBaseAsStartLocation hat Aufgaben, die eigentlich der Flug ausführen sollte.	In FLS.Data.Flight.shared.cs verschoben und auf SetStartAndLandingLocation umbenannt.
01.06.2012 lm, mf	CopyValuesFromFlightToTowFlight + SetFlightTypeToDefault + AddFlightCrew haben Aufgaben, die eigentlich der Flug ausführen sollte.	In FLS.Data.Flight.shared.cs verschoben.
01.06.2012 lm, mf	AddTowFlight ist überflüssig, es kann direkt CreateDefaultTowFlight benutzt werden.	Unnötige Aufrufe entfernt + Inline
01.06.2012 lm, mf	Das Erstellen des Schleppflugs und des normalen Flugs ist sehr ähnlich (douplicated code)	Gemeinsame Methode wird nun benutzt.

28.6 FLS.Client.ViewModels

28.6.1 StartListViewModel.cs

Datum/ Reviewer	Kommentar	Refactoring
01.06.2012 lm, mw	Die Methode StartTypeChanged benutzt ein Property IsCalledByRevertingStartType. Dieses ist dazu da, eine Endlosschleife zu beenden, wenn der Benutzer auf Cancel drückt und dadurch der Trigger dieser Methode abermals aufgerufen wird. Ist es möglich dies besser zu lösen?	Keine Möglichkeit gefunden.

28.6.2 AircraftViewModel.cs

Datum/ Reviewer	Kommentar	Refactoring
01.06.2012 lm, mw	Der Setter des Property SelectedAircraft hat eine hohe Komplexität	Durch Umsortieren + Löschen und Extract Method verringert.

28.6.3 UserViewModel.cs

Datum/ Reviewer	Kommentar	Refactoring
19.03.2012 lm, mf	Errormessage in Methode UserSave muss anders gemacht werden.	Methode wurde in ViewModelBase verschoben
19.03.2012 lm, mf	Methode SelectedItemChanged: Kann Reflection vermieden werden?	Methode wurde eliminiert
26.03.2012 lm, mw	Property SelectedItem: Update RolesUserMatched verbessern	Linq verwendet
05.04.2012 lm, mw	OnRolesLoadCompleted-Methode ist nicht mehr nötig.	Methode entfernt.

28.6.4 LoginViewModel.cs

Datum/ Reviewer	Kommentar	Refactoring
27.04.2012 lm, mf	Im Switch in InitErrorTextFromAccountStatus fehlt der default Case	ergänzt.
27.04.2012 lm, mf	Die Methode Login ist zu lang	

28.6.5 WelcomeViewModel.cs

Datum/ Reviewer	Kommentar	Refactoring
27.04.2012 lm, mf	Klasse ist überflüssig und kann entfernt werden	entfernt.

28.6.6 ViewModelBase.cs

Datum/ Reviewer	Kommentar	Refactoring
19.03.2012 lm, mf	Methode Save, die in mehreren Klassen gleich ist, soll in Oberklasse nur einmal implementiert werden.	Methode wurde in ViewModelBase verschoben

28.7 FLS.Data

28.7.1 Flight.shared.cs

Datum/ Reviewer	Kommentar	Refactoring
01.06.2012 lm, mw	Enums sollten in eigene Files ausgelagert werden.	Eigenes shared.cs für jeden Enum angelegt in einem neuen Namespace Enums.

28.7.2 FLSModel.Designer.cs

Datum/ Reviewer	Kommentar	Refactoring
26.03.2012 lm, mw	Daten (Insert, Update, Delete) sollen den einzelnen Tabellen vererbt werden (Überklasse erstellen).	Oberklasse in Entities wurde nicht umgesetzt, da zu wenig Nutzen und Aufwand zu gross. (Nicht über Designer machbar)

28.8 FLS.TestMiddleTier

28.9 FLS.Utilities

28.10 FLS.Web

28.10.1 CustomRoleProvider.cs

Datum/ Reviewer	Kommentar	Refactoring
01.06.2012 lm, mw	Diese Klasse wird nirgends gebraucht.	Klasse wurde gelöscht.

XIV

Systemtests

Datum: 15. Juni 2012

29 Voraussetzung

29.1 Infrastruktur

Es werden für die Tests die folgenden Computer verwendet.

Computer	Ausrüstung
Laptop1	Packard Bell, 4 GB RAM, Intel©Core™2 Solo CPU U355 1.40 GHz, 32 Bit-Betriebsystem, Windows Vista Home Premium, Internet Explorer 8
Laptop2	Mysn, 4GB RAM, Intel©Core™2 DUO CPU P9700 2.80 GHz, 64 Bit-Betriebsystem, Windows 7 Professional, Mozilla Firefox 11.0
Laptop3	Asus, 6 GB RAM, Intel©Core™i7-2630 QM CPU @ 2.00 GHz, 64 Bit-Betriebssystem, Windows 7 Professional, Google Chrome 17.0.963.46

29.2 Benutzer

Folgende Benutzer müssen im System angelegt sein:

Benutzername	Passwort	Rollen	Club
SegelflugVerband	SegelflugVerband	System Administrator	System Verein
Admin	Admin	Verein Administrator	Flugsportgruppe Zürcher Oberland
User	User	Flugdienstleiter	Flugsportgruppe Zürcher Oberland

30 Testspezifikation

30.1 Test 1: Mandant verwalten

Use Case: UC01: Mandant verwalten

30.1.1 Testdetail Rolle System Administrator

Voraussetzungen:

- Der Tester ist mit dem Benutzer Segelflugverband eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Vereine.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Vereins-Ansicht.
T03	Der Tester klickt auf den Button „Neuer Verein“.		Es erscheint ein leeres Eingabeformular.
T04	Der Tester klickt auf den Button „Speichern“.		Es erscheint eine Fehlermeldung, die nach dem Vereinsnamen und dem Land verlangt.

T05	Der Tester gibt folgende Daten ein und drückt auf den Button „Speichern“:	Name: Himmelstürmer Land: Schweiz	Der Verein wird gespeichert und erscheint in der Tabelle.
T06	Der Tester klickt auf den Button „Neuer Verein“.		Es erscheint ein leeres Eingabeformular.
T07	Der Tester gibt folgende Daten ein und speichert:	Verein: Himmelstürmer Land: Schweiz	Es erscheint eine Fehlermeldung die besagt, dass bereits ein Verein mit diesem Namen existiert.
T08	Der Tester gibt einen anderen Namen ein und speichert:	Name: Wolkenflieger	Der Verein wird gespeichert und erscheint in der Tabelle. Er erscheint die Meldung „Gespeichert“.
T09	Der Tester wählt im Menu Stammdaten Benutzer aus.		Die Benutzeransicht erscheint.
T10	Der Tester klickt auf den Button „Neuer Benutzer“.		Es erscheint eine leeres Eingabeformular.
T11	Der Tester erstellt einen Benutzer für den eben erstellten Verein Himmelstürmer und klickt auf den Button „Speichern“.	Benutzer: HimmelstürmerAdmin Rufname: HimmelstürmerAdmin E-Mail: h@h.ch Rolle: Verein Administrator Verein: Himmelstürmer	Der neue Benutzer erscheint in der Tabelle.

30.1.2 Testdetail Rolle Verein Administrator

Voraussetzungen:

- Der Tester ist mit dem Benutzer Admin eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Vereine.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Vereins-Ansicht. Es hat keinen „Neuer Verein“ Button.
T03	Der Tester wählt seinen eigenen Verein (Flugsportgruppe Zürcher Oberland) an.		Es erscheint die Detailansicht des Vereins.
T04	Der Tester kann die Daten des Vereins abändern. Er gibt folgende Daten ein und drückt auf den Button „Speichern“:	Telefon: 071 722 22 22	Die neue Telefonnummer wird gespeichert und erscheint in der Tabelle.
T05	Der Tester wählt den Verein „System Verein“ an.		Es erscheint die Detailansicht des Vereins. Der Tester kann die Daten des Vereins nicht abändern.

30.1.3 Testdetail Rolle Flugdienstleiter

Voraussetzungen:

- Der Tester ist mit dem Benutzer User eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Vereine.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Vereins-Ansicht. Es hat keinen Button „Neuer Verein“.
T03	Der Tester wählt seinen eigenen Verein (Flugsportgruppe Zürcher Oberland) an.		Es erscheint die Detailansicht des Vereins. Die Daten des Vereins sind nicht änderbar.
T04	Der Tester wählt den Verein „System Verein“ an.		Es erscheint die Detailansicht des Vereins. Die Daten des Vereins sind nicht änderbar.

30.2 Test 2: Person verwalten

Use Case: UC02 Person verwalten

30.2.1 Testdetail Rolle Verein Administrator

Voraussetzung:

- Der Tester ist mit dem Benutzer Admin eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Personen.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Personen-Ansicht.
T03	Der Tester klickt auf den Button „Neue Person“.		Es erscheint ein leeres Eingabeformular.
T04	Der Tester klickt auf den Button „Speichern“.		Es erscheint eine Fehlermeldung, die nach dem Nachname, Vorname und Land verlangt.
T05	Der Tester gibt folgende Daten ein und speichert:	Vorname: Hans Nachname: Müller Land: Schweiz	Die Person wird gespeichert und erscheint in der Tabelle.
T06	Der Tester wählt die eben erstellte Person aus, ändert ihre Daten ab und speichert.	Zweiter Vorname: Ueli	Die Person wird gespeichert. Es erscheint die Meldung „Gespeichert“.
T07	Der Tester wählt eine Person aus, die nicht in seinem Verein ist.		Es erscheint die Detailansicht.
T08	Der Tester sieht die Details zur Kommunikation nicht und kann das Rating nicht abändern, die anderen Details schon. Er gibt folgende Daten ein und speichert:	Nachname: Mustermann	Es erscheint die Meldung „Gespeichert“.

30.2.2 Testdetail Rolle Flugdienstleiter

Voraussetzung:

- Der Tester ist mit dem Benutzer User eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Personen.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Personen-Ansicht.
T03	Der Tester wählt eine Person aus seinem Club aus.		In der Detailview erscheinen die nicht sensiblen Daten der ausgewählten Person.
T04	Der Tester gibt folgende Daten ein und bricht danach ab:	Nachname: abc	Die Person wird zurückgesetzt.
T05	Der Tester wählt eine Person aus, die nicht in seinem Club ist.		Es erscheinen die Daten der ausgewählten Person. Es sind nur die Stammdaten änderbar, die anderen nicht.

30.3 Test 3: Benutzer verwalten

Use Case: UC03 Benutzer verwalten

30.3.1 Testdetail Rolle System Verein Administrator

Voraussetzung:

- Der Tester ist mit dem Benutzer Admin eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Benutzer.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Benutzer-Bearbeitungs-Ansicht.
T03	Der Tester klickt auf den Button „Neuer Benutzer“.		Es erscheint ein leeres Eingabeformular.
T04	Der Tester klickt auf den Button „Speichern“.		Es erscheint eine Fehlermeldung, die nach dem Benutzername, dem Rufname und der Emailadresse verlangt.
T05	Der Tester gibt folgende Daten ein und speichert:	Benutzer: ABenutzer, Rufname: ARufname, EMail: AMail@mail.ch	Es erscheint die Information „gespeichert“ und der neue Benutzer erscheint in der Tabelle mit den Benutzern.

T06	Der Tester klickt auf den Button „Neuer Benutzer“.		Es erscheint ein leeres Eingabeformular.
T07	Der Tester gibt folgende Daten ein und klickt danach auf „Speichern“.	Benutzer: BBenutzer, Rufname: BRufname, EMail: b@mail.ch	Es erscheint die Information „gespeichert“ und der neue Benutzer erscheint in der Tabelle mit den Benutzern.
T08	Der Tester wählt in der Tabelle mit den Benutzern den Benutzer ABenutzer aus.		Die Informationen von ABenutzer werden im Bearbeitungsforumlar angezeigt.
T09	Der Tester klickt auf den Button „Benutzer löschen“.		Der Status des Benutzers wird auf gelöscht gesetzt und verschwindet aus der Tabelle.

30.3.2 Testdetail Rolle Flugdienstleiter

Voraussetzung:

- Der Tester ist mit dem Benutzer User eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Benutzer.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Benutzer-Bearbeitungs-Ansicht.

T03		Der Tester hat keinen „Neuer Benutzer“ oder „Benutzer löschen“ Button.
T04	Der Tester wählt einen der Benutzer an.	Die Informationen des Benutzers werden in der Detailview angezeigt.

30.4 Test 4: Flugzeug verwalten

Use Case: UC04 Flugzeug verwalten

30.4.1 Testdetail Rolle System Administrator

Voraussetzung:

- Der Tester ist mit dem Benutzer Segelflugverband eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Flugzeuge.		Die Ansicht Benutzer erscheint.

T03	Der Tester klickt auf den Button „Neues Flugzeug“.		Es erscheint ein leeres Eingabeformular.
T04	Der Tester klickt auf den Button „Speichern“.		Es erscheint eine Fehlermeldung, die nach der Immatrikulation und dem Modell verlangt.
T05	Der Tester gibt folgende Daten ein und speichert	Immatrikulation: HB-1234 Modell: DG-100	Das Flugzeug wird gespeichert und erscheint in der Tabelle.
T06	Der Tester wählt ein bestehenden Flugzeug aus.		Das Flugzeug erscheint in der Detailview.
T07	Der Tester ändert den Verein des Flugzeugs auf Systemverein und speichert.		Das Flugzeug wird gespeichert und erscheint in der Tabelle.

30.4.2 Testdetail Rolle Flugdienstleiter

Voraussetzung:

- Der Tester ist mit dem Benutzer User eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Flugzeuge.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Personen-Ansicht.

T03	Der Tester klickt auf den Button „Neues Flugzeug“.		Es erscheint ein leeres Eingabeformular. Der Tester kann das Feld „Besitzer / Verwalteter Club“-Feld nicht ändern.
T04	Der Tester gibt folgende Daten ein und speichert:	Immatrikulation: HB-1234 Modell: DG-100	Das Flugzeug wird gespeichert und erscheint in der Tabelle.
T05	Der Tester wählt ein Flugzeug eines anderen Vereins aus.		Das Flugzeug erscheint in der Detailview. Der Tester kann das Flugzeug nicht ändern und hat auch keinen „Flugzeug löschen“-Button.

30.5 Test 5: Flugplatz verwalten

Use Case: UC05 Flugplatz verwalten

30.5.1 Testdetail Rolle Verein Administrator

Voraussetzung:

- Der Tester ist mit dem Benutzer Admin eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten

T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Flugplätze.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Flugplatz-Ansicht.
T03	Der Tester klickt auf den Button „Neuer Standort“.		Es erscheint ein leeres Eingabeformular.
T04	Der Tester klickt auf den Button „Speichern“.		Es erscheint eine Fehlermeldung, die nach dem Flugplatz, dem Land und dem Standort Typ verlangt.
T05	Der Tester gibt folgende Daten ein und speichert:	Flugplatz: GrünWiese Land: Schweiz Standort-Typ: Aussenlandefeld	Der Flugplatz wird gespeichert und erscheint in der Tabelle. Es erscheint die Meldung „Gespeichert“.
T06	Der Tester drückt auf den Button „Standort löschen“.		Der Flugplatz wird wieder gelöscht. Er verschwindet aus der Tabelle.

30.5.2 Testdetail Rolle Flugdienstleiter

Voraussetzung:

- Der Tester ist mit dem Benutzer User eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Stammdaten.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt in der Menüauswahl auf Flugplätze.		Das Menü schliesst sich, im Hauptbereich der Seite öffnet sich die Flugplatz-Ansicht.
T03			Der Tester hat keinen Button „Neuer Standort“.
T04	Der Tester wählt einen Standort aus.		Es erscheinen die Details in der Detailview. Der Tester kann den Standort nicht abändern.

30.6 Test 6: Startlisten und Flüge verwalten

Use Case: UC06 Startliste und Flüge verwalten

30.6.1 Testdetail Rolle Flugdienstleiter

Voraussetzung:

- Der Tester ist mit dem Benutzer User eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Startliste.		Die Startliste öffnet sich.
T02	Der Tester klickt auf den Button „Neuer Flug“.		Ein neuer Flug öffnet sich.
T03	Der Tester klickt auf den Button „Speichern“.		Es erscheint die Meldung, dass Segelfliege, Flugart, Schleppflieger ausgefüllt werden muss.
T04	Der Tester füllt folgende Daten ein und speichert:	Segelflieger: HB-1841, Flugart: Normaler Flug, Schlepper: HB-KCB, Pilot: Müller Heinz, Schlepppilot: Mustermann Robert	Der Flug wird gespeichert.
T05	Der Tester ändert die Startart auf Eigenstart.		Es erscheint die Meldung, ob der Schleppflieger wirklich gelöscht werden soll.
T06	Der Tester bestätigt die Meldung.		Es wird kein Schleppflug mehr angezeigt.
T07	Der Tester klickt auf den Button. „Speichern“.		Der Flug wird gespeichert.

30.7 Test 7: Flug-Summary BAZL

Use Case: UC07 Flug-Summary BAZL

30.7.1 Testdetail Rolle Flugdienstleiter

Voraussetzung:

- Der Tester ist mit dem Benutzer User eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Reports.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt auf den Menü-Punkt „Monats-Reports“.		Die Monats-Report-View öffnet sich.
T03	Der Tester wählt bei „Zusammenfassung für BAZL vom Monat“ den Monat Juni 2012 aus und klickt auf den Button „Report öffnen“.		Der Report erscheint als PDF und kann geöffnet/gespeichert werden.

30.8 Test 8: Startlisten-Summary Buchhaltung

Use Case: UC08 Startlisten-Summary Buchhaltung

30.8.1 Testdetail Rolle Flugdienstleiter

Voraussetzung:

- Der Tester ist mit dem Benutzer User eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Reports.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt auf den Menü-Punkt „Monats-Reports“.		Die Monats-Report-View öffnet sich.
T03	Der Tester wählt bei „Zusammenfassung für Rechnungsstellung vom Monat“ den Monat Juni 2012 aus und klickt auf den Button „Report öffnen“.		Der Report erscheint als PDF und kann geöffnet/gespeichert werden.

30.9 Test 9: Flug-Report für Piloten

Use Case: UC09 Flug-Report für Piloten

30.9.1 Testdetail Rolle Flugdienstleiter

Voraussetzung:

- Der Tester ist mit dem Benutzer User eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Reports.		Eine Menüauswahl öffnet sich.
T02	Der Tester klickt auf den Menü-Punkt „Flug-Report“.		Die Flug-Report-View öffnet sich.
T03	Der Tester geht auf den Button „Report öffnen“.		Es erscheint ein Tooltip, der darauf hinweist, dass eine Person ausgewählt sein muss.
T04	Der Tester klickt auf die Combobox neben „Person“.		Es erscheinen alle Personen, die im gleichen Verein wie der Tester sind.
T05	Der Tester wählt eine der Personen aus.		Button „Report öffnen“ wird aktiv.
T06	Der Tester klickt auf den Button „Report öffnen“.		Der Report erscheint als Pdf und kann geöffnet/gespeichert werden.

30.10 Test 10: Synchronisierung Adressdaten

Use Case: UC10 Synchronisierung Adressdaten

30.10.1 Testdetail Export Adressdaten (Rolle Verein Administrator)

Voraussetzung:

- Der Tester ist mit dem Benutzer Admin eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Import/Export, wählt Adressen und Export aus.		Die Ansicht „Export Adressen“ erscheint.
T02	Der Tester klickt auf den Button „Datei selektieren...“, wählt die Datei aus, in die er die Daten abspeichern will und drückt „Ausführen“. Die Selektion wird gemäss Einstellungen belassen.		Der Tester findet die Adressdaten der Personen vom eigenen Club in der Datei, die er selektiert hat.

30.10.2 Testdetail Import Adressen (Rolle Verein Administrator)

Voraussetzung:

- Der Tester ist mit dem Benutzer Admin eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Import/Export, wählt Adressen und Import aus.		Die Ansicht „Import Adressen“ erscheint.
T02	Der Tester klickt auf den Button „Datei selektieren...“, wählt die Datei aus, von der er die Daten einlesen will und drückt „Ausführen“.		Der Tester erhält eine Auflistung der importierten Daten, die er bearbeiten kann.
T03	Der Tester wählt eine Zeilen aus und drückt „Selektierte Standorte entfernen“.		Die zwei selektierten Zeilen verschwinden aus der Auflistung und es erscheint eine Rückmeldung, dass diese entfernt wurden.
T04	Der Tester wählt zwei Zeilen aus, bearbeitet diese so, dass sie keine Fehler mehr haben und drückt „Selektierte Standorte importieren“.		Die zwei selektierten Zeilen verschwinden aus der Auflistung und es erscheint eine Rückmeldung, dass zwei Adressen importiert wurden.
T05	Der Tester navigiert zu den Stammdaten / Personen.		Die zwei importierten Personen erscheinen neu in der Auflistung der Personen.

30.11 Test 11: Import/Export der Flugplätze im CUP-Format

Use Case: UC13 Import/Export der Flugplätze im CUP-Format

30.11.1 Testdetail Export Flugplätze im CUP-Format (Rolle Verein Administrator)

Voraussetzung:

- Der Tester ist mit dem Benutzer Admin eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Import/Export , wählt Standorte und Export aus.		Die Ansicht „Export Flugplätze / Standorte“ erscheint.
T02	Der Tester wählt die Option „Export Standorte in Cup-Format“ aus, drückt auf den Button „Datei selektieren...“, wählt die Datei aus, in die er die Daten exportieren will und drückt „Ausführen“. Die Selektion wird gemäss Einstellungen belassen.		Der Tester findet die Daten aller Standorte in der Datei, die selektiert wurde.

30.11.2 Testdetail Import Flugplätze im CUP-Format (Rolle Verein Administrator)

Voraussetzung:

- Der Tester ist mit dem Benutzer Admin eingeloggt.

Nr.	Aktion	Testdaten	Erwartetes Verhalten
T01	Der Tester klickt im Menü auf Import/Export, wählt Standorte und Import aus.		Die Ansicht „Import Flugplätze / Standorte“ erscheint.
T02	Der Tester klickt auf den Button „Datei selektieren...“, wählt die Datei aus, von der er die Daten einlesen will und drückt „Ausführen“. Die Selektion wird gemäss Einstellungen belassen.		Der Tester erhält eine Auflistung der importierten Daten, die er bearbeiten kann.
T03	Der Tester wählt zwei Zeilen aus und drückt „Selektierte Standorte entfernen“.		Die zwei selektierten Zeilen verschwinden aus der Auflistung und es erscheint eine Rückmeldung, dass diese entfernt wurden.
T04	Der Tester wählt zwei Zeilen aus, bearbeitet diese so, dass sie keine Fehler mehr haben und drückt „Selektierte Standorte importieren“.		Die zwei selektierten Zeilen verschwinden aus der Auflistung und es erscheint eine Rückmeldung, dass zwei Adressen importiert wurden.
T05	Der Tester navigiert zu den Stammdaten / Flugplätze.		Die zwei importierten Standorte erscheinen neu in der Auflistung der Flugplätze / Standorte.

31 Ausführungen

Die Ausführungen werden nach Ausführungsdatum und fortlaufender Nummer strukturiert.

31.1 Ausführung vom 08.06.2012

Für Ausführung der Tests wurden die drei oben spezifizierten Geräte benutzt.

Die Tests 1-6 wurden sowohl auf Laptop 1, wie auch auf Laptop 2 ausgeführt. Die Tests 7-9 wurden auf Laptop 3 ausgeführt. Die Tests 10-11 wurden auf Laptop 1 ausgeführt.

Test-Nr.	Test-schritt	Fehler/Unschönheiten	Mögliche Verbesserungen
1.1	T04	Es wird zuerst nur nach Clubname validiert. Erst wenn der Clubname ausgefüllt wird und wieder gespeichert wird, erscheint die Meldung, dass das Land ausgefüllt sein muss.	Sofort Meldung über alle Errors.
1.1	T05	Fehlerinfo in der rechten unteren Ecke verschwindet nicht beim Tippen, sondern erst beim Speichern	Fehlermeldung sollte bereits beim Tippen verschwinden.
2.1	T07	Es erscheint die Meldung, dass etwas geändert wurde, obwohl dies nicht der Fall ist.	Meldung soll nicht mehr erscheinen.
2.2	T11	Der Benutzer sollte aus der Tabelle verschwinden. Dies macht er erst beim nächsten einloggen oder wenn „Gelöschte Einträge anzeigen“ einmal ausgewählt und wieder deaktiviert wird.	Gelöschte User verschwinden sofort aus Tabelle
5.1	T04	Es erscheinen nicht alle Fehlermeldungen auf einmal sondern erst nach und nach.	Es sollten alle Fehler auf einmal erscheinen.
5.1	T06	Der Flugplatz verschwindet erst, nachdem neu eingeloggt wurde.	Gelöschte Einträge sollten sofort verschwinden.
6.1	T03	Bei Laptop 1 erschien keine Meldung, bei Laptop 2 erschien nur die Rückmeldung für das Flugzeug	Validierung verbessern.

XV

Usability Test

Datum: 15. Juni 2012

32 Ablauf des Usability-Tests

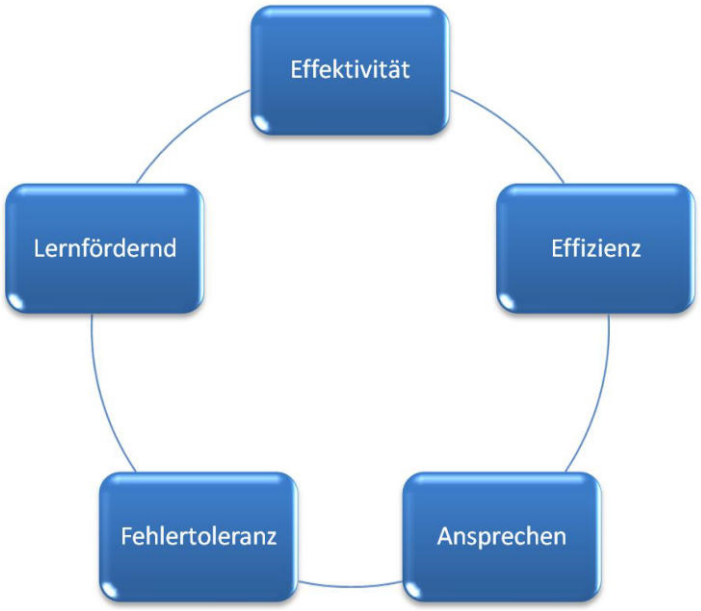
Der Usability-Test besteht aus 2 Phasen.

Als erstes wird die Test-Person gebeten, verschiedene Szenarien durchzuspielen und dabei laut zu denken. Der Entwickler beobachtet die Test-Person dabei genau und hilft nur mit kurzen Tipps, falls diese nicht mehr weiter weiss.

Danach wird die Test-Personen gebeten, ein Bewertungsformular auszufüllen.

32.1 Bewertungsformular

Das Bewertungsformular wurden gemäss Usability-Kriterien nach ISO 9241-11 und Quesenberry aufgebaut [UI110].



33 Usability Test

33.1 Usability Test mit Patrick Schuler

Datum des Tests: 16.05.2012, Dauer: 1 Stunde

33.1.1 Anmerkungen zur Test-Person

Patrick Schuler ist der Auftraggeber und wird später Benutzer dieser Applikation. Mit seiner Ausbildung im Bereich Informatik besitzt er sehr gute Computerkenntnisse. Als aktives Mitglied in einem Segelflugverein hat er sehr gute Kenntnisse über das Erfassen von Flügen und kennt sich mit den Fachbegriffen des Segelfliegens aus.

33.1.2 Anmerkungen zum Test

Der Usability-Test wurde mit einem Prototypen durchgeführt. Es sind noch nicht sämtliche Funktionalitäten vorhanden und das Programm enthält noch Fehler.

33.1.3 Szenarien

33.1.3.1 Szenario 1: Flug erfassen

Die Aufgabe war zu simulieren, wie Patrick als Flugdienstleiter einige Flüge erfasst, startet und landet.

Beobachtungen:

- Patrick fragte sich bei einem bestehenden Flug, der seit über 2 Tagen fliegt, was dies für eine Flugdauer sei. (Es muss besser angegeben werden, was Tage/Stunden usw. sind)
- Patrick frage sich, wie er nun einen Flug nachtragen kann, der gestern durchgeführt wurde.

- Patrick wünschte sich, da bei Segelfliegern die Immatrikulationsnummer meistens mit HB beginnt, dass HB nicht mehr eingegeben werden muss in der AutoCompleteBox.
- Bei der Auswahl von Flugart und Flugzeug störte sich Patrick daran, dass auch diejenigen Flugarten und Flugzeuge aufgeführt wurden, die nicht zu einem Segelflug bzw. Schleppflug passen.
- Patrick wünschte sich, dass er den Segelflug speichern kann, ohne dass der Schleppflug ausgefüllt ist.
- Die Schrift wäre besser etwas grösser und Wichtiges wie Immatrikulation, Pilot und Start/Landezeit sollte noch grösser sein.
- Beim Read-Only Flug ist die Trennung zwischen Flug und Schleppflug nicht eindeutig
- Pro Tag fliegt meistens derselbe Schleppflieger mit demselben Schlepper. Aus diesem Grund soll der Schlepppilot und Schleppflieger temporär gespeichert werden. Zusätzlich soll eine Map mit allen an diesem Tag ausgewählten Schlepppiloten und Schleppfliegern geführt werden. Wenn ein Schlepppilot oder ein Schleppflieger ausgwählt wird, wird das andere vorgeschlagen.

33.1.3.2 Szenario 2: Person erfassen

Patricks Aufgabe war es, eine anwesende Person im System zu erfassen.

Beobachtungen:

- Beim Einloggen wünschte sich Patrick, dass der Login-Name nicht case-sensitive sein soll, das Passwort aber schon.
- Beim Login wollte Patrick direkt nach der Eingabe von Benutzername und Passwort Enter drücken.
- Patrick suchte die Möglichkeit sich auszuloggen.
- Unter Kommunikation hat es eine Checkbox „Flugreport an Firmen-Email“. Hier fragte sich Patrick, was dies sei. Dies muss besser angeschrieben werden und ein Dropdown wäre vorteilhafter.
- Funktionsweise von „Personkategorie“ hinzufügen war nicht klar.

33.1.3.3 Szenario 3: Flugzeug erfassen

Patricks Aufgabe war es, ein Flugzeug zu erfassen.

Beobachtungen:

- Es war nicht klar, warum ein Flugzeug einen Verein hat. -> Verein und Person sollen als Besitzer gekennzeichnet werden.

33.1.3.4 Szenario 5: Verein erfassen

Patricks Aufgabe war es, einen Verein zu erfassen.

Beobachtungen:

- Basis sollte in Basisflugplatz umbenannt werden.

33.1.4 Bewertungsformular

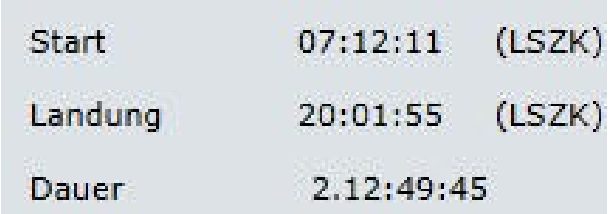
Abkürzungen			
5	Trifft in hohem Masse zu	**2**	Trifft in geringem Masse zu
4	Trifft grösstenteils zu	**1**	Trifft nicht zu
3	Trifft mehr oder weniger zu	**kinb**	Kann ich nicht beurteilen

Bewertungsformular	5	4	3	2	1	kinb
Effektiv						
Ich kann im FLS Stammdaten erfassen.		X				
Ich kann Flüge erfassen, starten und landen.		X				
Effizient						
Ich finde mich schnell in der Applikation zurecht.		X				
Ich kann schnell kleine Änderungen in den Stammdaten ausführen.		X				
Ich kann Flüge effizient erfassen.			X			
Ansprechend						
Das User Interface ist ansprechend / benutzerfreundlich.			X			
Das User Interface ist übersichtlich.			X			
Das User Interface ist farblich stimmig.		X				
Fehlertolerant						
Ich werde durch Applikationsfehler nicht beeinträchtigt.				X		
Bei falschen Eingaben stürzt die Applikation nicht ab.				X		
Die Applikation ist grundsätzlich fehlertolerant.			X			
Lernfördernd						
Die Applikation ist schnell lernbar.		X				
Die Bedienung ist einfach.			X			

34 Redesigns

34.1 Lesbarkeit der Flugdauer

34.1.1 Vorher



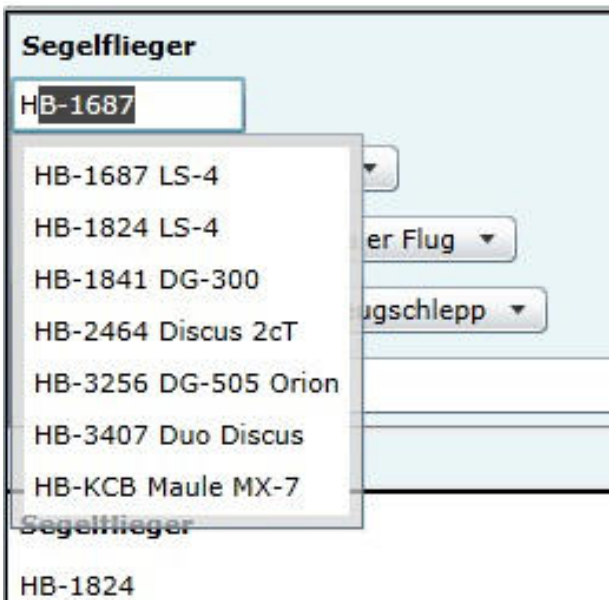
Der Benutzer konnte die Flugdauer nicht gut lesen, da nicht klar war, welches die Tage, Stunden, Minuten und Sekunden waren.

34.1.2 Nachher

Sekunden werden nicht mehr dargestellt. Tage, Stunden und Minuten werden leserlich angeschrieben.

34.2 Flexible AutocompleteBox

34.2.1 Vorher



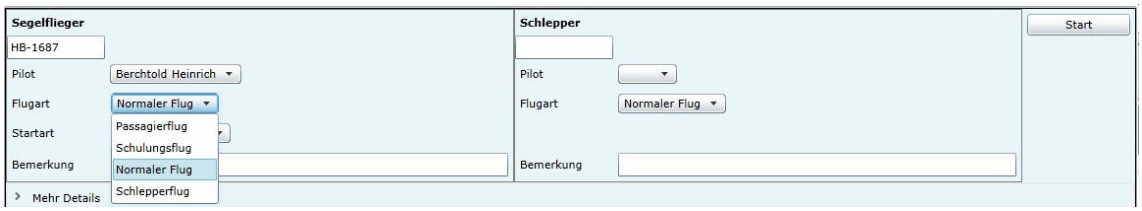
Die AutocompleteBox zum Auswählen eines Flugzeuges fand nur Resultate, die mit der Eingabe begannen.

34.2.2 Nachher

Die AutocompleteBox findet alle Flugzeuge, deren Immatrikulationsnummer die Eingabe enthält.

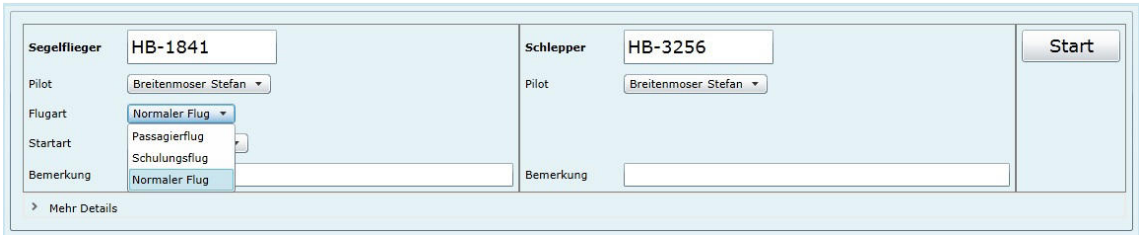
34.3 Auswahl der Flugart

34.3.1 Vorher



Sowohl beim Segelflieger wie auch beim Schleppflieger konnte eine Flugart ausgewählt werden. Dabei wurden bei beiden Auswahlmöglichkeiten alle Flugarten (Flugarten für Segelflieger und Flugarten für Schleppflieger) des Vereins aufgeführt.

34.3.2 Nachher



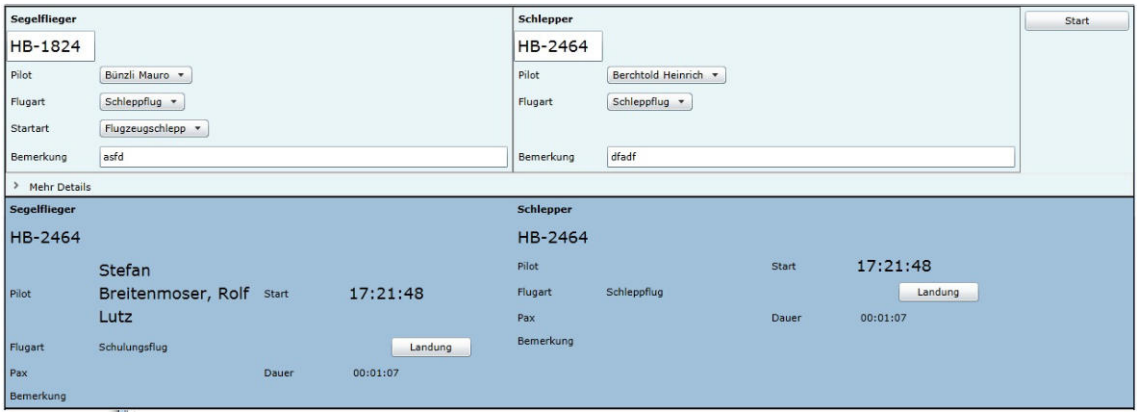
Da für den Schleppflieger die Flugart nicht relevant ist, wurde sie weggelassen. Es gibt nur noch Flugarten für Segelflieger.

34.4 Wichtiges Hervorheben

34.4.1 Vorher

Bei der Flugerfassung waren wichtige Informationen wie Flugzeug und Pilot nicht sofort ersichtlich.

34.4.2 Nachher



Der Flieger, die Piloten und die Start/Landezeit werden mittels Schriftvergrösserung hervorgehoben.

34.5 Trennung Segelflug / Schleppflug

34.5.1 Vorher

Segelflieger					Schlepper				
HB-3256					HB-KCB				
Pilot	Heinrich Berchtold, Anina Gysi	Start	15:13:16	(LSZK)	Pilot	Bruno Colombo	Start	15:53:16	(LSZK)
Flugart	Schulungsflug	Landung	15:44:28	(LSZK)	Flugart	Normaler Flug	Landung	15:43:40	(LSZK)
Pax		Dauer	00:31:13		Pax		Dauer	-00:09:36	
Bemerkung	Seilriss				Bemerkung				

Die Trennung zwischen Segelflug und Schleppflug ist nicht klar genug.

34.5.2 Nachher

Segelflieger	HB-1841			Schlepper	HB-3407		
Pilot	Heinz Bärfuss	Start	17:12:11	Pilot		Start	17:12:11
Flugart	Schleppflug	Landung	17:12:15	Flugart	Schleppflug	Landung	17:12:13
Pax		Dauer	00:00:04	Pax		Dauer	00:00:02
Bemerkung				Bemerkung			

Mittels Abtrennung mit einem Rahmen ist die Trennung klar ersichtlich.

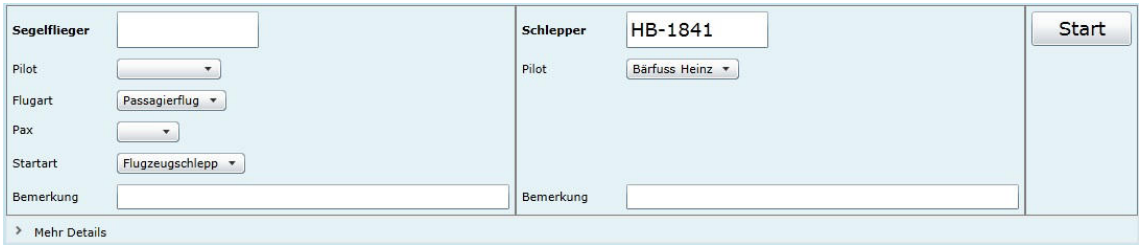
34.6 Neuer Flug vorausfüllen

34.6.1 Vorher

Segelflieger				Schlepper				Start
Pilot				Pilot				
Flugart	Normaler Flug			Flugart	Normaler Flug			
Startart	Flugzeugschlepp							
Bemerkung				Bemerkung				
Mehr Details								
Segelflieger				**Schlepper**				Start
HB-1687				HB-1824				
Pilot	Heinrich Berchtold			Pilot	Laurent Aebi			
Flugart	Normaler Flug			Flugart	Schleppflug			
Pax		Dauer	00:00:00	Pax		Dauer	00:00:00	
Bemerkung				Bemerkung				
Segelflieger				**Schlepper**				Start
HB-1824				HB-1824				
Pilot				Pilot	Laurent Aebi			
Flugart	Normaler Flug			Flugart	Schleppflug			
Pax		Dauer	00:00:00	Pax		Dauer	00:00:00	

Beim Neuerstellen eines Fluges, müssen sämtliche Felder neu ausgefüllt werden, obwohl sich der Flugtyp, Starttyp, Schlepppilot und Schleppflieger am selben Tag selten ändert.

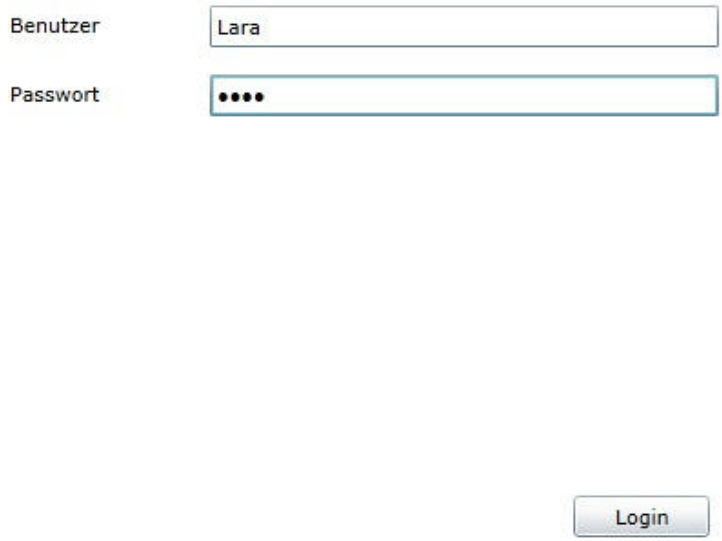
34.6.2 Nachher



Flugtyp, Starttyp, Schlepppilot und Schleppflieger werden zwischengespeichert und die Werte sind beim Neuanlegen eines Fluges bereits ausgefüllt.

34.7 Login case sensitive

34.7.1 Vorher



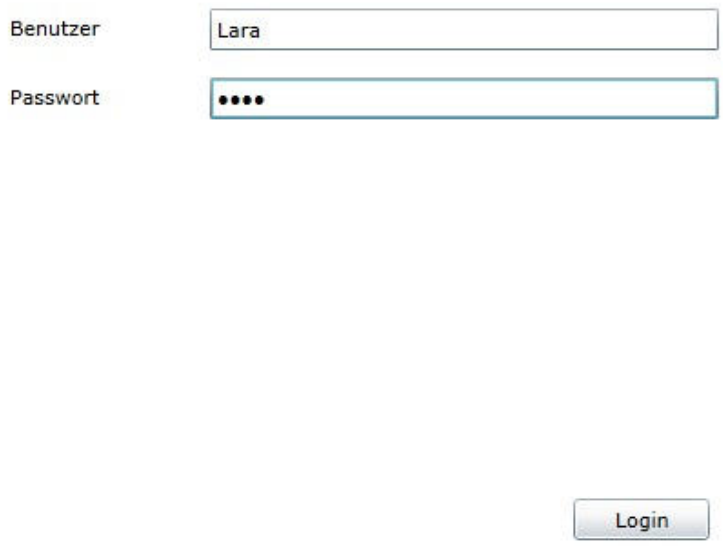
Der Benutzer möchte seinen Benutzername nach Belieben mit Gross- oder Kleinbuchstaben eingeben können. Aus Sicherheitsgründen muss bei der Eingabe des Passworts jedoch auf die Gross- und Kleinschreibung geachtet werden.

34.7.2 Nachher

Der Benutzername ist nicht mehr case-sensitive.

34.8 Login mit Enter

34.8.1 Vorher



Der Benutzer erwartet, dass er sich nach Eingabe von Benutzername und Passwort mit Enter einloggen kann, es passiert aber nichts.

34.8.2 Nachher

Durch drücken von Enter wird der Login-Vorgang ausgelöst.

34.9 Logout-Möglichkeit

34.9.1 Vorher

Der Benutzer hatte keine Möglichkeit sich auszuloggen.

34.9.2 Nachher

Das Menu wurde um einen Eintrag „Abmelden“ ergänzt.

34.10 E-Mail Adresse für Reportversand auswählen

34.10.1 Vorher

Die Funktionalität der Checkbox, die mit „Flugreport an Firmen-Email“ angeschrieben war, war nicht auf den ersten Blick ersichtlich.

34.10.2 Nachher

Die Checkbox wurde durch eine ComboBox ersetzt und mit „Report senden an:“ angeschrieben.

34.11 Besitzer des Flugzeuges beschriften

34.11.1 Vorher

Bei der Flugzeugbearbeitung kann für ein Flugzeug der Besitzer erfasst werden. Dies war mit Club und Person angeschrieben und somit nicht klar.

34.11.2 Nachher

Die Besitzer-Felder sind verständlich angeschrieben.

34.12 Basisflugplatz beschriften

34.12.1 Vorher

Der Basis-Flugplatz war bisher nur mit Basis beschriftet.

34.12.2 Nachher

Die Beschriftung wurde auf Basis-Flugplatz geändert.

XVI

Sitzungsprotokolle

Datum: 15. Juni 2012

35 Besprechungen mit dem internen Betreuer

35.1 Sitzungsprotokoll Woche 1 - 20.02.2012

35.1.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler
- Hansjörg Huser

35.1.2 Traktanden

- Vorstellung
- Einführung
- Ausblick

35.1.3 Beschlüsse und Diskussionen

- Die Sitzungen mit dem Betreuer Hansjörg Huser finden wöchentlich am Mittwoch von 15 bis 16 Uhr statt.
- Es wird ein Protokoll über die Beschlüsse in diesen Sitzungen geführt.
- Es wird sich an den Code-Styleguide auf dem Wiki gehalten.
- Bis Dienstag Abend vor der Sitzung werden die vorhandenen Unterlagen geschickt.
- LaTeX kann für die Dokumentation benutzt werden.
- Voraussichtlich wird RUP verwendet.

35.1.4 Nächster Termin

Datum:29.02.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.2 Sitzungsprotokoll Woche 2 - 29.02.2012

35.2.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.2.2 Traktanden

- Rückblick
- Aktuelles
- Ziele für nächste Woche

35.2.3 Beschlüsse und Diskussionen

- Projektauftrag genehmigt.
- Protokolle aus Sitzungen mit Patrick Schuler werden jeweils auch an Herr Huser gesendet.
- Es werden keine Dokumentinformationen (Änderungsgeschichte & Überprüfungdaten) in den Dokumenten geführt.
- Lässt sich das generierte Pdf stärker komprimieren?
- Authentifizierung lässt sich mit ASP.Net (Membership-Provider) verwirklichen
- Für Plug-In Architektur gibt es das MEF-Framework mit Dependency-Injection
- Für Prototypen muss keine Präsentation gemacht werden. Dieser soll einfach vorgeführt werden.
- Experte für Bachelor-Arbeit ist Stefan Zettel, Ascentiv AG.

35.2.4 Auf nächste Woche

- Domain-Modell erstellen und ausgedruckt mitbringen.

35.2.5 Nächster Termin

Datum:07.03.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.3 Sitzungsprotokoll Woche 3 - 07.03.2012

35.3.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.3.2 Traktanden

- Rückblick
- Aktuelles
- Ziele für nächste Woche

35.3.3 Beschlüsse und Diskussionen

- In Dokumenten können die Dokumentinformationen gelöscht werden.
- Die wöchentliche Abgabe soll auf die sich geänderten Dokumente beschränkt werden.
- Anforderungsspezifikation ist genehmigt.
- Domain-Model ist genehmigt.
- Users (im Domain-Model nicht aufgeführt) gehören nicht ins Domain-Model und könnten, falls nötig, in eigenem Modell für Berechtigung dargestellt werden.
- Für die folgenden Sitzungen wird, wenn frei, der Raum 6.110 reserviert.

- Auf nächste Sitzung wird die Architektur überdacht: Wie weit wird dem RIA entsprochen oder eigene Objekte anstatt Datenbank-Objekte an Client weitergegeben?

35.3.4 Auf nächste Woche

- Analyse über Architektur mitbringen (System Architecture Document).

35.3.5 Nächster Termin

Datum: 14.03.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.111

35.4 Sitzungsprotokoll Woche 4 - 14.03.2012

35.4.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.4.2 Traktanden

- Rückblick
- Aktuelles
- Ziele für nächste Woche

35.4.3 Beschlüsse und Diskussionen

- Der erste Prototyp wird ohne Interfaces umgesetzt. Falls die Interfaces von Patrick Schuler weiterhin erwünscht sind, wird später refactored.

35.4.4 Nächster Termin

Datum:21.03.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.5 Sitzungsprotokoll Woche 5 - 21.03.2012

35.5.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.5.2 Traktanden

- Präsentation des 1. Prototypen.

35.5.3 Beschlüsse und Diskussionen

- Der erste Prototyp ist genehmigt.

35.5.4 Nächster Termin

Datum:29.03.2012

Zeit: 14:00

Ort: HSR Rapperswil

Raum: 6.010

35.6 Sitzungsprotokoll Woche 6 - 29.03.2012

35.6.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.6.2 Traktanden

- Rückblick: Besprechung mit Patrick Schuler und Christian Moser

35.6.3 Beschlüsse und Diskussionen

- Themen-Austausch: Testing WCF RIA, Prism (Modularisierung, Dependency Injection), Architektur Client

35.6.4 Auf nächste Woche

- Aufbau des Clients erarbeiten (Service-Klassen, Zwischenspeicherung der Daten)

35.6.5 Nächster Termin

Datum:04.04.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.112

35.7 Sitzungsprotokoll Woche 7 - 04.04.2012

35.7.1 Teilnehmer

- Lara Mühlemann

- Marion Walser
- Marion Frei
- Hansjörg Huser

35.7.2 Traktanden

- Rückblick

35.7.3 Beschlüsse und Diskussionen

- Diskussion über Zwischenspeicherung von Daten in ClientModel.
- Daten werden sofort beim Speichern an den Server geschickt.

35.7.4 Nächster Termin

Datum:11.04.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.8 Sitzungsprotokoll Woche 8 - 11.04.2012

35.8.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.8.2 Traktanden

- Elaboration 2 abgeschlossen.
- Prototyp Version 2 vorgeführt.
- Ausblick: bis Ende Construction 1 sollen die Startliste sowie die Stammdatenverwaltung komplett beendet sein.

35.8.3 Beschlüsse und Diskussionen

- Einige Fragen diskutiert.

35.8.4 Nächster Termin

Datum: 18.04.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.9 Sitzungsprotokoll Woche 9 - 18.04.2012

35.9.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.9.2 Traktanden

- Arbeitsstand

35.9.3 Beschlüsse und Diskussionen

- Themen-Austausch: Enum, Event Bubbling und Command Pattern in XAML, Stand Prism
- Name der Bachelorarbeit ist Flight Logging System (Aktualisierung Ausschreibung)
- Herr Huser hat Anfrage von Patrick erhalten wegen Infragistics-Lizenz. Er wird abklären, ob die Lizenz für Silverlight Controls von der HSR erworben wird.

35.9.4 Nächster Termin

Datum:25.04.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.10 Sitzungsprotokoll Woche 10 - 26.04.2012

35.10.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.10.2 Traktanden

- Arbeitsstand

35.10.3 Beschlüsse und Diskussionen

- Die Popups werden momentan nicht getestet, da sie denselben Code benutzen wie bereits getestete UI's. Eine mögliche Lösung dazu wäre MessagePassing.

35.10.4 Nächster Termin

Datum:02.05.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.11 Sitzungsprotokoll Woche 11 - 02.05.2012

35.11.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.11.2 Traktanden

- Arbeitsstand

35.11.3 Beschlüsse und Diskussionen

- Bis Ende Construction 1 wird die Stammdatenverwaltung fertiggestellt, die Startliste bis auf kleine Details auch. Der Implementationsbeginn der Exportfunktionalitäten wird auf Construction 2 verschoben.
- Am 10.5.2012, 15:30 Uhr, findet eine Zwischenpräsentation vor dem Experten, Herr Zettel, statt.

35.11.4 Nächster Termin

Datum:09.05.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.12 Sitzungsprotokoll Woche 12 - 09.05.2012

35.12.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

- Hansjörg Huser

35.12.2 Traktanden

- Arbeitsstand

35.12.3 Beschlüsse und Diskussionen

- Am 15. Juni wird das Poster aufgehängt und präsentiert.
- An der Zwischenpräsentation morgen wird versucht, gleich den Termin für die Schlusspräsentation festzulegen.

35.12.4 Nachtrag

- An der Zwischenpräsentation wurde der 27./28. Juni 2012 als noch zu bestätigenden Zeitraum für die Schlusspräsentation festgelegt.

35.12.5 Nächster Termin

Datum: 16.05.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.13 Sitzungsprotokoll Woche 13 - 16.05.2012

35.13.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.13.2 Traktanden

- Arbeitsstand

35.13.3 Beschlüsse und Diskussionen

- Heute Abend wird mit Patrick besprochen, ob Reporting Services von Microsoft für die Reports verwendet wird.
- Falls die Entscheidung auf den ReportingService fällt, kann das Team Herrn Huser ein Mail schreiben, worauf er ihnen ein Tutorial schickt.
- Heute Abend wird mit Patrick besprochen, ob für Import/Export eine Plugin-Architekur verwendet werden soll oder ob dies gestrichen werden kann.

35.13.4 Nächster Termin

Datum:23.05.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.14 Sitzungsprotokoll Woche 14 - 23.05.2012

35.14.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.14.2 Traktanden

- Arbeitsstand

35.14.3 Beschlüsse und Diskussionen

- Herr Huser erhält bei Meilenstein „Abschluss Systemtests“ das Systemtest-Dokument.

35.14.4 Nächster Termin

Datum:30.05.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.15 Sitzungsprotokoll Woche 15 - 31.05.2012

35.15.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.15.2 Traktanden

- Arbeitsstand

35.15.3 Beschlüsse und Diskussionen

- Es ist das Abstrakt und das Poster bis nächsten Mittwoch zur Überprüfung zu erstellen und möglichst bis Dienstagabend an Herrn Huser zu schicken.

35.15.4 Nächster Termin

Datum:06.06.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.16 Sitzungsprotokoll Woche 16 - 06.06.2012

35.16.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.16.2 Traktanden

- Poster
- Abstrakt
- Fragen beantworten

35.16.3 Beschlüsse und Diskussionen

- Beim Poster kann eventuell SSRS ausgeschrieben werden. Um den nötigen Platz dazu zu erhalten kann die Schrift verkleinert werden.
- Das Abstrakt ist bis auf einen fehlenden Abstand und ein Wort das geändert werden muss gut so.
- Am Schluss abgegeben werden müssen:
  - 1 Ausdruck im Ordner
  - 3 CD's an Herr Huser, 1 CD in die Box
  - Begründung für fehlende Automation kann im Projektplan beschrieben werden.
  - Die Kontaktdaten von Herrn Huser müssen nicht aus der Dokumentation entfernt werden.

35.16.4 Nächster Termin

Datum: 13.06.2012

Zeit: 15:00

Ort: HSR Rapperswil

Raum: 6.010

35.17 Sitzungsprotokoll Woche 17 - 13.06.2012

35.17.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Hansjörg Huser

35.17.2 Traktanden

- Letze Fragen vor Abgabe

35.17.3 Beschlüsse und Diskussionen

- Herr Huser hat den technischen Bericht durchgesehen.
  - Einleitung ist etwas knapp geraten.
  - Übersicht geben (FLS ist eine Web-Applikation ...)
  - Aufgabenstellung sollte detaillierter formuliert werden.
  - Prozesse und Threads: Hervorheben dass Netzwerk-Kommunikation nicht von uns in eigenem Thread gestartet wird.
- Neue Erklärung eigenständige Arbeit mit zusätzlichem Punkt nehmen.
- Herr Huser hat Infragistic-Lizenz erworben. Es ist jedoch nicht sicher, dass diese noch bis Freitag ankommt.
- Der Ausstellungsplatz während des Forums wird uns zugewiesen und angeschrieben.
- Bachelorarbeit bei Herr Huser im INS abgeben.
- An die Schlusspräsentation kommen neben Herr Huser auch der Gegenleser Herr Bütler und der Experte Herr Zettel.
- Die Präsentation sollte ca. 20 (-30) min gehen und kann Redundanzen zur Zwischenpräsentation aufweisen.

36 Besprechungen mit dem externen Betreuer

36.1 Sitzungsprotokoll Woche 2 - 28.02.2012

36.1.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler

36.1.2 Beschlüsse und Diskussionen

- Es wird nur eine Sprache (Deutsch) implementiert.
- Das Produkt muss nicht komplett mit der Tastatur bedienbar sein.
- Patrick installiert eine Projektautomation auf dem Entwicklungsserver.
- Patrick organisiert eine Lizenz für Infragistic. Bis dahin wird eine Trial-Version benutzt.
- Es muss nicht, wie in den Coderichtlinien erwähnt, einen Logger benutzt werden.
- Bei der Wahl eines Code-Dokumentationswerkzeuges muss darauf geachtet werden, dass mit drei '/' dokumentiert wird.
- Patrick organisiert Beispieldaten für die Datenbank.
- Für die Migration von der Access-Datenbank zu der SQL Server 2008 Datenbank ist Patrick verantwortlich.
- Patrick erstellt die ersten Paper Prototypen, welche nächste Woche mit ihm besprochen werden.

36.1.3 Nächster Termin

Datum:07.03.2012

Zeit: 17:00

Ort: HSR Rapperswil

Raum: Cafeteria

36.2 Sitzungsprotokoll Woche 3 - 07.03.2012

36.2.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler

36.2.2 Beschlüsse und Diskussionen

- Domain-Modell: Invoice Mapping wird aus dem Domain-Modell genommen und Implementation dessen zurückgestellt.
- Implementation Login: Ein Ändern des Passwort soll vorerst zu keiner Zeit erzwungen werden. Das Passwort soll auf dem Server gehasht und gehasht in die Datenbank abgelegt werden.
- Implementation Kommunikationsschnittstelle: SSL-Verschlüsselung wird nicht verlangt.
- Installation der Projektautomation auf Entwicklungsserver ist noch offen.
- Beispieldaten können mit den Scripts https://svn.glider-fls.ch/svn/FLS/trunk /database/scripts/insert_masterdata.sql und https://svn.glider-fls.ch/svn/FLS/trunk/ database/scripts/insert_data.sql in die Datenbank eingefüllt werden.
- Beispiele von Bewegungsdaten werden noch erstellt.
- Aufbereiten der Paper-Prototypen ist offen.
- Es sollte ein Logo für die Applikation erstellt werden. Patrick überlegt sich etwas oder erstellt eines.
- Änderungen an der Datenbank von Patrick werden eingestellt. Sollte trotzdem eine Änderung von seiner Seite notwendig sein, wird diese vorher im Team besprochen.
- Anforderungsspezifikation und UseCases wurden teilweise durchgeschaut, nächste Woche werden noch weitere UseCases überprüft.
- Regions von Prism wird nicht benötigt. Für MVVM und Dependency Injection sollte entweder Prism (Unity Dlls) oder Prism Light verwendet werden.
- Der Server-Implementation wird mit Vorteil in drei Dlls aufgeteilt (nicht zwingend), in Service-Interface, Service-Implementation, Modell. Somit entsteht eine saubere Architektur und es kann auch sauber getestet werden.

36.2.3 Nächster Termin

Datum: 14.03.2012

Zeit: 17:00

Ort: HSR Rapperswil

Raum: Cafeteria

36.3 Sitzungsprotokoll Woche 4 - 14.03.2012

36.3.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler

36.3.2 Traktanden

- Prism: Zeigen der für dieses Projekt nützlichen Funktionalität.
- Besprechen der restlichen Use Cases.
- Fragen?

36.3.3 Beschlüsse und Diskussionen

- Die Kommunikation zwischen Client und Server erfolg nicht über Interfaces. Interfaces werden für die Kommunikation zwischen Modules und der BasisApplikation gebraucht.
- Wir müssen uns nicht darüber sorgen, dass die Internetverbindung auf einmal weg ist. Es kann davon ausgegangen werden, dass eine permanente Verbindung besteht. Allerdings soll darauf geachtet werden dass nicht zu viele Daten übertragen werden müssen.
- Min 1024*728 Auflösung, eher mehr.
- Infragistic-Lizenz: Patrick klärt mit Herr Huser ab, ob Lizenz über die Schule gekauft wird.
- Paper-Prototype: Ein Beispiel für Startliste wurde von Patrick erstellt. Momentan werden keine weiteren Paper-Prototypen von Patrick erstellt.

36.3.4 Offene Punkte

- Installation der Projektautomation auf Entwicklungsserver ist noch offen.
- Beispiele von Bewegungsdaten werden noch erstellt.
- Es sollte ein Logo für die Applikation erstellt werden. Patrick überlegt sich etwas oder erstellt eines.

36.3.5 Ausblick

Nächste Woche findet ein Code-Review und Erklärungen mit einem Kollegen von Patrick statt.

36.3.6 Nächster Termin

Datum:22.03.2012

Zeit: offen

Ort: HSR Rapperswil

Raum: Cafeteria

36.4 Sitzungsprotokoll Woche 5 - 22.03.2012

36.4.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler

36.4.2 Traktanden

- Prototyp Version 1
- Paper Prototypen
- Berechtigungstabelle
- Fragen

36.4.3 Beschlüsse und Diskussionen

- Paper Prototypen
  - Menü für Import / Export und Drucken nicht vergessen.
  - Filteroptionen stark vereinfachen.
  - Detailansichten in ausklappbare Gruppen unterteilen.
  - Dataform soll sofort bearbeitbar sein (kein Stift, welcher zuerst angewählt werden muss).
- Datenbank
  - Mandant soll eine Kontaktperson haben.
  - Aircraft: hasEngine durch isGliderWithMotor ersetzen.
  - Errors in Systemlog der Datenbank speichern.
  - Tabelle Systemdata ist mandantenunabhängig, mandantenabhängige Daten werden beim Mandant gespeichert (z.B. letzes Mal synchronisiert).
  - Rollen sind statisch.
  - Flugtyp kann null sein, was einem Fremdflug entspricht.
  - Ein Passagier ist nicht in der Membertabelle sondern wird nur mit Vor- und Nachname erfasst.
  - Passagiere haben keine PersonCategory, sie werden nur über Flugtyp ermittelt.

36.4.4 Offene Punkte

- Filter Personen
- Installation der Projektautomation auf Entwicklungsserver ist noch offen.
- Beispiele von Bewegungsdaten werden noch erstellt.
- Es sollte ein Logo für die Applikation erstellt werden. Patrick überlegt sich etwas oder erstellt eines.

36.4.5 Ausblick

Nächste Woche finden ein Code-Review und Erklärungen mit einem Kollegen von Patrick statt.

36.4.6 Nächster Termin

Datum:offen (nächste Woche)

Zeit: offen

Ort: HSR Rapperswil

Raum: Cafeteria

36.5 Sitzungsprotokoll Woche 6 - 27.03.2012

36.5.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler
- Christian Moser

36.5.2 Traktanden

- Dependency Injection in Prism
- Testing WCF RIA
- Menu Bar
- aktuelle kleinere Probleme

36.5.3 Beschlüsse und Diskussionen

- Architektur Client: Implementation von Services/Models möglich, um Daten zwischenzuspeichern und Zugriff von ViewModell zu kanalisieren.
- Christian hat Funktionsweise der DependencyInjection über Prism mittels DependencyContainer erklärt
- Modularisierung wird im Hauptteil der Anwendung nicht benötigt.
- Navigation mit Back/Forward-Button von Browser ist in dieser Applikation nicht nötig.

36.5.4 Offene Punkte

- Filter Personen
- Installation der Projektautomation auf Entwicklungsserver ist noch offen.
- Beispiele von Bewegungsdaten werden noch erstellt.
- Es sollte ein Logo für die Applikation erstellt werden. Patrick überlegt sich etwas oder erstellt eines.

36.5.5 Nächster Termin

Datum:offen

Zeit: offen

Ort: HSR Rapperswil

Raum: Cafeteria

36.6 Sitzungsprotokoll Woche 8 - 11.04.2012

36.6.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler per Skype

36.6.2 Traktanden

- Template-Verwendung noch sinnvoll und zeitgemäss?
- Mandantenfähigkeit: versch. Rating in versch. Clubs

36.6.3 Beschlüsse und Diskussionen

- Template mit #region für den Moment ganz weglassen oder Template soweit abändern, dass nur Header mit Klasseninfo etc. erstellt wird.
- Rating eines Pilots kann in jedem Verein, in dem er Mitglied ist, unterschiedlich sein. Es wird eine Zwischentabelle zwischen Club und Person erstellt. Diese enthällt MemberNr, GliderInstructorNr sowie die verschiedenen Ratings (isGliderPilot, isMotorPilot, etc). Die entsprechenden Felder in der Personen-Tabelle werden umbenannt, z.B. in hasMotorPilotLicense, hasGliderPilotLicense, ...
- Die GliderInstructorNr, die ein Fluglehrer hat, ist in jedem Verein, in dem er Mitglied ist, anders.

36.6.4 Offene Punkte

- Filter Personen

- Installation der Projektautomation auf Entwicklungsserver ist noch offen.
- Beispiele von Bewegungsdaten werden noch erstellt.
- Es sollte ein Logo für die Applikation erstellt werden. Patrick überlegt sich etwas oder erstellt eines.
- Anfrage Infragistics-Lizenz bei Herrn Huser (Menü von Infragistic in gebrauch).

36.6.5 Nächster Termin

Datum:offen

Zeit: offen

Ort: HSR Rapperswil

Raum: Cafeteria

36.7 Sitzungsprotokoll Woche 10 - 26.04.2012

36.7.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler

36.7.2 Traktanden

- Besprechen von Fragen

36.7.3 Beschlüsse und Diskussionen

- Patrick liefert den Aufbau des CSV für die Adresssynchronisation.
- Patrick liefert einen Flugbuchauszug oder den Aufbau des Reportes für die Piloten
- Patrick richtet einen SMTP Server ein.
- Es werden nur ein Report für das Flugbuch des Piloten, einen für die Rechnungen ans Sekretariat und einen für das BAZL erstellt. Weitere Reports sind optional.
- Der Export für das BAZL kann ein einfacher Report sein.
- Für die Reports wird der ReportingServer von Microsoft verwendet.
- In der Startliste wird wie im Paper-Prototyp ein Accordion benutzt.

- Einstellungen für den E-Mail-Versand werden in der DB gespeichert. Dafür wird die Tabelle Systemdaten verwendet. Zusätzlich werden dort Prozessdaten (z.B wann werden die Mails verschickt) gespeichert.
- Bei der Personenbearbeitung soll ausgewählt werden können, ob diese Person Emails will und es soll ausgewählt werden können, ob diese an die Privatadresse oder an die Geschäftsadresse geschickt werden.
- Bei der Flugerfassung soll ausgewählt/eingegeben werden können, welcher Pilot wieviel in % bezahlt. Dies könnte mit Slidebars gelöst werden. Zusätzlich muss ausgewählt werden können, ob der Fluglehrer ein Honorar will oder nicht.
- StartTyp wird bei der Flugerfassung nicht mehr in den Details, sondern im sofort sichtbaren Bereich angezeigt.
- Beim Ändern des Starttyps eines Fluges wird, falls dabei ein Schleppflug gelöscht wird eine Warnung ausgegeben.
- Bei der Flugerfassung kann bei einsitzigen Flugzeugen nur ein Pilot ausgewählt werden. Bei zweisitzigen Flugzeugen kann zusätzlich ein CoPilot ausgewählt werden. Bei Flugarten in denen ein Fluglehrer benötigt wird wird kein CoPilot angezeigt, aber es muss ein Fluglehrer ausgewählt werden.

36.7.4 Offene Punkte

- Patrick liefert den Aufbau des CSV für die Adresssynchronisation.
- Patrick liefert einen Flugbuchauszug oder den Aufbau des Reportes für die Piloten.
- Patrick richtet einen SMTP Server ein.
- Filter Personen
- Installation der Projektautomation auf Entwicklungsserver ist noch offen.
- Beispiele von Bewegungsdaten werden noch erstellt.
- Es sollte ein Logo für die Applikation erstellt werden. Patrick überlegt sich etwas oder erstellt eines.
- Infragistics-Lizenz

36.7.5 Nächster Termin

Datum:07.05.2012

Zeit: 16:30

Ort: Flugplatz Speck, Fehraltorf

36.8 Sitzungsprotokoll Woche 12 - 07.05.2012

36.8.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler

36.8.2 Traktanden

- Besprechen von Fragen
- Applikation testen

36.8.3 Beschlüsse und Diskussionen

- Schleppflugzeug soll nie zwei Personen haben, Spezialfälle können über Bemerkungen abgehandelt werden (Beispiel: Schleppausbildung).
- Es werden zwei Boolean in Tabelle Flugtyp benötigt: IsForTowFlights und IsForGlider.
- Im ClubView Felder Default-Startart (Bindung auf Id) und Default-Flugart (id) einrichten
- Anzahl Sitzplätze im Flugzeug soll nur auf 1 - 999 eingegrenzt werden, nicht auf 1 - 2.
- Patrick fragt Christian an, ob er nächste Woche Zeit für eine Besprechung hat.

36.8.4 Offene Punkte

- Patrick liefert den Aufbau des CSV für die Adresssynchronisation.
- Patrick richtet einen SMTP Server ein.
- Infragistics-Lizenz

36.8.5 Nächster Termin

Datum:offen

Zeit: offen

Ort: offen

Raum: offen

36.9 Sitzungsprotokoll Woche 13 - 16.05.2012

36.9.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler

36.9.2 Traktanden

- Besprechen von Fragen
- Usability Test

36.9.3 Beschlüsse und Diskussionen

- Usability Test wurde durchgeführt.
- Patrick liefert ein CUP-File aller Flugplätze der Schweiz.
- Die Imports/Exports müssen nicht als Plugins umgesetzt werden, aber eine saubere Abtrennung durch Interfaces ist gewünscht.
- Für die Adresssynchronisation wird der Export umgesetzt. Der Import wird umgesetzt, falls genug Zeit vorhanden ist.
- Beim Location-Import können Felder wie z.B. Latitude/Longitude ignoriert werden und nur der Name importiert werden.
- Alle Benutzer sehen nur die Flüge des eigenen Clubs. Auch kein Administrator sieht andere Flüge.
- Es wird ein zusätzlicher Flugstatus eingeführt: „gesichert“. Dieser wird vorerst aber nur in der Datenbank setzbar sein.
- Für die Reports werden Reporting Services von Microsoft verwendet.
- Vorerst wird auf lokalem SMTP-Server gearbeitet.

- Wichtiger als Report per Mail verschicken ist, dass der Report im UI angezeigt wird.
- Der wichtigste Report ist der Flugbuchreport des aktuellen Tages für die Piloten.

36.9.4 Offene Punkte

- Patrick liefert ein CUP-File aller Flugplätze der Schweiz
- Patrick liefert den Aufbau des CSV für die Adresssynchronisation.
- Patrick richtet einen SMTP Server ein.
- Infragistics-Lizenz

36.9.5 Nächster Termin

Datum:28.06.2012

Zeit: 18:00

Ort: HSR

Raum: Cafeteria

36.10 Sitzungsprotokoll Woche 14 - 24.05.2012

36.10.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei
- Patrick Schuler
- Christian Moser

36.10.2 Traktanden

- Besprechen von Fragen
- 1. Version Reports anschauen

36.10.3 Beschlüsse und Diskussionen

- BAZL-Report

  - Der Report für das BAZL wird jeweils 1x im Monat an den Motorflieger-Verein geliefert.
  - Die Schleppflüge werden für das BAZL nicht erfasst
  - Volte = Starten und gleich darauf wieder landen (ca. 2-3min in der Luft)
- Report für Buchhaltung
  - Unterscheidung Schleppdauer: erste 10min und über 10min
  - Pro Person zusätzlich noch Total von Anz. Landungen Schulungsflüge, Anz. Landungen normaler Flug, Stunden Schulung (solo, doppel, total), Anz. Stunden normaler Flug
- Beim Speichern-Dialog lässt sich Pfad nicht speichern oder mehrmals ohne neu auswählen benutzen aus Sicherheitsgründen.
- Read-only Template-Problem mit Landebutton klicken: Anstatt Grid mit 2 Templates eine List-Box mit 1 Template, welches sich ändern lässt. Christian Moser liefert eine einfache Beispiel-Solution dafür.
- Grids mit CollectionView direkt binden und nicht zusätzlich über ItemSelected.

36.10.4 Offene Punkte

- Patrick liefert ein CUP-File aller Flugplätze der Schweiz
- Patrick liefert den Aufbau des CSV für die Adresssynchronisation.
- Patrick richtet einen SMTP Server ein.
- Infragistics-Lizenz
- Patrick liefert Beispieldaten BAZL-Report
- Christian Moser liefert uns ein einfache Beispiel-Solution mit ListBox-Item und änderbarem Template

36.10.5 Nächster Termin

Datum:offen

Zeit: 18:00

Ort: HSR

Raum: Cafeteria

36.11 Sitzungsprotokoll Woche 15 - 30.05.2012

36.11.1 Teilnehmer

- Lara Mühlemann

- Marion Walser
- Marion Frei
- Patrick Schuler

36.11.2 Traktanden

- Besprechen von Fragen
- Reports überprüfen
- ToDo- und Bug-Liste durchschauen

36.11.3 Beschlüsse und Diskussionen

- Die Einstellung des Reporting-Server für den Echt-Betrieb werden von Patrick vorgenommen, so dass der Benutzer keine Administrator-Berechtigung auf seinem Browser benötigt, um Reports anzuschauen.

36.11.4 Offene Punkte

- Patrick liefert den Aufbau des CSV für die Adresssynchronisation.
- Patrick richtet einen SMTP Server ein.
- Infragistics-Lizenz
- Patrick liefert Beispieldaten BAZL-Report

36.11.5 Nächster Termin

Datum:offen

Zeit: 18:00

Ort: HSR

Raum: Cafeteria

36.12 Sitzungsprotokoll Woche 16 - 07.06.2012

36.12.1 Teilnehmer

- Lara Mühlemann

- Marion Walser
- Marion Frei
- Patrick Schuler

36.12.2 Traktanden

- Abschliessende Fragen besprechen.

36.12.3 Beschlüsse und Diskussionen

- Für Patrick müssen die Dokumente nicht gedruckt werden.
- Die SQL Scripts sind sauber aufzuteilen in Datenbankerstellung + Datenabfüllen etc.

37 Besprechungen im Team

37.1 Sitzungsprotokoll Woche 2 - 29.02.2012

37.1.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.1.2 Traktanden

- Aktuelles
- Ziele für den Rest der Woche

37.1.3 Beschlüsse und Diskussionen

- Lara Mühlemann erstellt zuerst das Domain-Modell und arbeitet danach an den fully-dressed Use Cases weiter.
- Marion Walser richtet ihre Arbeitsumgebung fertig ein und arbeitet sich in die neuen Technologien ein.
- Marion Frei erstellt / bearbeitet die Visual Studio Solution weiter.

37.1.4 Nächster Termin

Datum:07.03.2012

Zeit: 16:00

Ort: HSR Rapperswil

Raum: Cafeteria

37.2 Sitzungsprotokoll Woche 4 - 12.03.2012

37.2.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.2.2 Traktanden

- Aktuelles
- Ziele für die Woche

37.2.3 Beschlüsse und Diskussionen

- Einstellung im Visual Studio gemäss Code-Richtlinien: Lara.
- Anpassungen aus Sitzung mit Patrick am letzten Mittwoch: Lara, Montag.
- Beginnen mit SAD: Lara, Dienstag.
- Architektur-Analyse für Mittwoch: alle, Dienstag-Nachmittag 3-5 Uhr.
- MVVM-Modell für Prototyp und Benutzerverwaltung umsetzen: Marions, Montag.
- Marion Walser: Sequenzdiagramme beginnen zu erstellen (Benutzerverwaltung), Donnerstag beginnen.
- Donnerstag: alle zusammen Code Review für eine Stunde, 13.00 Uhr.
- Offizielle Team-Sitzung wird zukünftig jeweils montags um 10.00 Uhr stattfinden.

37.2.4 Nächster Termin

Datum: 19.03.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: Cafeteria

37.3 Sitzungsprotokoll Woche 5 - 19.03.2012

37.3.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.3.2 Traktanden

- Aktuelles
- Ziele für die Woche

37.3.3 Beschlüsse und Diskussionen

- Sequenzdiagramme und Operation Contracts, die letzte Woche geplant waren, wurden auf nächste Iteration verschoben, da die UseCases bis anhin genügend Informationen lieferten und die Entwicklung noch nicht so weit fortgeschritten war, dass Sequenzdiagramme nötig waren.
- Code-Review welches auf letzten Donnerstag geplant war, wurde auf heute, Montag verschoben.
- Iterationsplan erstellen: Marion W.
- Abgabe vorbereiten:
  - Dokumente durchlesen
  - Prototyp-Version wird jetzt von Marion W. erstellt.
- Prototypen: Marion F, Lara

37.3.4 Nächster Termin

Datum:26.03.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: Cafeteria

37.4 Sitzungsprotokoll Woche 6 - 26.03.2012

37.4.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.4.2 Traktanden

- Aktuelles
- Ziele der Woche

37.4.3 Beschlüsse und Diskussionen

- Für die Sitzungen am Donnerstag wird der Sitzungsraum 6.111 reserviert (Lara).
- Datenbank wird für die Mandantenfähigkeit angepassen (MarionW).
- Die Paper-Prototypen werden gemäss Feedback von Patrick angepasst (MarionF).
- Es wird weiter an der Stammdatenverwaltung und der Nativation zwischen den Views gearbeitet (Lara).
- Code-View des UserViews (Lara & Marion).

37.4.4 Nächster Termin

Datum:02.04.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: Cafeteria

37.5 Sitzungsprotokoll Woche 7 - 02.04.2012

37.5.1 Teilnehmer

- Lara Mühlemann
- Marion Walser

- Marion Frei

37.5.2 Traktanden

- Aktuelles
- Ziele der Woche

37.5.3 Beschlüsse und Diskussionen

- Marion F. erstellt ein Resource File für die Texte im UI.
- Marion F. erstellt ein Resource File für sonstige Magic Numbers.
- Stand Test: Ein Test läuft. Marion F. und Marion W. erweitern um weitere Tests.
- Lara beginnt das Systemtest-Dokument.
- Prototyp: Userverwaltung wird um die Personenverwaltung erweitert (Marion W.).
- Prototyp: Navigation soll zum Laufen gebracht werden(Lara).
- Buttons deaktivieren/aktivieren: Marion W.
- Optional: Mandantenfähigkeit Datenbank anpassen.
- Konzept Client Architektur (Services): Marion W.

37.5.4 Nächster Termin

Datum:11.04.2012

Zeit: 13:00

Ort: HSR Rapperswil

Raum: 1.258

37.6 Sitzungsprotokoll Woche 9 - 16.04.2012

37.6.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.6.2 Traktanden

- Aktuelles
- Ziele der Woche

37.6.3 Beschlüsse und Diskussionen

- Datenbankanpassung Person: MarionW
- PersonDataView: MarionF
- Tests: MarionF
- Flugerfassung: Lara
- MandantView: MarionW
- FlugzeugView: MarionW
- Separierung UserDataView: Marion F

37.6.4 Nächster Termin

Datum:23.04.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: 1.258

37.7 Sitzungsprotokoll Woche 10 - 23.04.2012

37.7.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.7.2 Traktanden

- Aktuelles
- Ziele der Woche

37.7.3 Beschlüsse und Diskussionen

- weiter an PersonDataView: MarionF
- Tests: Lara + MarionF
- weiter an Flugerfassung: Lara
- MandantView: erster Stand vorhanden. Wird später weiterbearbeitet.
- weiter an FlugzeugView: MarionW
- Separierung UserDataView: MarionF
- Validation: MarionW
- Refactoring MarionW + Lara am Freitag.
- Dokumente: gemäss Zeitplan

37.7.4 Nächster Termin

Datum:30.04.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: 1.258

37.8 Sitzungsprotokoll Woche 11 - 30.04.2012

37.8.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.8.2 Traktanden

- Aktuelles
- Ziele der Woche

37.8.3 Beschlüsse und Diskussionen

- Die versch. Exportfunktionen werden auf Beginn Construction2 verschoben.

- es wird an der Startliste weitergearbeitet, diese wird wahrscheinlich bis Ende Construction1 nicht fertig: Lara
- Validierung wird gemacht, die Validierungstexte kommen in die Services: MarionW
- Stammdatenverwaltung wird fertiggestellt (PersonCategory, Memberinfo): MarionW, MarionF
- UnitTests: MarionW, MarionF
- Ende Construction1: Dokumente gemäss Zeitplan: alle

37.8.4 Nächster Termin

Datum:07.05.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: 1.258

37.9 Sitzungsprotokoll Woche 12 - 07.05.2012

37.9.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.9.2 Traktanden

- Aktuelles
- Ziele der Woche

37.9.3 Beschlüsse und Diskussionen

- UnitTests: Lara
- Vorbereitung Präsentation Donnerstag: Dienstag, alle
- Iterationsplan Construction 2: Lara
- Import/Export Location beginnen: MarionW
- Flugerfassung weiterarbeiten: Lara (abhängig von heutigem Live-Einsatz)
- Plug-In-Architektur einarbeiten: MarionW

- Flugbuch-Report: MarionF
- Systemtests: MarionF
- Usability Tests: Alle, Montag Abend

37.9.4 Nächster Termin

Datum: 14.05.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: 1.258

37.10 Sitzungsprotokoll Woche 13 - 14.05.2012

37.10.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.10.2 Traktanden

- Aktuelles
- Ziele der Woche

37.10.3 Beschlüsse und Diskussionen

- Marion Frei arbeitet hauptsächlich an den Reports.
- Lara Mühlemann arbeitet hauptsächlich an der Authorisierung
- Marion Walser arbeitet hauptsächlich am Export.
- Alle beheben bekannte Fehler.

37.10.4 Nächster Termin

Datum:21.05.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: 1.258

37.11 Sitzungsprotokoll Woche 14 - 21.05.2012

37.11.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.11.2 Traktanden

- Aktuelles
- Ziele der Woche

37.11.3 Beschlüsse und Diskussionen

- Lara Mühlemann: Authentifizierung fertigstellen
- Marion Walser: Location export/import fertigstellen, export Adressdaten beginnen
- Marion Frei: Flugreport fertig, Summary für Rechnungsstellung, Bericht an BAZL
- MemberStatus, PersonCategories werden über den Club-Speichern-Button gespeichert.
- Es erscheint ein Popup, wenn die View gewechselt wird und ungespeicherte Änderungen vorhanden sind.

37.11.4 Nächster Termin

Datum:30.05.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: 1.258

37.12 Sitzungsprotokoll Woche 15 - 30.05.2012

37.12.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.12.2 Traktanden

- Aktuelles
- Ziele der Woche

37.12.3 Beschlüsse und Diskussionen

- Lara Mühlemann: Flugerfassung fertigstellen
- Lara Mühlemann: Doku: Operation, Sequenzdiagramm
- Marion Walser: Personen-Export (Adress-Export)
- Marion Walser: ClientService im Client erstellen
- Marion Walser: PersonView überarbeiten
- Marion Frei: Flugreport mit Versenden fertigstellen
- Marion Frei: Überarbeitung Summary für Rechnungsstellung und Bericht an BAZL
- Marion Frei: Systemtest-Dokumentation
- Alle: Bugfixing
- Lara und Marion Walser: Refactoring

37.12.4 Nächster Termin

Datum:04.06.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: 1.258

37.13 Sitzungsprotokoll Woche 16 - 04.06.2012

37.13.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.13.2 Traktanden

- Aktuelles
- Ziele der Woche

37.13.3 Beschlüsse und Diskussionen

- Marion Walser: Iterationsplan erstellen
- Marion Walser+Marion Frei: Use Cases überprüfen/ergänzen
- Alle: Abstrakt + Poster bis morgen
- Alle: SAD ergänzen/überprüfen
- Alle: Review am Mittwoch + Donnerstag immer zu zweit
- Alle: Refactoring
- Alle: Dokumente überprüfen
- Alle: Systemtests ergänzen + durchführen
- Alle: Unit Tests schreiben
- Alle: Bugfixen gemäss BugFixing+Todo Dokument.

37.13.4 Nächster Termin

Datum: 11.06.2012

Zeit: 10:00

Ort: HSR Rapperswil

Raum: 1.258

37.14 Sitzungsprotokoll Woche 17 - 11.06.2012

37.14.1 Teilnehmer

- Lara Mühlemann
- Marion Walser
- Marion Frei

37.14.2 Traktanden

- Ziele der Woche
- Endspurt

37.14.3 Beschlüsse und Diskussionen

- Titelblatt: MarionF
- Erklärung eigenständige Arbeit: MarionW
- Management Summary: Lara
- Technischer Bericht: Alle
- Persönliche Berichte: Alle
- Dokumente überprüfen: Alle
- Glossar: Alle, beim Überprüfen der Doku vorne zu ergänzen
- Literaturverzeichnis: Alle
- Klassendiagramme: MarionF
- Software Architektur Dokument: Alle
- Bedienungsanleitung: MarionF
- Code-Dokumentation: Alle
- Installationsanleitung für Patrick: MarionW

XVII

Anhang

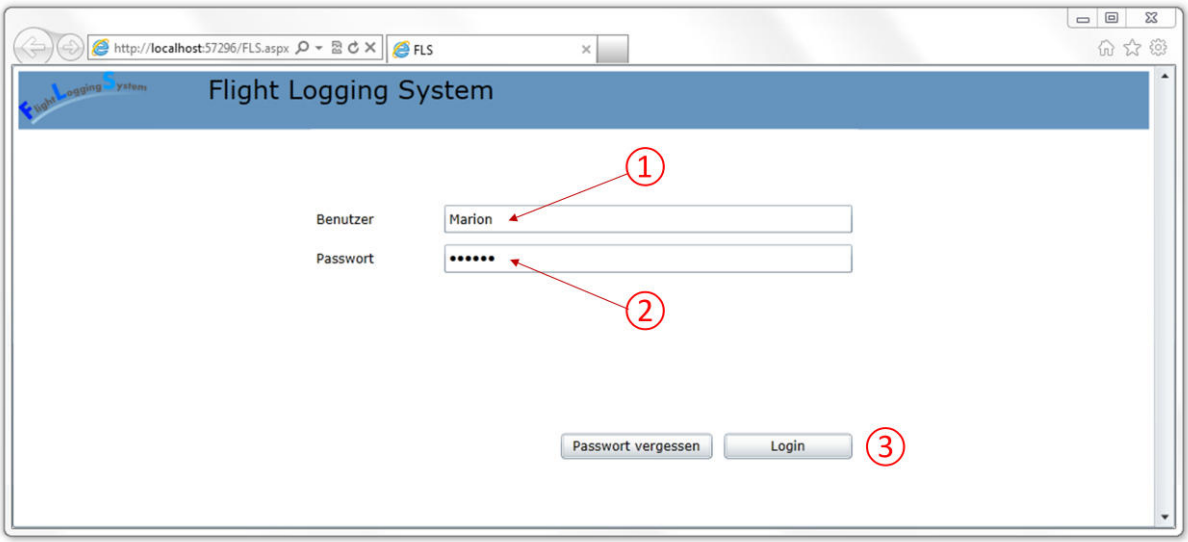
Bedienungsanleitung

Flight Logging System
Bedienungsanleitung

Inhalt

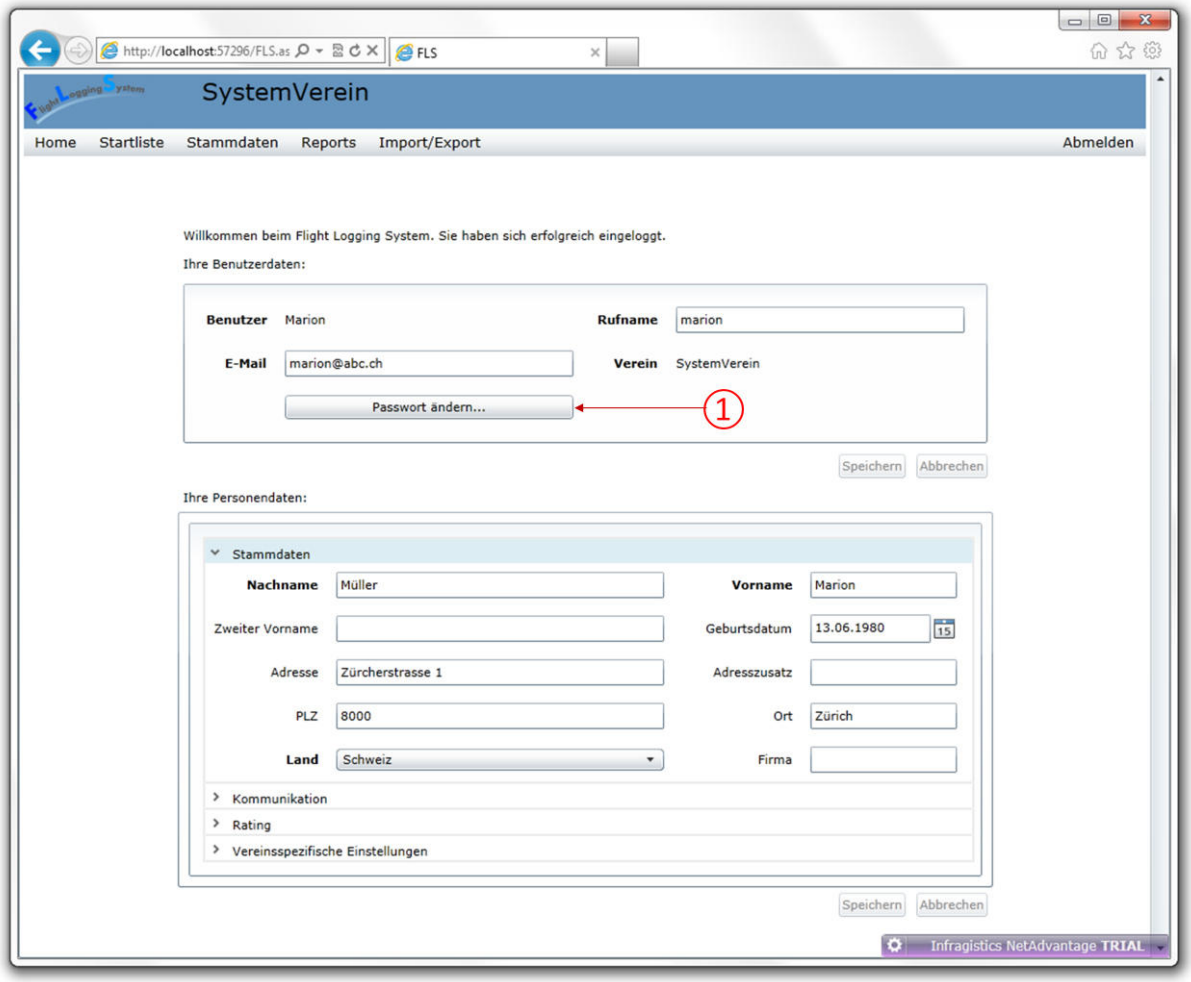
1. In das FLS einloggen

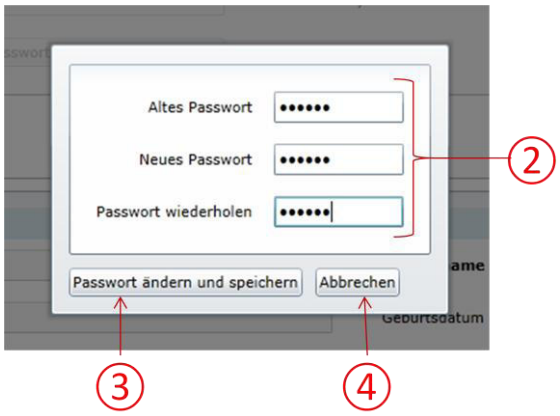
- Öffnen sie das Flight Logging System.
- Geben Sie im Startbildschirm ihren Benutzernamen ① und Ihr Passwort ② ein.
- Achten Sie bei der Eingabe des Passworts auf die Gross- und Kleinschreibung.
- Drücken Sie auf den "Login"-Button ③.
- Es öffnet sich die Home-Seite. Hier können Sie ihre Benutzer- und Personendaten direkt ändern. Dies können Sie aber auch über die Stammdaten-Verwaltung machen.



2. Passwort ändern

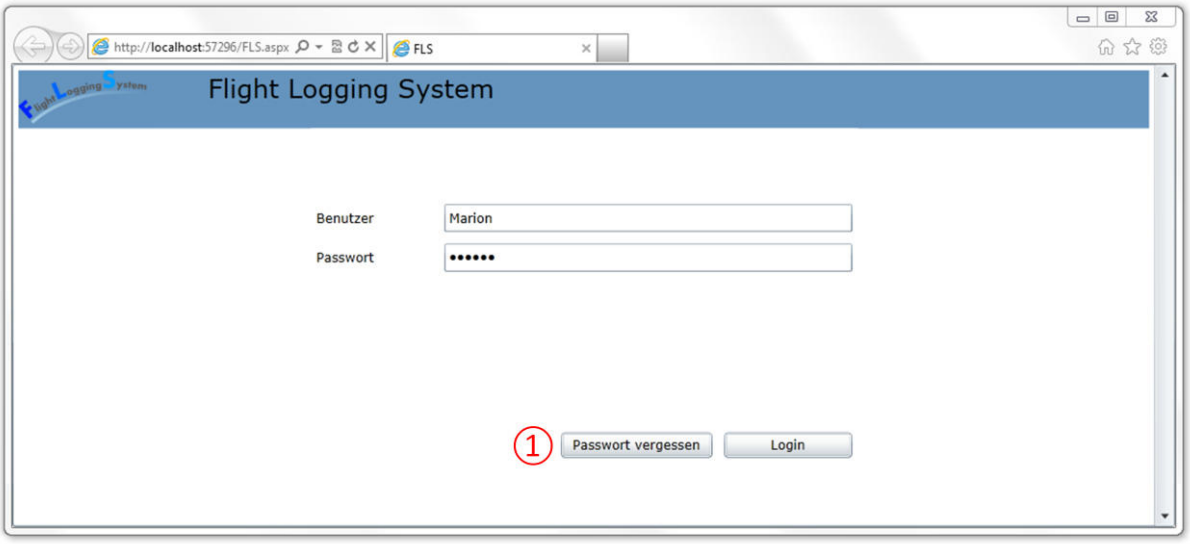
- Um Ihr Passwort zu ändern wechseln Sie auf die Home-Seite.
- Klicken Sie dort auf den "Passwort ändern"-Button ①.
- Geben Sie hier ② Ihr altes Passwort sowie zwei Mal ihr neues Passwort ein.
- Drücken Sie dann auf den "Passwort ändern und speichern"-Button ② , um Ihr Passwort zu speichern oder auf den "Abbrechen"-Button ③ , wenn Sie ihr Password nicht ändern möchten.

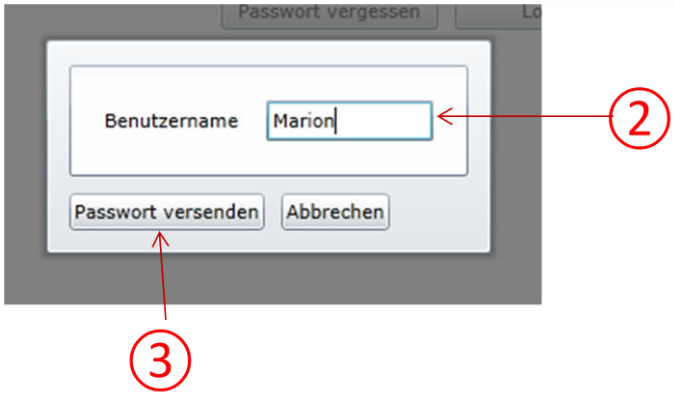




3. Neues Passwort anfordern

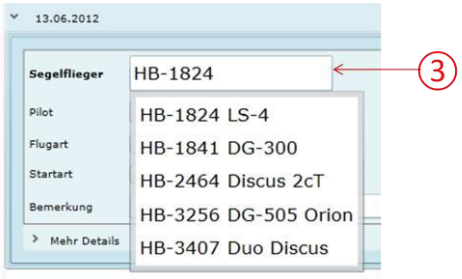
- Sollten Sie ihr Passwort vergessen haben, klicken Sie auf den "Passwort vergessen"-Button ①.
- Geben Sie danach Ihren Benutzernamen ein ②.
- Drücken Sie auf den "Passwort versenden"-Button ③.
  Sie erhalten eine E-Mail mit dem neuen Passwort.

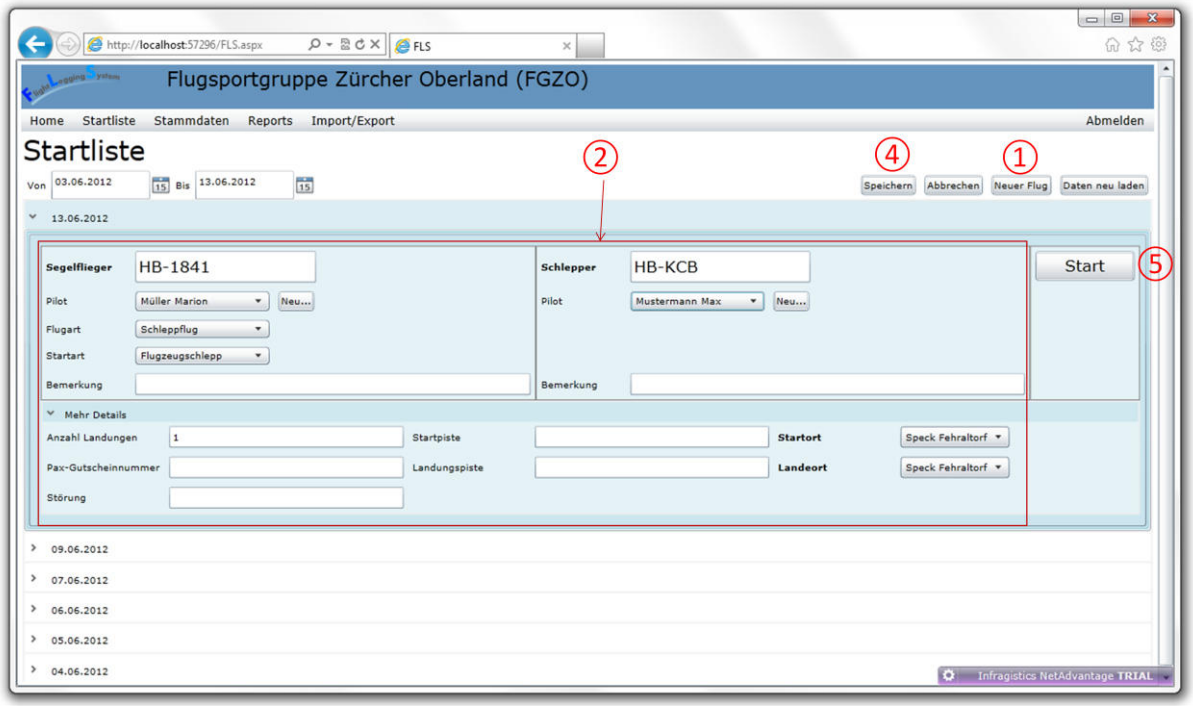




4. Flug erstellen

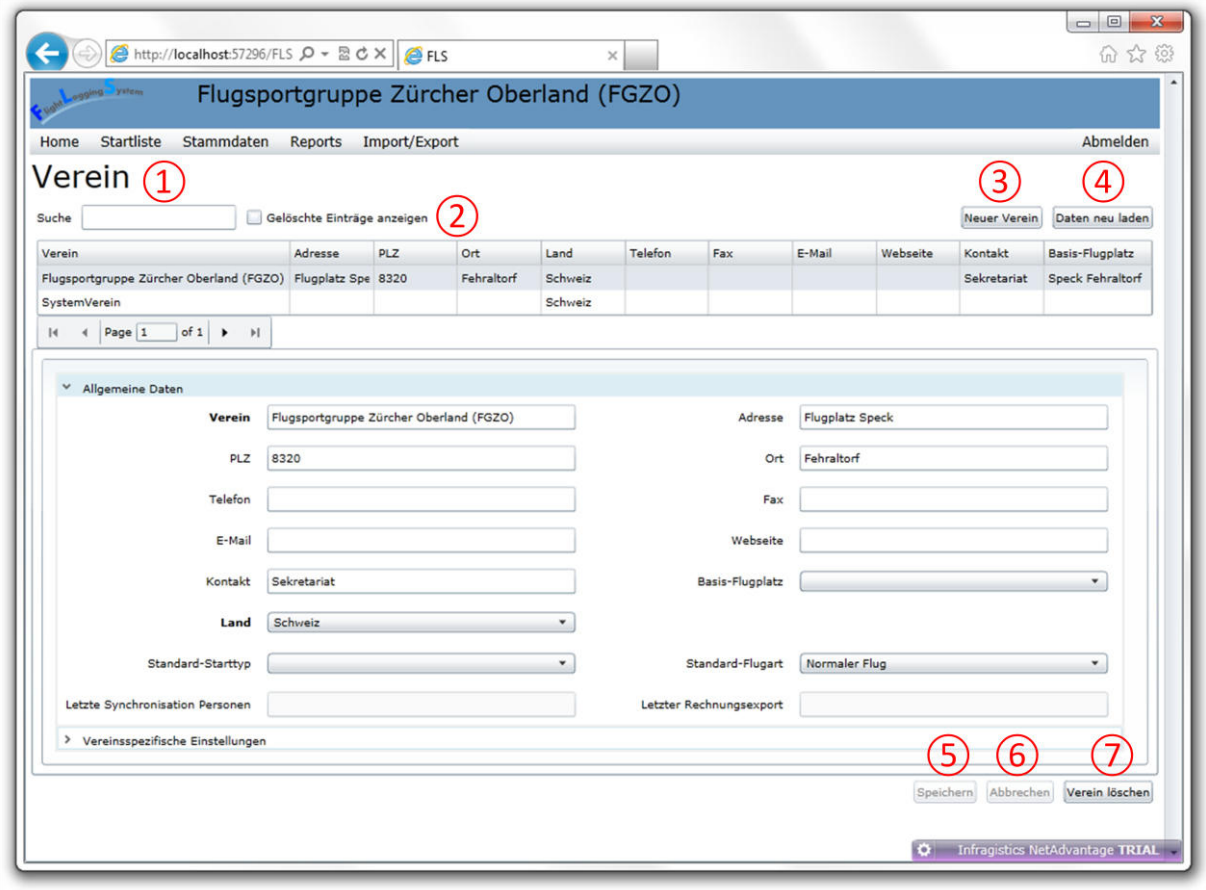
- Wechseln Sie in die Startlisten-Ansicht.
- Drücken Sie auf den "Neuen Flug"-Button ①.
- Füllen Sie die Felder im neu erstellten Flug aus ②. In den Feldern der Flieger können Sie den Namen oder die Nummer des Fliegers eintippen wobei Ihnen Vorschläge gemacht werden.③
- Sobald Sie die Pflichtfelder ausgefüllt haben können Sie wenn Sie möchten (Speichern ist jederzeit möglich, falls die Felder korrekt ausgefüllt sind) Speichern indem Sie auf den "Speichern"-Button ④ drücken.
- Um den Flug zu starten drücken Sie auf den "Start"-Button ⑤.
- Um den Schlepp- oder Segelflug zu Landen drücken Sie auf den "Landen"-Button, der erscheint, sobald der Flug gestartet ist.





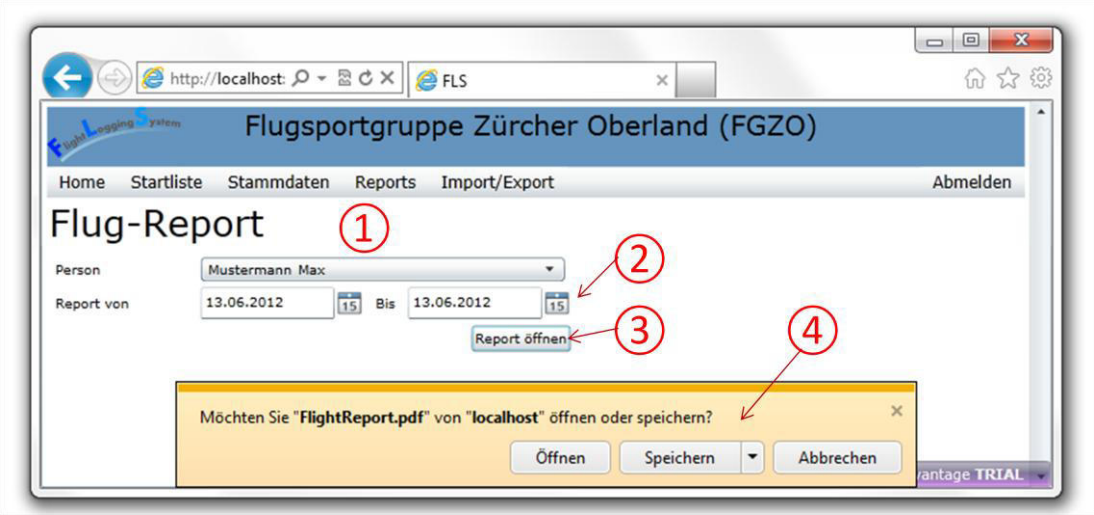
5. Stammdatenverwaltung

- Alle Ansichten in der Stammdatenverwaltung sind identisch aufgebaut. Im oberen Teil finden Sie jeweils eine Tabelle mit den wichtigsten Daten. Im unteren Teil können Sie sich die Details eines Eintrages ansehen und die Daten bearbeiten.
- Im Suchfeld ① können Sie nach einem bestimmten Eintrag suchen, daneben können Sie als gelöscht markierte Einträge ein-/ ausschalten ②.
- Mit diesem Button ③ können Sie einen neuen Eintrag erstellen.
- Wenn Sie diesen Button ④ drücken, werden die Daten neu geladen, falls diese von einem anderen System aus bearbeitet oder erstellt wurden.
- Hier können Sie Speichern ⑤ und Abbrechen ⑥ sowie einen Eintrag löschen ⑦.



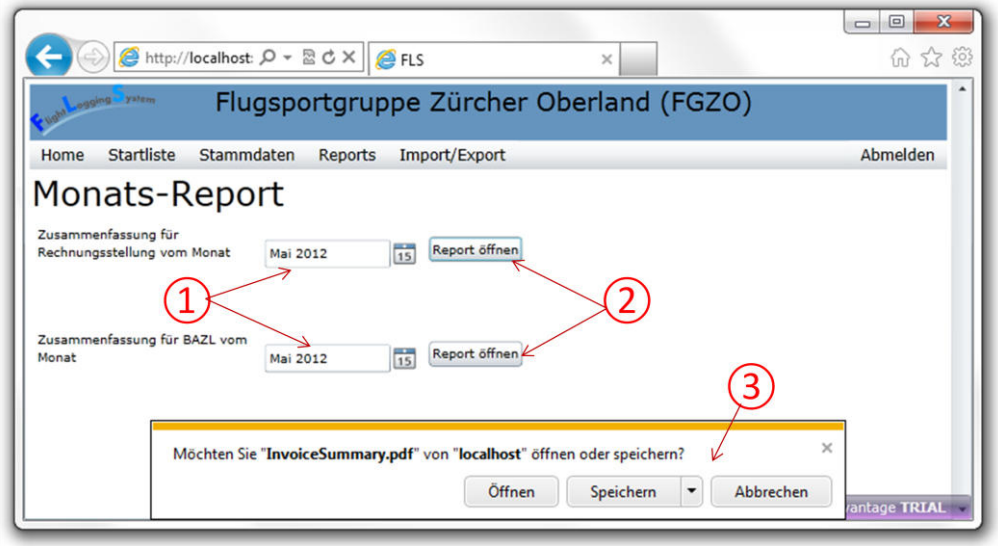
6. Flug-Reports manuell erstellen

- Wählen Sie die Person ① aus, von der Sie einen Report erstellen möchten. Beachten Sie dabei, dass Sie nur Reports von Personen erstellen können, die sich im selben Verein wie Sie befinden.
- Wählen Sie die Zeitspanne ② für den Report aus.
- Drücken Sie auf den Button "Report öffnen" ③.
- Sie erhalten den Report als PDF und werden gefragt, ob Sie dieses öffnen oder speichern wollen ④.



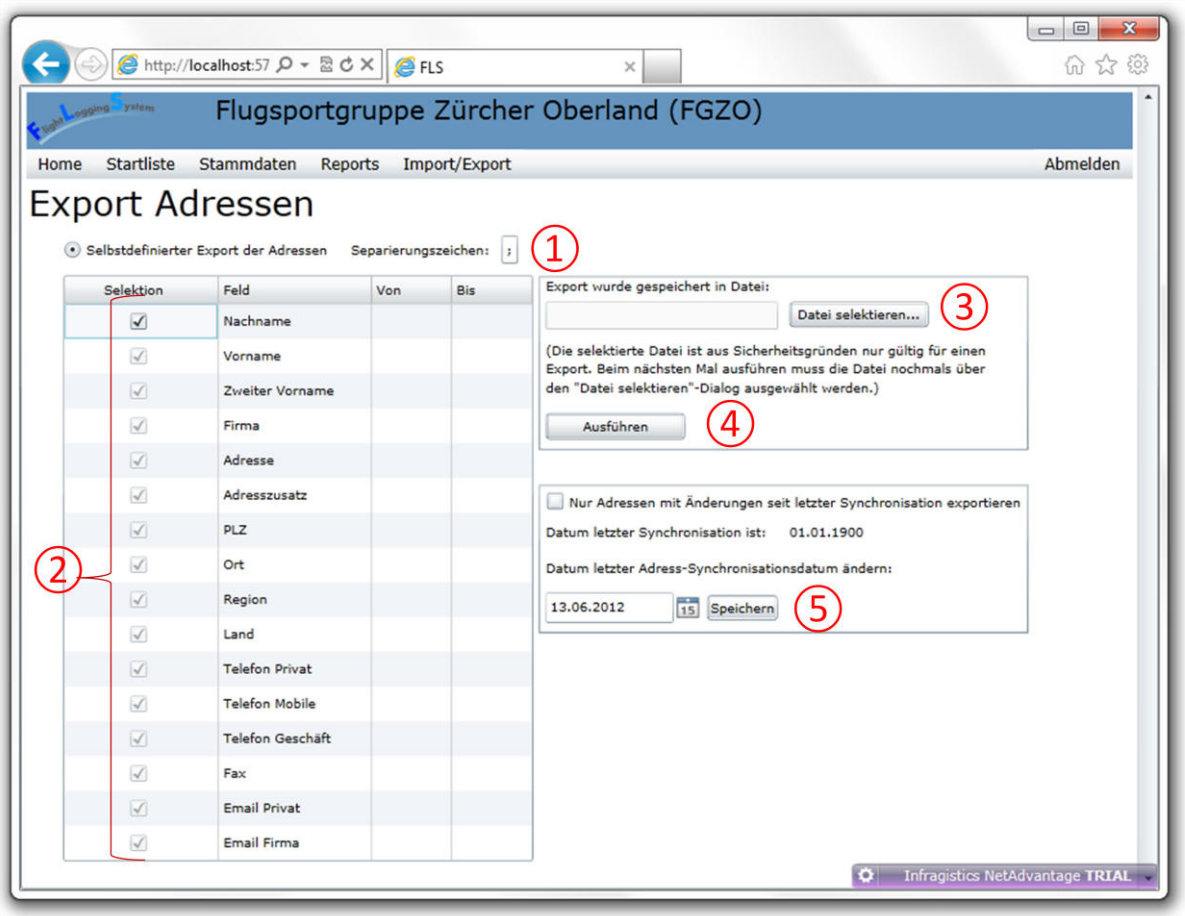
7. Monatliche Reports manuell erstellen

- Wählen Sie je nach gewünschtem Report den Monat ① , über den Sie den Report erstellen möchten, aus.
- Drücken Sie je nach Report auf den entsprechenden"Report öffnen"-Button ②.
  Der Report öffnet sich. ②



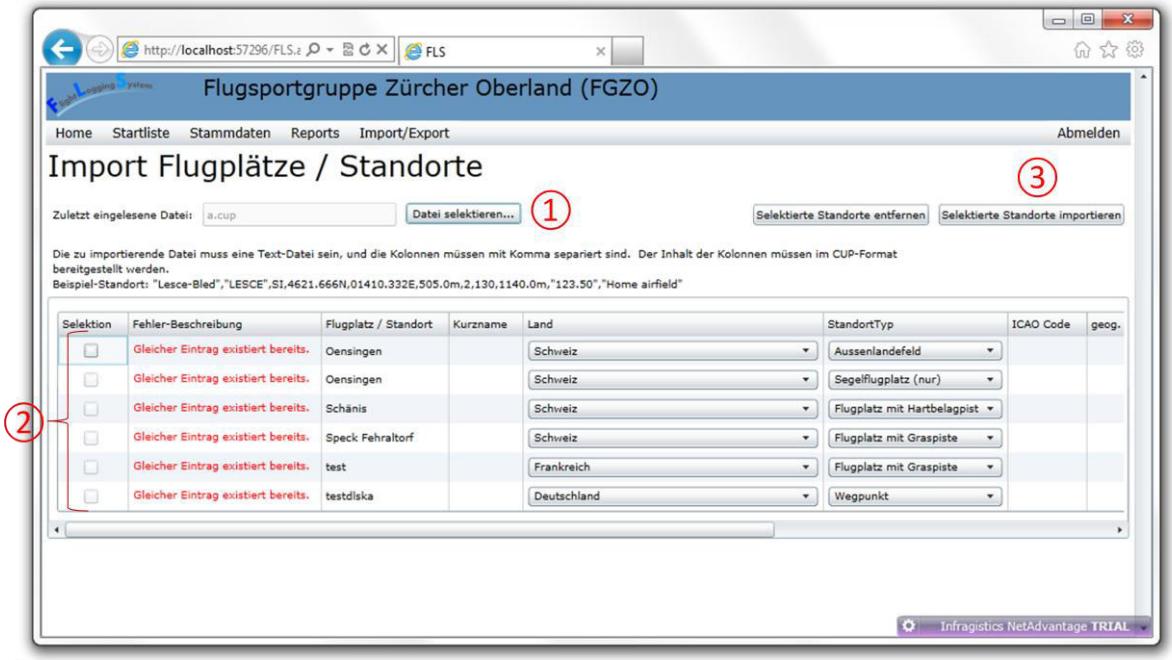
8. Adressen / Standorte exportieren

- Wählen Sie aus, ob Sie die zu exportierenden Daten als selbstdefiniertes Format oder in der Standort-Export-Ansicht als Cup-Format abspeichern möchten ①.
- Wählen Sie die zu exportierenden Felder aus ②.
- Selektieren Sie eine Datei ③, in der Sie die zu exportierenden Daten speichern möchten.
- Drücken Sie auf den "Ausführen"-Button ④.
- In der Adressen-Export-Ansicht können Sie abspeichern, dass Sie die Adressen exportiert haben ⑤.



9. Adressen / Standorte importieren

- Wählen Sie die einzulesende Datei aus ①.
- Wählen Sie die Einträge aus ②, die Sie importieren möchten.
- Drücken Sie auf den "Selektierte Standorte importieren"-Button③.



Glossar

Begriff	Beschreibung
Accordion	Ein aufklappbares GUI-Element
BAZL	Bundesamt für Zivilluftfahrt
CRUD-Operationen	Operationen zum Erstellen (Create), Lesen (Read), Aktualisieren (Update) und Löschen (Delete) von Daten.
CUP File Format	In diesem Format wird die Locations-Tabelle ins SeeYou importiert bzw. vom SeeYou exportiert.
DAeC-Index	Für Wettbewerbe - ermöglicht fairen Vergleich der Streckenleistungen.
DTO	Data Transfer Object
FLARM	Kollisionswarngerät
FLS	Flight Logging System (Name der Applikation)
Flugbewegung	„Start oder Landung eines Luftfahrzeugs“ [Flu12]
Flugbuch	Piloten sind verpflichtet einen Nachweis über ihre Flüge zu erbringen. Dazu erfassen sie ihre Flüge in einem Flugbuch.
Flugdienstleiter	Person welche die startenden Flüge erfasst.
Flugminuten	Anzahl Minuten, die ein Flugzeug in der Luft ist.
ICAO	International Civil Aviation Organization (deutsch: Internationale Zivilluftfahrtorganisation)
Immatrikulation	Individueller alphanumerischer Code, der ein Luftfahrzeug eindeutig identifiziert.
Landebahn	Strecke auf der ein Flugzeug landen kann.
Materialwart	Person, welche Wartungen durchführt
MVVM	Model-View-ViewModel - Architekturmuster für Software.
Piste	Start und Landebahn für ein Flugzeug
RIA	Rich Internet Application. Internetanwendung, die viele Interaktionsmöglichkeiten mit ihrer Benutzeroberfläche bietet.
RUP	Rational Unified Process
Schleppminuten	Anzahl Minuten, die das Schleppflugzeug in der Luft ist.

SeeYou	Streckenflugplanungs- und Analyseprogramm für Segelflieger, Drachenflieger und Gleitschirmflieger
Startbahn	Strecke auf der ein Flugzeug starten kann.
Startliste	Eine Erfassung aller Flüge, die von einem Flugplatz starten.
UC	Use Case. Anwendungsfall, der alle möglichen Szenarien bündelt.
Wettbewerbs-Zeichen	Um an einem Wettbewerb mitzumachen braucht ein Flieger ein Wettbewerbszeichen um identifizierbar zu sein.

Literaturverzeichnis

[Aut12] *WCF RIA Services Part 7 - Authentication and Authorization: http://www.silverlightshow.net/items/WCF-RIA-Services-Part-7-Authentication-and-Authorization.aspx*. 2012

[FGZ11] FGZO: *Original Startliste der Flugsportgruppe Zürcher Oberland (FGZO)*. 2011

[Flu12] *Flugbewegung Begrifferklärung:* `http://www.duden.de/rechtschreibung/Flugbewegung`. 2012

[Mor07] MORI, Lapo F.: *Tables in LATEX2: Packages and Methods:* `http://www.tug.org/pracjourn/2007-1/mori/mori.pdf`. 2007

[Qua] *Quartz.NET Scheduler Tutorial : http://quartznet.sourceforge.net/tutorial/index.html*

[Sch12] SCHULER, Patrick: *Programmier-Richtlinien:* `wiki.glider-fls.ch/images/CSharp_Programmier_Richtlinien.docx`. 2012

[Sil12] *WCF RIA Services Part 8 - Testing and Debugging : http://www.silverlightshow.net/items/WCF-RIA-Services-Part-8-Testing-and-Debugging.aspx*. 2012

[SSR05] *SSRS Tutorial : http://msdn.microsoft.com/en-us/library/ms170246%28v=sql.90%29*. 2005

[SSR12a] *Adding Query Parameters to SQL Server 2008 Reporting Services Reports : http://arcanecode.com/2010/07/13/adding-query-parameters-to-sql-server-2008-reporting-services-reports/*. 2012

[SSR12b] *Expression Examples (Report Builder and SSRS) : http://msdn.microsoft.com/en-us/library/ms157328.aspx*. 2012

[SSR12c] *Using Report Parameters in SQL Server Reporting Services : Using Report Parameters in SQL Server Reporting Services*. 2012

[Tes12] *Walkthrough: Creating Unit Tests for the Middle Tier : http://msdn.microsoft.com/de-at/library/ee707368%28v=VS.91%29.aspx*. 2012

[UI110] *UserInterface 1 Vorlesungsfolien LE01*. 2010

[Vor] *Vorlage für Tabellen:* `http://www.latexwiki.org/Loesungen:Tabellen`

Verweise auf verwendeter Webinhalte

1 Style für AccordionItem

Der Style sorgt dafür, dass das AccordionItem sich dynamisch dem Inhalt anpasst.

Quelle (zuletzt aufgerufen am 23.04.2012):

- http://stackoverflow.com/questions/3680985/how-to-get-accordion-region-to-expand-vertically-to-dynamic-content

2 DatePicker für Monate

Das DLL ermöglicht einen DatePicker, bei dem Monate ausgewählt werden können (Jahres-Ansicht). Der Zugriff auf die Monatsansicht (mit den Tagen) ist nicht möglich.

Quelle (zuletzt aufgerufen am 28.05.2012):

- http://netprogrammingodyssey.wordpress.com/2010/11/14/monthyear-only-datepicker/
- http://netprogrammingodyssey.wordpress.com/2011/04/17/monthyear-only-datepicker-revised/

3 Dynamische Zuordnung des Anzeige-Textes für ein Enum

Diese Source wurde verwendet, um die Bezeichnung der UserStatus über eine Ressource-Datei ermitteln zu können.

Quelle (zuletzt aufgerufen am 12.06.2012):

- http://stackoverflow.com/questions/3680985/how-to-get-accordion-region-to-expand-vertically-to-dynamic-content

4 Erkennung Codierung von importierten Daten

Diese Source wurde verwendet, um beim Importieren von Daten die Codierung zu erkennen.

Quelle (zuletzt aufgerufen am 12.06.2012):

- http://utf8checker.codeplex.com/SourceControl/list/changesets

Programmier-Richtlinien

Flight Logging System

C# Programmier Richtlinien

Guidelines und Best Practices

Projekt:	Flight Logging System (FLS)
Erstellt:	18. Februar 2012
Letzte Änderung:	22. Februar 2012
Version:	1.0
Ablage:	CSharp Programmier Richtlinien.docx

History:

Datum:	Autor:	Version:	Beschreibung:
18.02.2012	PS	1.0	Dokument erstellt

Qualitäts-Sicherung:

Review:	Datum:	Dokument:	Beschreibung:

	Name:	Datum:	Visum:
erstellt:	P. Schuler	18.02.2012	
geprüft:			
freigegeben:			

Inhalt

1 Einleitung

Eine konsistente Namensgebung ist eines der wichtigsten Elemente für einheitlichen, wartbaren Code. Dieses Dokument enthält Richtlinien und Konventionen für die Programmierung in C#.

2 Namenskonventionen

2.1 Pascal Schreibweise für Klassen- und Methodennamen

Bei der Pascal Schreibweise wird der erste Buchstabe des Bezeichners, sowie der erste Buchstabe aller angehängter Wörter gross geschrieben. Pascal Schreibweise für alle Typ- und Methodennamen verwenden.

Bsp.:

```
public class SomeClass
{
   public SomeMethode( );
}
```

2.2 Kamel Schreibweise für Parameter und lokale Variabeln

Bei der Kamel Schreibweise wird der erste Buchstabe des Bezeichners klein, und der erste Buchstabe aller angehängter Wörter gross geschrieben. Kamel Schreibweise für alle lokalen Variabeln und Parameter verwenden.

Bsp.:

```
private int _number;
private string _lastName;
void SomeMethod(int someNumber)
{
}
```

2.3 Namen von Interfaces haben ein 'I' Präfix

Interfaces zur speziellen Erkennbarkeit immer mit einem 'I' Präfix benennen.

Bsp.:

```
interface IDisposable;
```

2.4 Präfix '_' für private member Variablen

Private member Variablen (Fields) werden zur speziellen Erkennbarkeit mit dem Präfix '_' benannt.

Bsp.:

```
private int _number;
```

2.5 Konstanten

Wir verwenden für Konstanten, welche sich nie ändern, konstante Felder und verwenden die Pascal Schreibweise. Für value types kann das Schlüsselwort const genommen werden. Für Objekte oder Strukturen muss static readonly voran gestellt werden.

Bsp.:

```
public const int MaxValue = 0x7fffffff;
public static readonly object Lock = new object();
```

2.6 Controls

Controls auf einer Form (oder einem UserControl) werden ohne Präfix benannt.

Bsp.:

```
private System.Windows.Forms.Button okBtn;
```

2.7 Ungarische Schreibweise vermeiden

Die Ungarische Schreibweise kommt in Programmiersprachen wie C++ mit seinen primitiven Datentypen zur Anwendung. Dabei werden Namen von Variabeln mit einem Präfix versehen, der anzeigt, welchen Datentyp die Variable besitzt. (z.B. iNumber, pWindow, nCount, bFlag)
In einer reinen objektorientierten Programmiersprache, (was C++ nicht ist) ist die ungarische Schreibweise unnötig.

2.8 Methoden und member Variablen ordnen

Alle member Variablen sind am Anfang der Klassendeklaration zu platzieren, gefolgt von den Methoden und Properties.

Bsp.:

```
public class SomeClass
{
   int _number;
   string _lastName;

   public void Methode( ) { }
   public void Methode( ) { }
}
```

2.9 Variablendefinitionen solange wie möglich aufschieben

Durch das Aufschieben von Variablendefinitionen wird die Effizienz der Programme erhöht, sie werden leichter verständlich und die Notwendigkeit den Sinn der Variablen zu dokumentieren wird reduziert.

2.10 Namespace Benennung

Nach Microsoft Guideline:

```
Firma.Technologie[.Feature][.Design]
```

Angepasst für FLS:

```
FLS.<Produkt>.<Domain/Subjekt>[.Weitere Ordner]
```

Für das FLS-Projekt sieht dies zum Beispiel folgendermassen aus:

Für Klassen, welche direkt im Root vom Server-Workflow-Management liegen:

```
namespace FLS.Server.WorkflowManagement
```

Für Klassen, welche im Ordner vom Shared-Timing liegen:

```
namespace FLS.Shared.Timing
```

2.11 Assembly Benennung

Ein Assembly soll wie folgt benennt werden:

```
FLS.<Produkt>.< Domain/Subjekt>.dll
```

Für das Server-Projekt sieht dies zum Beispiel folgendermassen aus:

```
FLS.Server.dll
```

Weitere Beispiele:

```
FLS.Shared.Common.dll
FLS.Shared.Diagnostics.dll
FLS.Shared.Security.dll
FLS.Shared.DTO.dll
FLS.Shared.Workflow.dll s
FLS.Shared.Timing.dll
FLS.Server.Plugin.dll
```

2.12 Voll qualifizierte Typnamen vermeiden

Voll qualifizierte Typnamen sind zu vermeiden. → Das using Schlüsselwort verwenden.

2.13 Die `using namespace` Statements gruppieren

Alle framework namespaces zusammen auflisten, gefolgt von third party namespaces.

Bsp.:

```
using System;
using System.Collection;
using FLS.Shared.Common;
```

2.14 Eine Klasse pro Datei mit entsprechendem Dateinamen

In der Regel soll eine Datei nur eine Klassendefinition enthalten. Der Dateinamen entspricht dem Klassennamen.

3 Code-Konventionen

3.1 Geschweifte Klammern '{' auf separaten Zeilen

3.2 Codezeilen mit mehr als 110 Zeichen vermeiden.

Für VisualStudio 2008:

Visual Studio bietet die Möglichkeit eine vertikale Linie im Editor darzustellen. Diese kann einem sofort deutlich machen, wenn Codezeilen zu lang werden. Um diese Linie einzuschalten ist ein kleiner Eingriff in der Registry nötig:

```
[HKEY_CURRENT_USER\Software\Microsoft\VisualStudio\9.0\Text Editor]
"Guides"="RGB(192,192,192) 110"
```

Die Werte für die RGB-Funktion bestimmen die Farbe der dargestellten Linie, die Nummer zuhinterst bestimmt, wie weit hinten die Linie erscheinen soll (z.B. beim 110. Zeichen).

Für Visual Studio 2010

Das oben beschriebene guideline feature war nie offiziell unterstützt und funktioniert im Visual Studio 2010 nicht. Die Funktion wird aber von der Microsoft Studio Extension „Productivity Power Tools“ angeboten.

3.3 Klassen nur wenn nötig public machen, sonst internal

3.3.1 Public Members in Public Klassen möglichst kurz halten

Grund: die mit „public“ deklarierten Funktionen und Properties werden mit den Dotfuscatoren nicht verschleiert (→ keine Veränderung der Interfaces durch den Dotfuscator).
Abhilfe: so schnell wie möglich eine private oder internal Funktion aus einer public Funktion aufrufen.

3.4 Arrays: auf 0 basierenden Index verwenden.

3.5 Schlüsselwort `goto` nicht verwenden

Das Schlüsselwort goto soll nicht verwendet werden. Ausnahme: expliziter Fall-Through in einer switch Anweisung.

3.6 Für den Benutzer sichtbare Texte nicht hart codieren

Für sichtbare Texte müssen Ressourcen verwendet werden, damit Übersetzungen in andere Sprachen möglich sind.

3.7 Keine public oder protected member Variablen → Properties verwenden

3.8 Nicht mehr als 20 members pro Interface

Ein Interface mit zu vielen members zeugt von schwachem Design. 12 members pro Interface ist etwa eine praktikable Limite.

3.9 public und protected Methoden mit `virtual` deklarieren

Ausnahme: Abgeschlossene Klassen, von denen sowieso nicht mehr abgeleitet werden kann (sealed classes).

3.10 Geschweifte Klammern auch für einzeilige if-Blocks verwenden

Bsp.:

```
if (true)
{
   DoSomething();
}
```

3.11 In `switch` Statements immer einen `default` Pfad verwenden

Bsp.:

```
switch (number)
{
   case 1:
      Trace.WriteLine("Case 1:");
      break;
   default:
      Debug.Assert(false);
  break;
}
```

3.12 Properties

3.12.1 Property Name

Für Properties, die vom Typ bool sind, folgende Konvention verwenden:

```
public bool IsReady (nicht nur Ready)
public bool HasData (nicht nur Data)
```

3.12.2 Property Kommentar

Der Hauptkommentar gilt immer nur für den sichtbarsten Modifier (meistens den public). Das heisst, wenn der get public ist und der set nur protected, so ist der Hauptkommentar nur für den get zu schreiben.
Der set Kommentar wird separat über dem set geschrieben.

Beispiel:

```
/// <summary>
   /// Gets true, if the object is ready to manipulate.
   /// </summary>
   /// <value>True or False.</value>
   public bool IsReady
   {
      get
      {
         return _isReady;
      }
      /// <summary>
      /// Sets a value that indicates, if the object is ready to
      /// manipulate.
      /// </summary>
      /// <value>True or False.</value>
      internal set
      {
         _isReady = value;
      }
   }
```

3.12.3 Benutzen von Fields / Properties

Für jedes Field existiert auch immer ein Property. Auch innerhalb der Klasse wird NUR über dieses Property zugegriffen (Ausnahme Constructor, siehe 3.13).

3.13 Constructor

Beim Werte zuweisen im Constructor sollte immer direkt auf die Fields zugewiesen werden und nicht via Properties.
So kann vermieden werden, dass bereits schon Code durchlaufen wird, ehe das Objekt die nötigen Werte gesetzt hat (z.B. beim Validieren).

Beispiel:

```
public ExampleClass(string name)
{
   _name = name;
}
```

3.14 Enums

3.14.1 Namen

- Möglichst keine Abkürzungen im Namen.

3.14.2 Typ

Sobald ein Enum gespeichert wird, sollten seine Werte fixiert sein. Ansonsten kann der ursprüngliche Zustand nicht zwingend wiederhergestellt werden.

Beispiel:

```
/// <summary>
/// The days for a time window.
/// </summary>
[Flags]
public enum TimeWindowWeekday : byte
{
   /// <summary>
   /// Never. Assign this value when no days are valid
   /// </summary>
   Never = 0,
   /// <summary>
   /// The time window is valid on a Sunday
   /// </summary>
   Sunday = 1,
```

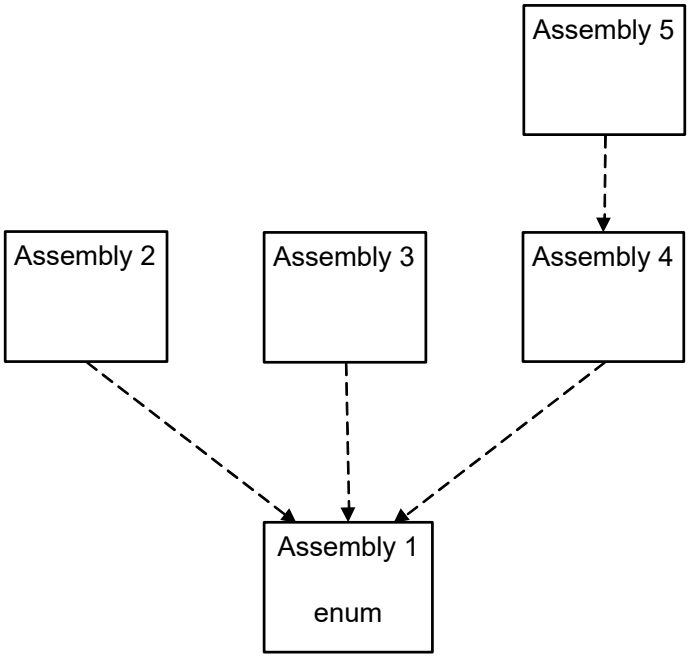
Entsprechend der Menge von Zuständen, welche definiert werden sollen, folgende Typen verwenden:

Mögliche Zustände CLR data type (.NET Framework) SQL Server data type (CLR data type in System.Data.SqlTypes)

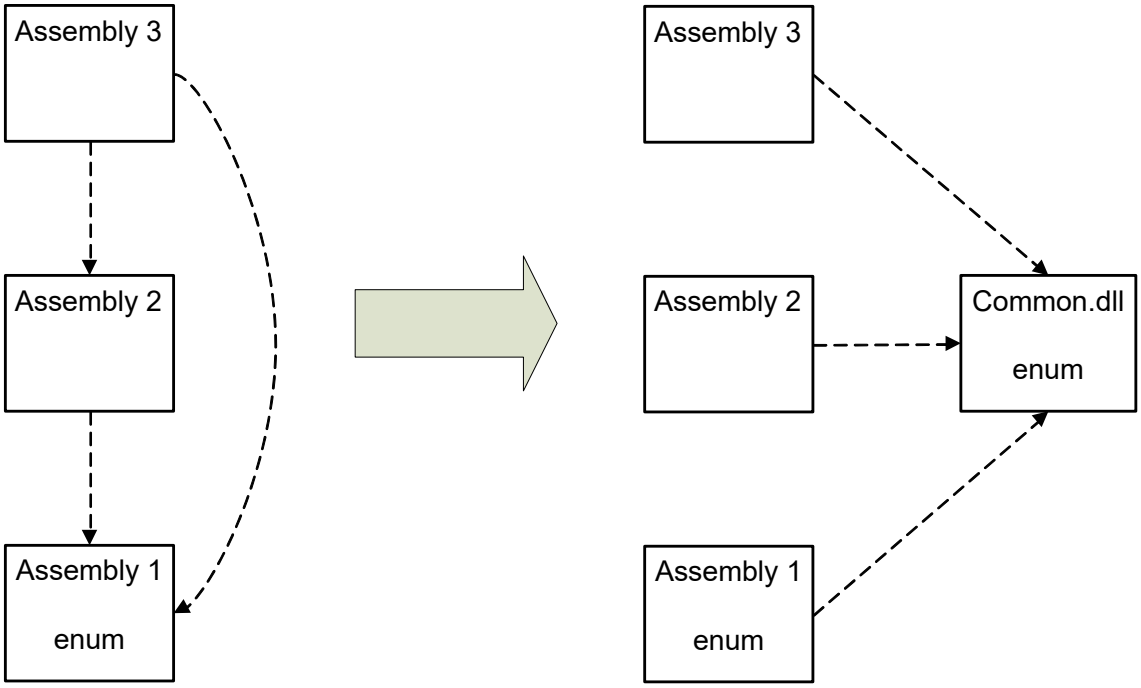
Mögliche Zustände	CLR data type (.NET Framework)	SQL Server data type (CLR data type in System.Data.SqlTypes)
.. 256	Byte	Tinyint (SqlByte)
.. 32’767	Int16	Smallint (SqlInt16)
.. 2’147’483’647	Int32	Int (SqlInt32)
> 2’147’483’647	Int64	Bigint (SqlInt64)

3.14.3 Platzierung

- Wird der Enum nur in einem Assembly benutzt wird er auch gleich in diesem Assembly definiert.
- Wird der Enum von einem oder mehreren Assemblies „direkt“ benutzt, wird dieser direkt im referenzierten Assembly definiert.



- Wird der Enum „indirekt“ benutzt, wird dieser im FLS.Shared.Common Assembly definiert.



- Ist der Enum ‚allgemeiner’, so wird pro Enum ein File erstellt (Common Assembly im Folder ‚Enumerations’).
- Enums nicht in Klasse einbetten. Ausnahme: der Enum wird ausschliesslich von dieser Klasse benutzt.

3.15 Event Deklaration

Die Signatur von Eventhandler ist gemäss folgender Konvention:

Rückgabewert: `void`
Erster Parameter: `System.Object sender`
Zweiter Parameter: `System.EventArgs e`

Dabei wird für den zweiten Parameter meistens eine eigene Spezialisierung der Klasse EventArgs verwendet.
Nun muss aber für diese Signatur mit der eigenen EventArgs Klasse kein eigener Delegate mehr definiert werden (seit .Net 2.0). Es kann der generische Delegate `EventHandler<TEventArgs>` aus dem Namespace System verwendet werden.

Beispiel:

```
        /// <summary>
        /// The argument to the ViewChangedEventHandler.
        /// </summary>
        public class ViewChangedEventArgs : EventArgs
        {
        }
        /// <summary>
        /// Occurs when the view changes.
        /// </summary>
        public event EventHandler<ViewChangedEventArgs> ViewChanged;

...
        // Subscribe to ViewChanged event.
        ViewChanged += new EventHandler<ViewChangedEventArgs>(OnViewChanged);
        ...

        /// <summary>
        /// Called when the View has changed.
        /// </summary>
        /// <param name="sender">The sender.</param>
        /// <param name="e">The <see cref=" ViewChangedEventArgs"/>
        /// instance containing the event data.</param>
        void OnViewChanged(object sender, ViewChangedEventArgs e)
        {
        }
```

4 Kommentar

4.1 Schlüsselwörter

Zum einfachen Auffinden von speziellen Codestellen werden folgende Schlüsselwörter verwendet.

Schlüsselwort	Beispiel
TODO	// TODO: PS, 22.07.2011 und dann ein Text

Offener Punkt. Muss im Rahmen des aktuellen Haupt-Release bearbeitet werden.

ISSUE	// ISSUE: PS, 22.07.2011 und dann ein Text

Pendenz, welche nicht im aktuellen Haupt-Release bearbeitet werden muss.

4.2 Versionsverwaltungskommentar

Beispiele:

MAS#0106	Mantis FLS Server – Nummer
MAC#0321	Mantis FLS Client – Nummer

5 Exceptions

5.1 Exceptions anstatt Error codes verwenden

Faustregel zum Verwenden von Exceptions:
Wenn eine Methode nicht erfolgreich ausführen kann, was ihr Name verspricht, wirft sie eine Exception.

5.2 Performance beachten beim Verwenden von Exceptions

Wenn Exceptions in Methoden verwendet werden, die sehr häufig fehlschlagen, kann es zu Performanzproblemen kommen.

5.2.1 Exception vermeiden durch vorgängiges Testen

Um Exceptions zu vermeiden kann vor dem Aufruf einer Methode getestet werden, ob sie erfolgreich sein kann.

Beispiel:

```
Icollection<int> numbers = …
// Icollection<T>.Add() wirft Exception,
// wenn die collection readonly ist.
If (numbers.IsReadOnly == false)
{
   numbers.Add(1);
}
```

Falls ein solcher Test umfangreich ist, kann er in einer zusätzlichen Methode angeboten werden. Für die Benennung einer solchen Methoden kommen zwei Varianten in Frage:

- „Is“ Präfix und „able“ Postfix Beispiel: `Public bool IsParsable();`
- Präfix “CanHandle“ verwenden: Beispiel: `Public bool CanHandleUid();`

Der Rückgabewert kann vom Typ bool sein (true für erfolgreich, false für fehlschlagen). Normalerweise wird aber ein enum Wert als Rückgabewert verwendet. Dieser kann zusätzlich zum erfolgreichen Wert (OK) mehrere Werte für verschiedene Ursachen von Fehlschlägen definieren.

5.2.2 Exception vermeiden durch das Anbieten einer zusätzlichen Try-Methode

Um Exceptions zu vermeiden kann eine zusätzliche Try-Methode angeboten werden. Diese hat „Try“ als Präfix im Namen. Sie wirft keine Exception, wenn die im Methodennamen suggerierte Funktion fehlschlägt. Ein Fehlschlagen wird mit dem Return Wert signalisiert. Eigentliche Resultate der Funktion werden mittels Out-Parameter zurückgegeben.
Der Rückgabewert ist vom Typ bool (true für erfolgreich, false für fehlgeschlagen). Alternativ kann auch ein enum Wert als Rückgabewert verwendet werden. Dieser kann zusätzlich zum erfolgreichen Wert (OK) mehrere Werte für verschiedene Ursachen von Fehlschlägen definieren.

Beispiel:

```
public struct DateTime
{
   // Wirft Exception, wenn string nicht geparst werden kann.
   Public static DateTime Parse(string dateTime)
   {...}
   // Gibt das Resultat in einem out Parameter.
   // Return-Wert ist false, wenn string nicht geparst werden kann.
   // Kann bei anderen Fehlern immer noch Exception werfen.
   Public static bool TryParse(string dateTime, out DateTime result)
   {...}
}
```

5.3 Keine reservierten Exceptions werfen

Es gibt einige Exceptions, die reserviert sind und nicht vom eigenen Code geworfen werden sollten:

5.3.1 Keine ‚CLR-spezifischen' Exceptions werfen

Folgende Exceptions sind reserviert für die Common Language Runtime (CLR) und sollten nicht geworfen werden:

```
System.OutOfMemoryException
System.IndexOutOfRangeException
System.ExecutionEngineException
System.NullReferenceException
```

5.3.2 Keine generellen Exceptions werfen

Einige Exceptions sollten nicht geworfen werden, da sie zu generell sind um hilfreiche Informationen zu liefern. Ausserdem kann es problematisch werden im Zusammenhang mit NUnit, wenn diese generelle Exception die Basisklasse einer NUnit-Exception ist.

```
System.Exception
System.SystemException
System.ApplicationException
```

6 Loggen von Debug-Informationen

Dieses Kapitel ist spezifisch für den eigen entwickelten Logger.

6.1 Public Methoden Loggen

Um den Programmfluss der Applikation zu Loggen kann folgende Codezeile zu Beginn einer Funktion eingefügt werden:

```
if (Logger.IsOn) { Logger.Log(MethodBase.GetCurrentMethod()); }
```

Der dabei generierte Logeintrag wird mit dem Level DEBUG gemacht und enthält automatisch Informationen über den Methodennamen und die Signatur der Methode.

Soll eine zusätzliche Meldung oder eine anderer Log Level verwendet werden, sieht der Aufruf wie folgt aus:

```
if (Logger.IsOn) { Logger.Log("My message", LogLevel.ERROR, MethodBase.GetCurrentMethod()); }
```

Da Namen durch Obfuszieren verändert werden können, erhält man nur für public Members lesbare Informationen über die Methoden.

6.1.1 Loggen von Funktionszeiten

Falls die bei länger dauernden Methoden die Funktionszeit geloggt werden soll, kann dies mit Hilfe eines Stopwatch Objekts gemacht werden:

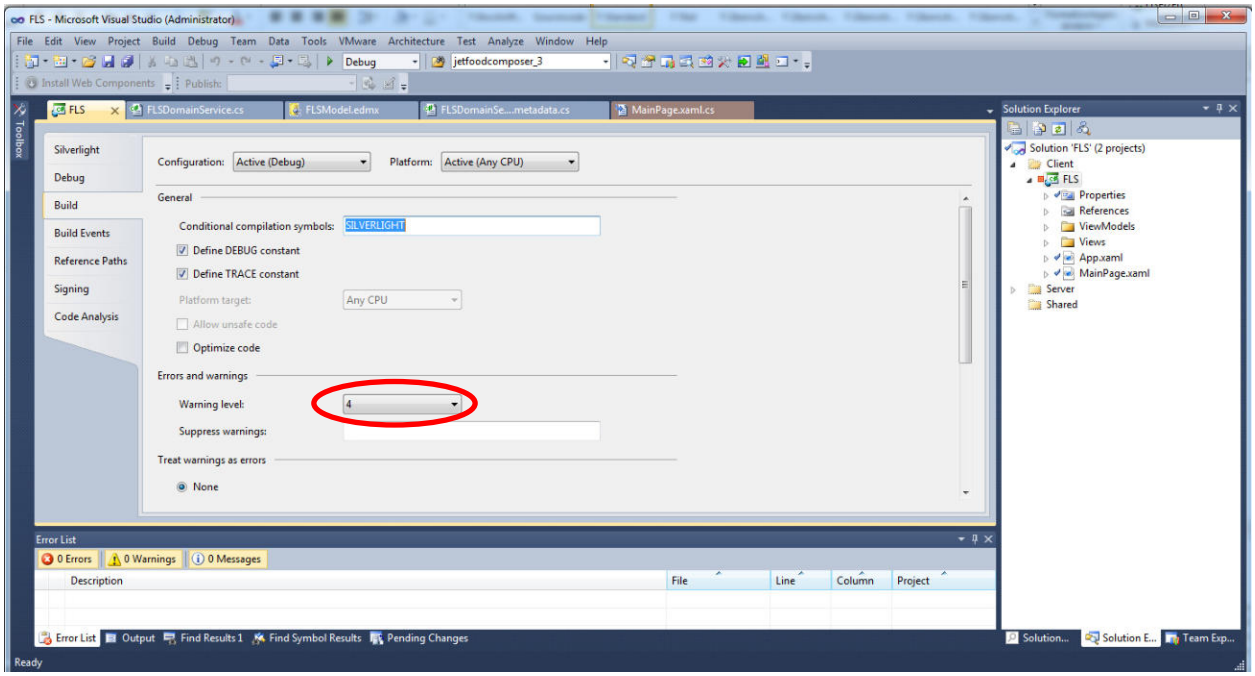
```
Stopwatch stopWatch = null;
if (Logger.IsOn)
{
   Logger.Log(MethodBase.GetCurrentMethod());
   stopWatch = new Stopwatch();
   stopWatch.Start();
}

…

if (Logger.IsOn)
{
   stopWatch.Stop();
   Logger.Log(MethodBase.GetCurrentMethod(), stopWatch);
}
```

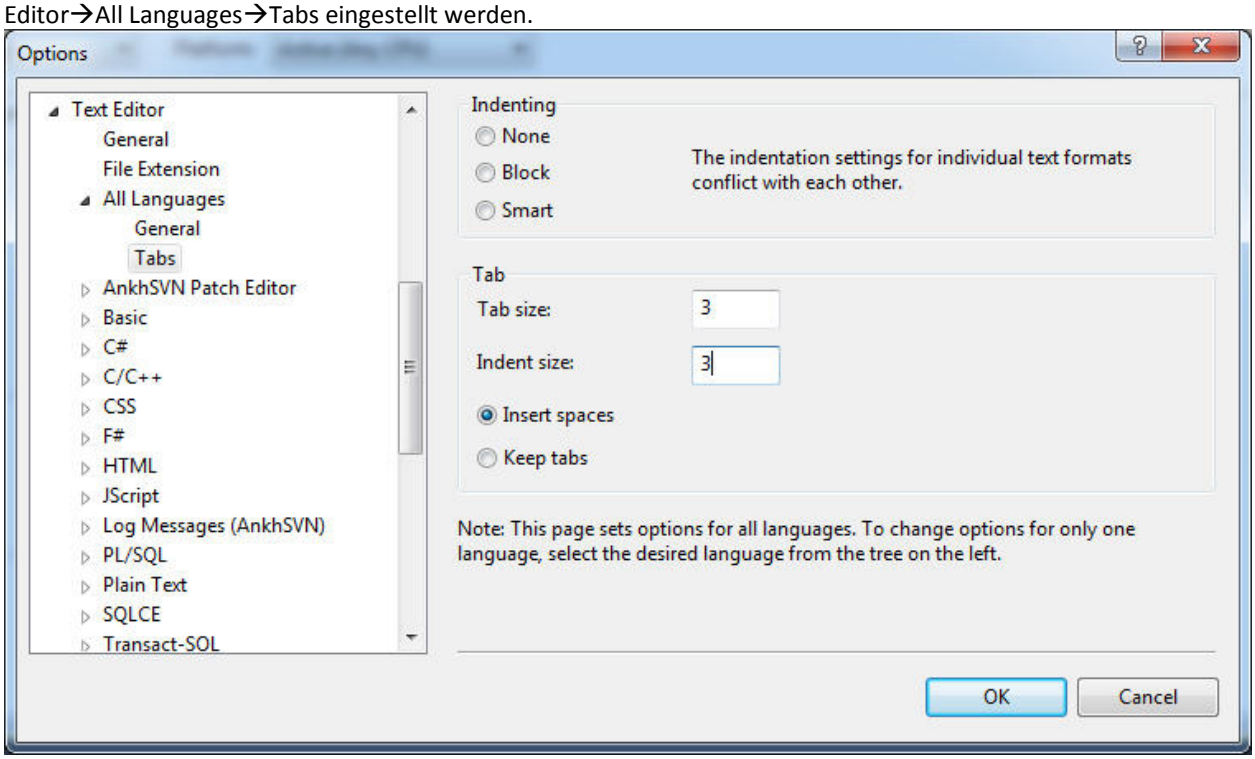
7 Projekt Einstellungen

7.1 Projekte bilden mit Warning Level 4

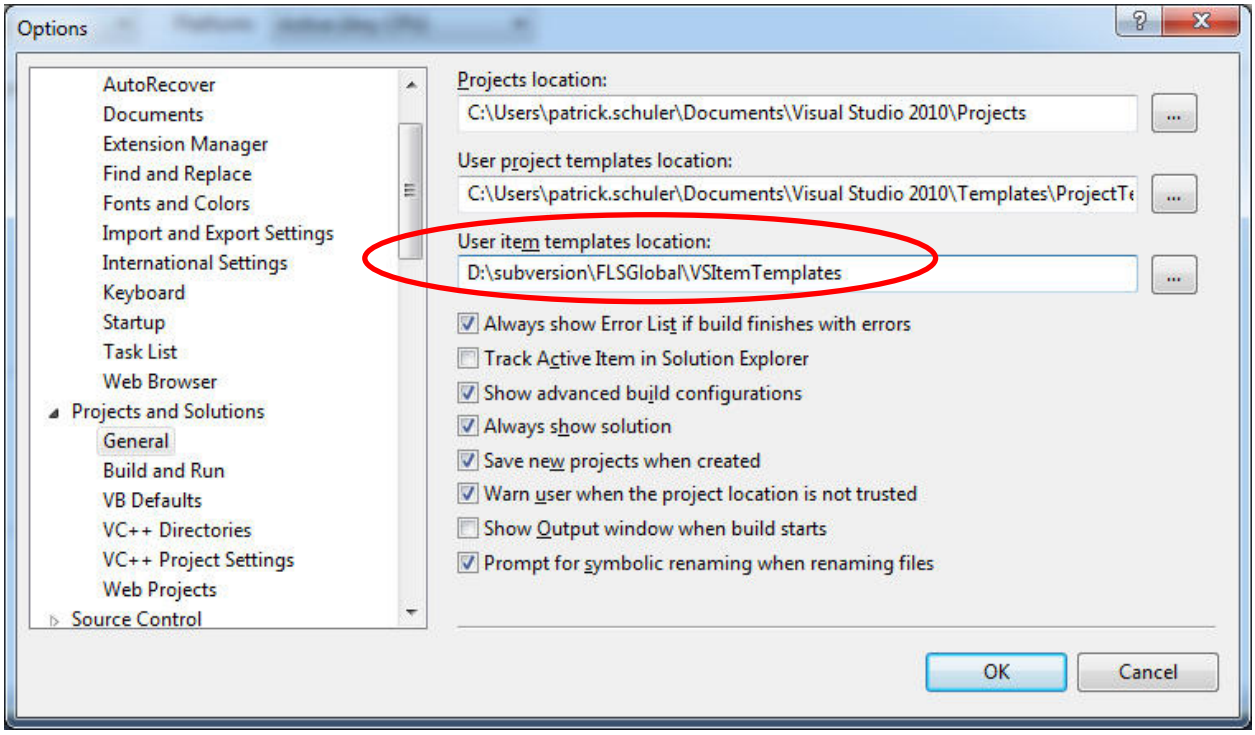


7.2 3 spaces für Tabs

Die Tabs sollen durch 3 Leerschläge (Spaces) ersetzt werden. Dies kann unter dem Menü: Tools→Options→Text Editor→All Languages→Tabs eingestellt werden.



7.3 Integration von Standard Klassen Templates



Unter Tools/Options kann man im Bereich „Projects and Solutions“ die Pfade für die unterschiedlichen Templates angeben. Hier können alle selber generierten Templates abgelegt werden. Allgemeine Templates können dann auch ins SVN übernommen und den anderen Entwicklern zur Verfügung gestellt werden.

Der Pfad „User item templates location“ sollte auf den Pfad:„/Subversion/FLSGlobal/VSItemTemplates“ gesetzt werden.

7.4 StandardClass Template

```
// ***********************************************************************
// SVN Info:
// $HeadURL: http://svn.glider-fls.ch/svn/FLS/trunk/Server/Class1.cs $
// $Revision: 5 $
// $Date: 2010-10-18 14:33:53 +0200 (Mo, 18 Okt 2010) $
// $Author: pschuler $
// -----------------------------------------------------------------------
//    (c) Copyright   FLS   Switzerland
// ***********************************************************************

using System;

namespace $rootnamespace$
{
   /// <summary>
   /// $safeitemname$ class.
   /// </summary>
   internal class $safeitemname$
   {
      #region Events
      #endregion Events

      #region Fields
      #endregion Fields

      #region Properties
      #endregion Properties

      #region Construction
       /// <summary>
       /// Constructor.
       /// </summary>
       public StandardClass1()
       {
       }
      #endregion Construction

      #region Public Methods
      #endregion Public Methods

      #region Internal Methods
      #endregion Internal Methods

      #region Protected Methods
      #endregion Protected Methods

      #region Private Methods
      #endregion Private Methods

      #region Nested Items
      #endregion Nested Items
   }
}
```

8 Referenzen

[1] Naming Guidelines, Microsoft