HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

INFORMATIK

ins institute for networked solutions

# SURVEAST Technologiestudie mit SharePoint

## Studienarbeit

Abteilung Informatik
Hochschule für Technik Rapperswil

Herbstsemester 2010


| | |
|---|---|
| Autoren: | Clemens Meier |
| | Silvan Gacond |
| Betreuer: | Prof. Hansjörg Huser |
| Projektpartner: | Kantonsspital St.Gallen |
| | Abteilung Infektiologie |

## Inhaltsverzeichnis

# 1. Aufgabenstellung

## Erfassung nosokomiale Infektionen für das Kantonsspital St.Gallen

## Aufgabenstellung für Clemens Meier und Silvan Gacond

**Einführung**
Im Spital erworbene Infektionen (=nosokomiale Infektionen (NI)) und multiresistente Keime sind grosse Herausforderungen der modernen Medizin. Im Schnitt kommt es in mindestens 5-15% der Fälle während Behandlungen im Spital zu infektiösen Ansteckungen von Patienten.
Die Abteilung Spitalhygiene der Infektiologie am Kantonsspital St. Gallen (KSSG) sucht nun nach einer Möglichkeit, Häufigkeit und die Ursachen von noskomialen Infektionen zu erfassen mit dem Ziele, diese einzudämmen. Dies bedingt die Erfassung und Auswertung aller diesbezüglich relevanten Daten.
Als Pilotsystem plant die Abteilung Spitalhygiene konkret die Erfassung nosokomialer Infekte bei Leukämiepatienten. Im Verlauf ist bei erfolgreichem Praxistest die Weiterentwicklung des Systems mit einer Ausdehnung auf einen breiten Abteilungsübergreifenden Einsatz am KSSG vorgesehen.
Gemäss einem Grundsatzentscheid am KSSG, ist das Ziel, Neuentwicklungen im Microsoft .Net Bereich mit der Umgebung Microsoft SharePoint abzuwickeln.
Es existieren bereits erste Konzeptpapiere, welche von den Verantwortlichen der Infektiologie am KSSG ausgearbeitet wurden. Im Laufe der Arbeit soll genau konkretisiert werden, welche Teile daraus im Rahmen der Semesterarbeit entwickelt werden. Der Gesamtumfang der Idee würde zusätzlich sogar eine eventuelle Nachfolgearbeit als Bachelorarbeit ermöglichen.

**Aufgabenstellung**
- Analyse der Anforderungen
- Spezifikation des zu erstellenden Systemes
- Definition der Systemarchitektur
- Implementation eines Kernsystems
- Systemtest

### Resultate
- Ausführbare Software mit Dokumentation

### Projektpartner

**Auftraggeber**

Dr. Matthias Schlegel, Leiter Spitalhygiene, KSSG
Dr. Christian Kahlert, Oberarzt Infektiologie und Spitalhygiene KSSG und Ostschweizer Kinderspital

**Projektteam**

Clemens Meier (cmeier@hsr.ch)
Silvan Gacond (sgacond@hsr.ch)

**Betreuung HSR**

Prof. Hansjörg Huser

### Projektabwicklung

**Termine:**

- Beginn der Arbeit: M**o., 20. Sept. 2010**
- Abgabetermin Kurzfassung/Poster zum Review: Di. **20.12.2010**
- Abgabetermin (ink. Poster): **Fr. 23.12.2010**, 17.00 Uhr
- Zwischenbesprechung/Review mit Auftraggeber nach Projektplan

**Arbeitsaufwand**

Für die erfolgreich abgeschlossene Arbeit werden 8 ECTS angerechnet. Dies entspricht einer Arbeitsleistung von 240 Stunden pro Student. Bei einer 14-wöchigen Laufzeit sind dies ca. 2 Arbeitstage pro Woche.

***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 (spätestens in der Mitte des Projektes) 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 (oder auf einer vergleichbaren Umgebung). Stellen Sie sicher, dass der Betreuer jederzeit Zugriff auf das Repository hat und dass das Projekt anhand des Repositories jederzeit wiederhergestellt werden kann.
- Achten Sie auf die Einhaltung guter Programmier- und Designprinzipien
- 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.

***Form der Dokumentation:***

- Bericht (Struktur gemäss Beschreibung) in Ordner(1 Exemplar für HSR)
- Alle Dokumente und Quellen der erstellten SW auf CD, CD's sauber angeschrieben (2 Ex.).

***Fortschrittsbesprechung:***

Regelmässig findet zu einem fixen Zeitpunkt eine Fortschrittsbesprechung statt.
Teilnehmer: Dozent und Studenten, bei Bedarf auch Vertreter der Auftraggeber
Termin: nach Vereinbarung, 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. Sept.2010
Hansjörg Huser

## 2. Erklärung über die eigenständige Arbeit

**Erklärung**

Ich erkläre hiermit,

- dass ich die vorliegende Arbeit selber und ohne fremde Hilfe durchgeführt habe, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit dem Betreuer schriftlich vereinbart wurde,
- dass ich sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben habe.

Rapperswil, 23.12.2010:

Clemens Meier Silvan Gacond

## 3. Abstract

# Kurzfassung der Studienarbeit

| Abteilung | **Informatik** |
|---|---|
| **Name[n] der Studierenden** | **Clemens Meier**<br>**Silvan Gacond** |
| **Studienjahr** | **HS 2010** |
| **Titel der Studienarbeit** | **SURVEAST – Technologiestudie mit SharePoint** |
| ***Examinatorin / Examinator*** | **Prof. Hansjörg Huser** |
| ***Themengebiet*** | **Software** |
| ***Projektpartner*** | **Kantonsspital St.Gallen, Abteilung Infektologie** |
| ***Institut*** | **Institut für vernetzte Systeme** |

**Kurzfassung**

**In der Medizin sind Infektionen, welche während stationären Behandlungen auftreten, ein grosses Problem. Mit SURVEAST soll eine Surveillance Plattform geschaffen werden, welche mögliche Ursachen von Infektionen sowie deren Eintritt so erfasst, dass sie systematisch ausgewertet werden können.**

**Am Kantonsspital St.Gallen wurde der Grundsatzentscheid gefällt, neue Applikationen grundsätzlich auf der Microsoft SharePoint Plattform zu entwickeln. Dies betrifft auch SURVEAST.**

**Das ursprüngliche Ziel der Arbeit, ein Pilotmodul zu SURVEAST zu entwickeln wurde während der Arbeit im Einverständnis aller Beteiligten zu Gunsten einer Technologiestudie abgeändert. Dazu wurden mögliche Integrationsmöglichkeiten abgewogen und beleuchtet. Daraus entstand ein Technologieprototyp mit voll integrierter Architektur welche auf den Business Connectivity Services von SharePoint aufbaut und die damit verbundenen Probleme und Schwierigkeiten aufzeigt. Zusätzlich wurde die entstandene Lösung mit anderen Technologien verglichen, dies um aufzuzeigen wie erheblich der tatsächliche Mehraufwand durch die SharePoint Integration wirklich ist.**

**Das Gesamtprojekt SURVEAST wird im Anschluss an die Arbeit am Kantonsspital nun öffentlich ausgeschrieben. Die Arbeit dient als Basis, die Einschränkung auf die SharePoint Technologie nochmals zu überdenken und gegenüber dem Informatikdienst und der Geschäftsleitung des Spitals den Mehraufwand aufzuzeigen.**

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL
INFORMATIK

# SURVEAST

## Technologiestudie mit SharePoint

---

## Technischer Bericht

Semesterarbeit HS 2010 / 2011
Datum: 23-12-2010

Clemens Meier (cmeier@hsr.ch)
Silvan Gacond (sgacond@hsr.ch)

# 4. Einführung

## 4.1. Management Summary

In der Medizin sind Infektionen, welche während einer stationären Behandlung auftreten, ein grosses Problem. Die Infektionsrate ist trotz modernster Mittel nach wie vor sehr gross. Die Verantwortlichen der Infektiologie des Kantonsspitals St.Gallen möchten darum mit einer Software die Möglichkeit haben, aufgetretene Infektionen sowie deren mögliche Ursachen systematisch zu erfassen und auszuwerten. Aus diesem Problem heraus entstand die Idee der Surveillance Plattform SURVEAST. Vor einiger Zeit wurde diese Idee konkretisiert und ein Konzeptpapier ausgearbeitet. Dies geschah in Zusammenarbeit mit einer Softwarefirma, welche mit den Verantwortlichen des Spitals Anforderungen für das Gesamtsystem aufnahm und ein Konzeptdokument dazu geschrieben hat. Die Grundidee dieses Konzeptes ist, SURVEAST in Zukunft zu einer umfassenden Gesamtlösung ausbauen zu können, welche im Spitalverbund Ostschweiz eingesetzt werden könnte.

Die Geschäftsleitung und die IT-Abteilung des Spitalverbundes fällten zusätzlich den Strategieentscheid, ihre zukünftigen Softwareprojekte auf die sich im Aufbau befindende Microsoft SharePoint 2010 Plattform aufzubauen. Erste kleinere Projekte wurden bereits damit realisiert und der Vorteil aus Sicht der IT Abteilung liegt durch eine einfache Installation und Wartung klar auf der Hand. Dies hat zur Folge, dass eine SharePoint Integration für das Projekt SURVEAST eine klare Anforderung darstellt.

Um eine Benutzeroberfläche zur Verfügung stellen zu können, welche sich in SharePoint einbinden lässt, sowie die möglichst einfache und schnelle Erfassung von möglichen Parametern für die gewünschte Auswertung ermöglicht, wurden verschiedene Möglichkeiten abgewogen. Grundsätzlich sollte eine solche Oberfläche so intuitiv und einfach sein, dass die Erfassung möglichst leicht fällt. Ist die Erfassung eine zu grosse Hürde, werden automatisch zu wenig Daten erfasst und das Ergebnis der Auswertungen wird weniger Aussagekräftig. Darum wurde sehr früh schon entschieden, die Erfassung und später auch die Auswertung, welche eine eher komplexe Konfigurationsoberfläche erfordert, mit der Microsoft Silverlight Technologie zu realisieren. Sie ermöglicht die Realisierung von komplexeren und agilen Benutzeroberflächen für den Web-Browser.

Bezüglich der Integration einer Applikation in SharePoint gibt es einige verschiedene Ansätze. Der technisch sinnvollste und einfachste wäre, die Applikation möglichst losgelöst von SharePoint zu entwickeln und ausschliesslich die Oberfläche der Applikation innerhalb eines SharePoint Webparts zu betreiben. Dies erfordert aber neben der normalen SharePoint Infrastruktur zusätzliche Komponenten und viele der betriebstechnischen Vorteile von SharePoint wären hinfällig. Darum wurde von den Informatikdiensten des Kantonsspitals ganz klar entschieden eine vollständige Integration in SharePoint anzustreben. Um diese Anforderung zu erfüllen wurde der Weg eingeschlagen mit den Business Connectivity Services von SharePoint externe Daten in Form einer externen Datenbank einzubinden.

Während der Entwicklung des ersten Prototyps wurde schnell klar, dass sich das geplante Pilotmodul nicht innert der gewünschten Zeit realisieren lässt. Die Hürden welche die vollständige SharePoint Integration stellte, erwiesen sich als erheblich. So wurde während der Projektarbeit das Gespräch mit den Verantwortlichen gesucht und ihnen zwei Vorschläge unterbreitet. Entweder soll die Funktionalität im Vordergrund stehen und die Anforderung mit dem SharePoint fallen gelassen werden oder man konzentriert sich auf die Integration in SharePoint und erstellt einen Technologieprototyp mit Dokumentation, welcher alle Punkte und Probleme zeigt sowie eine Beispielarchitektur vorgibt, wie das Projekt unter diesen Umständen realisierbar wäre. Die Verantwortlichen entschieden sich für diese Lösung, da eine andere Technologie von dem Informatikdienst nicht akzeptiert würde.

Am Kantonsspital wurde während des weiteren Verlaufs entschieden, mit dem Projekt einen neuen Weg einzuschlagen. Durch die Dimension der ganzen SURVEAST Projekts sowie die Probleme mit der SharePoint Integration nahm das Projekt ein Ausmass an, welches die Infektiologie nicht mehr intern zu tragen vermochte. So wurde entschieden, das Gesamtprojekt SURVEAST auf offiziellem Weg auszuschreiben. Um aber gegenüber der Informatikdienste die Probleme mit SharePoint aufzeigen zu können, sollte die Technologiestudie wie besprochen weitergeführt werden. Der

Mehraufwand der Entwicklung eines solchen Projekts steht nicht im Verhältnis zu dem Nutzen der SharePoint Integration.

Zusätzlich zum Aufzeigen der Probleme mit der Integration in SharePoint wurde in der Technologiestudie zum Schluss noch ein weiterer Punkt beleuchtet. Microsoft bietet nämlich als andere Plattformlösung für solche Applikationen mit der Dynamics Produktreihe eine weitere Lösung an, welche sich mit diesem Beispiel sehr gut mit SharePoint vergleichen lässt und die stärken und schwächen beider Lösungen aufzeigt.

Abschliessend kann gesagt werden, dass es mit Microsoft SharePoint 2010 zusammen mit den Business Connectivity Services möglich ist eine Applikation mit eigenständigem Datenmodell darauf zu implementieren. Die verwendete Architektur ist erweiterbar und bietet grundsätzlich die Voraussetzung dazu. Der Aufwand aber, steht in keinem Verhältnis zu den Vorteilen, welche die SharePoint Plattform bietet, ihre Stärken liegen ganz klar in einem anderen Bereich. Anders sieht die Situation aus wenn eine Applikation zusätzlich die Stärken der kollaborativen Dokumentenverwaltung in SharePoint ausnützen soll. Ist dies aber nicht der Fall, macht eine Integration einer eigenständigen Applikation in SharePoint keinen Sinn.

# 5. Anforderungsanlyse

Die nachfolgende Analyse zeigt die ursprünglich geplanten Anforderungen an das Pilotmodul von SURVEAST. Da die Zielsetzung der Studienarbeit während es Verlaufs des Projektes geändert wurde, wurde diese Analyse nicht mehr weiter angepasst. Sie zeigt lediglich die ursprüngliche Idee des Projektes und die Anforderungen, welche im weiteren Verlauf des SURVEAST Projektes gestellt werden.

## 5.1. Allgemeine Beschreibung

### 5.1.1. Ausgangslage

Im Spital erworbene Infektionen und multiresistente Keime sind eine grosse Herausforderung der modernen Medizin. Die Überwachung am KSSG im Sinne eines Registers aufgetretener Infekte mit ermittelten oder vermuteten Hintergründen und Folgen ist ungenügend und basiert auf Einzelinitiativen.

### 5.1.2. Produkt Funktion

Das Projekt „SURVEAST“ soll eine Surveillance, also eine systematische Erfassung der Verlaufsdaten der Patienten ermöglichen. Anhand dieser Daten sollen Rückschlüsse gewonnen werden, welche zur Vermeidung solcher Infektionen beitragen sollen.

Schlussendlich sollen konkret die Häufigkeit und die Umstände des Auftretens von Infektionen mit deren Zusammenhängen und Risikofaktoren aufgezeigt werden können.

Speziell Patienten mit myeloablativen Therapien wie Hochdosis-Chemotherapie mit autologem Stammzellersatz oder Behandlungskonzepte von akuten Leukämien haben ein erhöhtes Risiko für Infektionen mit komplizierten Verläufen und hoher Sterblichkeit. Darum werden mit dem zu entwickelnden Pilotmodul speziell die Behandlungsdaten der Onkologie in diese Erfassungen einbezogen.

Es sollen in diesem Modul im speziellen Risikofaktoren, welche während Chemotherapien zu Infekten führen deutlich gemacht und die Massnahmen zur Verminderung möglicher Infekte überprüft werden können.

### 5.1.3. Weiterentwicklung

Das SURVEAST Pilotmodul „Surveillance myeloablativer Therapien bei hämatoonkologischen Patienten“, welches aus der Semesterarbeit von Silvan Gacond und Clemens Meier entsteht, soll später zu SURVEAST, einer umfassenden Lösung für die Erfassung und Auswertung von verschiedensten Therapien in den verschiedenen Standorten der Spitalregion St. Gallen ausgebaut werden.

### 5.1.4. Wartung

Die Wartung des Systems wird durch den Informatikdienst des Kantonsspitals vorgenommen. Es ist deshalb ein enger Kontakt mit dem Informatikdienst während der Entwicklung des Pilotprojektes erwünscht.

### 5.1.5. Benutzercharakteristik

Das System wird abteilungsübergreifend von Ärzten und administrativem Personal der Onkologie/Hämatologie und Infektiologie/Spitalhygiene und Plegepersonal zur Dateneingabe verwendet. Standardmässige Auswertungen sollen direkt durch die Ärzte eingesehen werden können.

## 5.2. Spezifische Anforderungen

Die Anforderungen an das Modul basieren auf den Konzeptdokumenten des Kantonsspital St.Gallen bezüglich des geplanten Gesamtsystems „SURVEAST“ sowie des zu entwickelnden Moduls zur Surveillance myeloablativer Therapien bei hämatoonkologischen Patienten.

### 5.2.1. Funktionale Anforderungen

#### *5.2.1.1. Datenerfassung*

Ziel der Erfassung der Patientendaten und Behandlungsdaten ist die Nachverfolgbarkeit des Verlaufes von aufgetretenen Infekten während einer Behandlung im Kantonsspital und deren Resultat. Im Modul zur Surveillance myeoablativer Therapien werden dazu die Behandlungsdaten der Onkologie erfasst und in die Auswertung beigezogen.

Die gesamte Patientengeschichte beginnt mit Geburt und kann auch Daten über Ereignisse umfassen, die bereits vor dem Spitalaufenthalt respektive vor der eigentlichen Therapie stattgefunden haben. Als Ereignisse werden alle Diagnosen, Therapien etc. definiert die mit dem Auftreten von Infektionen und deren Behandlungen zusammenhängen.

Um Zusammenhänge von infektologischen Ereignissen bei Patienten während Chemotherapien aufzuzeigen, werden Start und Ende eines Chemotherapiezyklus mit der daraus erfolgten Aplasie (Aussetzen der Bildung der Blutzellen) erfasst. Infektiologische Ereignisse können anhand des Datums diesen Zyklen zugeordnet werden. Die Daten müssen dazu interdisziplinär zwischen Oknologie/Hämatoologie und Infektiologie/Spitalhygiene erfasst und eingesehen werden können.

Eine Übersicht über die zu erfassenden Daten ist im Kapitel 5.3.1 zu finden.

#### *5.2.1.2. Auswertung*

Die Auswertung der Daten soll Rückschlüsse ermöglichen, unter welchen Umständen die einzelnen Infekte aufgetreten sind. Dazu werden die Daten aus der Onkologie sowie die vorangegangenen Daten der Infektiologie verwendet. So kann ermittelt werden, welche vorangegangenen Behandlungen, Messdaten oder Falldaten wie oft zu welchen Infektionen führen.

Einfaches Beispiel: Patienten die während einer Chemotherapiebehandlung in einem Viererzimmer untergebracht sind, erleiden häufiger eine Pilzinfektion als Patienten die einzeln untergebracht sind.

### 5.2.2. Erweiterbarkeit

Die Menge der zu erfassenden Daten muss zwar erweiterbar sein, sollte aber im Grundgerüst weitgehend konstant bleiben, damit die Daten auch sinnvoll ausgewertet werden können.

### 5.2.3. Schnittstellen

Um die Stammdaten konsistent zu halten, sollen Schnittstellen zu Drittsystemen ermöglicht werden. Welche Drittsysteme angeschlossen werden können ist zu diesem Zeitpunkt noch nicht genau definiert, das Kantonsspital verfügt aber über einen Schnittstellenservice der die Daten aus den Drittsystemen aufbereiten kann, damit sie sich einfach importieren lassen. Die genaue Spezifikation der Schnittstellen erfolgt zu einem späteren Zeitpunkt.

## 5.3. Struktur

### 5.3.1. Strukturierung der Daten

#### *5.3.1.1. Administrative Daten*

Die Administrativen Daten können folgendermassen gruppiert werden.

**Stammdaten:** Die Stammdaten werden einmal pro Patient erfasst und umfassen Daten zum Patienten wie Geburtsdatum oder Patientennummer.

**Fall**: Ein Patient erhält mehrere Fallnummern (meist eine pro Jahr). Ereignisse werden auf eine bestimmte Fallnummer gebucht.

#### *5.3.1.2. Ereignis*

Ereignisse haben immer einen Zeitbezug und können kontinuierlich aufgenommen werden. Anstatt eines einzelnen Datums können auch Zeiträume angegeben werden.
Eine Gruppierung der Ereignisdaden ermöglicht später eine automatische Auswertung der Daten nach Regelmässigkeiten. Regelmässigkeiten von Handlungen, welche oft auf gewissen Ereignisse folgen können so automatisch ausgeschlossen werden.

Ereignisse werden folgendermassen gruppiert:

**Zustand (passives Ereignis):** Zustände sind Symptombeschreibungen oder Messdaten und können Daten von Routinemässigen Messungen oder Handlungen wie Messung der Körpertemperatur, des Blutdruckes oder Messdaten von einmaligen Untersuchungen enthalten. Im Modul Onkologie können dies zum Beispiel Neutropiemessungen sein. Zustände können zu vordefinierten, standardmässigen Gruppen von Zuständen gebündelt werden und mit Diagnosen oder Behandlungen verknüpft werden um zum Beispiel eine Wirkung einer Handlung aufzuzeigen.

**Diagnose (analysiert):** Diagnosen beinhalten Daten zu belegten oder vermuteten Infekten oder anderen Erkrankungen. Es sind also Schlüsse aus Zuständen und vorgehenden Handlungen. Zum Beispiel könnte dies die Angabe eines bestimmte Bakterienstammes bei einer Bakteriämie (Bakterien im Blut) sein oder die Art der Leukämie bei einem Chemotherapiepatienten. Diagnosen enthalten Verknüpfungen zu Zuständen.

**Handlung (aktives Ereignis):** Handlungen sind Massnahmen, welche *bewusst ausgeführt* wurden um ein bestimmtes Ziel zu erreichen. Es können Daten zu prophylaktische Behandlungen oder Behandlung einer diagnostizierten Krankheit (z.B. spezieller bakterieller Infekt) sein. Zu den Behandlungsdaten gehören zum Beispiel Daten zur medikamentösen Behandlung mit der angewendeten Dosis oder speziell bei der Onkologie Angaben zu einem während der Behandlung verwendeten Katheter. Eine Handlung kann mit einer Diagnose verknüpft werden um den Zusammenhang aufzuzeigen. Stationäre Behandlungen enthalten ausserdem Datumangaben zur Einlieferung und Entlassung sowie Angaben zum Zimmer.

### 5.3.2. Domain Model

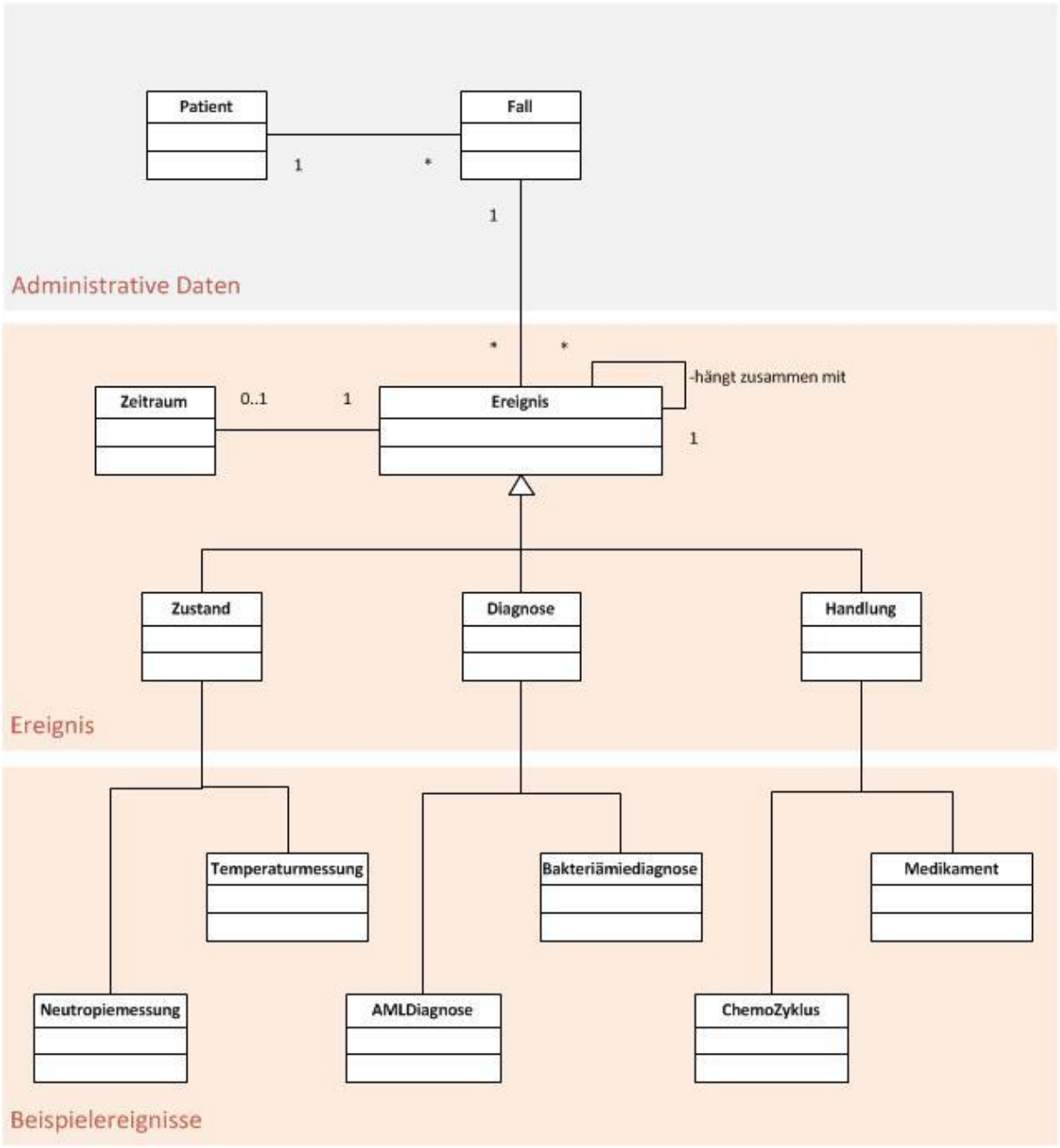



Abbildung 1: Domain Model

#### *5.3.2.1. Bemerkungen zum Domain Model*

Es wurden bewusst nicht alle möglichen Ereignisse im Domain Model aufgeführt, da diese einerseits zum jetzigen Zeitpunkt noch nicht definiert sind und andererseits den Rahmen für eine gute Übersicht sprengen würden.

In einem ersten Schritt soll grundsätzlich nur das Grundmodell entwickelt werden, genauere Ereignisse variieren von Modul zu Modul.

### 5.3.3. Darstellung der Daten auf der Zeitachse (Beispielzyklus)

Die nachfolgende Darstellung zeigt den zeitlichen Ablauf der Datenerfassung anhand eines Beispiels für einen Behandlungszyklus. Ereignisse können auch ausserhalb des Zyklus auftreten und dort erfasst werden.

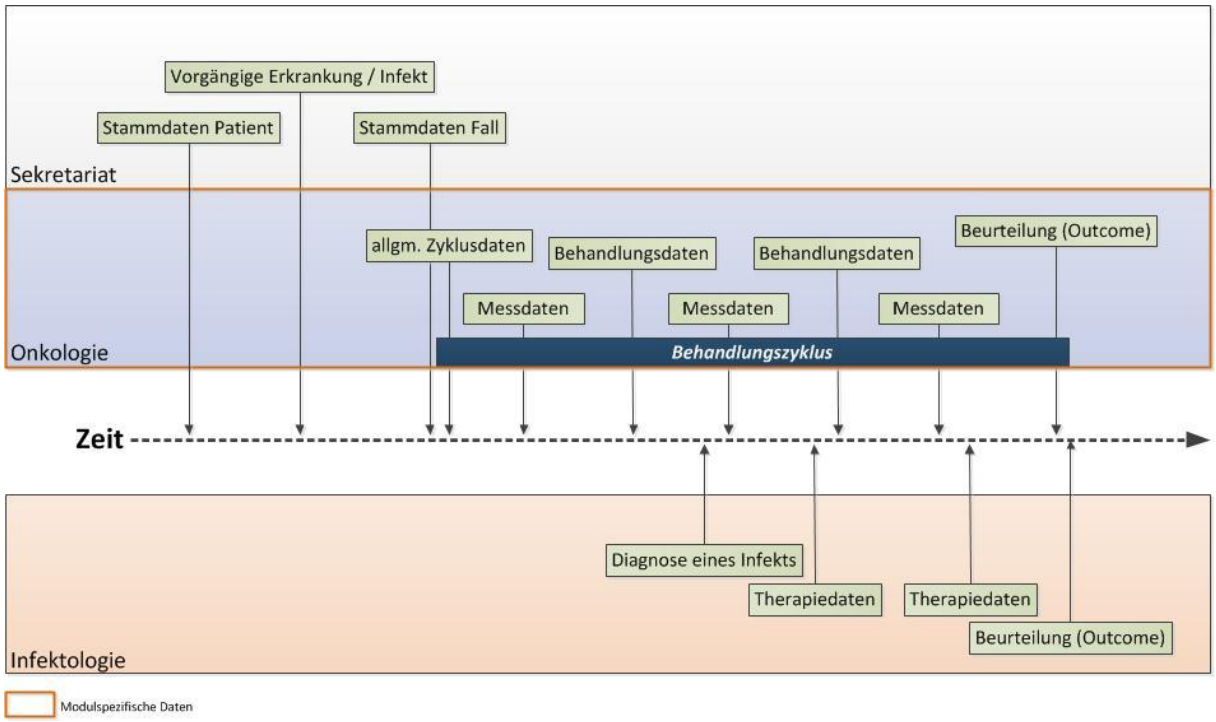



Abbildung 2: Ereignisse auf der Zeitachse dargestellt

## 5.4. Use Cases

### 5.4.1. UC1: Administrative Daten erfassen / aktualisieren

| | |
|---|---|
| Umfang | SURVEAST |
| Ebene | Anwenderziel |
| Primärakteur | Administratives Personal<br>Arzt/Ärztin (alternativ, falls nicht in Sekretariat erfasst) |
| Stakeholder und Interessen | |
| Administratives Personal:<br>• Will schnelle Eingabe ohne grossen Zusatzaufwand.<br>Arzt/Ärztin Infektiologie & Onkologie:<br>• Will korrekte und komplette Daten welche zur Analyse beigezogen werden können. | |
| Vorbedingungen | Anwender wird identifiziert und authentifiziert |
| Nachbedingungen | Die Daten Stammdaten sind gespeichert und können für weitere Datenerfassungen oder Auswertungen verwendet werden. |
| Standardablauf | |
| 1. Akteur erhält den Auftrag, Stammdaten zu erfassen oder zu ändern.<br>2. Akteur überprüft ob Patient in der Datenbank bereits erfasst ist.<br>3. Akteur überprüft Patienten Id. Wurde der Patient noch nie im Kantonsspital behandelt, so muss eine neue Patientennummer im SAP generiert und in SURVEAST übernommen werden. Dazu müssen Name, Vorname, Geburtstag und Geschlecht aus SAP übernommen werden.<br>4. System fügt die neuen Stammdaten in das System ein oder speichert die geänderten Daten.<br>5. Wurde der Patient im laufenden Jahr noch nicht behandelt, muss eine neue Fallnummer im SAP generiert werden, welche in SURVEAST übernommen werden muss.<br>6. Die Fallnummer wird (sofern benötigt) gespeichert. | |
| Liste der Technik- und Datenvariationen | |
| 1. Stammdaten könnten (sofern erlaubt) aus mit einer Datenschnittstelle aus SAP importiert werden. | |
| Häufigkeit des Auftretens | Einmal pro Patient<br>Fallnummer jährlich |
| Offene Fragen: | |
| Datenimport SAP | |

### 5.4.2. UC2: Patientengeschichte suchen / analysieren

| | |
|---|---|
| Umfang | SURVEAST |
| Ebene | Subfunktion |
| Primärakteur | Arzt/Ärztin |
| Stakeholder und Interessen | |
| Arzt/Ärztin Infektiologie:<br>• Will sich eine Übersicht über die Patientengeschichte verschaffen | |
| Vorbedingungen | 1. Anwender wird identifiziert und authentifiziert<br>2. Stammdaten des betreffenden Patienten sind erfasst |
| Standardablauf | |
| 1. Akteur gibt die Patientennummer des untersuchten Patienten ein und gelangt zur Übersicht über die Daten des betreffenden Patienten<br>a. Alternativ kann der Patient über eine Suche nach dem Namen gefunden werden.<br>2. Akteur gewinnt eine Übersicht über die erfassten Daten zum Patienten | |
| Häufigkeit des Auftretens | Laufend |

### 5.4.3. UC3: Ereignis erfassen

| | |
|---|---|
| Umfang | SURVEAST |
| Ebene | Anwenderziel |
| Primärakteur | Arzt/Ärztin |
| Stakeholder und Interessen | |
| Arzt/Ärztin Infektiologie:<br>• Will schnelle Eingabe ohne grossen Zusatzaufwand.<br>• Will komplett dokumentierte Patientengeschichte für spätere Auswertung<br>• Will Zusammenhänge zwischen Handlungen und infektiösen Ansteckungen genau erkennen und durch die Daten belegen können.<br>Arzt/Ärztin Onkologie: | |

| | |
|---|---|
| • Will eine Verminderung von Risikofaktoren bei infektiösen Ansteckungen während der Chemotherapie. | |
| Vorbedingungen | 1. Anwender wird identifiziert und authentifiziert<br>2. Stammdaten des betreffenden Patienten sind erfasst<br>3. Die Patientengeschichte des betreffenden Patienten ist geöffnet |
| Nachbedingungen | Die Ereignisdaten sind gespeichert und können für weitere Auswertungen verwendet werden. |
| Standardablauf (Für alle Ereignistypen gleich) | |
| 1. Akteur wählt die Fallnummer welcher er das Ereignis zuordnen will.<br>2. Akteur erfasst die Ereignisdaten (In den folgenden Unterusecases detailiert beschrieben)<br>3. Zeitliche Angaben<br>a. Falls in der Vergangenheit, wird das Datum angegeben.<br>b. Falls nötig wird ein Zeitraum zum Ereignis angegeben.<br>4. System speicher die Daten und fügt das aktuelle Datum ein, falls keine Angabe zum Datum gemacht wurde. | |
| Häufigkeit des Auftretens | Laufend während eines Behandlungszyklus |

#### *5.4.3.1. UC3-1: Zustand erfassen*

| Unterscheidung |
|---|
| Zustände sind Daten welche den Zustand eines Patienten mit klaren Fakten (ohne Diagnose) beschreiben. Dazu gehören z.B. Labordaten oder Messdaten. |
| Spezielle Interessen |
| Arzt/Ärztin:<br>• Benötigt die Daten später zur Diagnose |
| Vorbedingungen |
| Akteur hat Daten während einer Untersuchung eines Patienten ermittelt oder aus einer Laboruntersuchung ermittelte Daten erhalten. |
| Standardablauf (Abweichung von Standard UC2) |
| Speziell bei Schritt 4 UC2 (Ereignis erfassen):<br>1. Akteur wählt Typ des zu erfassenden Zustandes (Bsp: Körpertemperatur) |

#### *5.4.3.2. UC3-2: Diagnose erfassen*

| Unterscheidung |
|---|
| Diagnosen sind Annahmen und Rückschlüsse welche während einer Untersuchung oder bei der nachträglichen Analyse ermittelt werden. |
| Spezielle Interessen |
| Arzt/Ärztin:<br>• Benötigt die Daten später zur Auswertung |
| Vorbedingungen |
| Akteur hat Gesamtzustand des Patienten analysiert und Schlüsse gezogen. |
| Standardablauf (Abweichung von Standard UC2) |
| Speziell bei Schritt 4 UC2 (Ereignis erfassen):<br>1. Akteur wählt den Typ der zu erfassenden Diagnose (z.B. katheterbedingte Bakteriämie bei Infektiologie, bestimmter Leukämietyp bei Onkologie)<br>2. Akteur verknüpft bereits erfasste Ereignisse (vorhergehende Diagnosen, Zustände und Handlungen) um deren Zusammenhänge und Relevanz für die Diagnosen aufzuzeigen.<br>3. Akteur gibt Relevanz bei den Verknüpfungen an.<br>4. Akteur notiert weitere Annahmen und Rückschlüsse als Freitext. |

#### *5.4.3.3. UC3-3: Handlung erfassen*

| Unterscheidung |
|---|
| Handlungen sind bewusste aktive Einflussnahmen auf den Patienten wie z.B. medikamentöse Behandlungen oder Daten zum Einsatz eines Katheters während einer Chemotherapie. |
| Spezielle Interessen |
| Arzt/Ärztin:<br>• Benötigt die Daten bei der Auswertung um die Wirkung einer Handlung aufzuzeigen |
| Standardablauf (Abweichung von Standard UC2) |
| Speziell bei Schritt 4 UC2 (Ereignis erfassen):<br>1. Akteur wählt den Typ der zu erfassenden Handlung (z.B. Einsatz eines Medikamentes, Daten |

| |
|---|
| zum eingesetzten Katheter während der Chemotherapie)<br>2. Akteur verknüpft Handlung mit dazugehörender Diagnose<br>3. Akteur gibt erhofftes Resultat der Handlung an |

### 5.4.4. UC5: Auswertungen

| | |
|---|---|
| Umfang | SURVEAST |
| Ebene | Anwenderziel |
| Primärakteur | Arzt/Ärztin |
| Stakeholder und Interessen | |
| Arzt/Ärztin Infektiologie:<br>• Will die Erfolge und Risikofaktoren von verschiedenen Behandlungen aufzeigen und mit Daten belegen können.<br>Arzt/Ärztin Onkologie:<br>• Will durch die ermittelten und eliminierten Risikofaktoren für infektöse Ansteckungen die Sterblichkeit von Patienten während Chemotherapien vermindern.<br>Spital:<br>• Ist an einer Qualitätsverbesserung interessiert. | |
| Vorbedingungen | 1. Anwender wird identifiziert und authentifiziert<br>2. Es sind Ereignisse zu mehreren Patienten erfasst.<br>3. Der Arzt hat eine Vermutung über Erfolgs- oder Risikofaktoren einer gewissen Behandlung. |
| Standardablauf | |
| 1. Der Akteur filtert die Ereignisse nach bestimmten Kriterien und wertet diese selbst aus. | |
| Alternative Abläufe | |
| 1. Das System sucht selbst nach Regelmässigkeiten und zeigt diese auf | |
| Häufigkeit des Auftretens | Einmal pro Behandlungsphase |

## 5.5. Entwürfe Benutzeroberfläche

### 5.5.1. Erfassung der Daten

Die Benutzeroberfläche des Prototypen zur Erfassung der Daten soll möglichst einfach, aber benutzerfreundlich gestaltet sein. Pro Ereignis muss die Benutzeroberfläche variieren können. Die Unterscheidung zwischen den verschiedenen Ereignisgruppen (Zustand, Diagnose, Handlung) ist auf dieser Ebene nicht sichtbar. Sie wird durch den gewählten Ereignistyp bestimmt.


**Abbildung 3: Entwurf der Benutzeroberfläche für die Erfassung von Daten**

### 5.5.2. Auswertung erstellen

Für die Erstellung von Auswertungen können Ereignisse per Drag&Drop als Endpunkte oder Modifikatoren definiert werden. Ereignisse können auch mit und Operatoren verknüpft werden. Dazu können zu jedem Ereignis vordefinierte Parameter angegeben werden. Es ist damit für einen Arzt möglich, direkt über die Benutzeroberfläche Hypothesen zum Zusammenhang gewisser Modifikatoren zu gewissen Endpunkten zu stellen und diese gleich mit den Daten aus dem Datenbestand zu belegen.

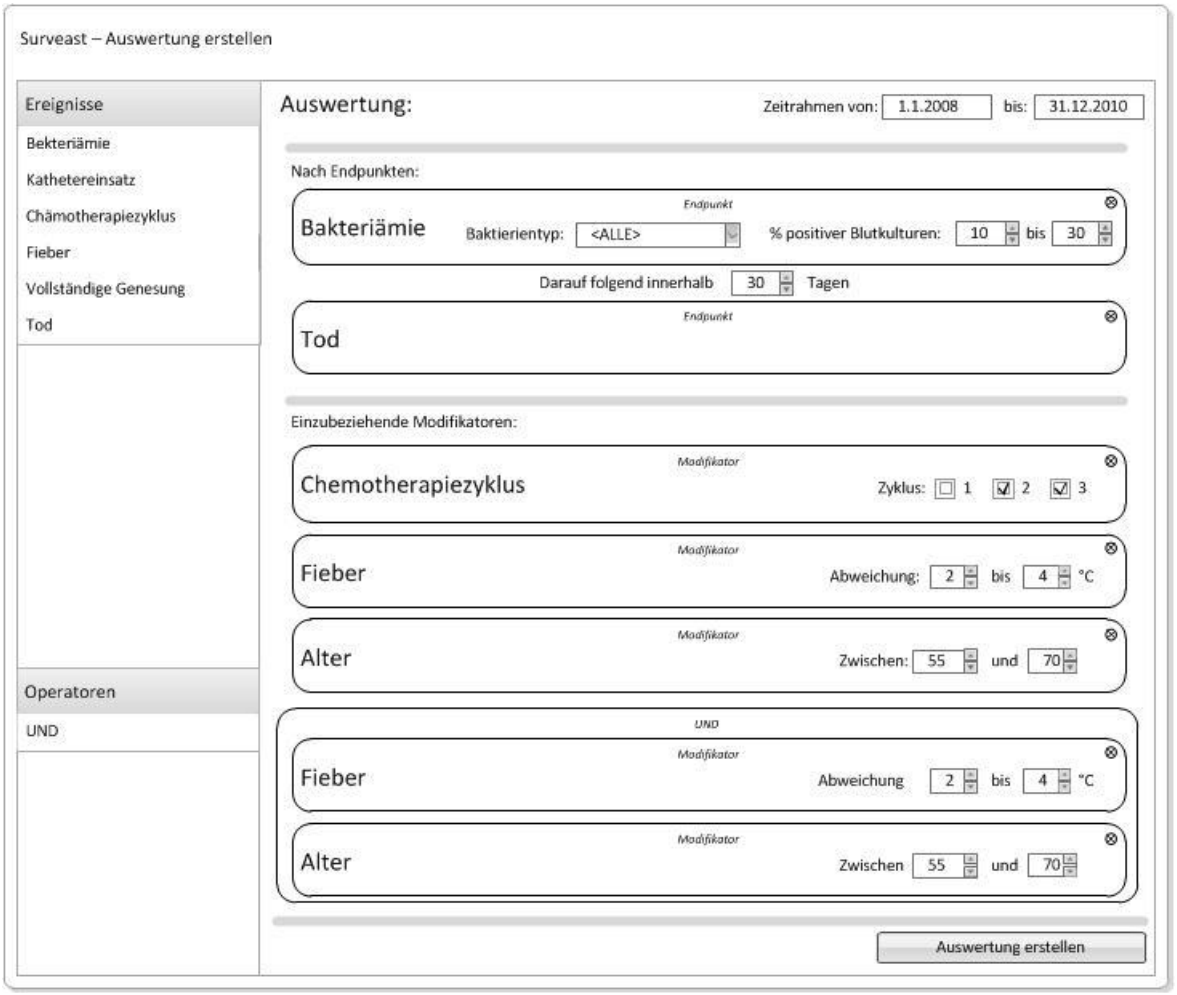



**Abbildung 4: Entwurf für die Benutzeroberfläche für die Erstellung von Auswertungen**

### 5.5.3. Auswertung

Eine Auswertung könnte folgendermassen aussehen:

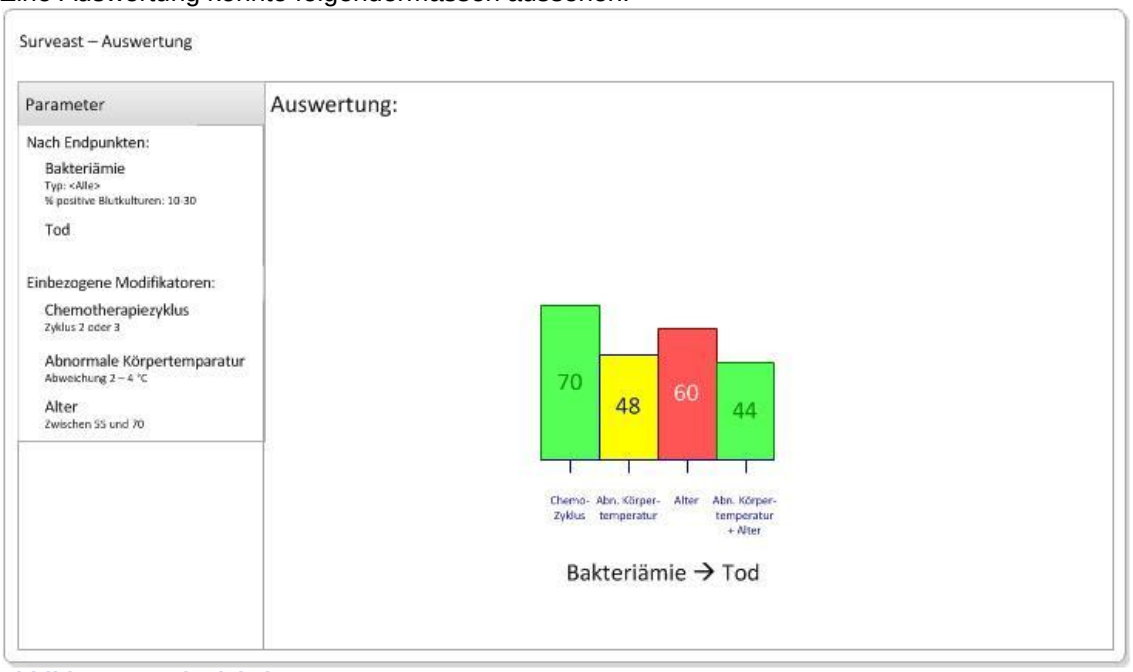



**Abbildung 5: Beispiel einer Auswertung**

# 6. Architekturfindung

## 6.1. Ziele der verwendeten Architektur

Die verwendete Architektur soll eine Erfassung strukturierter Daten ermöglichen, aus deren Auswertung sinnvolle Schlüsse gezogen werden können. Sie darf keine internen Hürden oder Hindernisse bei grossen Datenmengen darstellen.

Besonders die verschiedenen zu erfassenden Ereignisse (siehe Anforderungsanalyse 5) können strukturmässig stark variieren und müssen auch in Zukunft einigermassen einfach (mit möglichst wenig Programmieraufwand) erweiterbar sein.

Da sich eine möglichst automatisch auswertbare, klar definierte und gleichzeitig möglichst flexible Datenstruktur teilweise widersprechen, muss hier ein sinnvoller Kompromiss gefunden werden.

## 6.2. Einschränkungen

Das Kantonsspital St.Gallen hat mit der Integration von SharePoint 2010 in ihre IT-Infrastruktur den Strategieentscheid gefällt, sämtliche neu zu entwickelnde interne Applikationen auf dieser Plattform zu betreiben. Dies erspart den Informatikdiensten des Spitalverbundes einerseits Administrationsaufwand, andererseits können sie so auf konzentriertes Know-How setzen. Dies hat zur Folge, dass die Applikation SURVEAST möglichst komplett in SharePoint integriert werden soll.

## 6.3. Architekturvarianten

### 6.3.1. SharePoint Silverlight Webpart mit RIA Services

Um die Flexibilität und Einfachheit der Architektur möglichst hoch zu halten, wurde folgende Lösung evaluiert:
SharePoint wird lediglich als Hosting Plattform für die in Microsoft Silverlight implementierte Benutzeroberfläche verwendet. Die restlichen Schichten der Applikation werden in einem Silverlight RIA Services Webservice implementiert, welcher mit der Silverlight Client Applikation kommuniziert.

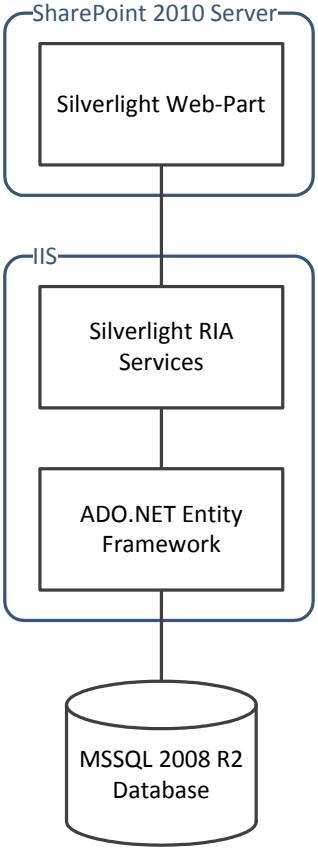



**Abbildung 6: Architekturübersicht der Silverlight / RIA Lösung**

RIA Services ist im Silverlight 4 SDK integriert, welches ein Visual Studio Projekt Template „Silverlight Business Application“ enthält. Dieses Template ermöglicht eine effiziente Entwicklung von datenlastigen verteilten Webapplikationen mit Silverlight.

Um die erforderte Sicherheit zu gewährleisten, wird die ganze Authentifizierung auf dem Webserver (Microsoft Internet Information Services - IIS) mit Integrated Security über das bestehende Active Directory (Windows Authentifizierung) realisiert.

Die Silverlight-Komponente welche in einem SharePoint Webpart bereitgestellt wird, kommuniziert via Windows Communication Foundation (WCF) mit den RIA Services Komponenten welche auf IIS installiert und betrieben werden. ADO.NET Entity Framework kann mit RIA Services verwendet werden, was eine einmalige Implementierung der Entity Objekte und eine transparente Kommunikation zwischen Silverlight Client und der Webservice Komponente bei voller Typensicherheit ermöglicht.
Dadurch wird der Aufwand für zukünftige Erweiterungen der Applikation gering gehalten, was einer wichtigen Anforderung an das Gesamtsystem entspricht.

Da die ganze Applikation SharePoint nur als Einstiegsplattform verwendet, wird die Integration der Datenstruktur in SharePoint Listen hinfällig.

Diese Lösung hat aber den Nachteil, dass der Webservice in einem eigenen Prozess in IIS betrieben werden muss. Ausserdem ist für Entity Framework 4.0 und RIA Services ist eine .NET 4 Runtime notwendig, welche eingerichtet und betreut werden muss. Microsoft SharePoint 2010 verwendet die .NET 3.5 Runtime.

Dies widerspricht dem Konzept der integrierten SharePoint Applikationen welche auf der gegebenen SharePoint Infrastruktur bereitgestellt und zentral betreut werden können und wird so vom Informatikdienst des Kantonspital St.Gallen nicht bewilligt.

#### 6.3.2. Silverlight WebPart mit in SharePoint integriertem WCF Service

Als Variante zu obiger Implementation wäre es denkbar, die RIA Services durch einen konventionellen WCF Service zu ersetzen. Dies wäre zwar in der Implementation aufwändiger und die Definitionen der Schnittstellen würden nicht mehr durch das RIA Services Framework automatisiert, dafür könnte man den WCF Service innerhalb SharePoint betreiben. Dies wäre zwar keine vollständige Integration, hätte aber den Vorteil der möglichen Auslieferung als SharePoint Solution.

#### 6.3.3. Silverlight WebPart mit vollständiger Integration der Daten in SharePoint

Um eine Applikation komplett in SharePoint zu integrieren, muss die Datenhaltung entweder in internen Listen erfolgen oder es müssen mit den Business Connectivity Services externe Datenquellen in SharePoint integriert werden.

Eine genauere Analyse der Möglichkeiten zur vollständigen Integration in SharePoint ist im Kapitel 7 beschrieben.

### 6.4. Architekturentscheid

Durch die Anforderung der Informatikdienste die ganze Applikation ohne weitere Komponenten ausschliesslich auf dem SharePoint Server zu betreiben war die Entscheidung der zu wählenden Architektur klar:

Die Entwicklung der Applikation hat als vollständig integrierte Lösung zu erfolgen. Als einzige eigenständige Komponente wurde eine eigene SQL Server Instanz in Betracht gezogen.

# 7. Möglichkeiten zur vollständigen Integration in SharePoint auf Datenebene

Um ein Datenmodell vollständig in SharePoint zu integrieren existieren grundsätzlich zwei Möglichkeiten. Einerseits ist es möglich, die Daten mit internen Listen komplett von SharePoint selbst verwalten zu lassen, andererseits können mit Hilfe der Business Connectivity Services (BCS) externe Systeme wie zum Beispiel eine Datenbank oder ein SAP angekoppelt werden.

## 7.1. Interne Listen

Microsoft SharePoint verwendet für die Speicherung von flachen und semistrukturierten Daten ein Konzept namens SharePoint Listen. Diese Listen gleichen im weitesten Sinne einer Datenbanktabelle und werden in der „Content Database“ der SharePoint Site abgelegt. Sie wurden aber in erster Linie für lineare Datenstrukturen konzipiert.

### 7.1.1. Datenbasis

Eine normale SharePoint 2010 Server Installation verwaltet praktisch die gesamten Inhalte mittels Microsoft SQL Server 2008. Dazu werden auf einem zur Installationszeit festgelegten SQL-Server mehrere Datenbanken angelegt. Standardmässig mit einem eindeutigen Namen, also mit dem eigentlichen Datenbanknamen und einer angehängten GUID (global unique identifier, siehe Glossar: Kapitel 14) als Suffix. Eine Ausnahme bildet hier die „WSS_Search_*“ Datenbank die den Namen des Servers als Suffix besitzt. Sie beinhaltet auch ausschliesslich Indizierungsinformationen zum aktuellen Server (in einem Farm Szenario mit mehreren Servern nachvollziehbar). [1]

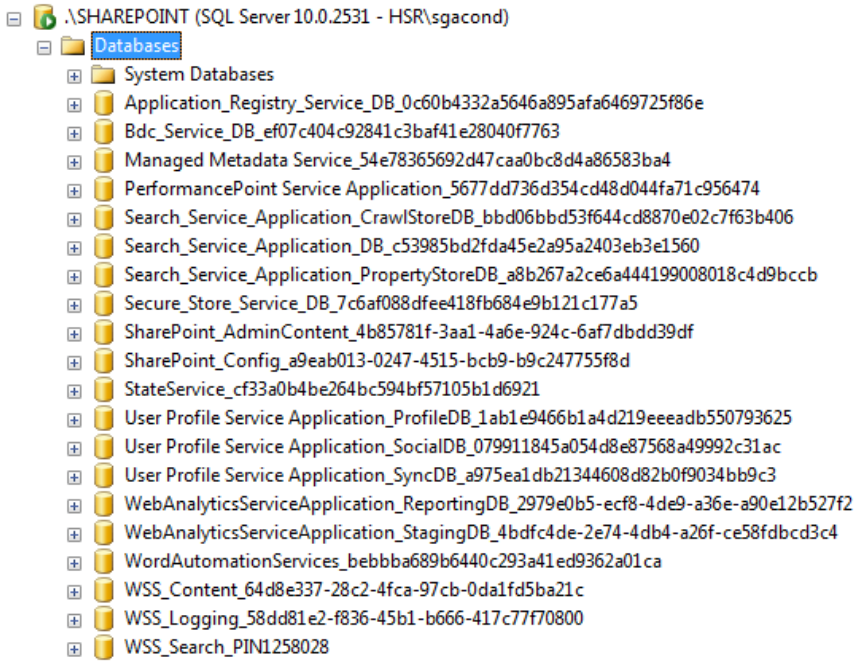



**Abbildung 7: SharePoint Datenbanken mit GUID**

Arbeitet man innerhalb SharePoint mit internen Listen (siehe 7.1), werden diese Daten in der WSS_Content Datenbank abgelegt. Versuche mit dem SQL Server Profiler haben gezeigt, warum diese Art Daten abzulegen bei grösseren Datenmengen nicht performant sein kann. Für einen einfachen Detail-Aufruf eines List Items werden mehrere hundert Anfragen abgesetzt. Dass dieses Prinzip überhaupt so gut funktioniert, ist eigentlich nur der guten Optimierung und der hohen Performance des SQL Servers zu verdanken.

Die WSS_Content Datenbank beinhaltet den ganzen User Content (Lists, Events, Features, Solutions, etc.) und besteht aus 114 Tabellen, teilweise mit verschiedenen Collations verteilt auf verschiedene Tabellen. Teile der Daten sind auch redundant abgelegt um eine Performancesteigerung zu erreichen, der Zugriff erfolgt über StoredProcedures damit diese Redundanzen nicht zu Anomalien führen.

[1] Introduction to the Microsoft SharePoint SharePoint 2010 Database Layer, http://blogs.technet.com/b/wbaer/archive/2009/11/30/introduction-to-the-microsoft-sharepoint-sharepoint-2010-database-layer.aspx, letzter Zugriff 23.12.2010

### 7.1.2. Vorteile

**Rapid Prototyping:**
SharePoint Listen können im SharePoint Designer oder mit den entsprechenden Rechten direkt über den Webbrowser auf dem SharePoint Server zusammengeklickt werden. Funktionalität zum Erstellen, Lesen, Bearbeiten und Löschen von Listendaten kann per Knopfdruck erstellt werden. Auch Workflows können per Mausklick definiert werden. Bei einfachen Anwendungen kommt man so sehr schnell zum Ziel. Ausserdem sind dazu keine Programmierkenntnisse erforderlich. Mit Microsoft InfoPath kann noch ein weiteres Tool beigezogen werden, welches die Erstellung und Verschönerung von SharePoint Formularen erleichtert.

**Flexibilität der Datenstrukturen:**
Da die Daten nicht als fixe Tabellen in der Datenbank abgelegt werden, ist es möglich das List Datenmodell ständig zu erweitern. Allerdings verliert man hier durch die gewonnene Flexibilität meistens die Möglichkeit, die Daten später sinnvoll auszuwerten. Auch Abstriche bei der Performance gibt es aus diesem Grund, da etliche Datenbankaufrufe nötig sind um die Daten aus der Datenbank zu laden.

### 7.1.3. Nachteile

**Performance:**
Kumulative Abfragen auf SharePoint Listen welche zum Beispiel Summen oder Mittelwerte aus grossen Datenmengen berechnen sind sehr ineffizient. Auch Except, Intercept oder GroupBy sind bei grossen Datenmengen ein Problem. Die Verwendung von Linq2SharePoint (Linq Adapter für SharePoint Listen) erleichtert die Erstellung solcher Abfragen, trotzdem müssen aber zuerst alle Daten aus der Datenbank geladen werden, um solche im Prinzip einfache Berechnungen durchzuführen. Allgemein ist Linq2Sharepoint nur für die einfachere Generierung von CAML-Abfragen (Collaborative Application Markup Language) konzipiert und hilft dem Entwickler nicht, Abfragen auf Listen zu optimieren. Bei grossen Datenmengen stösst man hier schnell an die Grenzen und generiert eine hohe Last auf der Farm. Würde man für SURVEAST SharePoint Listen verwenden, müsste mindestens für die Auswertung der Daten eine weitere Lösung entwickelt werden, welche die Daten aus SharePoint exportiert und auswertet.

**Benutzerfreundlichkeit:**
Mit der Benutzerfreundlichkeit kommt man bei generierten List Views und Forms schnell an die Grenzen, wenn man Datenmasken für die Erfassung von Daten mit Beziehungen erstellen will. Es ist ein beachtlicher Mehraufwand erforderlich um die gewünschte Usabiltity zu erreichen.

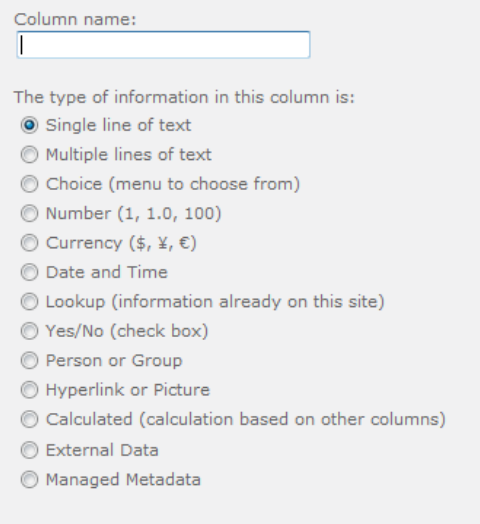


**Abbildung 8: Datentypen für interne Listen**

**Rapid Prototyping:**
Wenn man alternative Möglichkeiten zur Erstellung eines UI's betrachtet, ist Rapid Prototyping auch mit ScetchFlow und Silverlight möglich.

**Flexibilität:**
SharePoint Listen können nur eine kleine Anzahl vorbestimmter Datentypen verwenden (siehe Abbildung). Ausserdem sind Beziehungen zwischen Listen nicht auf Datenebene optimiert und nur für einfache Relationen ausgelegt.

**Wartbarkeit und Testbarkeit:**
Um eine bessere Wartbarkeit und Testbarkeit der Software zu erreichen, ist eine Trennung der Datenstruktur vom Datenprovider notwendig (Repository Pattern[2]). Diese Arbeit wird dem Softwareentwickler bei der Verwendung des .NET Frameworks durch moderne Tools wie das ADO.NET Entity Framework wesentlich erleichtert. Mit SharePoint Listen und SPMetal[3] ist eine solche Trennung zwar auch möglich, aber mit einem grösseren Aufwand verbunden.

[2] Repository Pattern http://msdn.microsoft.com/en-us/library/ee413961.aspx, letzter Zugriff 22.12.2010
[3] Repository Design Pattern mit Microsoft SharePoint Listen und SPMetal: http://msdn.microsoft.com/en-us/library/ff648864.aspx, letzter Zugriff 22.12.2010

**Datenintegrität:**
Transaktionen stellen sicher, dass Daten welche in mehreren Schritten geschrieben werden müssen entweder vollständig oder gar nicht geschrieben werden. So wird die Konsistenz von zusammenhängenden Daten im Datenmodell gewährleistet. SharePoint Listen unterstützen keine Transaktionen, was bei der Anwendung von SURVEAST am Anfang noch vernachlässigbar sein kann. Bei einem breiten Einsatz und einer wachsenden Anzahl von Benutzern welche Daten bearbeiten, wird dies aber zu einem Problem.

**Sicherheit:**
Es besteht keine direkte Möglichkeit die Daten in der Datenbank zu verschlüsseln. Administrative Dokumente und Patientendaten würden an ein und demselben Ort abgelegt. Es existieren Tools von Drittfirmen, welche eine solche Verschlüsselung ermöglichen. Da diese aber in die Content Database eingreifen, erlöschen die Supportansprüche bei Microsoft für diese Installation. Dies ist in einem produktiven Umfeld keine akzeptable Lösung.

### 7.1.4. Schlussfolgerunen

SharePoint Listen sind schon wegen der Performanceprobleme bereits bei einfacheren kumulativen Abfragen (Max, Avg, GroupBy) überfordert für unsere Anwendung. Sinnvolle dynamische Auswertungen wären nur mit vorhergehendem Export der Daten möglich.
Die Analyse der Datenbank und die Versuche mit dem Profiler haben unsere Entscheidung zusätzlich unterstützt, dass wir für unsere datenlastige Applikation unbedingt mit einer externen Datenbank arbeiten sollten.

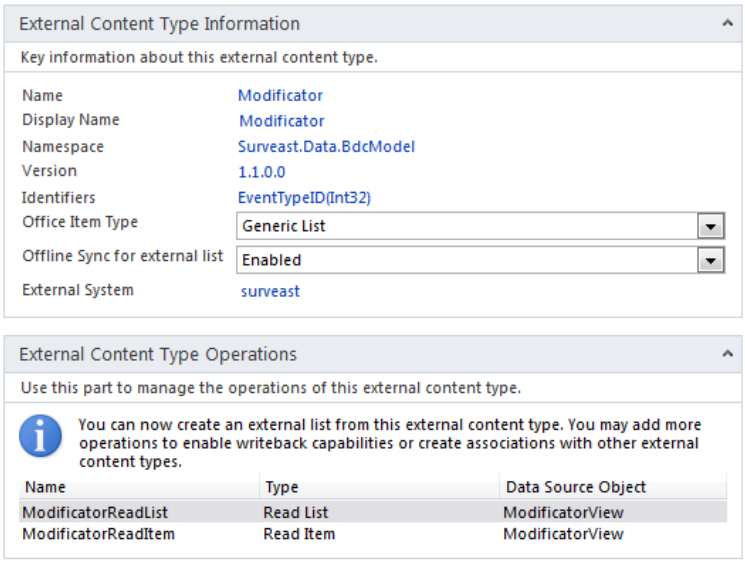
## 7.2. Business Connectivity Services

Um Daten aus einer externen Datenquelle in SharePoint zu integrieren, bietet SharePoint 2010 die sogenannten „Business Connectivity Services“ (BCS). Sie sind eine Weiterentwicklung der „Business Data Catalog“ (BDC) Schnittstelle in SharePoint 2007, welche dazu diente, Daten aus Drittsystemen in SharePoint zu laden und anzuzeigen. In SharePoint 2010 ist es nun möglich mit BCS auch Daten in externe Systeme zu schreiben. BCS unterstützt neben direkten Verbindungen zu MS-SQL Server, OLE-DB Datenquellen und WCF Services (Windows Communication Foundation) auch den Zugriff auf .NET Assemblies, welche ein vordefiniertes Interface implementieren. Damit kann man beliebige Systeme (z.B. SAP) über ihre Serviceschnittstellen ansprechen.

Wir betrachten im Folgenden die Verwendung von BCS mit einem MS-SQL Server, da ein solcher mit der gegebenen Infrastruktur am Kantonspital verwendet werden könnte.

### 7.2.1. Externe Inhaltstypen (External Content Types)

Mit Hilfe der Business Connectivity Services lassen sich externe Inhaltstypen (external content types) in SharePoint integrieren. Diese werden durch vorgegebene Methoden definiert (operations), welche in etwa die Funktionalität von SharePoint Listen abbilden:




Folgende Operationstypen sind vorgesehen:

- Read List
- Read Item
- Create
- Update
- Delete

Ausserdem können zu Read List noch Filter mit zugehörigen Parametern definiert werden.

**Abbildung 9: Definition des externen Inhaltstypen**

### 7.2.2. BDC Model

Definiert werden diese Operationen und Filter in einem Business Data Connectivity Model, welches als Teil eines Farm-Level Features auf die SharePoint Farm installiert wird (siehe 8.4). Dieses BDC Model kann mittels eines XML-Files definiert werden, oder von einer bestehenden SharePoint Farm (wo es z.B. mit dem SharePoint Designer angelegt wurde) als ein solches exportiert werden. Die aktuelle Schemadefinition des BDC Models ist mit dem „alten" Standard des SharePoint 2007 Business Data Catalog (BDC) identisch. Daraus lässt sich schliessen, dass die Business Connectivity Services abgesehen von den neuen Insert/Update/Delete Operationen keine grösseren Änderungen gegenüber dem alten Business Data Catalog aufweisen.

Die nachfolgende Darstellung bietet eine Übersicht über die wichtigsten BDC Metainformationen, welche im BDC Model eingetragen werden:

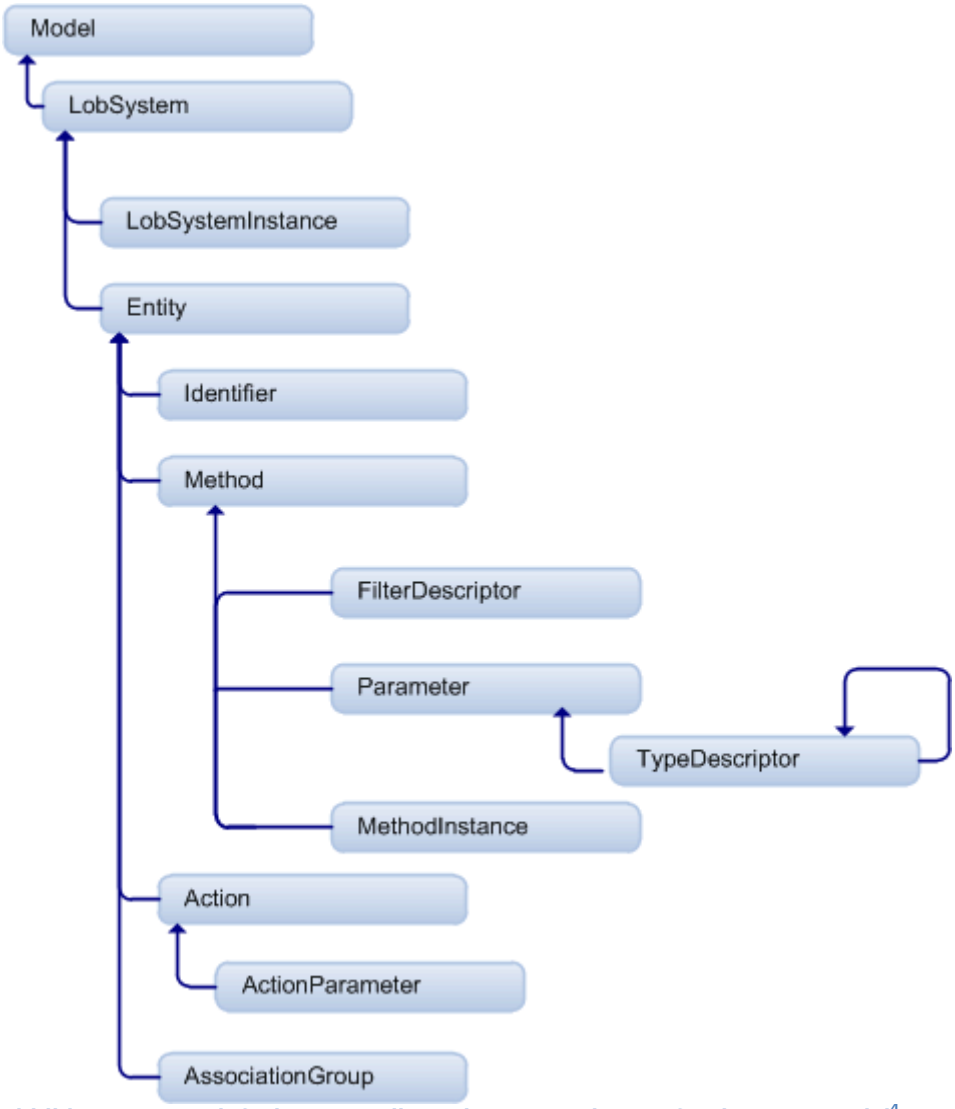



**Abbildung 10: Vereinfachte Darstellung des XML-Schemas für das BDC-Model**[4]

**Model:**
Model ist der Root Tag welcher neben den XML-Schemaangaben den Namen (im Attribut „Name") des BDC Models enthält.

**LobSystem:**
Definiert das externe System (Line of Business System), aus welchem Daten in SharePoint verfügbar gemacht werden können. Es muss der Typ und ein Name für das System angegeben werden. Als

[4] Quelle: BDC-Model Infrastructure: http://msdn.microsoft.com/en-us/library/ee556378.aspx, letzter Zugriff, 22.12.2010

Typen können hier „Value“, „Database“, „DotNetAssembly“, „WCF Webservice“ oder „Custom“ angegeben werden.

**LobSystemInstances:**
Definiert die verwendete Instanz für den Zugriff auf das externe System. Das Attribut Name bestimmt, unter welchem Namen die Instanz schlussendlich in Sharepoint erreichbar ist. In unserem Fall mit der Direktverbindung zum SQL Server sind hier auch der Connection String und die dazugehörigen Sicherheitsattribute definiert.

**Entity:**
Ein Entity Eintrag stellt ein einzelnes Business Objekt dar. Die Attribute Name, Namespace und Version sind verlangt. Die Definitionen für die Methoden für Abfrage, Filterung, Aktualisierung und Löschung von Daten sind ebenfalls in diesem Eintrag anzufügen.

**Identifier:**
Identifier gibt den Primary Key mit dessen Datentyp des Business Objektes an. Über den Wert des Primary Keys ist ein Business Objekt eindeutig identifizierbar.

**Method:**
Im Method Tag können Methoden mit Filtern, Parametern mit deren Datentypen und deren Ausführbare Instanzen definiert  werden.

**AssociationGroup:**
Hier können Beziehungen zwischen verschiedenen Business Objekten angegeben werden.

### 7.2.3. Externe Listen

Um externe Daten aus dem BCS Service in einer SharePoint Applikation zu verwenden, gibt es sogenannte externe Listen. Diese verhalten sich grundsätzlich wie interne Listen in SharePoint, können also eigene Views angeschlossen haben oder direkt auf der Seite bearbeitet werden.

Die externen Inhaltstypen stellen bereits ReadList, ReadItem etc. Methoden zur Verfügung. Diese könnten auf dem SharePoint Server direkt angesprochen werden (Assembly: Microsoft.BusinessData). Verwendet man aber für die Benutzeroberfläche Silverlight, so müssen die Daten trotz eigener View-Schicht von einer externen Liste geladen werden. Die API für die direkte Verwendung des BCS wurde nicht im SharePoint Silverlight Client Object Model integriert.

Externe Listen können mit dem SharePoint Designer per Mausklick aus den vorhandenen external content types der BCS erstellt werden. Diese externen Listen verhalten sich fast wie normale SharePoint Listen, leider mit einigen Einschränkungen. Will man zum Beispiel für eine bessere Wart- und Testbarkeit eine Trennung der Datenschicht vom Datenprovider erreichen, so stehen dafür keinerlei Hilfsmittel zur Verfügung. Das in SharePoint integrierte Entwicklungstool SPMetal (7.1.3) würde hier Unterstützung anbieten, funktioniert aber leider nur mit internen Listen.

### 7.2.4. Abfrage von Daten über externe Listen

Um Daten aus einer SharePoint Liste abzufragen kann Linq2Sharepoint verwendet werden. Linq2Sharepoint ist ein Linq Adapter für CAML (Collaborative Application Markup Language)[5], der die Erstellung von Abfragen auf SharePoint Listen wesentlich vereinfacht, indem er CAML-Queries im Hintergrund erstellt. Leider funktioniert Linq2Sharepoint aber nicht mit externen Listen. Abfragen auf externe Listen können nur direkt mit CAML erstellt werden.

Als bekannt wurde, dass Microsoft Linq2Sharepoint in Sharepoint 2010 integriert, wurde das Opensource Projekt Linq2Caml[6] eingestellt. Es ist auf Codeplex keine Version mehr zum Download verfügbar. Dass Linq2Caml mit externen Listen funktioniert hätte, ist ausserdem nicht anzunehmen.

[5] CAML Referenz http://msdn.microsoft.com/en-us/library/ms467521.aspx, letzter Zugriff 22.12.2010
[6] Linq2Caml http://linq2caml.codeplex.com/, letzter Zugriff 22.12.2010

Eine Abfrage in CAML um den Patienten mit dem Namen Müller zu finden sieht folgendermassen aus:

```
<View>
  <Method Name='PatientReadList'/>
  <ViewFields>
    <FieldRef Name='PatientID' />
    <FieldRef Name='Firstname'/>
    <FieldRef Name='Lastname'/>
    <FieldRef Name='DateOfBirth'/>
  </ViewFields>
  <Query>
    <Where>
      <Eq>
        <FieldRef Name='Lastname'/>
        <Value Type='Text'>Müller</Value>
      </Eq>
    </Where>
  </Query>
</View>
```

Diese Abfrage ist leider sehr ineffizient, da SharePoint zuerst über den BCS alle Daten aus der Tabelle <Patients> holt und diese erst danach filtert.

Effizienter (bei 1000 Einträgen mehr als 10x schneller!) kann man dies folgendermassen machen:

```
<View>
  <Method Name='PatientReadList'>
    <Filter Name='ByLastname' Value='Müller'/>
  </Method>
  <ViewFields>
    <FieldRef Name='PatientID' />
    <FieldRef Name='Firstname'/>
    <FieldRef Name='Lastname'/>
    <FieldRef Name='DateOfBirth'/>
  </ViewFields>
</View>
```

Dies erfordert aber, dass im BDC-Model der Filter ByLastname definiert wurde. Queries auf externe Listen können also nur mit zusätzlichem Aufwand effizient gemacht werden.

Führt man CAML Queries mit dem Client Object Model auf externen Listen aus, so muss eine Liste der Felder an den Client Context mitgegeben werden. Dazu ist ein Umweg nötig, der im Artikel „Code Snippet: Get Item Data from an External List on the Client“ [7] auf MSDN beschrieben wird.

### 7.2.5. Tools von Drittherstellern

Für die Datenanbindung mit BCS gibt es Tools von Drittherstellern. Zu beachten ist hier sicher LightningTools[8]. Diese Firma stellte schon Produkte für die einfachere Integration von Daten mit BDC in SharePoint 2007 her. Leider sind diese Komponenten einerseits sehr teuer, andererseits auch immer mit Einschränkungen verbunden. Man ist unweigerlich von der Versionierung und vom Support der Dritthersteller abhängig, was je nach Hersteller durchaus ein Risiko sein kann.

### 7.2.6. Vorteile

Die Verwendung der BCS mit MS-SQL als Datenquelle bietet natürlich auf unterster Ebene die volle Leistungsfähigkeit eines SQL-Servers. Sie ermöglicht Auswertungen der Daten, die mit internen SharePoint listen gar nicht, oder nur sehr inperformant möglich wären. Dies ohne die Daten zuerst exportieren zu müssen.

### 7.2.7. Nachteile

Es muss beachtet werden, dass auch mit BCS kumulative Abfragen wie Gruppierungen, Summierungen oder Mittelwerte nur mittels einer .NET Assembly oder bei einer per BDC-Model direkt angegebenen SqlServer Datenquelle direkt mit Stored Procedures in der Datenbank effizient möglich sind. Sonst werden für eine Abfrage zuerst alle Daten geladen und erst dann die Berechnungen darauf ausgeführt, was noch mehr Overhead verursacht als bei internen Listen.

[7] Code Snippet: Get Item Data from an External List on the Client, http://msdn.microsoft.com/en-us/library/ff464384.aspx, letzter Zugriff 22.12.2010
[8] LighningTools, http://www.lightningtools.com, letzter Zugriff 22.12.2010

Viele Tools welche mit internen Listen funktionieren, funktionieren mit externen Listen (welche über BCS auf die Daten zugreifen) nicht. Darunter fallen unter anderem Linq2SharePoint und SPMetal.

Gegenüber SharePoint Listen haben externe Datenquellen mit BCS noch weitere Nachteile, welche bei SURVEAST aber weniger relevant wären. Mehr-zu-mehr Beziehungen sind nur umständlich implementierbar und gewisse SharePoint List Funktionalitäten wie Workflows, Events, Alerts oder automatische Erstellung eines RSS-Feeds können auch über externe Listen (7.2.3) mit den BCS nicht direkt verwendet werden.

### 7.2.8. Schlussfolgerungen

Die Verwendung von BCS ist aufwändiger, als eine Implementation mit SharePoint Listen. Effiziente Queries müssen im BCS vordefiniert sein. Linq2SharePoint würde das erstellen von Abfragen allgemein erleichtern, funktioniert nicht mit externen Listen. Ausserdem muss jede einzelne Datenbearbeitungsmethode im BDCM definiert werden.

Die Implementation einer Erfassung von komplexeren Daten und vor allem die Implementation einer Möglichkeit, Daten über eine grafische Benutzeroberfläche flexibel abzufragen verkomplizieren sich dadurch stark.

Dennoch bietet diese Lösung die benötigte Flexibilität bei den Auswertungen, welche mit internen Listen nur durch Datenexport und Verarbeitung in einem Drittsystem gewährleistet werden könnte. Ausserdem ist die Kontrolle über die erfassten Daten einfacher (Patientendaten), da diese zentral in einer SQL Server Datenbank liegen und nicht unüberschaubar neben anderem User Content in der Content Datenbank des SharePoint Servers.

# 8. Umsetzung der Architektur

## 8.1. Architekturkonzepte

### 8.1.1. Datenbank

Die SURVEAST Datenbank kann wahlweise auf einer eigenen SQL Server Instanz betrieben werden oder auf derselben wie die SharePoint Content Datenbanken. Wichtig hierbei ist, dass die im BDC Model (siehe 7.2.2) hinterlegten Active Directory Identitäten auch die richtigen Berechtigungen auf dem SQL Server besitzen.

Um sicherzustellen, dass die externe SURVEAST Datenbank wirklich vom SharePoint getrennt werden kann, wurden während den Tests die Instanzen getrennt. Dies verursacht grundsätzlich keine Probleme oder Seiteneffekte.

Die logische Struktur der Datenbank wird im Kapitel (8.2.2) genauer beschrieben.

### 8.1.2. Datenbankanbindung des SURVEAST BCS Service

Wie im Abschnitt 7.2 beschrieben gibt es mehrere Möglichkeiten eine Datenbank mit dem BCS zu verbinden: Direkt über eine angegebene Treiberschnittstelle, über einen WCF Service oder über eine .NET Assembly.

Als erste Alternative, wurde der Weg über die .NET Assembly und Linq2Sql geprüft. Während der Erstellung eines Prototypen hat sich aber herausgestellt, dass diese Indirektion schlussendlich keine wesentlichen Vorteile bringt. Die Daten sind im Weiteren nur über das Interface des BCS Services in SharePoint verwendbar, wo Relationen zwischen den Daten nur sehr mühsam wiederverwendet werden können. Da die Security direkt über das ActiveDirectory realisiert wird, erübrigt sich auch die Einbindung eines Security-Layers in dieser Schicht.

So wurde beschlossen die Daten direkt durch eine im BDC-Model angegebene SQL-Server Treiberschnittstelle zu laden. Die Abstraktion der konkreten Ereignisse wird direkt auf der Datenbank behandelt (siehe 8.1.1). So hält sich auch die Komplexität des BDC-Models in Grenzen und das Deployment funktioniert via BDC Model in der SURVEST Solution zuverlässig.

### 8.1.3. Umsetzung der Vererbungshierarchie als BCS-Service

Da das SURVEAST Datenmodell für die verschiedenen Ereignistypen eine speziellere Struktur aufweist wird hier bereits auf der Business Data Connectivity Schicht eine Abstraktion vorgenommen. Konkret geht es darum, die konkreten Ereignistypen zwar redundanzfrei in der Datenbank abzubilden, diese aber trotzdem im Sharepoint als einzelne Entitäten darzustellen. Dies wurde wie folgt umgesetzt:

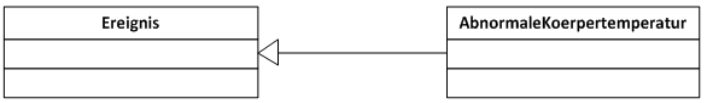



Abbildung 11: Beispiel für die Vererbung der Ereignisse

Aus Sicht des Domain Models ist das Ereignis „abnormale Körpertemperatur“ eine Konkretisierung des Typs Ereignis, was logisch gesehen natürlich Sinn macht. Es besitzt also alle Parameter eines Ereignisses sowie einige zusätzliche, welche zum konkreten Ereigenis „abnormale Körpertemperatur“ gehören. Auf Datenbankebene ist dies nun mit einer Aggregation, also mit einer 1:1 Beziehung zwischen den Tabellen „Event“ und „AbnormalBodyTemperature“ realisiert:

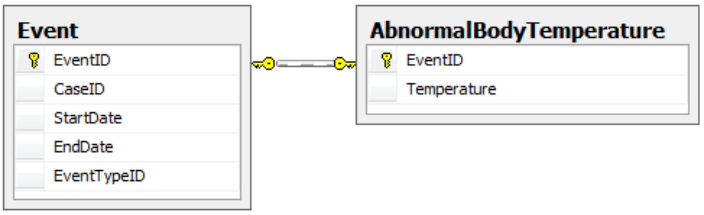



Abbildung 12: Vererbung in der Datenbank als Assosiation

Da Assoziationen in SharePoint ziemlich kompliziert umgesetzt werden müssen (GetParentByChild(…) und GetChildByParent(…) Methoden in beide Richtungen), wurde dies bereits auf Datenbankebene umgangen und folgende Abstraktion eingebaut: Gegenüber SharePoint wird ein externer Inhaltstyp pro Ereignistyp eingeführt. Zusätzlich existiert auch ein externer Inhaltstyp für alle Ereignisse.

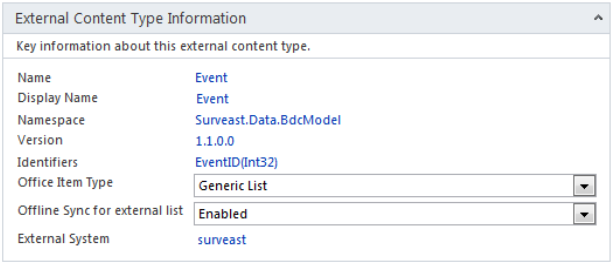



Abbildung 14: Externer Inhaltstyp "Event"

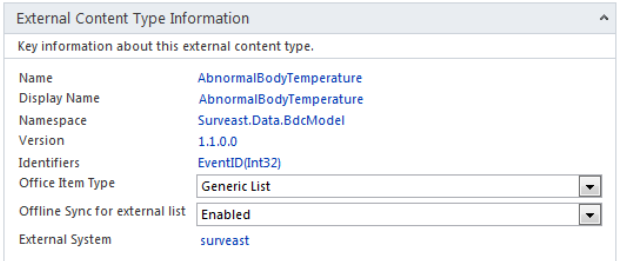



Abbildung 13: externer Inhaltstyp "AbnormalBodyTemperature"

Wie in den Screenshots ersichtlich ist, dient die Liste der Ereignisse (Event) nur zur Anzeige, zur Erfassung muss ein konkreter Ereignistyp erfasst werden (hier abnormale Körpertemperatur). Auf Datenbankebene ist diese Abstraktion mit Views und Stored Procedures umgesetzt, auf SharePoint-Ebene aber sehen die Inhaltstypen aus, als ob sie getrennt wären. Mit den Stored Procedures für Create, Update und Delete wird gewährleistet, dass die Daten im verknüpften Datenmodell redundanzfrei abgelegt werden.

### 8.1.4. Umsetzung der Benutzeroberfläche für die Erfassung von Ereignissen

Für die Erfassung der Ereignisse in SURVEAST ist bereits ein kleiner Workflow notwendig, da die erfassende Person zuerst den jeweiligen Patient und das zu erfassende Ereignis auswählen muss.

Da SharePoint 2010 eine Workflowunterstützung bietet, würde sich hier die Verwendung dieser Workflow Engine anbieten. Leider unterstützen diese Workflowkomponenten aber keine externen Listen.

Ein weiterer Ansatz war, die Erfassung mit einem „Custom Workflow“ und normalen externen Listen zu lösen. Nach einigen Versuchen musste man aber zum Schluss kommen, dass hier der Aufwand um einen solchen Workflow herzustellen, sowie die Views der einzelnen Listen anzupassen, nicht gerechtfertigt ist. Mit Silverlight lässt sich dieser einfache Ablauf (2 Workflow-Steps) schlanker und einfacher abbilden. Zusätzlich existiert eine Anforderung, gegebenenfalls ganze Gruppen von Ereignissen aufs Mal erfassen zu können, was mit Silverlight massiv einfacher umzusetzen ist, als mit den Views der SharePoint Listen.

### 8.1.5. Umsetzung der Benutzeroberfläche für die Auswertung

Um die Auswertung der Daten sinnvoll konfigurieren zu können ist einiges an Interaktion und Zwischenspeicherung auf dem Client gefragt. Der Benutzer sollte die Endpunkte sowie die Modifikatoren für seine Auswertungen frei aus den möglichen Ereignissen auswählen können und diesen noch zusätzliche Parameter angeben können.

Auch hier kristallisierte sich Silverlight als einzige sinnvolle Möglichkeit heraus, eine solche Oberfläche zu realisieren. Die Möglichkeiten die Konfiguration über die normalen Views der SharePoint Listen zu erstellen sind massiv eingeschränkter und die Benutzerfreundlichkeit lässt dabei zu wünschen übrig (zu viele Page-Postbacks).

## 8.2. Logische Architektur

### 8.2.1. Übersicht

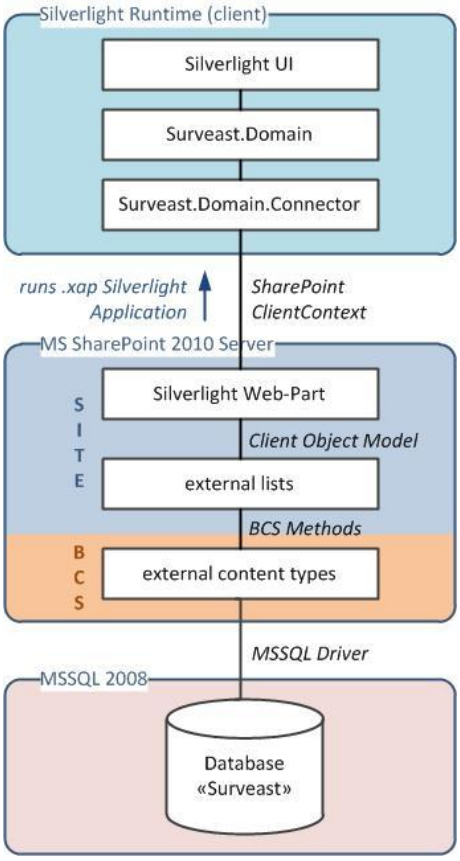



Abbildung 15: Architekturübersicht

Das Bild zeigt einen Überblick über die verwendeten Architekturkomponenten:

Die oberste Ebene repräsentiert die View inkl. ViewModel. Hier wurde mit standard Silverlight 4.0 Komponenten gearbeitet, welche über das SharePoint Client Object Model mit SharePoint kommunizieren. Die Silverlight Komponente ist in einem SharePoint Modul gekapselt, welches beim Deployment der Solution automatisch auf den SharePoint Server installiert wird.

Die angesprochene Silverlight-Komponente kommuniziert via Domain und Domain.Connector über das Client Object Model mit externen Listen, welche die externen Inhaltstypen repräsentieren. Der Domain Layer mit dem Domain.Connector ist in einer Silverlight class library implementiert. Mehr zur Architektur dieser Komponenten im Punkt 8.2.5 und 8.2.6.

Unter den externen Listen werden die Methoden der externen Inhaltstypen aufgerufen welche wiederum über die Business Connectivity Services mit der SQL Server Datenbank kommunizieren (Siehe 8.2.3).

Die Datenbank selber beinhaltet neben den Tabellen auch Views und Stored Procedures um die Abstraktion der einzelnen Inhaltstypen gegenüber den oberen Schichten vorzunehmen. Dieses Konzept ist im Kapitel 8.1.3 beschrieben.

### 8.2.2. Datenbank

Wie bei den Architekturkonzepten (8.1.1) angesprochen, wurde die Umfasst die Datenbank neben einem einfachen Datenmodell über Patient, Fall und Ereignistyp die 1:1 Assoziation von einem konkreten Ereignistyp. Kommen mehr Ereignistypen hinzu, wird das Modell natürlich entsprechend erweitert.

Zu jedem konkreten Ereignistypen existiert zusätzlich eine View. Im Beispielfall mit der abnormalen Körpertemperatur sind dies die AbnormalBodyTemperatureView für die Anzeige und die dazugehörigen Stored Procedures für die schreibenden Operationen:

- InsertAbnormalBodyTemperature (@CaseId, @StartDate, @EndDate, @Temperature)
- UpdateAbnormalBodyTemperature (@EventId, @CaseId, @StartDate, @EndDate, @Temperature)
- DeleteAbnormalBodyTemperature (@Temperature)

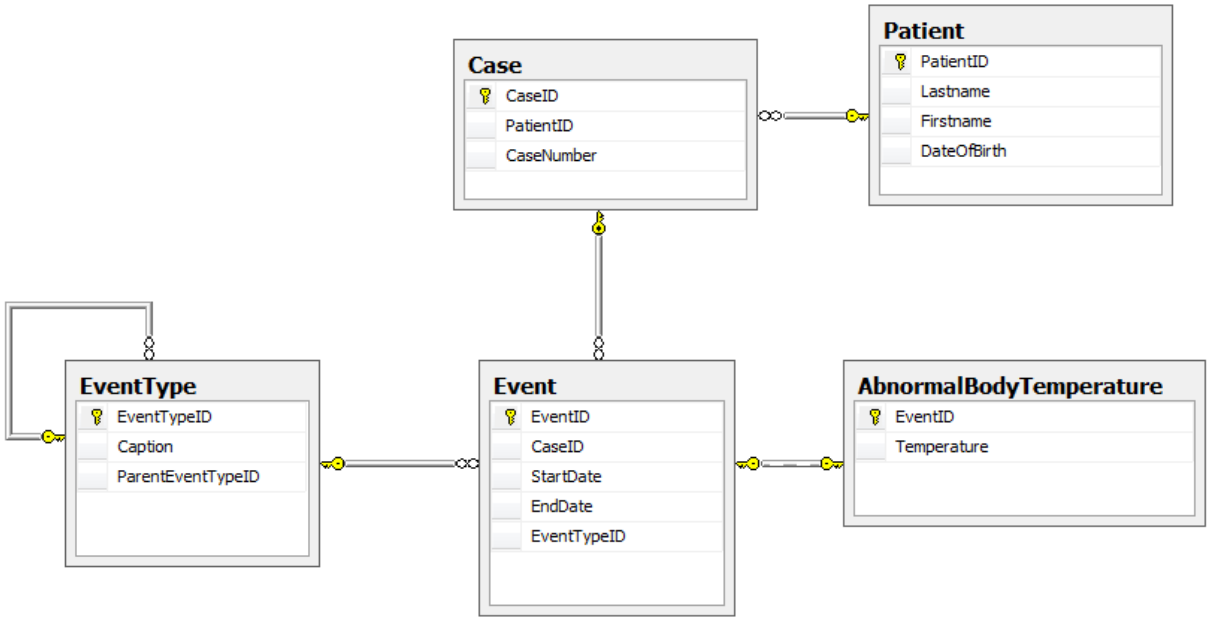



Abbildung 16: Datenbankmodell des Prototypen

### 8.2.3. Business Connectivity Service (External Content Types)

Das BDC-Model greift über die SqlServer Schnittstelle auf die Datenbank zu und wird durch ein XML definiert (siehe 7.2.2).

```
<?xml version="1.0" encoding="utf-16" standalone="yes"?>
<Model xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://schemas.microsoft.com/windows/2007/BusinessDataCatalog BDCMetadata.xsd"
Name="SurveastBdcModel" xmlns="http://schemas.microsoft.com/windows/2007/BusinessDataCatalog">
  <LobSystems>
    <LobSystem Type="Database" Name="surveast">
      <Properties>
        <Property Name="WildcardCharacter" Type="System.String">%</Property>
      </Properties>
      <Proxy />
      <LobSystemInstances>
        <LobSystemInstance Name="surveast">
          <Properties>
            <Property Name="AuthenticationMode" Type="System.String">PassThrough</Property>
            ...
          </Properties>
        </LobSystemInstance>
      </LobSystemInstances>
      <Entities>
        <Entity Namespace="Surveast.Data.BdcModel" Version="1.0.0.0" EstimatedInstanceCount="10000"
                Name="Patient" DefaultDisplayName="Patient">
          <Identifiers>
            <Identifier TypeName="System.Int32" Name="PatientID" />
          </Identifiers>
          <Methods>
            <Method IsStatic="false" Name="PatientReadList">
              <Properties>
                ...
              </Properties>
              ...
```

Abbildung 17: Stark gekürztes XML Beispiel des SURVEAST BDC Models

In unserem Fall ist als LOBSystem die SQL Server Datenbank definiert, welche die Daten des SURVEAST Systems beinhaltet. Es handelt sich also um einen externen Inhaltstyp, welcher direkt via eine Treiberschnittstelle mit der MS-SQL Datenbank verbunden ist.

Achtung: Die Versionsnummer (-> Attribut „Version“) sollte nach jeder Änderung an den Entity Klassen geändert werden. Bei Strukturänderungen muss die Versionsnummer erhöht werden, damit die neuere Version dieser Entity im BDC Model beim Deployment-Vorgang richtig installiert wird.

### 8.2.4. External Lists

Externe Listen müssen erstellt werden um über das Silverlight Client Object Model auf den BCS zugreifen zu können (siehe 7.2.3).

### 8.2.5. SURVEAST Domain Connector

Die Klassen im Namespace Surveast.Domain.Connector kümmern sich um das Laden der Daten über eine externe SharePoint Liste und den Business Connectivity Service und trennen den Domain Layer vom Datenprovider (BCS).

Das folgende Beispiel zeigt den Vorgang am Beispiel des Ladens von Patienten über eine externe SharePoint Liste (pl : PatientList).

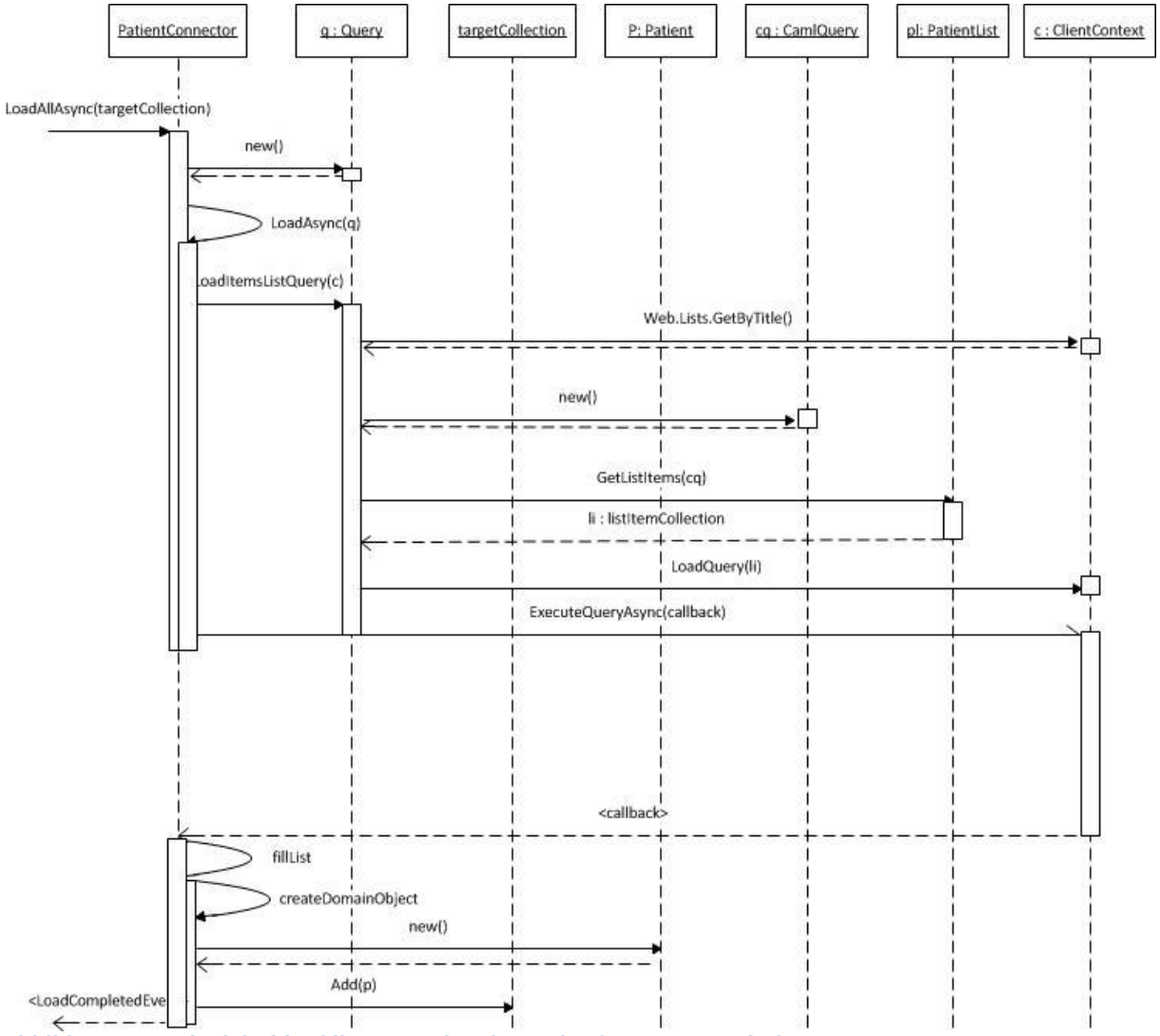



**Abbildung 18: Beispiel Ablaufdiagramm für einen Abruf von Daten mit dem Connector**

### 8.2.6. SURVEAST Domain

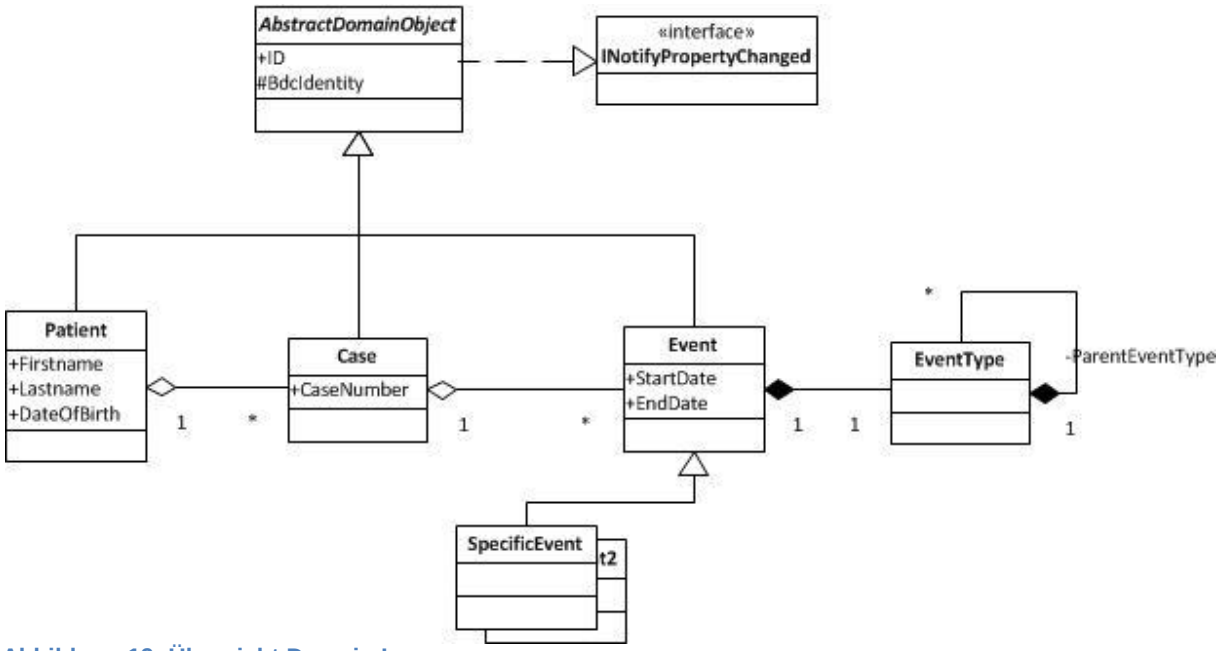



Abbildung 19: Übersicht Domain Layer

Der Domainlayer wurde für das Pilotmodul so einfach wie möglich gehalten. Ereignisgruppen werden nur noch durch eine Angabe im Eventtypen (EventType) unterschieden. Durch Anfügen weiterer SpecificEvents (von Event abgeleitete Klassen) kann das System für zukünftige Erfassungen und Auswertungen erweitert werden. Allerdings müssen dafür alle weiteren Schichten unterhalb des Domainlayers (Domain.Connector, Externe Listen, BDCModel und Stored Procedures auf der Datenbank) auch angepasst werden.

### 8.2.7. Silverlight Webpart

Der WebPart dient grundsätzlich dazu das Silverlight XAP-File zur Verfügung zu stellen, damit dies vom Browser geladen werden kann. Dazu wird der SharePoint eigene, generische Silverlight Webpart verwendet und entsprechend konfiguriert.

### 8.2.8. Silverligth UI

Gemäss dem MVVM Pattern wird für XAML-Technologien wie Silverlight oft die Logik der Views in eine weitere Zwischenschicht gekapselt. Dieses ViewModel wird dann direkt als DataContext an die View gebunden. Dazu muss es das INotifyPropertyChanged Interface implementieren, damit allfällige Änderungen direkt an die anderen Schichten weitergereicht werden können.

Wichtig hierbei ist, dass die View-Controls, welche per DataBinding ans ViewModel gebunden sind, bei jeder Änderung den PropertyChanged Event auslösen. Die Daten sollten aber nicht bei diesem Event persistiert werden, sonst hat einerseits der Benutzer die Kontrolle über die Persistierung („Save/Discard“) verloren und andererseits wird viel teilweise ungewollter Traffic zum SharePoint verursacht.

Interaktionen die vom Benutzer ausgelöst werden (z.B. Button-Clicks), werden mit dem sogenannten CommandBinding realisiert. Dazu wird eine im ViewModel eingebettete Klasse verwendet, die das ICommand Interface implementiert. Diese wird dann im XAML Code an das entsprechende Steuerelement gebunden. So ist das gesamte ViewModel mit allen Interaktionen von aussen automatisiert testbar (siehe 8.3).

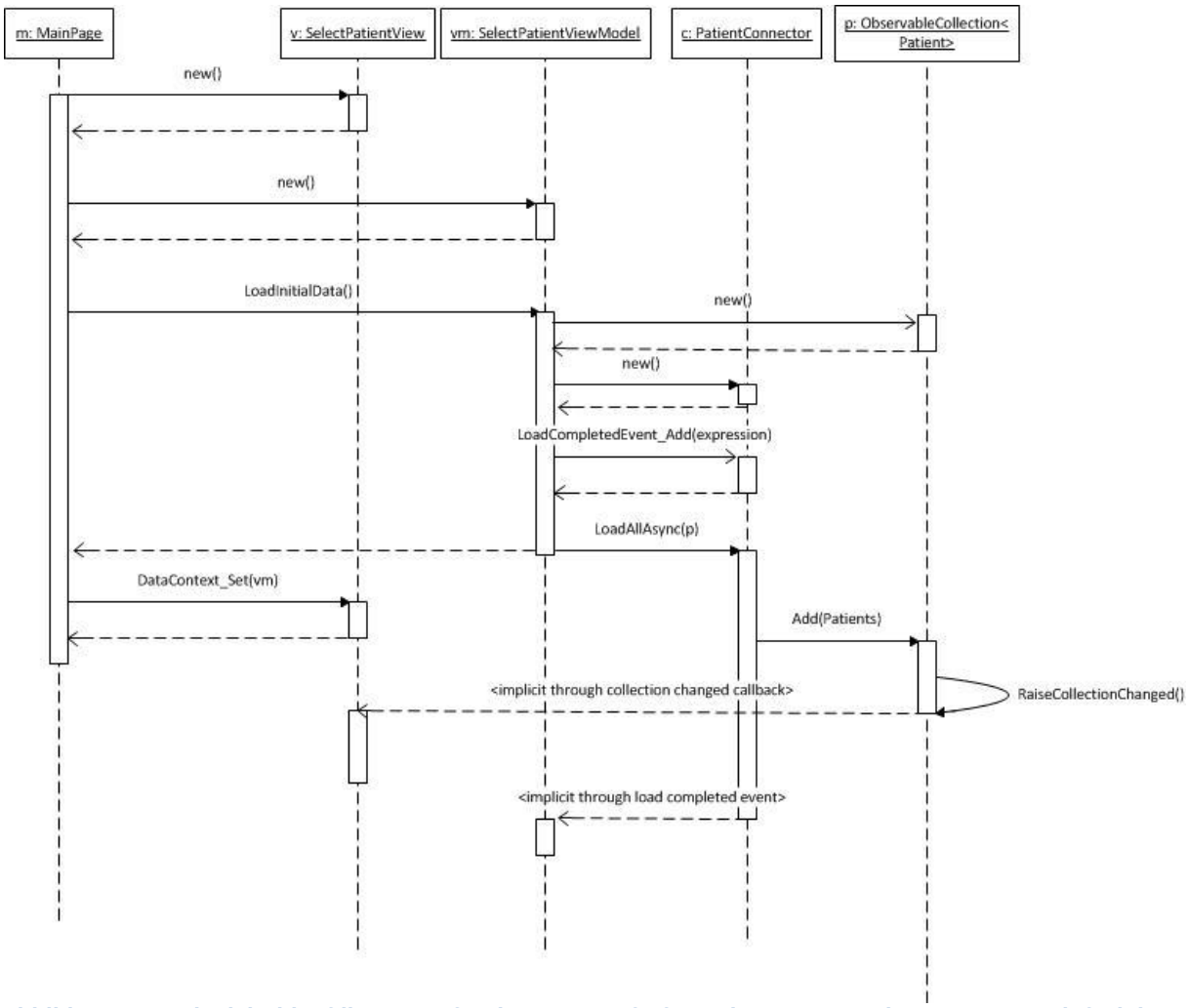



**Abbildung 20: Beispiel Ablaufdiagramm für den Datenaufruf aus dem UI Layer (Connector vereinfacht)**

Das Sequenzdiagramm zeigt den Erstellungsablauf und einen Load Aufruf auf die erste Collection der View. Der Aufruf der Load Methoden und die Konstruktion des Domain Connectors erfolgt asynchron. Durch das DataBinding der View und die PropertyChanged Events werden Callback Methoden hinfällig. Die Aktualisierung der View nach erfolgtem Ladevorgang wird durch die Events automatisch ausgelöst.

Im Diagramm ist gut zu erkennen, dass das Laden der Daten aus dem Data Access Layer immer asynchron erfolgt. Dies ist eine Vorgabe, die von Microsoft bewusst in den Schnittstellen integriert wurde, um eine Blockierung der Benutzeroberfläche durch Aufrufe übers Netzwerk zu verhindern. Da die Silverlight Applikation Clientseitig läuft, lösen solche Ladevorgänge gezwungenermassen immer Aufrufe übers Netzwerk aus und erfolgen darum asynchron. Der Aufruf wird also abgesetzt und die aufrufende Funktion läuft weiter. Die mit <implicit> bezeichneten Methoden werden ausgeführt sobald der asynchrone Aufruf ein Ergebnis geliefert hat.

## 8.3. Testing

Um die Qualität der Software garantieren zu können, wurde eine Möglichkeit gesucht, die verwendete Architektur automatisiert testen zu können. Leider sind normale Unit-Tests in VisualStudio nicht geeignet die Silverlight / SharePoint / BCS Architektur zu testen da der auszuführende Testcode auch auf dem SharePoint Server ablaufen müsste.

Mit dem Silverlight 4 Toolkit, bietet sich nun die Möglichkeit von der obersten Schicht der Applikation aus Unit Tests auszuführen. Die Testumgebung wird in eine normale Silverlight Komponente gepackt und kann so auch in einem Silverlight Webpart auf dem SharePoint Server installiert werden.

Damit die Unit Tests möglichst Zugriff auf alle Schichten der Applikation haben, wurde die Silverlight-Test-Applikation in einen separaten Webpart eingegliedert und auch auf den SharePoint installiert:

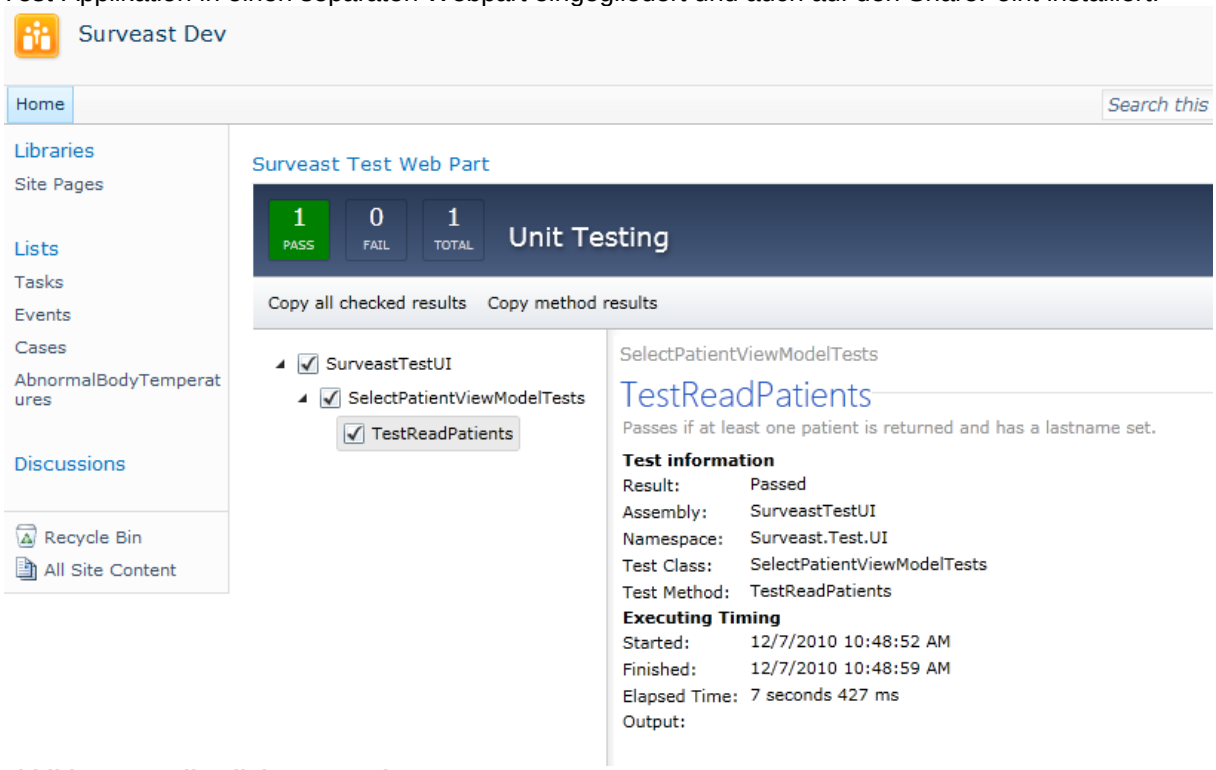



**Abbildung 21: Silverlight Test Webpart**

Diese Möglichkeit lässt es zu, Tests auf alle Schichten abzusetzen, welche in die Silverlight Application (*.xap) kompiliert und auf dem Client ausgeführt werden:

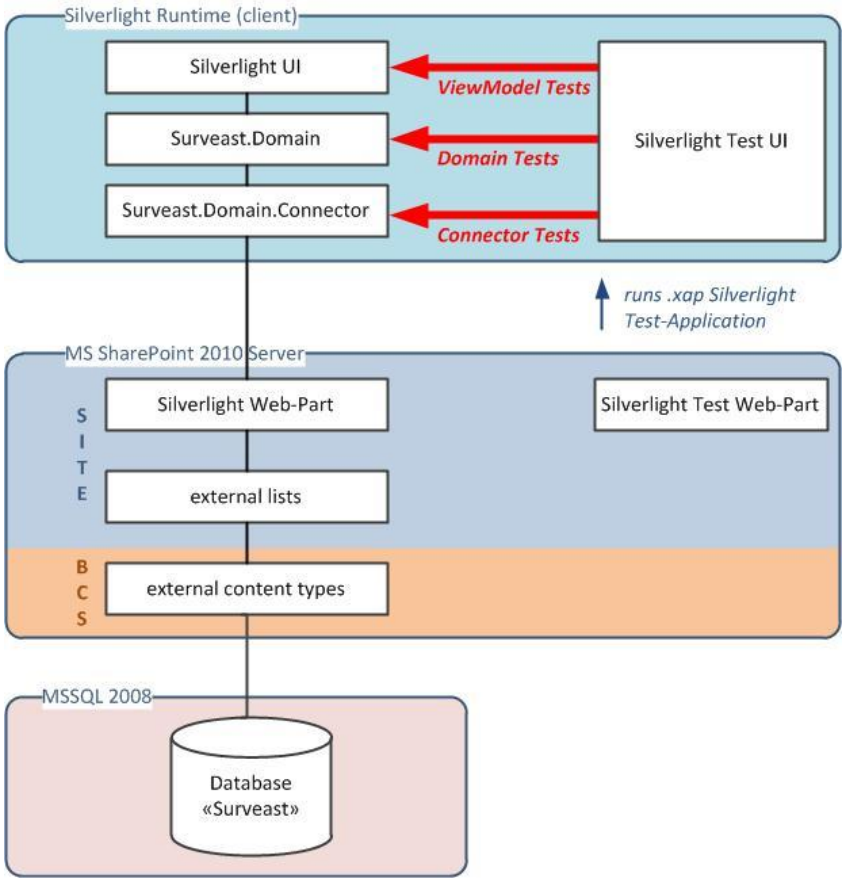



**Abbildung 22: Architekturübersicht inkl. Testing**

Durch die asynchronen Aufrufe in Silverlight muss auch in den Unit Tests dieses Konzept berücksichtigt werden. Dies funktioniert mit dem Silverlight-Toolkit eignen Testframework, welches Methoden zur Verfügung stellt, um die Tests mit asynchronen Aufrufen durchführen zu können:

*EnqueueConditional:* Mit EnqueueConditional können Bedingungen festgelegt werden, wann der Test weiterlaufen kann. Dabei drängt sich beim Testen des ViewModels im MVVM Pattern natürlich auf, zu prüfen, ob die PropertyChanged Events ausgelöst wurden.

*EnqueueCallback:* Über EnqueueCallback kann ein delegate angegeben werden, welches ausgeführt wird, sobald die EnqueueConditional Bedingung eingetroffen ist. Sinnvollerweise werden darin die Assert's gemacht und das Testergebnis überprüft.

*EnqueueTestComplete:* Wartet mit dem Beenden des Tests bis die EnqueueConditional Bedingung eingetroffen ist.

```
 [TestMethod]
[Asynchronous]
[Description("Passes if at least one patient is returned and has a lastname set.")]
public void TestReadPatients()
{
      SelectPatientViewModel viewModel =
            new SelectPatientViewModel(Deployment.Current.Dispatcher, null);

      bool viewModelChanged = false;

      viewModel.PropertyChanged += ((sender, args) => { viewModelChanged = true; });
      EnqueueConditional(() => viewModelChanged);

      viewModel.LoadInitialData();

      EnqueueCallback(() =>
      {
          Assert.IsTrue(viewModel.Patients.Count > 0, "Some Patients loaded.");
          Assert.IsNotNull(viewModel.Patients[0].Lastname, "P0 has a Lastname set.");
      });

      EnqueueTestComplete();
}
```

**Abbildung 23: Beispieltest welcher mit diesen Enqueue Funktionalität arbeitet:**

Der gezeigte Test prüft auf Stufe ViewModel das Laden der Daten alle Schichten über den BCS bis zur Datenbank. Er prüft ob die Collection richtig gefüllt wird, welche schlussendlich an ein Steuerelement auf der Oberfläche gebunden wird. Da die Oberfläche keinerlei CodeBehind verwendet, ist so sichergestellt dass hier keine weiteren Fehler zu erwarten sind. Einzig das deklarative Binding im XAML Code ist so nicht testbar.

Um nur die oberen Schichten bis zum Domain-Layer zu testen, könnten die betreffenden Domain.Connector durch Testmocks ersetzt werden, welche Domain Objekte erstellen ohne über externe Listen und BCS auf die Datenbank zu gehen. Dies würde es ermöglichen, den Code in den oberen Schichten isoliert und effizient zu testen.

Im vorliegenden Prototyp wurden vier Beispieltests implementiert. Ein Test setzt oben auf dem ViewModel an, drei direkt auf dem Domain Connector. Dies soll zeigen wie die Applikation auf den verschiedenen Schichten testbar ist.

## 8.4. Komponenten und Deployment

Die verwendete Architektur gibt die Aufteilung der Komponenten innerhalb des SharePoint Servers weitgehend vor.

### 8.4.1. Übersicht

Der Data Access Layer baut auf den Business Connectivity Services auf. Diese laufen als sogenannte Service Application auf der Share Point Farm, sind also Komponenten die nicht einer einzelnen SharePoint Seite zugeordnet sind. Da der Data Access Layer von SURVEAST in dieser BCS Komponente betrieben wird, muss auch dieser im „Farm-Scope“, also auf die ganze Farm, installiert werden.

Die restlichen Layer sowie die visuellen Komponenten der Applikation sind nach dem Buildvorgang alle in der jeweiligen Silverlight Applikation (.xap Package) gekapselt und können als Silverlight Webpart auf eine SharePoint Seite installiert werden. Da es sich hier um Webparts, also modulare UI Komponenten handelt, werden diese auf die jeweilige SharePoint Seite und nicht auf die ganze Farm installiert.

Eigene sowie mitgelieferte SharePoint Komponenten können als sogenannte „Features“ einzeln installiert und deinstalliert werden. Diese Features haben einen definierten Gültigkeitsbereich (Scope) und können so nach Bedarf ausgelegt werden. Ausserdem können Abhängigkeiten von anderen Features festgelegt werden, die erfüllt sein müssen damit das aktuelle Feature aktiviert werden kann (Details zum Installations und Aktivationsvorgang siehe 8.4.5). Die SURVEAST Architektur teilt sich in zwei Features auf: SurveastData und SurveastUI.

### 8.4.2. SurveastData

SurveastData beinhaltet das BDC Model mit der Definition des externen Systems inklusive der externen Datentypen. Der Gültigkeitsbereich wurde als Farm-Level-Feature ausgelegt, da die Business Connectivity Services wie oben angesprochen als Service Application auf der ganzen Farm laufen.

### 8.4.3. SurveastUI

Anders als beim SurveastData Feature bezieht sich der Gültigkeitsbereich des SurveastUI Features nur auf die entsprechende Site Collection. Es beinhaltet die jeweiligen SharePoint Komponenten die nicht innerhalb des Business Connectivity Services laufen müssen wie zum Beispiel die Webparts.

Grundsätzliche wäre es möglich, auch die externen Listen in diesem Feature mitzuliefern. Darauf wurde aber in der aktuellen Phase des Prototyps verzichtet, da der Aufwand erheblich grösser ist, als die externe Liste nach dem Deployment manuell zu erstellen. Diese kann auch bestehen bleiben, wenn die anderen Teile der Applikation geändert werden, sie muss also nicht bei jedem Deploymentvorgang neu erstellt werden.

Die angesprochenen Webparts sind modulare UI-Bausteine die in SharePoint auf die gewünschten Seiten platziert werden können. Solche Webparts bauen auf normalem ASP.net auf (in unserem Fall mit eingebetteter Silverlight Komponente) und können auch untereinander wiederverwendet werden. Für die Silverlight Komponenten von SURVEAST wurde der eingebaute SilverlightWebPart verwendet, welcher eigentlich nur instanziert wird und das zugehörige XAP File zugewiesen bekommt.

Pro Silverlight Webpart werden zwei SharePoint Komponenten benötigt. Einerseits der Webpart, welcher sich ausschliesslich auf die Konfiguration des instanzierten SilverlightWebparts beschränkt und andererseits ein sogenanntes Modul welches dafür sorgt das der Build-Output des Silverlight Projektes (schlussendlich das XAP File) mit dem Feature mit installiert wird.

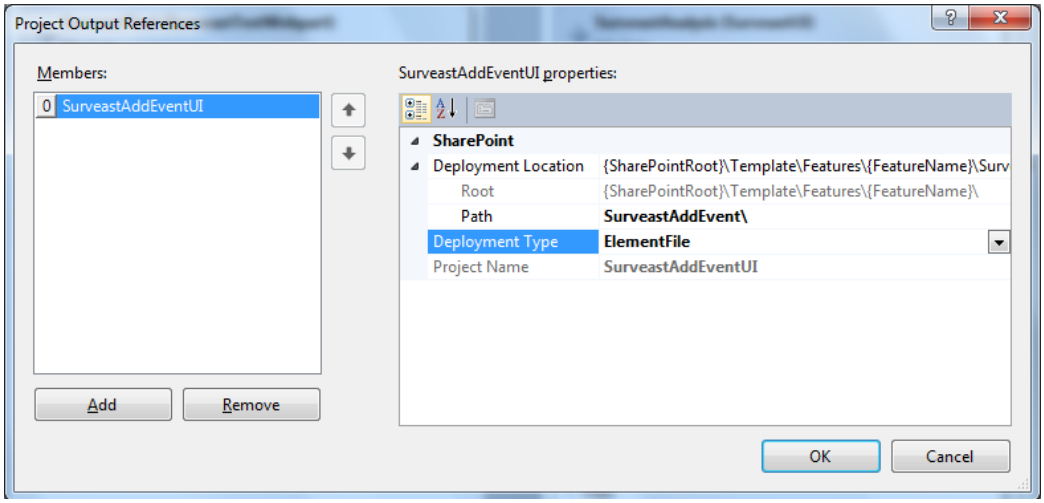



**Abbildung 24: Project Output Referenz eines oben angesprochenen Moduls**

### 8.4.4. Packetierung

Um die Applikation ausliefern und sinnvoll auf eine andere SharePoint Farm installieren zu können werden die beiden Features SurveastUI und SurveastData in sogenannte SharePoint Solution Packages (*.wsp) gepackt. Dies sind im Grunde genommen normale CAB Archive welche die Struktur eines gesamten Features enthalten.

Die Struktur der beiden Features sieht folgendermassen aus:

SurveastUI:

| | |
|---|---|
| SurveastUI.wsp | |
| .\manifest.xml | Beinhaltet Metadaten zum Package. In diesem Fall einfach einen Verweis zum zu installierenden Feature. |
| .\1033\xml\webtemp_SurveastSiteDefinition.xml | Site Definition, falls solche mit dem Site-Feature mitgeliefert warden. |
| .\SiteDefinition\default.aspx | Start-View für die entsprechende SharePoint Site. |
| .\SurveastUI_Surveast | Feature Haupverzeichnis, wird so auf die Farm kopiert. |

| | |
|---|---|
| .\.\Feature.xml | XML Definitionsfile mit Metadaten zum Feature |
| .\.\SurveastAddEvent | Basisverzeichnis für den AddEventWebpart. |
| .\.\SurveastAddEvent\Elements.xml | Definitionsfile, beinhaltet alle zugehörige Files des Verzeichnisses. |
| .\.\SurveastAddEvent\SurveastAddEventUI.xap | Silverlight Application Package – die kompilierte Benutzeroberfläche. |
| .\.\SurveastAddEventWebpart | Basisverzeichnis für die Webpart Metadaten. |
| .\.\SurveastAddEventWebpart\Elements.xml | Definitionsfile, beinhaltet alle zugehörige Files des Verzeichnisses. |
| .\.\SurveastAddEventWebpart\SurveastAddEvent Webpart.webpart | Webpart Konfigurationsfile (der bestehende generische Silverlight-Webpart wurde verwendet). |

SurveastData:

| | |
|---|---|
| SurveastUI.wsp | |
| .\manifest.xml | Beinhaltet Metadaten zum Package. In diesem Fall einfach einen Verweis zum zu installierenden Feature. |
| .\SurveastData_SurveastData | Feature Haupverzeichnis, wird so auf die Farm kopiert. |
| .\.\Feature.xml | XML Definitionsfile mit Metadaten zum Feature |
| .\.\BdcAssemblies | BDC Assemblies (in SURVEAST nicht benötigt, wird aber trotzdem ins Packet genommen). |
| .\.\BdcAssemblies\SurveastData.dll | |
| .\.\SurveastBdcModel | |
| .\.\SurveastBdcModel\SurveastBdcModel.bdcm | BDC Model (XML Definition) |

### 8.4.5. Deployment

Um ein Feature auf eine SharePoint Farm zu installieren sind einige einzelne Schritte nötig. Diese sind je nach Featuretyp für bestimmte Gültigkeitsbereiche durchführbar. Das Farm-Feature SurveastData kann also nur auf der ganzen Farm installiert werden, während das Site-Feature SurveastUI pro Site installiert oder aktiviert werden kann.

Folgende Schritte müssen auf der SharePoint Management Shell mit Hilfe eigener SharePoint cmdlets, des „stsadm“ Kommandozeilenprogramms (eigentlich ein Relikt aus SharePoint 2007) oder der „SharePoint Central Administration“ Website ausgeführt werden:

#### *8.4.5.1. Add Solution (Add-SPSolution / stsadm –o addsolution …)*

Das SharePoint Solution Package wird auf die Farm kopiert (in den sogenannten „solution store“).

#### *8.4.5.2. Deploy Solution (Install-SPSolution / stsadm –o deploysolution …)*

Die Inhalte des Solution Packages werden gemäss dem manifest.xml an den entsprechenden Ort kopiert. Beinhaltet die Solution Assemblies, die in den „Global Assembly Cache“ kopiert werden müssen, muss dies explizit vom Benutzer mit dem Parameter „-allowgacdeploy“ erlaubt werden.

#### *8.4.5.3. Activate Feature (Enable-SPFeature / stsadm –o activatefeature …)*

Die installierten Features werden nun im gewünschten Gültigkeitsbereich aktiviert. Farm-Features automatisch auf der ganzen Farm, Site-Features in der entsprechenden SharePoint Site (muss als Parameter angegeben werden).

Dieser Ablauf inklusive Deinstallation der vorherigen Version („Retract Solution“) sowie Neustart des Application Pools auf dem Webserver wird während der Entwicklung von VisualStudio automatisch gemacht. Über Pre-Deployment Scripts könnte man hier noch äussere Komponenten in den Ablauf integrieren wie z.B. das Zurücksetzen der Testdatenbank oder Anpassungen an den Berechtigungen im BDC Model (siehe 7.2.2) für einen produktiven Build.

# 9. Zusätzlich aufgetretene Probleme

## 9.1. SharePoint BDC Export

Da die SharePoint BDC Models trotz minimaler Implementation der BDC Schnittstelle (keine nicht direkt benötigten Methoden, Inhaltstypen oder Filter) mit ihrer XML Beschreibung sehr komplexe Konstrukte sind, wurde bei der Umsetzung oft der SharePoint Designer zu Hilfe genommen und das Model wieder exportiert. Hier scheint aber der SharePoint Server noch einige Bugs zu haben, wurden in den exportierten XML-Files teilweise solche Stellen entdeckt:


Abbildung 25: Screenshot Exportproblem

Schlussendlich mussten also alle Änderungen direkt im XML erfolgen. Hier gilt noch zu beachten, dass beim Deploy-Vorgang die Versionsnummer alleine darüber entscheidet ob das Model erneut auf den Server installiert wird.

## 9.2. Fehlermeldungen

Ausführliche Fehlermeldungen erleichtern die Entwicklung sehr. Leider sind Fehlermeldungen in SharePoint oft schwer lesbar, oder völlig unbrauchbar.


Abbildung 26: Beispiel eines Fehlers ohne Fehlermeldung bei der Verwendung des Kommandozeiletools stsadm zur Administration von SharePoint in der PowerShell

# 10. Mögliche Alternative: Microsoft Dynamics CRM/xRM

Da die Entwicklung von Anwendungen mit relationalen Datenmodellen für SharePoint sehr aufwändig ist, betrachten wir in diesem Kapitel ein alternatives Produkt, welches für solche Anwendungen besser geeignet ist.

Microsoft bietet für die für „Line of Business“ Applikationen das Produkt Microsoft Dynamics CRM und dessen Framework Microsoft Dynamics xRM.

## 10.1. Überblick

Microsoft Dynamics CRM und Microsoft SharePoint sind fast gleichzeitig, unabhängig und mit anderen Zielen entstanden. Während Microsoft SharePoint ein anfangs ein reines Dokumentemanagement System mit Funktionen für Zusammenarbeit und Inhaltssuche war, war Microsoft Dynamics CRM eine Software zur Verwaltung von Verkaufs- und Kundeninformationen. Beide Produkte sind inzwischen aber Produkte für die tägliche Erfassung von flexibel definierbaren Geschäftsdaten und weisen viele Gemeinsamkeiten auf.

Dennoch gibt es einen bedeutenden Unterschied zwischen den beiden Produkten, weshalb Microsoft auch auf beide Produkte setzt. Die beiden Produkte ergänzen sich aber auch und können durchaus nebeneinander verwendet werden.

### 10.1.1. Ursprüngliche Kerngebiete von SharePoint

SharePoint hat seine Stärken bei Dokumenteverwaltung, Zusammenarbeit und der öffentlichen oder internen Präsentation der Daten auf einem integrierten Web-Portal.
Daten können mit den entsprechenden Berechtigungen auch von Zuhause aus Online ohne VPN-Client erreicht werden. Dazu bietet SharePoint eine Versionskontrolle und eine leistungsfähige Volltextsuche um die Dokumente zu durchsuchen.

### 10.1.2. Ursprüngliche Kerngebiete von Dynamics CRM

Da Microsoft Dynamics CRM ursprünglich eine Software zur Verwaltung von Kundeninformationen war, ist die Software vor allem bei Kontaktemanagement, Controlling, Auswertung und Analyse grosser Mengen geschäftsrelevanter Daten besonders stark.
Das Kontaktemanagement kann direkt als Erweiterung in Microsoft Outlook verwendet werden.

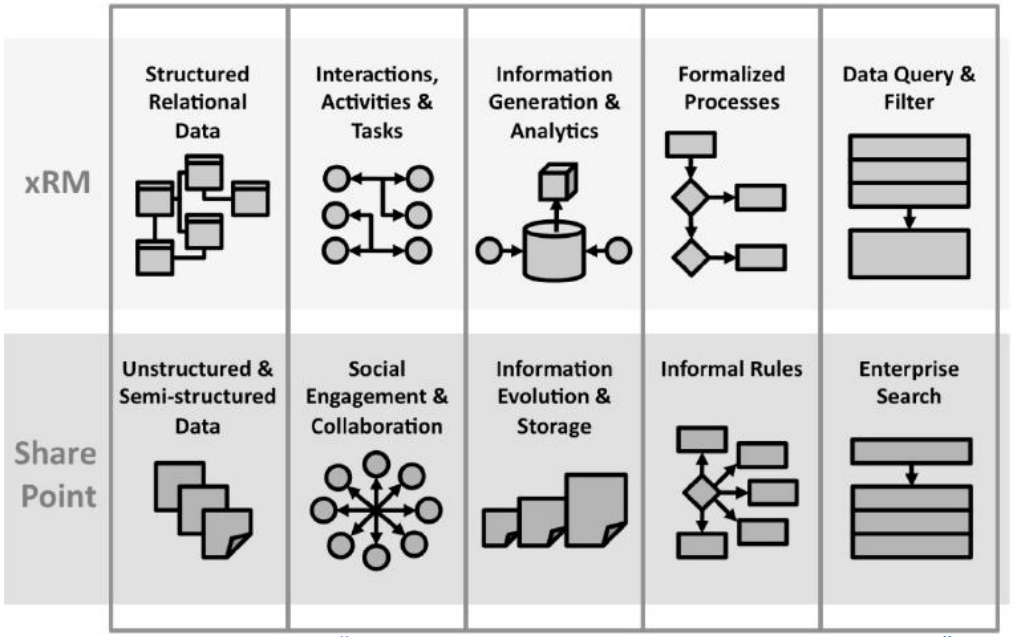



**Abbildung 27: Schematische Übersicht über der Kerngebiete der beiden Systemen[9]**

[9] Quelle: xRM Framework and Microsoft SharePoint http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Components-PostAttachments/00-09-98-03-25/xRM-Framework-and-Microsoft-SharePoint.pdf

## 10.2. Vergleich der Funktionalität

### 10.2.1. Relationale Daten

SharePoint ist auf Linear strukturierte Daten wie Dokumente und Webseiten ausgelegt, bietet aber auch minimale Möglichkeiten Relationen zwischen den Daten ohne viel Aufwand zu erstellen (z.B. Lookup Felder). Weitere Funktionalität für relationale Daten erfordert aber einen wesentlichen Mehraufwand, verglichen mit von SharePoint losgelösten und mit Einsatz von modernen Tools entwickelten Applikationen.

Microsoft Dynamics bietet die Möglichkeit ohne grossen Aufwand Datenobjekte so zu modellieren, dass im Hintergrund ein relationales Datenbankmodell erstellt wird, mit welchem die vollen Möglichkeiten von OLAP (Online Analytical Processing) ausgeschöpft werden können. Genau diese Stärken wären bei SURVEAST erwünscht.

### 10.2.2. Daten Import und Export

Microsoft SharePoint zeigt klare Stärken beim Import und Export von Daten mit Microsoft Excel. Für die Integration von Daten aus Drittsystemen bieten sich die BCS an.

Microsoft Dynamics bietet ein Import Wizard Interface welches bei der Integration von Excel oder Access Daten ähnliches ermöglicht. Dieses ist aber in erster Linie für den Administrator gedacht. Sonst sind im Microsoft Dynamics Framework etliche Möglichkeiten zur Integration von Webservices und Datenquellen vorhanden, welche die Möglichkeiten der BCS bei weitem übersteigen.

### 10.2.3. Administration

Dynamics wird üblicherweise zentral administriert, während Microsoft SharePoint darauf ausgelegt ist, dass jede Organisationseinheit im Betrieb eine eigene Seite erhält, welche durch den „Site Owner“ administriert wird.

### 10.2.4. Sicherheit

Dynamics hat verglichen mit SharePoint ein von Grund auf verschiedenes Security Model. Während bei SharePoint die Rechte auf Seiten und oder Datentypen festgelegt werden, bietet Dynamics „RowLevel Security“. Das heisst, dass Daten des gleichen Typs pro Eintrag für verschiedene Gruppen Berechtigungen haben können. Dies wäre vor allem bei Ereignissen in SURVEAST sinnvoll.

### 10.2.5. Softwareentwicklung

Geht man über die standardmässig integrierten Funktionalitäten von SharePoint hinaus, welche man bequem mit SharePoint Designer konfigurieren kann, bietet Microsoft SharePoint 2010 die Möglichkeit mit Hilfe von Visual Studio 2010 Erweiterungen zu entwickeln. Leider sind diese Entwicklungstools aber vor allem im Zusammenhang mit den Business Connectivity Services und Silverlight noch nicht sehr weit fortgeschritten.

Microsoft Dynamics CRM baut auf dem Microsoft Dynamics xRM Framework auf, welches es einem Entwickler ermöglicht „Line Of Business“ Applikationen leicht in Microsoft Dynamics CRM zu integrieren. Microsoft entwickelt dieses Framework, überlässt aber die Entwicklung von Endlösungen ihren Partnern. Ähnlich wie bei Microsoft Access sind in vielen Fällen bei Microsoft Dynamics nur minime Programmierkenntisse nötig um Erweiterungen zu erstellen. Dennoch ist Wissen zur Planung einer relationalen Datenbank erforderlich.

### 10.2.6. Offline und mobile Verwendung

Microsoft SharePoint kann offline nicht verwendet werden. Für Mobilgeräte ist es nicht optimiert. Microsoft Dynamics bietet einen Offlinemodus. Daten können ohne eine Verbindung zum Internet laufend bearbeitet werden. Bei der nächsten Gelegenheit werden die Daten wieder synchronisiert. Ausserdem bietet Microsoft Dynamics die Möglichkeit zur Bedienung mit Windows Mobile PocketPCs oder über ein optimiertes Mobilebrowser UserInterface mit beliebigen Smart Phones.

## 10.3. Zusammenarbeit der beiden Systeme

Eine Zusammenarbeit der beiden Systeme könnte folgendermassen aussehen:

In Dynamics werden die Daten verwaltet und Analysiert, SharePoint dient als Einstiegspunkt um die Daten für den ganzen Betrieb oder gewisse Abteilungen zur Betrachtung zur Verfügung zu stellen.

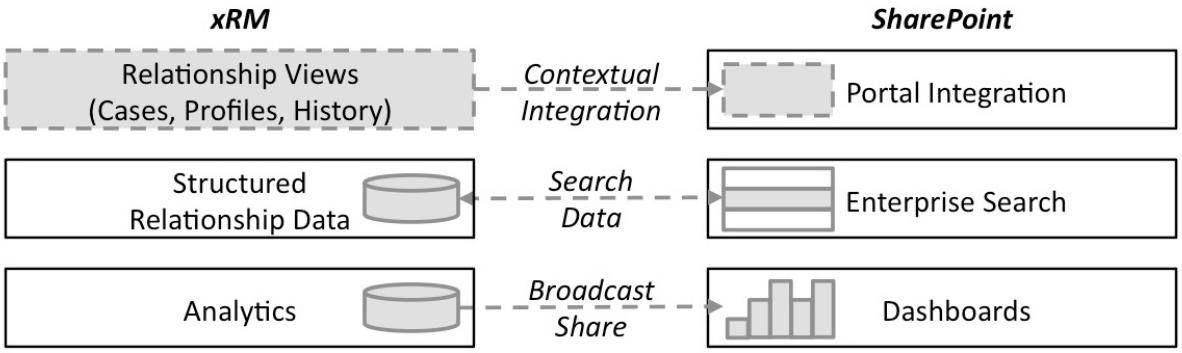



**Abbildung 28: Mögliche Zusammenarbeit von SharePoint und Dynamics xRM[10]**

## 10.4. Zusammenfassung

SURVEAST ist zwar nicht eine Line of Business Applikation im klassischen Sinn, dennoch stellt es Anforderungen welche nur mit einer solchen Funktionalität zu erfüllen sind. Microsoft SharePoint ist kein relationales Datenbanksystem. Es ist zwar möglich, aber sicher nicht sinnvoll so eine Software mit Microsoft SharePoint umzusetzen.

"*Dynamics CRM is a stronger platform for most line of business applications than SharePoint alone, unless your goal is to provide full employment for developers. It is a truism of software development that anything is possible with enough time and money, but you probably do not want to prove this with your own time or money*."[11] – James J. Townsend, Infostrat.com, Microsoft Gold Certified Partner

[10] Quelle: xRM Framework and Microsoft SharePoint http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Components-PostAttachments/00-09-98-03-25/xRM-Framework-and-Microsoft-SharePoint.pdf, letzter Zugriff 22.12.2010

[11] Microsoft Dynamics CRM and SharePoint Platforms - Choosing the Right Tools for the Solution http://www.infostrat.com/NR/rdonlyres/424C528E-0C5F-4BFB-BE32-1B726D8EB90B/166/CRMSharePointWhitePaperBooklet1.pdf, letzter Zugriff 22.12.2010

# 11. Ergebnisse

Die Benutzung von SharePoint als Architekturgrundlage für eine Business Applikation wie SURVEAST war für dieses Projekt ein gegebener Umstand. Nach den Erfahrungen und Versuchen sollen in diesem Kapitel nochmals alle Aspekte der Architektur mit einer konventionellen Business Applikation verglichen werden, wie sie mit modernen Frameworks machbar ist.

## 11.1. Daten- / Persistenzschicht

Die gängigen SharePoint Listen haben wegen ihrer Struktur bei komplexeren Auswertungen wesentliche Nachteile gegenüber einer klassischen relationalen Datenbank wie MS-SQL. Ausserdem ist die Performance ab einer gewissen Grösse vor allem bei kumulativen Abfragen (SUM, MAX, AVG, GroupBy) problematisch (siehe 7.1.3).

Für die Integration einer MS-SQL Datenbank in SharePoint werden die Business Connectivity Services benötigt. Hier werden die Entitäten der relationalen Datenbank mit externen Inhaltstypen und passenden Methoden abgebildet (siehe 7.2). Microsoft SharePoint 2010 bietet Linq2SharePoint, womit man komfortabel Daten aus SharePoint Listen abfragen kann. Leider bietet Linq2SharePoint diese Funktionalität nur für klassische interne Listen. Abfragen auf externe Listen welche Daten aus den BCS verwenden können, können nur mit CAML (7.2.4) formuliert werden.

Um eine bessere Erweiterbarkeit und Testbarkeit zu erreichen ist eine Trennung der Datenobjekte vom Datenprovider nötig. Dafür kann das gegebene Hilfsmittel SPMetal nicht verwendet werden, da dieses nur mit internen Listen funktioniert (7.2.3). Vergleicht den Aufwand zur Erstellung eines Domain Layers mit BCS mit dem Aufwand mittels eines modernen Frameworks (z.B. ADO.NET Entity Framework), so ist beim Einsatz des BCS sehr viel Zusatzaufwand erforderlich. Beim Technologie Prototyp für SURVEAST wird hier mit Connector-Klassen gearbeitet, die für jedes Domain Objekt einzeln implementiert werden müssen. Diese Connector-Klassen bieten aber nur grundlegende Funktionalität zum Laden, Schreiben und Lesen von Daten. Änderungen am Datenmodel müssen in allen Schichten (UI, Domain, Domain Connector, externe Listen, BCS, Datenbank) nachgezogen werden.

Solange also die eigentliche Kernfunktionalität von SharePoint nicht benutzt wird und die Business Applikation ganz auf eigenständige Datenmodelle setzt, ist hier kein Vorteil aber ein beachtlicher Mehraufwand aus der Integration in SharePoint erkennbar.

## 11.2. Security

Das Sicherheitskonzept von SharePoint basiert gänzlich auf dem Active Directory. Dies macht für interne Applikationen durchaus Sinn und lässt sich auch so in der Applikation implementieren.

Dieses Konzept könnte man aber ohne grossen Aufwand auch in eine eigenständige .NET Applikation einbinden. Alle benötigten Libraries werden vom .NET Framework zur Verfügung gestellt. SharePoint stellt hingegen eine zusätzliche Hürde dar, da auf den benötigten Zwischenschichten (BCS, externe Listen) die Berechtigungen auch entsprechend gesetzt werden müssen. Da die Änderungen an diversen Stellen redundant vollzogen werden müssen, wächst das Risiko für Fehler.

Zugriffschutz auf Datensatzebene ist mit externen Listen nur sehr schwer implementierbar. Zum Beispiel Microsoft Dynamics CRM bietet dies schon als Standardfunktionalität.

Im Bereich Security bringt die Integration in Microsoft SharePoint speziell mit der Verwendung der BCS keine Vorteile.

## 11.3. User Interface

Wegen der Anforderungen an die Bedienbarkeit wurde entschieden, die für den Technologie Prototypen die Kompontenten der Benutzeroberfläche mit Silverlight Webparts umzusetzen. Der Aufwand bei SURVEAST, die normalen List Views von SharePoint auf unseren Anwendungsfall anzupassen wäre beträchtlich höher, als direkt eigene Silverlight Komponenten zu verwenden. Zusätzlich wird mit Silverlight durch die reduzierte Anzahl an Page PostBacks der Bedienkomfort weiter erhöht.

Die Verwendung von normalen List Views mit externen Listen ist dadurch eingeschränkt, dass List Workflows mit externen Listen nicht direkt funktionieren. Die SharePoint Workflow Engine zu

verwenden, macht ausserdem keinen Sinn, da der Workflow vom Benutzer nicht anpassbar sein muss. Ausserdem werden keine standardmässigen SharePoint Datenstrukturen (z.B. Kalender) verwendet.

Durch das Client Object Model stellt SharePoint eine API zur Verfügung, mit welcher Teile der SharePoint Funktionalität direkt auf einem Client verwendet werden können. Die „Microsoft.SharePoint.Client.Silverlight“ Assembly von Silverlight bietet ein Subset des Client Object Models an. Leider fehlen in der Assembly für Silverlight aber viele Funktionen welche das Client Object Model sonst bieten würde. Es fehlt beispielsweise die Möglichkeit externe Inhaltstypen von BCS direkt anzusprechen, weshalb ein weiterer Umweg über externe Listen gemacht werden muss, welcher wieder einen Mehraufwand verursacht.

Auch hier ist also kein direkter Nutzen durch die Integration in SharePoint erkennbar.

## 11.4. Deployment und Wartung

In diesem Punkt liegt der Vorteil der SharePoint Integration klar auf der Hand. Durch die Auslieferung als Solution kann fast die ganze Lösung in einem Schritt installiert werden. Lediglich die Datenbank muss separat auf einem SQL Server vorhanden sein. Läuft diese können sowohl die Farm-Level-Features (wie z.B. die externen Inhaltstypen) als auch die Site-Features (Webparts) der Applikation aus dem Visual Studio auf eine SharePoint Farm installiert werden. Es ist also keine zusätzliche Webserverinstanz nötig, was bei einer unabhängigen Webapplikation der Fall wäre.

Auch Updates oder eine allfällige Deinstallation kann so zentral pro Feature erfolgen.

## 11.5. Tests

Um jede Schicht der Applikation automatisiert testen zu können, wurden diverse Versuche mit UnitTests durchgeführt. Leider lässt sich die BCS Schicht sehr schlecht automatisiert testen. Durch die direkte Anbindung des BCS an die Datenbank besteht hier keine Möglichkeit direkt in den Code einzugreifen um diesen getrennt von der Datenbank zu testen.

Durch die Silverlight Test Komponenten des Silverlight 4 Frameworks lassen sich aber Tests durchführen, die an den externen Listen ansetzen können. Diese Tests sind leider sehr langsam und erschweren die konstante Integration der Tests in die Entwicklung deshalb wesentlich. Die höheren Schichten (Domain und ViewModel) lassen sich ebenfalls durch das Silverlight Testframework testen. Darin unterscheidet sich die Architektur mit SharePoint kaum von einer normalen Webapplikation mit Silverlight.

# 12. Schlusswort

Microsoft SharePoint ist als Dokumentenverwaltung und für die Verwaltung von linearen Daten konzipiert und dafür auch sehr geeignet. Einfache Formulare, Listenansichten und Workflows können leicht und ohne Programmierkenntnisse erstellt werden. Es wird sehr viel Funktionalität geboten um die tägliche Zusammenarbeit in einer Firma zu erleichtern. Vor allem die hervorragende Zusammenarbeit von SharePoint mit Microsoft Office Produkten ist speziell hervorzuheben.

Müssen die Daten aber komplex ausgewertet werden (z.B. um Regelmässigkeiten festzustellen), so sind relationale Datenmodelle besser geeignet. Diese lassen sich aber nur mit bedeutendem Mehraufwand sinnvoll in SharePoint 2010 integrieren. Es kann durchaus sein, dass SharePoint und die dazugehörenden Entwicklungstools für solche Anwendungen in den nächsten Versionen weiter ausgebaut werden. Allerdings bietet Microsoft selbst bereits für die Erfassung, Verwaltung und Auswertung strukturierter Daten eine ausgewachsene und geeignete Produktreihe: Microsoft Dynamics.

Für die Forschung am Kantonsspital sind Möglichkeiten zur effizienten Auswertung der Daten essentiell. Es ist sehr zu empfehlen den Entscheid nur noch SharePoint als Plattform für sämtliche Applikationen zu verwenden, nochmals zu überdenken. Für die Implementation von Applikationen wie SURVEAST gibt es Plattformen, welche eindeutig geeigneter sind.

Abbildung 29: Das falsche Werkzeug verwenden.[12]

[12] Quelle: Microsoft Dynamics CRM and SharePoint Platforms - Choosing the Right Tools for the Solution http://www.infostrat.com/NR/rdonlyres/424C528E-0C5F-4BFB-BE32-1B726D8EB90B/166/CRMSharePointWhitePaperBooklet1.pdf, letzter Zugriff 22.12.2010

# 13. Erfahrungsberichte

## 13.1. Erfahrungsbericht von Clemens Meier

Schon seit mehreren Jahren bearbeite ich als Nebenerwerb für die Abteilung Infektiologie des Kantonsspitals St. Gallen kleinere Software Projekte auf verschiedenen Technologien. Im Herbst 2009 wurde angefragt, ob ich meine Semesterarbeit für das Kantonsspital schreiben würde.
Ich wollte mich in erster Linie ausgiebig mit Microsoft .NET Technologien beschäftigen, da ich mich auch nach dem Studium möglichst in diese Richtung weiterentwickeln möchte, aber erst Grundkenntnisse besass. Da das Kantonsspital zukünftig voll auf Microsoft Technologien setzen will, klang die Idee für mich sehr Interessant.

Erste Konzeptpapiere der Software SURVEAST zur Auswertung von Ursachen für im Spital erworbene Infektionen, entstanden schon vor 2 Jahren. Die Software konnte aber wegen Personalproblemen nicht entwickelt werden. Die Ursprüngliche Idee war nun, als Semesterarbeit ein Pilotmodul zu entwickeln, welches sich auf die Erfassung und Auswertung von Infekten bei Leukämiepatienten konzentriert.

Mit Silvan Gacond fand sich ein Projektpartner, welcher sich ebenfalls für ein Projekt mit Microsoft Technologien interessierte. Ausserdem absolviert er genau wie ich das Studium im Teilzeitpensum. Wegen dem klaren Fokus auf Microsoft .NET Technologien fragten wir Herrn Huser an, ob er unsere Arbeit betreuen und ausschreiben würde.

Schon die Anforderungsanalyse stellte eine erste Herausforderung dar. Einerseits erschwerten die vielen medizinischen Begriffe den Einstieg in die Materie, andererseits erwies sich das bestehende Konzeptdokument als so für uns nicht direkt verwendbar. Nach der vierten Woche konnten wir planmässig unsere Anforderungsanalyse für das Pilotmodul abschliessen.

Der Informatikdienst des Kantonsspitals gab von Anfang an Microsoft SharePoint als Plattform für SURVEAST vor. Doch während dem Technologiestudium in den ersten 4 Wochen unserer Arbeit wurde uns immer bewusster, dass SharePoint eigentlich nicht für die Implementation einer solchen Software gemacht ist. So suchten wir zuerst alternative Technologien für die Umsetzung des Projektes, welche mit SharePoint zusammen arbeiten könnten. Wir beschäftigten uns unter anderem mit ASP.NET MVC, WebForms und Silverlight und wollten die Applikation als WebPart in SharePoint integrieren.

Ende der Woche 4 des Projektes kam vom Informatikdienst des KSSG aber ganz klar die Richtlinie, dass die Applikation voll in SharePoint integriert werden müsse. Der Informatikdienst teilte unsere Ansicht nicht, dass SharePoint die Anforderungen an SURVEAST nicht erfüllen kann.

Die vollständige Integration in Microsoft Sharepoint bedeutet leider einen beachtlichen Mehraufwand. Unsere Auftraggeber wollten in erster Linie eine Applikation, mit welcher sie Daten erfassen und auswerten können. Der Informatikdienst entscheidet aber schlussendlich, ob die Software eingesetzt wird. So wurde entschieden, einen Architekturprototypen mit Hilfe der Business Connectivity Services (BCS) zu erstellen, welcher die grundlegende Funktionalität erfüllt.

Die Implementation des Prototypen ging allerdings nur schleppend voran. Wir versuchten es mit mehreren verschiedenen Architekturmöglichkeiten. Am Schluss mussten wir realisieren, dass sich unserer Vermutungen bestätigt haben und die Technologie (BCS) nicht für diesen Zweck gedacht ist. Sie ist lediglich dafür konzipiert Daten aus Fremdsystemen anzuzeigen und kleine Mengen an Daten zu schreiben. Darum mussten wir immer wieder Umwege gehen, die nur schwer nachzuvollziehen sind. Technische Hilfe im Internet zu finden war oft fast nicht möglich, da anscheinend fast niemand solche Applikationen mit BCS implementiert. Ausserdem sind wir auf sehr viele Unwahrheiten oder Halbwahrheiten im Internet gestossen und mussten darum speziell darauf achten, nur vertrauenswürdige Quellen zu verwenden.

Schlussendlich konzentrierten wir uns darauf, die Schwierigkeiten zu beschreiben welche eine Implementation einer Software wie SURVEAST mit Microsoft SharePoint stellt. Die Infektiologie des Kantonspitals ist darauf angewiesen, dass auch in Zukunft ähnliche Systeme für die Forschung verwendet werden können. Der Entscheid nur Microsoft SharePoint als Plattform zu verwenden, ist aus technischer Sicht sicher zu überdenken. Vor allem bei der Betrachtung von Microsoft Dynamics, wird klar, dass es für solche Anwendungen geeignetere Plattformen gibt.

Während dieser Semesterarbeit habe ich durch die Beschäftigung mit verschiedensten Microsoft Technologien sehr viel gelernt. Dass das Verhältnis von Aufwand und Resultat vor allem beim implementierten Prototypen eher ernüchternd war, ist etwas frustrierend. Neben den technischen Erfahrungen, waren aber auch sicher die Erfahrungen in der Teamarbeit sehr wichtig. Silvan und ich haben es geschafft sehr viele Hürden zusammen zu überwinden. Wir sind zwei Personen die beide sehr unterschiedlich Handeln und Denken, was sich aber während der gesamten Arbeit oft als Vorteil erwies, da wir uns recht gut ergänzen. Wir haben uns in schwierigen Situationen gegenseitig unterstützt.

Zum Schluss möchte ich noch allen Personen Danken, die uns bei dieser Arbeit unterstützt haben. Namentlich Herr Huser der unsere gesamte Arbeit betreut und viel Verständnis für unsere Kursänderungen während der Arbeit aufbrachte. Ausserdem möchten ich Dr. Matthias Schlegel und Dr. Christian Kahlert danken, welche uns eine interessante Aufgabenstellung und konstruktive Sitzungen boten. Aber auch Herr Jucker, der unsere Fragen zu Microsoft SharePoint immer kompetent beantworten konnte.

## 13.2. Erfahrungsbericht von Silvan Gacond

Bereits einige Zeit im Vorfeld der Studienarbeit, die sich nun langsam dem Ende zu neigt, hatten Clemens Meier und ich den Entschluss gefasst, zusammen diese Arbeit zu schreiben. Da wir beide in der selben Situation sind und neben dem Studium noch einem Erwerb nachgehen und beide sonst fast keine Module mehr zu absolvieren hatten, standen die Vorzeichen für eine gute Zusammenarbeit sehr gut. Durch Clemen Meiers Kontakte zum Kantonsspital St.Gallen konnten wir schon früh eine Arbeit finden, die im Bezug auf die Aufgabe sehr interessant klang. Das vom Auftraggeber der Fokus klar auf Microsoft .NET Technologien gesetzt wurde, war mir persönlich mehr als recht. Dieses Gebiet interessiert mich sehr und ich möchte mich möglichst auch nach dem Studium in diese Richtung entwickeln. Also haben wir Herrn Huser angefragt, ob er bereit wäre, diese Arbeit mit uns durchzuführen.

Nach den üblichen Vorbereitungen begannen wir die Arbeit mit einer ersten Anforderungsanalyse, welche sich natürlich durch die vielen Medizinischen Begriffe und Gegebenheiten schon als erste Herausforderung herausstellte. Dennoch waren wir guten Mutes am Ende der Arbeit ein komplettes Pilotmodul abgeben zu können. Dass dieses Modul schlussendlich in einer SharePoint Umgebung betrieben werden sollte, war von vorne rein klar. In wie fern diese Integration aber stattfinden sollte, war am Anfang noch ziemlich offen, also konzentrierten wir uns zunächst auf den funktionalen Teil, während parallel einige mögliche Technologien zur Umsetzung angeschaut wurden.

Gegen Ende der vierten Semesterwoche, nahm das Projekt eine einschneidende Wendung: Von den Informatikdiensten des Kantonsspitals kam die klare Anforderung, das Projekt müsse vollständig in SharePoint integriert werden. Wir hatten kurz danach eine Sitzung mit einem SharePoint Entwickler des Kantonsspitals, der uns mögliche Wege zur Integration aufzeigte. Seine Botschaft war ganz klar: Unser bisheriger Eindruck, dass SharePoint nicht unbedingt die beste Plattform für solche eigenständige Applikationen sei, sei falsch.

Nach intensivem Studium der möglichen Technologien zur vollständigen SharePoint Integration wurde schnell klar, dass diese Integration ein beachtlicher Mehraufwand darstellt. Also suchten wir das Gespräch mit unseren Auftraggebern. Es wurde entschieden, den Fokus der Arbeit auf der Technologie zu belassen und eine Architektur abzuliefern, mit welcher eine Entwicklung des gesamten Projektes nach den Vorgaben der Informatikdienste möglich wäre. Leider war es in diesem Zeitrahmen nicht mehr möglich, die ursprünglich geplante Funktionalität umzusetzen, dies war allen Beteiligten klar.

Rückblickend war die Arbeit auch nach dieser Wendung zwar sehr spannend, teilweise aber auch sehr demotivierend. Angetrieben von der Aussage, SharePoint sei eine sehr geeignete Plattform für die Entwicklung einer solchen Applikation, haben wir oft gezweifelt, wirklich den richtigen Weg zur Umsetzung eingeschlagen zu haben. Je tiefer die Recherchen aber gingen, desto mehr Belege wurden gefunden um zu zeigen, dass unsere Eindrücke durchaus der Realität entsprachen. Wie oft in der Informatik scheint hier aber leider das falsche Werkzeug für die falsche Anwendung eingesetzt worden zu sein. In diesem Sinne war es trotzdem motivierend die Arbeit so weiter zu ziehen, um wenigstens mit handfesten Tatsachen und Beispielen zeigen zu können, dass hier ein falscher

Strategieentscheid gefällt wurde. Da das Gesamtprojekt SURVEAST innerhalb des Kantonsspitals weiter gezogen werden soll, wird unsere Arbeit wohl nun als Grundlage genommen diesen Entscheid für ein derart umfangreiches Projekt nochmals zu überdenken.

Nicht unerwähnt bleiben darf natürlich auch die Zusammenarbeit mit allen, die an dem Projekt beteiligt waren. Auch wenn es nicht das erste Projekt war, welches Clemens und ich zusammen absolviert haben, ist dies immer ein springender Punkt welcher schlussendlich einen wichtigen Teil über Erfolg oder Misserfolg ausmacht, gerade wenn die eigentliche Arbeit doch die eine oder andere unerwartete Hürde mit sich bringt. Hier kann man mit gutem Gewissen sagen, dass wir zwar in einigen Punkten nicht gleich denken und handeln, dies aber überhaupt keinen negativen sondern eher einen positiven Einfluss auf die Arbeit hatte. Wir haben beide unsere Stärken und haben uns so sehr gut ergänzt. Auch die Unterstützung von Herrn Huser, welcher uns auch in der Phase unterstütze, als ein Umdenken und Umplanen der Arbeit nötig war, wer sehr wertvoll.

Darum möchte ich mich bei allen Beteiligten für die Unterstützung Bedanken. Namentlich natürlich Herr Huser der uns sehr unkompliziert und effizient betreut hat, Herr Jucker seinem Mitarbeiter im INS der uns kompetent Auskunft über die SharePoint technischen Fragen geben konnte und auch Dr. Matthias Schlegel und Dr. Christian Kahlert für die spannende Aufgabenstellung und die konstruktiven Sitzungen.

## 14. Glossar

| | |
|---|---|
| API | Application Programming Interface - Bibliothek mit Funktionen die von anderen Applikationen aus verwendet werden können |
| Assembly | Deployment-Einheit |
| BCS | Business Connectivity Services |
| BDCM | Business Data Connectivity Metadata |
| CAB Archiv | „Cabinet“ Archivformat, ursprünglich von Microsoft eingeführtes Archivformat. |
| CAML | Collaborative Application Markup Language, XML basierte Abfragesprache für Abfragen auf Sharepoint Listen |
| Cmdlet | „Commandlet“: Kommandozeilenprogramm zum Aufruf in der Microsoft PowerShell. |
| Codeplex | Microsofts Plattform für Opensource Projekte |
| Collation (SQL Server) | Verwendeter Zeichensatz in der betreffenden SQL Server Tabelle |
| Data Binding | Verknüpfung der Businesslogik mit dem User Interface |
| Entity Framework | Moderner Object-Relational Mapper von Microsoft |
| External List | Eingeschränktes List Interface für die Verwendung von Daten von Drittsystemen under Verwendung des BCS |
| Feature (SharePoint) | Deployment-Einheit für SharePoint Inhalte |
| GUI | Graphical User Interface |
| GUID | Global Unique Identifier, pseudozufällig generierte, 128bit lange Zahl. Die grosse Länge kommt daher, dass eine solche Zahl theoretisch Weltweit möglichst eindeutig sein sollte. |
| IIS | Internet Information Services, Webserver von Microsoft, integriert in Windows 2008 Server und Windows 7 |
| Linq | Language Integrated Query – Abfragesprache für C# mit Syntaxüberprüfung zur Kompilierzeit, es bestehen verschiedene Linq Adapter welche unter anderem Abfragen auf SQL Datenbanken (Linq2SQL) oder XML (Linq2XML) ermöglichen. |
| Linq2Caml | Opensource Projekt, Linq Adapter für CAML – wird seit 2009 nicht mehr weiterentwickelt und kann nicht mehr von Codeplex heruntergeladen werden |
| Linq2Sharepoint | Linq Adapter welcher CAML Queries im Hintergrund erstellt und ausführt, eingeführt mit SharePoint 2010, funktioniert nicht mit externen Listen |
| LOB-System | Line of Business System - |
| MSDN | Microsoft Developer Network, Informationsplattform für Microsoft Entwickler http://www.msdn.com |
| MVVM | Model-View-ViewModel: „Best Practice“ für die Implementierung von grafischen Benutzeroberflächen mit WPF & Silverlight |
| ODBC | Open Database Connectivity, standardisierte Datenbankschnittstelle |
| RIA Services | Rich Internet Application Services |
| SharePoint Client Object Model | API für den Zugriff eines Clients auf die Funktionalität des SharePoint Servers. Unterstützt .NET Managed Clients, Silverlight und ECMA-Script |
| SharePoint Farm | SharePoint Server Installation - kann auf einem oder mehreren physikalischen Servern betrieben werden |
| Sharepoint List | Standard Datenstruktur in Microsoft SharePoint |
| Silverlight | Browser Plugin von Microsoft für die Erstellung von Rich Internet Applications, Implementiert ein Subset der .NET Runtime (unter anderem auch WPF) |
| SPMetal | Tool zur automatischen Erstellung von Domain Klassen und Trennung des Domain Layers von Daten Provider (Repository Pattern), unterstützt nur interne SharePoint Listen |
| WCF | Windows Communication Foundation |
| Workflow | Vordefinierte Abfolge von Aktivitäten in einem Programm / einer Organisation |
| WPF | Windows Presentation Foundation |
| XAML | Extensible Application Markup Language, XML-Sprache von Microsoft, unter anderem verwendet für die deklarative Beschreibung von GUI’s |

## 15. Referenzen


- Microsoft Developer Network – SharePoint Developer Center http://msdn.microsoft.com/en-us/sharepoint/default, letzter Zugriff 22.12.2010
- Microsoft Developer Network – Developing Applications for SharePoint 2010 http://msdn.microsoft.com/en-us/library/ff798432.aspx, letzter Zugriff 22.12.2010
- Microsoft Developer Network – Internal Lists vs. BCS http://msdn.microsoft.com/en-us/library/ff798319.aspx, letzter Zugriff 22.12.2010
- SharePoint 2010 – BCS Limitationen und Alternativen zu externen Listen, http://www.layer2.de/de/community/blog/archive/2010/08/29/sharepoint-2010-bcs-limitationen-und-alternativen-zu-externen-listen.aspx, letzter Zugriff 22.12.2010
- Everything you need to know about BDC (SharePoint 2007) http://sharepointmagazine.net/technical/administration/everything-you-need-to-know-about-bdc-part-1-of-8, letzter Zugriff 22.12.2010
- BDC-Model Infrastructure http://msdn.microsoft.com/en-us/library/ee556378.aspx, letzter Zugriff, 22.12.2010
- CAML Referenz http://msdn.microsoft.com/en-us/library/ms467521.aspx, letzter Zugriff 22.12.2010
- Microsoft Dynamics CRM and SharePoint Platforms - Choosing the Right Tools for the Solution http://www.infostrat.com/NR/rdonlyres/424C528E-0C5F-4BFB-BE32-1B726D8EB90B/166/CRMSharePointWhitePaperBooklet1.pdf, letzter Zugriff 22.12.2010
- xRM Framework and Microsoft SharePoint http://blogs.msdn.com/cfs-file.ashx/__key/CommunityServer-Components-PostAttachments/00-09-98-03-25/xRM-Framework-and-Microsoft-SharePoint.pdf, letzter Zugriff 22.12.2010
- SharePoint is not a Database http://chriswoodill.blogspot.com/2009/05/sharepoint-is-not-database.html, letzter Zugriff 22.12.2010
- Leukämie-Lexikon http://www.leukaemie-online.de/index.php?option=com_glossary&Itemid=34, letzter Zugriff 22.12.2010
- Folien: ISV-Workshop: SharePoint 2010 (Reiner Ganser, Microsoft Innovation Center Rapperswil)
- Steve Fox, “Beginning SharePoint 2010 Development”, John Wiley & Sons, 1. Auflage (4. Juni 2010)
- Melanie Schmidt - Britta Seidler, „Microsoft SharePoint 2010 - Das Praxisbuch für Anwender“, Addison-Wesley, München, 1. Auflage (10. August 2010)
- Bill Evjen - Scott Hanselman - Devin Rader, “Professional ASP.NET 4 in C# and VB”, John Wiley & Sons, 1. Auflage (5. März 2010)

# SURVEAST

## Technologiestudie mit SharePoint

---

## *Anhang*

# 16. Projektplanung und Projektverlauf

Um die einzelnen Iterationen der Studienarbeit zu planen, wurde zu Beginn eine erste Grobplanung erstellt. Das ursprüngliche Ziel war, ein agiles Projektmanagement durchzuführen und die eigentlichen Ziele in kleinen Iterationen (3 – 4 Wochen) zu überprüfen und neu zu setzen.

Durch den Entscheid in der 4. Woche, eine vollständige SharePoint Integration anzustreben, wurde die eigentliche Zielformulierung entscheidend umgestellt. Die Projektplanung tangierte dies insofern, dass neue Ziele festgelegt wurden.

Die laufende Planung wurde intern mit einer ständig aktualisierten Todo-Liste nachgeführt. Darauf wurde auch vermerkt, wenn ein Punkt aufgegeben oder verschoben wurde. Diese Liste sowie die ursprüngliche Grobplanung sind auf den nächsten Seiten zu finden.

Folgende Übersicht soll die wichtigsten Phasen und Entscheide, sowie die neu festgelegten Ziele kurz aufzeigen:

**Wochen 1 – 4**
*Anfangsphase der Studienarbeit:*
- Auf Rat hin von Herrn Huser und Herrn Jucker (Mitarbeiter Institut für vernetzte Systeme), wird der Fokus der Arbeit zunächst mal auf die Anforderungsanalyse und die Entwicklung, möglichst losgelöst von SharePoint, gelegt.
- Aus Technologiesicht wurde damit gerechnet, die Applikation auf einer modernen Technologie für Rich Internet Applications mit Silverlight zu entwickeln und lediglich das Front-End der Applikation in SharePoint anzuzeigen.

**Woche 4, 14.10.2010**
*Treffen von Clemens Meier und Brian Douglas von den Informatikdiensten des Kantonsspital St.Gallen:*
- Die angestrebte Architektur mit einer eigenständigen Applikation welche in SharePoint lediglich angezeigt wird, wird von den Informatikdiensten des Kantonsspitals nicht akzeptiert.
- Es wird entschieden ein Treffen mit Ramon Brülisauer, SharePoint Entwickler am Kantonsspital zu vereinbaren.

**Woche 5, 18.10.2010**
*Treffen mit Ramon Brülisauer (SharePoint Entwickler, Kantosspital St.Gallen, Informatikdienste):*
- Die komplette Integration in SharePoint ist eine zwingende Anforderung der Informatikdienste des Kantonsspitals.
- Er empfiehlt uns die Verwendung der Business Connectivity Services zur Integration von SQL Server Daten innerhalb SharePoint.
- Silverlight UI Komponenten seien kein Problem, die aktuellste Laufzeitumgebung ist auf den Rechnern im Kantonsspital installiert.

**Wochen 5 – 8**
*Evaluation und Test von diversen Technologien zur vollständigen Integration in SharePoint:*
- Diverse Versuche mit verschiedenen Technologien zur SharePoint Integration haben gezeigt, dass die Umsetzung der ursprünglichen Anforderungsanalyse mit vollständiger SharePoint Integration nicht in der vorgegebenen Zeit möglich ist.
- In Absprache mit Herrn Huser wurde beschlossen, den Verantwortlichen des Kantonsspitals zwei Vorschläge zu präsentieren: Entweder die möglichst fortgeschrittene Implementation der ursprünglichen Anforderungen mit einer anderen Technologie (z.B. Silverlight RIA Services) oder der Aufbau einer vollständigen Architektur und eines einfachen Prototyps auf SharePoint.

**Woche 8, 8.11.2010**
*Sitzung mit Dr. Matthias Schlegel und Dr. Christian Kahlert:*
- Die ausgearbeiteten Vorschläge wurden den Verantwortlichen der Infektiologie, dem eigentlichen Auftraggeber des Projektes, dargelegt.
- Da eine andere Technologie aus dem derzeitigen Standpunkt des Informatikdienstes nicht bewilligt werden würde, wurde entschieden den Technologieprototypen mit SharePoint weiter zu entwickeln.

- Da es sich auch bei der ursprünglichen Anforderungsanalyse sowieso nur um ein Pilotmodul handelte, wird sich die vollständige Entwicklung von SURVEAST noch länger hinziehen.
- Es soll in den nächsten Wochen entschieden werden, wie das Projekt nach der Studienarbeit weitergezogen wird. Eventuell sollten Zivildienstleistende im Kantonsspital beteiligt werden. Dies würde erfordern, eine Art „How-To“ Dokument mitzuliefern, welches die Weiterentwicklung beschreibt.

**Wochen 8 - 11**
*Weiterentwicklung des Prototypen, Entwickeln einer erweiterbaren und möglichst flexiblen Architektur:*
- Die Erfahrungen und Erkenntnisse aus den letzten Wochen Entwicklung bestätigen sich immer mehr: Die vollständige Integration stellt ein erheblicher Mehraufwand dar.
- Es kann immer besser belegt werden, wo die Hauptprobleme der Integration liegen und warum der Mehraufwand entsteht.
- Der Prototyp mit Lese- und Schreibfunktionalität, sauberer Anwendung des MVVM Patterns in Silverlight und Versuche mit dem Deployment und Packaging zeigen dass die entwickelte Architektur in der Praxis mit SharePoint funktioniert.

**Woche 11, 1.12.2010**
*Sitzung mit Dr. Matthias Schlegel und Dr. Christian Kahlert:*
- Am Kantonsspital wurde entschieden, das Gesamtprojekt SURVEAST anfangs 2011 öffentlich auszuschreiben.
- Die Verantwortlichen der Infektiologie sehen den entstandenen Mehraufwand und den harzigen Fortschritt der Umsetzung. Darum möchten sie vor der öffentlichen Ausschreibung gegenüber den Informatikdiensten belegen können, dass die Entwicklung einer eigenständigen Applikation mit SharePoint einen Mehraufwand bedeutet, welcher in keinem Verhältnis zum Nutzen der Integration steht.

**Wochen 11 - 14**
*Endphase der Studienarbeit:*
- In Absprache mit Herrn Huser wird die verbleibende Zeit für die Fertigstellung eines einfachen Prototypen mit Lese- und Schreibfunktionalität sowie die Dokumentation der entstandenen Architektur verwendet.
- Ein Review der Dokumente und der Architektur durch Herrn Jucker bestätigt die gemachten Erfahrungen. Zusätzlich zeigt er uns mit Microsoft Dynamics CRM noch eine alternative Plattformlösung auf, welche als guter Vergleich in die Dokumentation einfliessen kann.

# 17. Ursprünglicher Projektplan

| | | Sprint 1 | | | | Sprint 2 | | | Sprint 3 | | | Sprint 4 | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | SW 1 | SW 2 | SW 3 | SW 4 | SW 5 | SW 6 | SW 7 | SW 8 | SW 9 | SW 10 | SW 11 | SW 12 | SW 13 | SW 14 |
| | | *20.9.-26.9.* | *27.9.-3.10.* | *4.10.-10.10.* | *11.10.-17.10.* | *18.10.-24.10.* | *25.10.-31.10.* | *1.11.-7.11.* | *8.11.-14.11.* | *15.11.-21.11.* | *22.11.-28.11.* | *29.11.-5.12.* | *6.12.-12.12.* | *13.12.-19.12.* | *20.12.-24.12.* |
| *Allgemeine Arbeiten* | | | | | | | | | | | | | | | |
| | Aufsetzen Infrastruktur | | | | | | | | | | | | | | |
| | Technologiestudium | | | | | | | | | | | | | | |
| *Anforderungsanalyse* | | | | | | | | | | | | | | | |
| | Identifizieren der Anforderungen an das Gesamtsystem | | | | | | | | | | | | | | |
| | **MS: Anforderungen an das Gesamtsystem klar** | | | | **Sitzung KSSG, 13.10.** | | | | | | | | | | |
| | UseCases | | | | | | | | | | | | | | |
| | Domain Model | | | | | | | | | | | | | | |
| | Auswertungen | | | | | | | | | | | | | | |
| | Externes Design | | | | | | | | | | | | | | |
| *Design* | | | | | | | | | | | | | | | |
| | Logische Architektur | | | | | | | | | | | | | | |
| | Internes Design Datenhaltung | | | | | | | | | | | | | | |
| | Internes Design Problem Domain | | | | | | | | | | | | | | |
| | Internes Design User Interface | | | | | | | | | | | | | | |
| *Implementation* | | | | | | | | | | | | | | | |
| | Implementation Datenbank | | | | | | | | | | | | | | |
| | Implementation Erfassung (SharePoint Modul) | | | | | | | | | | | | | | |
| | Implementation Auswertungen | | | | | | | | | | | | | | |
| | **MS: Implementation beendet** | | | | | | | | | | | | | **Sitzung, tbd** | |
| *Integration* | | | | | | | | | | | | | | | |
| | Integrationsplanung und Test | | | | | | | | | | | | | | |
| *Testing* | | | | | | | | | | | | | | | |
| | Unit Tests | | | | | | | | | | | | | | |
| | Systemtests | | | | | | | | | | | | | | |
| *Prototypen* | | | | | | | | | | | | | | | |
| | **MS: Erster Prototyp** | | | | | | | **Sitzung, tbd** | | | | | | | |
| | **MS: Zweiter Prototyp** | | | | | | | | | | | **Sitzung, tbd** | | | |
| *Dokumentation* | | | | | | | | | | | | | | | |
| | Dokumentation Code | | | | | | | | | | | | | | |
| | Überarbeitung und Fertigstellung der restlichen Dokumente | | | | | | | | | | | | | | |
| | Poster, Abstract, Reflexion, etc. | | | | | | | | | | | | | | |

| Sprint | Bereich | Aufgabe | Zuständig | Priorität | Erl.-Soll | Erl.-Ist | v/a *[13] | Kommentar |
|---|---|---|---|---|---|---|---|---|
| 1 | Aufsetzen Infrastruktur | Installation Entwicklungsrechner | sg / cm | 1 | 22.09.2010 | 22.09.2010 | | |
| 1 | Organisation | Sitzungsprotokoll Sitzung KSSG | sg /cm | 2 | 22.09.2010 | 21.09.2010 | | |
| 1 | Organisation | Grober Projektplan | sg / cm | 2 | 22.09.2010 | 22.09.2010 | | |
| 1 | Aufsetzen Infrastruktur | Installation Server | sg / cm | 1 | 22.09.2010 | 22.09.2010 | | |
| 1 | Dokumentation | Grunddokumentstruktur Anforderungsanalyse | cm | 1 | 27.09.2010 | 27.09.2010 | | |
| 1 | Anforderungsanalyse | Anforderungen an das Gesamtsystem | cm / sg | 1 | 11.10.2010 | 11.10.2010 | | |
| 1 | Anforderungsanalyse | Use Cases | cm | 2 | 11.10.2010 | 11.10.2010 | | |
| 1 | Anforderungsanalyse | Domain Model | sg | 2 | 11.10.2010 | 11.10.2010 | | |
| 1 | Anforderungsanalyse | Externes Design | cm / sg | 2 | 11.10.2010 | 11.10.2010 | | |
| 1 | Anforderungsanalyse | Auswertungen | cm / sg | 2 | 11.10.2010 | 11.10.2010 | | |
| 1 | Dokumentation | Fertigstellen Anforderungsanalyse | cm / sg | 1 | 11.10.2010 | 11.10.2010 | | |
| 1 | Dokumentation | Review Anforderungsanalyse | sg | 1 | 11.10.2010 | 11.10.2010 | | |
| 1 | Technologiestudium | Sharepoint | sg / cm | 2 | 13.10.2010 | 18.10.2010 | | |
| 1 | Technologiestudium | asp.net webforms / mvc | cm | 2 | 13.10.2010 | 18.10.2010 | a | |
| 1 | Technologiestudium | Silverlight / RIA Services | cm / sg | 2 | 13.10.2010 | 18.10.2010 | a | |
| 1 | Technologiestudium | Technologie Prototyp SharePoint Listen | cm / sg | 2 | 13.10.2010 | 18.10.2010 | a | Wegen Usablility und sonstigen Einschrenkungen sinnlos |
| 1 | Dokumentation | Technologieentscheid Dokument | cm | 2 | 13.10.2010 | 18.10.2010 | a | Durch gegebene Umstände nicht mehr nötig |
| 1 | Organisation | Vorbereitung Sitzung KSSG | cm / sg | 1 | 13.10.2010 | 13.10.2010 | | |
| 1 | Organisation | Sitzungsprotokoll | sg | 1 | 13.10.2010 | 13.10.2010 | | |
| | | | | | | | | |
| 2 | Organisation | Zusammenfassung Sitzung KSSG (Ramon Brülisaurer) | sg | 1 | 18.10.2010 | 18.10.2010 | | |
| 2 | Internes Design | Datenhaltung | cm / sg | 1 | 20.10.2010 | 20.10.2010 | | |
| 2 | Implementation | Datenbankmodell | sg | 1 | 20.10.2010 | 20.10.2010 | | |
| 2 | Internes Design | Problem Domain | cm | 1 | 20.10.2010 | 20.10.2010 | | Problem Daten über BCS laden! |
| 2 | Technologiestudium | BCS | cm / sg | 2 | 20.10.2010 | 20.10.2010 | | |
| 2 | Technologiestudium | Internetrecherche BCS/Silverlight | cm | 2 | 20.10.2010 | 05.11.2010 | | Immer wieder neue Probleme -> Verspätung |

[13] verschoben (v) / aufgegeben (a)

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 2 | Implementation | BCS / External Content Types | cm / sg | 1 | 28.10.2010 | 26.11.2010 | v | sehr aufwändig -> Verspätung, Verschiebung aug Sprint 3 |
| 2 | Implementation | BCS mit .NET Assembly | cm | 1 | 28.10.2010 | | a | Nicht sinnvoll da keine Funktionalität in dieser Schicht nötig. |
| 2 | Implementation | Stored Procedures | sg | 1 | 01.11.2010 | 01.11.2010 | | |
| 2 | Implementation | BCS mit BCDM (XML-Definition) | cm / sg | 1 | 03.11.2010 | 26.11.2010 | v | Probleme mit BCS -> Verschiebung auf Sprint 3 |
| 2 | Implementation | Datenhaltung + Domain Layer | cm | 1 | 03.11.2010 | 26.11.2010 | v | Probleme mit BCS -> Verschiebung auf Sprint 3 |
| 2 | Prototypen | Paper Prototype Erfassung (UI) | sg / cm | 2 | 02.11.2010 | 02.11.2010 | | |
| 2 | Prototypen | Implementierung Prototyp Erfassung (UI) | sg / cm | 1 | 08.11.2010 | | v | BCS / Domain / Silverlight nicht funktionstüchtig bis zum Sprintende -> Verschiebung auf Sprint 3 |
| 2 | Prototypen | Paper Prototype Analyse (UI) | sg | 1 | 02.11.2010 | 02.11.2010 | | |
| 2 | Prototypen | Implementierung Prototyp Analyse (UI) | sg | 2 | 08.11.2010 | | v | BCS / Domain / Silverlight nicht funktionstüchtig bis zum Sprintende -> Verschiebung auf Sprint 3 |
| 2 | Implementation | Problem BCS/Silverlight ohne externe Listen | cm | 2 | 08.11.2010 | 05.11.2010 | a | geht nicht |
| 2 | Testing | Planung Unit Tests | sg | 1 | 08.11.2010 | | v | noch keine testbare Architektur -> Verschiebung auf Sprint3 |
| 2 | Organisation | Sitzungsprotokoll Sitzung KSSG | sg | 1 | 09.11.2010 | 09.11.2010 | | |
| 2 | Design | Logische Architektur Sprint 2 | cm /sg | 1 | 08.11.2010 | | v | |
| | | | | | | | | |
| 3 | Testing | Unit Tests | sg | 1 | 26.11.2010 | 26.11.2010 | | |
| 3 | Prototypen | Implementierung Prototyp Erfassung (UI) | sg | 1 | 26.11.2010 | 26.11.2010 | | Funktionsreduktion ! Nur CRUD funktionalität |
| 3 | Prototypen | Implementierung Prototyp Analyse (UI) | sg | 1 | 26.11.2010 | | a | zu aufwändig |
| 3 | Impementation | Datenhaltung + Domain Layer | cm | 1 | 26.11.2010 | 26.11.2010 | | |
| 3 | Dokumentation | How-To -> Anleitung für Erstellung von Erweiterungen | cm /sg | | 26.11.2010 | | a | nicht mehr nötig! Applikation soll extern entwickelt werden |
| 3 | Impementation | Technologistudium CAML | cm | 2 | 12.11.2010 | 12.11.2010 | | |
| 3 | Impementation | Performance Probleme CAML | cm | 2 | 12.11.2010 | 12.11.2010 | | |
| 3 | Impementation | Update Methoden BCS | cm | 1 | 12.11.2010 | 23.11.2010 | | |
| 3 | Dokumentation | Struktur SAD | sg | 1 | 10.11.2010 | 10.11.2010 | | |
| 3 | Dokumentation | Entwurf SAD | sg | 1 | 12.11.2010 | 12.11.2010 | | |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 3 | Logische Architektur | Allgemeine Beschreibung der Architektur | sg | 1 | | | | |
| 3 | Technologiestudium | Technologiestudium Silverlight / MVVM | sg | 2 | 01.12.2010 | 03.12.2010 | | |
| 3 | Logische Architektur | SAD - 1. Version für KSSG | sg / cm | 1 | 30.11.2010 | 30.11.2010 | | |
| 3 | Impementation | Refactoring Domain Connector | cm | 2 | 22.11.2010 | 26.11.2010 | | |
| 3 | Internes Design Datenhaltung | SAD Beschreibung Domain Connector | cm | 2 | 29.11.2010 | 30.11.2010 | | |
| 3 | Internes Design User Interface | SAD Beschreibung UI | sg | 1 | | | v | |
| 3 | Infrastruktur | Testserver reparieren | sg / cm | 1 | 29.11.2010 | | a | Deploymentprobleme, Packaging dafür repariert, aber leider zur Sitzung noch nicht einsatzbereit. |
| 3 | Organisation | Vorbereitung Sitzung KSSG | sg / cm | 1 | 01.12.2010 | 01.12.2010 | | |
| 3 | Organisation | Sitzungsprotokoll KSSG | cm | 1 | 03.12.2010 | 03.12.2010 | | |
| | | | | | | | | |
| 4 | Prototyp | Komplettierung Technologie Prototyp CRUD | cm /sg | 1 | 17.12.2010 | 17.12.2010 | | |
| 4 | Testing | Unit Tests | cm | 1 | 17.12.2010 | 20.12.2010 | | |
| 4 | Dokumentation | SAD Deployment | sg | | 17.12.2010 | 17.12.2010 | | |
| 4 | Dokumentation | SAD GUI | sg | | 17.12.2010 | 17.12.2010 | | |
| 4 | Dokumentation | SAD Glossar | cm | | 17.12.2010 | 17.12.2010 | | |
| 4 | Dokumentation | Logische Architektur: Beschreibung der Probleme | cm | | 17.12.2010 | 17.12.2010 | | |
| 4 | Dokumentation | Entwurf Poster | cm /sg | | 19.12.2010 | | v | |
| 4 | Dokumentation | Entwurf Erfahrungsbericht | cm /sg | | 19.12.2010 | | v | |
| 4 | Dokumentation | Entwurf Abstract | cm /sg | | 19.12.2010 | | v | |
| 4 | Dokumentation | Code Dokumentation | cm | | 19.12.2010 | | v | aus Zeitgründen verschoben |
| 4 | Technologiestudium | Sandcastle (Code Dokumentation) | cm | | 08.12.2010 | | a | weggelassen |
| 4 | Dokumentation | SAD Beschreibung Testing | sg | | 19.12.2010 | 19.12.2010 | | |
| 4 | Dokumentation | SAD Gliederung verbessern (Trennen, Technischer Bericht?) | sg | | 22.12.2010 | | v | aus Zeitgründen verschoben |
| 4 | Implementation | Domain Connector (Refactoring) | cm | | 19.12.2010 | 17.12.2010 | | |
| | | | | | | | | |
| 5 | Dokumentation | Erklärung Eigenständige Arbeit | cm/sg | | 23.12.2010 | 22.12.2010 | | |
| 5 | Dokumentation | Titelblatt | cm/sg | | 23.12.2010 | 22.12.2010 | | |
| 5 | Dokumentation | Abstract | cm/sg | | 23.12.2010 | 20.12.2010 | | |
| 5 | Dokumentation | Poster | sg | | 23.12.2010 | 20.12.2010 | | |

| | | | | | |
|---|---|---|---|---|---|
| 5 | Dokumentation | Technischer Bericht (SAD + Anforderungsanalyse zusammen) | cm/sg | 23.12.2010 | 23.12.2010 |
| 5 | Dokumentation | Projektmanagement Dokumentation (TODO Liste, Sitzungen) | sg | 23.12.2010 | 22.12.2010 |
| 5 | Organisation | Drucken | cm/sg | 23.12.2010 | 23.12.2010 |
| 5 | Organisation | CD Brennen | cm/sg | 23.12.2010 | 23.12.2010 |
| 5 | Dokumentation | Erfahrungsbericht | cm/sg | 23.12.2010 | 22.12.2010 |
| 5 | Organisation | Dokumentereview | cm/sg | 23.12.2010 | 22.12.2010 |
| 5 | Dokumentation | Technischer Bericht (Referenzen) | cm | 23.12.2010 | 22.12.2010 |

# 18. Sitzungsprotokolle

## 18.1. Kick Off Meeting, HSR & KSSG

Datum: 1.9.2010
Ort: Rapperswil
Anwesend: M. Schlegel, KSSG
C. Kahlert, KSSG
H. Huser, Projektbetreuer HSR
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden:**

***1. Vorstellung der Personen***
Vorstellung der Personen des KSSG und H. Huser.
Die Studenten hatten bereits ein Orientierungsmeeting mit den Verantwortlichen des KSSG und kannten sich bereits.

***2. Erwartung der HSR an die Arbeit***
Genaue Aufgabenstellung wir am Anfang der Arbeit von den Studenten erarbeitet. Ansonsten gelten die Rahmenbedingungen der normalen Semesterarbeit.
Es dürfen keine Spesen oder Honorare für die Arbeit ausbezahlt werden (Reglement HSR), dafür existieren auch keine Garantien von Seiten der HSR bezüglich Endstand der Arbeit.

***3. Erwartung des Kantonsspitals an die Arbeit***
Für das Kantonsspital St.Gallen ist diese Arbeit ein wichtiges Pilotprojekt für die geplante SURVEAST Lösung zur „Surveillance" von Patienten. Die Verantwortlichen heben hervor, dass sie bereit sind Zeit zu investieren und mit ihren medizinischen Fachkenntnissen uns so gut als möglich unterstützen möchten.

***4. Erwartungen der Studierenden an die Arbeit***
Die Hauptherausforderung liegt für uns in der realen Aufgabenstellung. Es soll wirklich ein Projekt entstehen dass nachher in seiner Funktion produktiv genutzt wird. Zusätzlich werden wir mit neuen und spannenden Technologien und anschliessend mit deren Integration in die Infrastruktur konfrontiert.

***5. Kurzdefinition der Arbeit***
Hauptdokument „Surveillance myeloablativer Therapien bei hämatoonkologischen Patienten – Konzept" des KSSG. Teilprojekt / Modul für Gesamtprojekt SURVEAST. Wichtig ist die Umsetzung auf SharePoint. SharePoint wurde als Schlüsseltechnologie am KSSG definiert und man will zukünftig vermehrt Applikationen darauf aufbauen.

***6. Besprechung: Provisorisches Konzeptdokument***
Genaue Besprechung und Erklärung durch KSSG an der nächsten Sitzung bei Start der Arbeit.

***7. Infrastruktur, Ressourcen und zuständige Kontaktpersonen***
Die Entwicklungsinfrastruktur wird innerhalb der HSR bereitgestellt.
Auf Seiten KSSG kommt eine weitere Kontaktperson dazu, Brian Douglas von der IT, verantwortlich für die SharePoint Infrastruktur.

***8. Festlegen erster Termine***
Besprechung Konzeptdokument und Anforderungen mit KSSG, 20.9. in St.Gallen.
Kickoff mit H.Huser: 22.9. an der HSR.

***9. Weiteres***
Keine weiteren Punkte.

## 18.2. 1. Meeting im Kantonsspital

Datum: 20.09.2010
Ort: Kantonsspital St. Gallen
Anwesend: M. Schlegel, KSSG
C. Kahlert, KSSG
B. Douglas, Informatiker KSSG
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. Vorstellung von Brian Douglas***
Brian Douglas ist am Kantonsspital St.Gallen unsere Ansprechperson für Fragen an die Infrastruktur. Enger Kontakt mit Brian Douglas ist während der ganzen Arbeit erwünscht. Sein Team wird die Software nach der Fertigstellung warten müssen.

***2. Besprechung der wesentlichen funktionalen Anforderungen***

**Patientengeschichte:**
Patientengeschichte muss (soweit erfasst) über die gesamte Timeline mit allen Ereignissen nachverfolgbar und auswertbar sein.
**Flexibilität der Strukturierung der Daten:**
Es werden hohe Anforderungen an die Flexibilität der Strukturierung der Daten verlangt.

***3. Technische Anforderungen***

Die Software soll in die bereits existierende Infrastruktur Sharepoint 2010 integriert werden. Die Infrastruktur darf durch die Software aber nicht merklich verlangsamt werden.
Silverlight ist im Kantonspital standardmässig auf allen Clients installiert und kann verwendet werden.

***4. Schnittstellen***

SAP: Am wichtigsten wird die Schnittstelle zu SAP sein. Die ist im Moment in erster Linie noch betriebspolitisch ein Problem. Brian Douglas wird sich darum bemühen, dass man die benötigten Daten aus SAP abrufen kann.
Onkologische Daten: Hier muss noch festgelegt werden welche Daten überhaupt verwendet werden dürfen und sollen. Darum wird sich Matthias Schlegel kümmern.

***5. Prioritäten für den ersten Meilenstein***

1. Aufsetzen der Infrastruktur und der Testumgebung
2. Konzept für
    a. flexible Strukturierung der Daten
    b. Übersicht über Krankengeschichte eines Patienten
    c. Workflow
    d. Ereignisse/Faktoren und deren Verknüpfungen
3. Intensive Auseinandersetzung mit Microsoft Sharepoint

***6. Festlegen der nächsten Sitzung im Kantonsspital***

Die nächste Sitzung im Kantonsspital wurde auf Mittwoch den 13.10. um 09:30 gelegt.

## 18.3. Zwischenstandssitzung, HSR

Datum: 22.09.2010
Ort: HSR, Rapperswil
Anwesend: H. Huser, Projektbetreuer HSR
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. Rückblick Sitzung KSSG, 20.9.***

Kurze Erwähnung der an der Sitzung besprochenen Punkte abgesehen der Anforderungen. Ansprechpartner Brian Douglas auf der Technologie-Seite, SharePoint Infrastruktur, etc.

***2. Besprechung der Anforderungen von Seiten KSSG***

Die aus der Sitzung gewonnen Erkenntnisse wurden besprochen. Wir stehen vor dem Hauptproblem, dass die gewünschte Flexibilität sich nicht mit den eigentlichen Zielen der Arbeit verträgt. Natürlich ist es möglich eine ganz generische Lösung zur Erfassung der Daten zu entwickeln, diese Daten sind aber eigentlich nach der Erfassung unnütz, da sie nicht sinnvoll ausgewertet werden können.

Es soll nun eine genaue Anforderungsanalyse folgen, in der man die wesentlichen Teile identifizieren muss, die für die späteren Auswertungen wichtig sind. Ohne diese Auswertungen macht auch die Erfassung keinen Sinn. Natürlich sind zusätzliche Informationen mit generischer Struktur denkbar, diese sind aber nur bedingt auswertbar.

Die Anforderungsanalyse wird in der nächsten Zeit zusammen mit den Ansprechpartnern im KSSG erarbeitet.

***3. Weiteres Vorgehen***

Mit dem Kantonsspital werden nun im 3 – 4 Wochenrythmus Sitzungen abgehalten. Dieses Zeitfenster beinhaltet ungefähr jeweils den nächsten Meilenstein. An diesen Sitzungen werden zusätzlich auch immer die nächsten Schritte besprochen.

Nächste Sitzung an der HSR am 29.9.2010, 14:00. Bis dann soll der Projektplan und eine erste Fassung des Anforderungsdokuments stehen.

## 18.4. Zwischenstandssitzung, HSR

Datum: 29.09.2010
Ort: HSR, Rapperswil
Anwesend: H. Huser, Projektbetreuer HSR
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. Grobentwurf Anforderungsanalyse***

Der erste Grobentwurf der Anforderungsanalyse wurde kurz durchgesehen und die Rückmeldungen von Christian Kahlert (mehr Konzentration auf Infektologie als auf Onkologie) besprochen.

***2. Logische Strukturierung der Domaindaten***

Die logische Strukturierung der Daten ist ein guter Anfang, muss aber in der nächsten Woche noch weiter ausgearbeitet werden. Einige Fragen müssen noch mit Matthias Schlegel und Christian Kahlert besprochen werden. Es wird eine Liste mit Fragen erstellt die wir alle auf einmal klären wollen. Konkret müssen vor allem die Beziehungen zwischen den Daten geklärt werden.

***3. Technische Integration***

Die technische Orientierungslosigkeit betreffend Sharepoint wurde erwähnt.
Herr Huser kümmert sich darum, dass innerhalb der nächsten Wochen mit einem Sharepoint Spezialisten des Instituts zusammensitzen können um Ratschläge zu holen. Bis dahin werden wir uns beim Technologiestudium nicht weiter mit Sharepoint sondern nur mit asp.net befassen.

***4. Weiteres Vorgehen***

Bis zur nächsten Sitzung wird die Domainanalyse (Strukturierung der Domaindaten) weiter konkretisiert und Fragen betreffend der Anforderungen und der Domain geklärt. Das Dokument Anforderungsanalyse wird so weit erweitert, dass Nach der Sitzung nur noch kleine Korrekturen nötig sein sollten.

Nächste Sitzung an der HSR am 11.10.2010, 13:00.

## 18.5. Zwischenstandssitzung, HSR

Datum: 12.10.2010
Ort: HSR, Rapperswil
Anwesend: H. Huser, Projektbetreuer HSR
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. Review Anforderungen***

Wir hatten in der Zwischenzeit wieder Kontakt mit den Verantwortlichen am KSSG und konnten nun eine Grundstruktur formulieren die aus unserer Sicht funktionieren sollte.
An der Sitzung morgen werden wir versuchen, dass so von den Verantwortlichen Absegnen zu lassen.

***2. Technologieprototyp***

Nach den Recherchen von letzter Woche tendieren wir in Richtung Silverlight / WCF. Um die Sharepoint Integration und die Security Anforderungen mit der KSSG-Informatik abzuklären, wäre aus unserer Sicht einer der ersten Schritte einen kleinen Technologieprototypen zu bauen und ihn dann sozusagen beim KSSG in die Vernehmlassung zu schicken. Dieser Prototyp sollte alle Layers und die Integration in SharePoint beinhalten.

***3. Abgrenzung erster Schritt Design / Realisierung***

Wir würden den Verantwortlichen gerne an der Sitzung einen Vorschlag für die nächsten Schritte, resp. eine Abgrenzung des ersten "Sprints" machen:

- Sprint 2:
  - Technologieprototyp
  - Design und Implementation Grundstruktur:
    Je ein Beispiel für einen Ereignistypen: 1 Diagnose, 1 Zustand, 1 Massnahme
  - Spezifikation weiterer Diagnosen, Zustände, Massnahmen

- Sprint 3: Weitere Zustände, Massnahmen, Diagnosen, Verknüpfungen
- Sprint 4: Auswertung und Import

## 18.6. 2. Meeting im Kantonsspital

Datum: 13.10.2010
Ort: Kantonsspital St. Gallen
Anwesend: M. Schlegel, KSSG
C. Kahlert, KSSG
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. Review und Überarbeitung Anforderungsspezifikation***

Lange Diskussionen über Terminologie und Verknüpfungen. Ereignistypen Handlung, Zustand und Diagnose werden gestrichen, da die Kategorisierung teilweise eher schwierig ausfallen würde. Dafür wurden folgende Begriffe eingeführt:

*Modifikator:* Ereignis das den Krankheitsverlauf im Bezug auf einen möglichen Endpunkt beeinflussen kann.

*Endpunkt* (vorher: Outcome): Ergebnis des Verlaufs und später Ausgangspunkt für Auswertungen. Beispiel: Wir möchten auswerten, welche Modifikatoren bei den Patienten zum Endpunkt „Bakteriämie“ geführt haben.

Genaueres in der neusten Version des Anforderungsdokuments.

***2. Nächste Schritte***

Unser Vorschlag:

1. Technologieprototyp
2. Design und Implementation Grundstruktur:
   Je ein Beispiel für einen Ereignistypen: 1 Diagnose, 1 Zustand, 1 Massnahme
3. Spezifikation weiterer Diagnosen, Zustände, Massnahmen

Termin nächste Sitzung wird per Mail abgesprochen.

***3. Grobplanung weitere Sprints***

Unser Vorschlag:

- Sprint 2:
  - Technologieprototyp
  - Design und Implementation Grundstruktur:
    Je ein Beispiel für einen Ereignistypen: 1 Diagnose, 1 Zustand, 1 Massnahme
  - Spezifikation weiterer Diagnosen, Zustände, Massnahmen
- Sprint 3: Weitere Modifikatoren, Endpunkte, Auswertungsmöglichkeiten, Verknüpfungen
- Sprint 4: Auswertungen und Import

## 18.7. 3. Meeting im Kantonsspital (Zwischenstand nach Woche 7)

Datum: 08.11.2010
Ort: Kantonsspital St. Gallen
Anwesend: M. Schlegel, KSSG
C. Kahlert, KSSG
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. Rückblick auf die letzten 4 Wochen***

- Input Informatikdienst bezgl. SharePoint
- „Klar geht das mit SharePoint, aber 14 Wochen sind viel zu kurz“
- Gegensätzliche Meinung von Jürg Jucker (INS): SharePoint eher ungeeignet für komplexes Datenmodell. Schon möglich, aber viel Aufwand für praktisch keinen Vorteil.
- Sitzung mit Ramon Brülisauer
- → klare Aussage ID: SharePoint! Reines Hosting wird nicht akzeptiert.

- Aufbau eines Prototypen mit SharePoint: Diverse Ansätze mit BCS ausprobiert, einige Ungeeignet, langsamer Fortschritt: dafür jetzt eine Architektur beisammen die funktionieren sollte!

***2. Nächste Schritte: Technologie***

*Variante A:*
Weiterführen der Arbeit mit SharePoint. Reduzierter Funktionsumfang dafür dokumentierte Architektur als Basis für die Weiterentwicklung: + keine Konflikte mit Informatikdienst, - Aufwand hoch, - Flexibilität eingeschränkt

*Variante B:*
Technologiewechsel z.B: SL RIA Services. Später Zeitpunkt um umzusteigen: Aussage von ID war sehr deutlich. Dafür mehr Funktionalität implementierbar in dieser Zeit. Geht einfacher als SharePoint.

Entscheidung fällt auf Variante A.

***3. Nächste Schritte: Anforderungen***

Anschauen UI Prototypen. Grundstruktur besprechen. Auswertungen besprechen.
Definition 2 Endpunkte und 2 Modifikatoren.

## 18.8. 4. Meeting im Kantonsspital (Zwischenstand nach Woche 11)

Datum: 1.12.2010
Ort: Kantonsspital St. Gallen
Anwesend: M. Schlegel, KSSG
C. Kahlert, KSSG
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. Rückblick auf die letzten 4 Wochen***

Es wurden die Probleme angesprochen, die durch die komplexe Architektur entstanden sind.
Ein Prototyp konnte leider nicht gezeigt werden, da es am Tag vor der Sitzung Probleme mit dem Server gab und dieser nicht einsatzbereit war.

***2. Resultat der Sitzung zum Technologieentscheid des KSSG zum breiten Einsatz von Sharepoint***

Der Informatikdienst des KSSG hält weiterhin am Technologieentscheid fest Sharepoint als einzige Plattform für zukünftige Softwareprojekte zu verwenden. Der Entscheid wird akzeptiert, stösst aber auf Unverständnis da auch bei Matthias Schlegel und Christian Kahlert einsehen, dass Sharepoint nicht die geeignetste Plattform für die Implementation von datenlastigen Anwendungen wie SURVEAST sein kann. Gerade am Kantonsspital wird es in Zukunft sicher noch mehr ähnliche Projekte geben.

***3. Nächste Schritte: Prototyp***

Der Prototyp soll sich nicht mehr an der Funktionalität aus der Anforderungsanalyse orientieren sondern nur noch ein Architekturprototyp werden, welcher die Probleme und Schwierigkeiten bei der Verwendung des BCS zusammen mit Silverlight aufzeigt.

***4. Nächste Schritte: Dokumentation***

Das geplante HOW-TO welches sich an Zivildienstleistende richten sollte, wird nicht erstellt, da die Ausmasse der Entwicklung erkannt wurden und dies nicht mit internen Zivildienstleistenden machbar ist.
M. Schlegel und C. Kahlert sind vor allem an der Dokumentation als Technologiestudie interessiert. Das Softwarearchitekturdokument soll in Zukunft verwendet werden, um den Aufwand einer solchen Applikation aufzuzeigen. Es soll in den nächsten Wochen noch erweitert und verbessert werden.

***5. Nächste Schritte: Surveast***

Surveast soll als Projekt nun offiziell ausgeschrieben werden. Allerdings muss weiter in Betracht gezogen werden, die Applikation auf einer anderen Plattform als Sharepoint zu entwickeln. Auch für weitere Entwicklungen von datenlastigen Applikationen muss unter Umständen eine andere Technologie in Betracht gezogen werden. Davon möchten die Auftraggeber am Kantonsspital vor allem ihre interne IT überzeugen und auch gegenüber der Spitalleitung aufzeigen können, dass eine solche Entwicklung mit SharePoint ein massiv grösseres Budget benötigen würde, als eine Entwicklung auf einer anderen Plattform.

Eine letzte Sitzung vor der Abgabe der Arbeit ist nicht nötig. Wir werden nach der Abgabe aber nochmals zusammensitzen. Wenn möglich sollte Herr Jucker dabei sein. Es soll vor allem um Einschätzungen für eine externe Vergabe des Projektes gehen.

## 18.9. Zwischenstandssitzung, HSR

Datum: 13.12.2010
Ort: HSR, Rapperswil
Anwesend: H. Huser, Projektbetreuer HSR
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. Dokumentation***

- Die Anforderungsanalyse wird als separates Dokument abgegeben.
- Das SAD wird umgetauft in Technischer Bericht und noch etwas umstrukturiert
- Die UI Prototypen werden in die bestehenden Dokumente integriert -> Anforderungs Analyse
- Projektplan, Taskplanung & Zeiterfassung werden so weitergeführt und abgegeben
- Ressourcen zum Technologiestudium werden auf der CD mitgeliefert

- Abgegeben wird:
  - Poster
  - Abstract
  - Titelblatt
  - 2CDs mit Code, Dokumente in Elektronischer Form (ganze Arbeit)
  - Anforderungsanalyse (1 Druck)
  - Technischer Bericht (1 Druck)
  - Sitzungsprotokolle
  - Projektplan & Taskplanung
  - Zeiterfassung

***2. Implementation***

- Es wird die CRUD Funktionalität für Patient fertiggestellt.
- Tests auf mehreren Layers
- Code wird aufgeräumt
- Code Dokumentation als /// direkt im Code
- Deployment (Lists, Scripts, …), wird beschreiben wo nicht automatisiert

***3. Präsentation***

- Gibt es nicht

## 18.10. Besprechung Technologie SharePoint & SAD, HSR

Datum: 13.12.2010
Ort: HSR, Rapperswil
Anwesend: J.Jucker, Mitarbeiter INS HSR
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. SharePoint Listen:***

- **Performance**
  Herr Jucker bestätigte unsere Performancebedenken. Unter anderem stösst man bei komplexeren Relationen schnell auf Probleme. SharePoint Listen sind als flache Datenstruktur konzipiert.

- **Usability**
  Bestätigung: Die Automatisch erzeugten ListViews und Forms sind nicht brauchbar sobald mehrere Relationen dargestellt werden müssen.

- **Rapid Prototyping**
  Input: Rapid Prototyping ist auch kein schlagendes Argument für SharePoint Listen. Expression Blend bietet SketchFlow, woraus direkt ein Silverlight UI generiert werden kann.

- **Struktur Content Database**
  Alle Informationen über Forms sind in der Content Database.
  Content Database ist nicht verschlüsselt -> Security

***2. Erfahrungen mit BCS:***

- **Credential Management**
  Herr Jucker bestätigt unsere Bedenken wegen Security Problemen.
  Ausserdem: Row Security ist mit SharePoint eigenen Mitteln gar nicht möglich.

***3. Weiteres:***

- **Architekturvarianten:**
  Die Möglichkeit der Verwendung eines WCF Services mit .NET 3.5 sollte noch erwähnt sein.

- **Reporting Services:**
  Die Reporting Services in Microsoft SharePoint bieten nur graphische Elemente zur Darstellung von Zahlen.
  Die Daten sollten vorher extern verarbeitet werden.

- **Alternativen:**
  Als ergänzende Alternative zu SharePoint könnte für das Kantonsspital Dynamics CRM eine Lösung sein.

## 18.11. Zwischenstandssitzung, HSR

Datum: 22.12.2010
Ort: HSR, Rapperswil
Anwesend: H. Huser, Projektbetreuer HSR
C. Meier, Student HSR
S. Gacond, Student HSR

**Traktanden**

***1. Zusammenfassung Sitzung J. Jucker***

- Auch Herr Jucker ist der Meinung, dass SharePoint für unsere Anwendung nicht geeignet ist.
- Zusätzliche Inputs warum SharePoint für unsere Anwendung nicht geeignet ist.
- Alternativen (Microsoft Dynamics)
- Interesse an Ausschreibung

***2. Titel der Arbeit***

- Vorschläge:
  SURVEAST – Technologiestudie zu Pilotmodul
  SURVEAST – Technologiestudie für ein Pilotmodul mit Microsoft SharePoint 2010

- Entscheidung:
  SURVEAST – Technologiestudie mit SharePoint

***3. Technischer Bericht***

- Neue Struktur fast fertig
- Neu: Anforderungsanalyse im Technischen Bericht
- Neu: Alternative - Microsoft Dynamics am Schluss

TODO:
- Referenzverzeichnis
- Einzelne Stellen noch zu überarbeiten
- Sprachliches

***4. Weiteres***

- Das Poster sollte weniger Text enthalten. -> wird überarbeitet
- Abstract weniger negativ
- Die Arbeit wird am Donnerstag einem Mitarbeiter im Büro des INS abgegeben.

# 19. Zeiterfassung

## 19.1. Zeiterfassung Clemens Meier

| Woche | Tag | Datum | Tätigkeit | Zeit | Zeit/Wo | Total | Soll |
|---|---|---|---|---|---|---|---|
| Woche 1 | Mo | 20.09.2010 | Sitzung mit KSSG inkl. Reise | 4.0 | **23.5** | **23.5** | 17.2 |
| | | | Installation Entwicklungsrechner | 3.0 | | | |
| | Di | 21.09.2010 | Technologiestudium | 3.0 | | | |
| | | | Aufbau Infrastruktur / Installation Server | 4.0 | | | |
| | | | Sitzungsprotokoll | 1.0 | | | |
| | Mi | 22.09.2010 | Sitzung mit H.Huser | 1.0 | | | |
| | | | Vorbereiten Sitzung, Besprechen | 1.0 | | | |
| | | | Sitzungprotokoll | 0.5 | | | |
| | | | Projektplan | 2.0 | | | |
| | | | Installation Server | 4.0 | | | |
| Woche 2 | Mo | 27.09.2010 | Anforderungsanalyse | 6.0 | **19.0** | **42.5** | 34.4 |
| | Di | 28.09.2010 | Anforderungsanalyse | 2.0 | | | |
| | | | Technologiestudium | 4.0 | | | |
| | Mi | 29.09.2010 | Anforderungsanalyse | 3.0 | | | |
| | | | Technologiestudium | 4.0 | | | |
| Woche 3 | Mo | 04.10.2010 | Technologiestudium asp.net | 7.0 | **18.0** | **60.5** | 51.6 |
| | Di | 05.10.2010 | Technologiestudium asp.net | 5.0 | | | |
| | Mi | 06.10.2010 | Technologiestudium | 2.0 | | | |
| | | | Anforderungsanalyse | 4.0 | | | |
| Woche 4 | Mo | 11.10.2010 | Anforderungsanalyse | 3.0 | **26.0** | **86.5** | 68.8 |
| | | | Technologie Silverlight RIA | 4.0 | | | |
| | Di | 12.10.2010 | Technologie Silverlight RIA | 7 | | | |
| | Mi | 13.10.2010 | Sitzung KSSG (inkl Reise) | 3.0 | | | |
| | | | Anforderungen | 2.0 | | | |
| | Fr | 15.10.2010 | Technologie Sharepoint Listen | 4 | | | |
| | Sa | 16.10.2010 | Technologie Sharepoint Listen | 3.0 | | | |
| Woche 5 | Mo | 18.10.2010 | Siztung Ramon Brülisauer KSSG | 4 | **25.0** | **111.5** | 86.0 |
| | | | BCS Technologiestudium | 2.0 | | | |
| | | | BCS Ausprobieren | 2.0 | | | |
| | Di | 19.10.2010 | Implementation BCS + xml Definition | 8.0 | | | |
| | Mi | 20.10.2010 | Implementation BCS | 9.0 | | | |
| Woche 6 | Mo | 26.10.2010 | BCS über .net Assembly (Technologiestudium) | 7.0 | **27.0** | **138.5** | 103.2 |
| | Di | 27.10.2010 | Implementation BCS + .net Assembly | 8.0 | | | |
| | Mi | 28.10.2010 | Implementation BCS + .net Assembly | 8.0 | | | |
| | Sa | 30.10.2010 | BCS | 4.0 | | | |
| Woche 7 | Mo | 01.11.2010 | BCS direkt auf SQL | 8.0 | **25.0** | **163.5** | 120.4 |
| | Di | 02.11.2010 | BCS direkt auf SQL + Domain Layer | 8.0 | | | |
| | Mi | 03.11.2010 | Erfassungs GUI (Silverlight) | 7.0 | | | |
| | Fr | 05.11.2010 | GUI Prototy Erfassung | 2.0 | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| **Woche 8** | Mo | 08.11.2010 | Paper Prototypen für Sitzung | 2.0 | **28.0** | **191.5** | 137.6 |
| | | | Sitzung KSSG (inkl Reise) | 5.0 | | | |
| | Di | 09.11.2010 | CamlQueries auf externe Listen | 7.0 | | | |
| | Mi | 10.11.2010 | CamlQueries auf externe Listen | 4.0 | | | |
| | | | Performance (BCS-Methoden) | 3.0 | | | |
| | Fr | 12.11.2010 | BCS (Update Methoden) | 7.0 | | | |
| **W9** | | | | | **0.0** | **191.5** | 154.8 |
| **Woche 10** | Mo | 22.11.2010 | GUI Prototy Erfassung | 3.0 | **27.0** | **218.5** | 172.0 |
| | | | Domain.Connector (Refactoring) | 3.0 | | | |
| | Di | 23.11.2010 | Domain.Connector BCS | 7.0 | | | |
| | Mi | 24.11.2010 | Software Architektur Dokument | 4.0 | | | |
| | | | BCS / Externe Listen / Domain | 3.0 | | | |
| | Fr | 26.11.2010 | Software Architektur Dokument | 2.0 | | | |
| | | | Vorbereiten Sitzung KSSG | 1.0 | | | |
| | | | Domain.Connector BCS | 4.0 | | | |
| **Woche 11** | Mo | 29.11.2010 | Architektur | 2.0 | **27.5** | **246.0** | 189.2 |
| | | | Silverlight MVVM | 2.0 | | | |
| | | | SAD Beschreibung Domain.Connector | 2.0 | | | |
| | Di | 30.11.2010 | SAD | 2.0 | | | |
| | | | Testserver reparieren | 9.0 | | | |
| | | | Vorbereiten Sitzung KSSG | 2.0 | | | |
| | | | GUI Prototyp | 1.0 | | | |
| | Mi | 01.12.2010 | Sitzung KSSG (inkl Reise) | 4.0 | | | |
| | Fr. | 03.12.2010 | Sitzungsprotokoll | 0.5 | | | |
| | | | Domumentation | 3.0 | | | |
| **Woche 12** | Mo | 06.12.2010 | Dokumentation SAD | 7.0 | **24.0** | **270.0** | 206.4 |
| | Di | 07.12.2010 | SAD | 5.0 | | | |
| | | | Code Dokumentation | 1.0 | | | |
| | Fr | 10.12.2010 | Nachführung Dokumente | 1.0 | | | |
| | | | SAD | 6.0 | | | |
| | Sa | 11.12.2010 | Ergänzungen SAD | 4.0 | | | |
| **Woche 13** | Mo | 13.12.2010 | Domain Connector | 4 | **28.0** | **298.0** | 223.6 |
| | | | Code Dokumentation | 1 | | | |
| | | | SAD, Belege und Referenzen | 2 | | | |
| | Di | 14.12.2010 | SAD, Review | 1 | | | |
| | | | SAD Weiterführung | 2 | | | |
| | | | Unit-Tests | 2 | | | |
| | | | Sitzung J.Jucker | 1.0 | | | |
| | | | Research Microsoft Dynamics | 2.0 | | | |
| | Mi | 15.12.2010 | Unit-Tests | 2.0 | | | |
| | | | Dokumentation Microsoft Dynamics | 3.0 | | | |
| | Fr | 17.12.2010 | Technischer Bericht | 5.0 | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Woche 14 | Mo | 20.12.2010 | Dokumentation | 4.0 | **23.0** | **321.0** | 240.8 |
| | | | Code Dokumentation | 1.0 | | | |
| | Di | 21.12.2010 | Korrekturen Dokumentation | 6.0 | | | |
| | Mi | 22.12.2010 | Sitzung mit H.Huser | 1.0 | | | |
| | | | Fertigstellen Dokumente | 7.0 | | | |
| | Do | 23.12.2010 | Drucken, Vorbereitung für Abgabe | 4.0 | | | |

**Total Ist - Arbeitszeit:** **321.0**
**Total Soll - Arbeitszeit:** **240.8**
**Differenz:** **80.2**

## 19.2. Zeiterfassung Silvan Gacond

| Woche | Tag | Datum | Tätigkeit | Zeit | Zeit/Wo | Total | Soll |
|---|---|---|---|---|---|---|---|
| Woche 1 | Mo | 20-09-2010 | Sitzung mit KSSG inkl. Reise | 4.0 | **23.5** | **23.5** | 17.2 |
| | | | Installation Entwicklungsrechner | 3.0 | | | |
| | Di | 21-09-2010 | Technologiestudium | 4.0 | | | |
| | | | Aufbau Infrastruktur / Installation Server | 4.0 | | | |
| | | | Sitzungsprotokoll | 1.0 | | | |
| | Mi | 22-09-2010 | Sitzung mit H.Huser | 1.0 | | | |
| | | | Vorbereiten Sitzung, Besprechen | 1.0 | | | |
| | | | Sitzungprotokoll | 0.5 | | | |
| | | | Projektplan | 2.0 | | | |
| | | | Installation Server | 3.0 | | | |
| Woche 2 | Mo | 27-09-2010 | Anforderungsanalyse | 6.0 | **21.5** | **45.0** | 34.4 |
| | Di | 28-09-2010 | Anforderungsanalyse | 3.0 | | | |
| | | | Technologie / WebParts | 4.5 | | | |
| | Mi | 29-09-2010 | Anforderungsanalyse | 4.0 | | | |
| | | | Technologie | 4.0 | | | |
| W3 | Sa | 09-10-2010 | Review Anf.Analyse vor Sitzung | 3.0 | **3.0** | **48.0** | 51.6 |
| Woche 4 | Mo | 11-10-2010 | Anfoderungen, Silverlight RIA | 8.0 | **33.0** | **81.0** | 68.8 |
| | Di | 12-10-2010 | Anfoderungen, Silverlight RIA | 8.0 | | | |
| | Mi | 13.10.2010 | Sitzung KSSG (inkl Reise) | 4.0 | | | |
| | | | Anforderungen umstrukturieren gem Sitzung, Technologie (SharePoint Listen) | 2.0 | | | |
| | Fr | 15-10-2010 | Technologie, Umbau auf SharePoint Listen | 6.0 | | | |
| | Sa | 16-10-2010 | Technologie, Umbau auf SharePoint Listen | 5.0 | | | |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Woche 5 | Mo | 18.10.2010 | Sitzung Ramon Brülisauer (KSSG Informatik) bzgl Technologie (inkl Reise) | 4.0 | **24.5** | **105.5** | 86.0 |
| | | | BCS Einlesen, Ausprobieren | 4.0 | | | |
| | Di | 19.10.2010 | Umbau auf BCS ContentTypes | 7.0 | | | |
| | Mi | 20-10-2010 | Umbau auf BCS ContentTypes | 9.5 | | | |
| Woche 6 | Di | 26-10-2010 | Umbau auf BCS ContentTypes mit Umweg über .net Assembly & UI Prototype (Analyse, Silverlight) | 8.0 | **21.0** | **126.5** | 103.2 |
| | Mi | 27-10-2010 | | 9.0 | | | |
| | Sa | 30-10-2010 | | 4.0 | | | |
| Woche 7 | Mo | 01-11-2010 | Zurückbauen auf direkte BCS -> SQL Verbindung, Aufbau Domain Layer und Domain <-> BCS Connectivity. | 9.0 | **25.0** | **151.5** | 120.4 |
| | Di | 02-11-2010 | | 9.0 | | | |
| | Mi | 03-11-2010 | | 7.0 | | | |
| Woche 8 | Mo | 08.11.2010 | Vorbereitung Sitzung | 2.0 | **31.0** | **182.5** | 137.6 |
| | | | Sitzung KSSG (inkl Reise) | 5.0 | | | |
| | Di | 09.11.2010 | Architektur Datalink Layer / BCS | 8.0 | | | |
| | Mi | 10.11.2010 | SAD, Entwurf | 8.0 | | | |
| | Fr | 12.11.2010 | Silverlight / UI / MVVM | 8.0 | | | |
| W 9 | | | | | **0.0** | **182.5** | 154.8 |
| Woche 10 | Mo | 22.11.2010 | Silverlight / UI / MVVM Umbau | 5.0 | **29.0** | **211.5** | 172.0 |
| | | | UI's einbauen in Connector Arch. | 2.0 | | | |
| | Di | 23.11.2010 | SAD, UI Layer, etc. | 7.0 | | | |
| | Mi | 24.11.2010 | SAD, Vererbungskonzept, etc. | 8.0 | | | |
| | Fr | 26.11.2010 | Umbau MVVM -> Command | 7.0 | | | |
| Woche 11 | Mo | 29.11.2010 | Versuch Umbau AnalysisUI | 3.0 | **28.0** | **239.5** | 189.2 |
| | | | MVVM & Drag'n'Drop & Interactivity | 4.0 | | | |
| | Di | 30.11.2010 | Vorbereiten Sitzung | 1.0 | | | |
| | | | Implementation "write" auf UI | 4.0 | | | |
| | | | Bereitmachen für Preview, Serverprobleme und Deploymentversuche bis spät… | 6.0 | | | |
| | Mi | 01.12.2010 | Sitzung KSSG (inkl Reise) | 4.0 | | | |
| | Fr. | 03.12.2010 | Überarbeiten Doku & Aufräumen .sln | 6.0 | | | |
| Woche 12 | Mo | | Prototyp kompletieren, SAD | 7.0 | **23.0** | **262.5** | 206.4 |
| | Di | | einige Tests / Research für SAD | 7.0 | | | |
| | Mi | | SAD, Research Content DB, etc. | 4.0 | | | |
| | Do | | SAD, Research Content DB, etc. | 5.0 | | | |
| Woche 13 | Mo | 13-12-2010 | Erweiterungen SAD, Review einflechten, sammeln von Belegen für gemachte Erfahrungen | 7.0 | **23.0** | **285.5** | 223.6 |
| | Di | 14-12-2020 | Weiterführen Dokumentation, SAD | 5 | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| | | | Sitzung J.Jucker | 1.0 | | |
| | Mi | 15-12-2010 | Unit Testing, Zusätzliche Punkte SAD | 6.0 | | |
| | Fr | 17-12-2010 | Umbau SAD - Techn. Bericht | 2.0 | | |
| **Woche 14** | Mo | 20-12-2010 | Umbau SAD - Techn. Bericht, Erweiterungen, Korrekturen | 6.0 | **25.0** | **310.5** | 240.8 |
| | Di | 21-12-2010 | Poster, Abstract, Erweiterung Techn. Bericht | 7.0 | | |
| | Mi | 22-12-2010 | Sitzung mit H.Huser | 1.0 | | |
| | | | Fertigstellung Dokumente, Korrekturen | 7.0 | | |
| | Do | 23-12-2010 | Drucken, Vorbereiten f. Abgabe | 4.0 | | |

| | | | |
|---|---|---|---|
| **Total Ist - Arbeitszeit:** | | **310.5** | |
| **Total Soll - Arbeitszeit:** | | | **240.8** |
| **Differenz:** | **69.7** | | |