

Prüfungskandidatin: Nicole Stadelmann

Fachvorgesetzter: Benjamin Stettler

Berufsbildner: Sandra Walzer

Erstexperte: Andreas Auer

Zweitexperte: Kurt Flury

Prototyp: Webanwendung für GP Stammdaten

Individuelle Praktische Arbeit



Startdatum: 03.04.2018

Enddatum: 18.04.2018

Inhaltsverzeichnis

[Teil 1 4](file:///C:\Users\stadni\Desktop\IPA\IPA%20Bericht.docx#_Toc511709026)

[1 Aufgabestellung 5](#_Toc511709027)

[1.1 Ausgangslage 5](#_Toc511709028)

[1.2 Detaillierte Aufgabenstellung 5](#_Toc511709029)

[1.2.1 Ursprung des Prototyps 5](#_Toc511709030)

[1.2.2 Systemarchitektur 5](#_Toc511709031)

[1.2.3 Musskriterien 6](#_Toc511709032)

[1.2.4 Technische Vorgaben 6](#_Toc511709033)

[1.2.5 Abgrenzungskriterien 7](#_Toc511709034)

[1.2.6 Zielgruppen 7](#_Toc511709035)

[1.2.7 Betriebsbedingungen 7](#_Toc511709036)

[1.2.8 Testumfang 10](#_Toc511709037)

[1.2.9 Dokumentation 10](#_Toc511709038)

[1.3 Mittel und Methoden 11](#_Toc511709039)

[1.4 Vorkenntnisse 11](#_Toc511709040)

[1.5 Vorarbeiten 11](#_Toc511709041)

[1.6 Neue Lerninhalte 11](#_Toc511709042)

[1.7 Arbeiten in den letzten 6 Monaten 12](#_Toc511709043)

[2 Projektorganisation 13](#_Toc511709044)

[2.1 Projektrollen 13](#_Toc511709045)

[2.2 Projektmethode 14](#_Toc511709046)

[2.2.1 Umsetzung der Projektmethode 15](#_Toc511709047)

[3 Zeitplan 16](#_Toc511709048)

[4 Arbeitsjournal 17](#_Toc511709049)

[Teil 2 18](file:///C:\Users\stadni\Desktop\IPA\IPA%20Bericht.docx#_Toc511709050)

[5 Kurzfassung des IPA Berichts (Management Summary) 19](#_Toc511709051)

[6 Vorbereitung 20](#_Toc511709052)

[6.1 Infrastruktur und Organisatorisches 20](#_Toc511709053)

[6.2 Risikomanagement 20](#_Toc511709054)

[6.3 Analyse des Projektauftrags 20](#_Toc511709055)

[7 Entwurf Webservice 21](#_Toc511709056)

[7.1 Überlegungen 21](#_Toc511709057)

[7.2 Benötigte Funktionsbausteine und Tabellen 21](#_Toc511709058)

[7.2.1 Funktionsbauseine 21](#_Toc511709059)

[7.2.2 SAP Tabellen 22](#_Toc511709060)

[7.3 Entwurf Webservice 1: Kunden suche 23](#_Toc511709061)

[7.4 Entwurf Webservice 2: Kundendaten auslesen 24](#_Toc511709062)

[7.5 Entwurf Webservice 3: QMC Produkte auslesen 25](#_Toc511709063)

[7.6 Entwurf Webservice 4: Kundendaten ändern 26](#_Toc511709064)

[7.7 Entscheidungen 26](#_Toc511709065)

[8 Realisierung Webservice 27](#_Toc511709066)

[8.1 Enterprise Services Builder 27](#_Toc511709067)

[8.1.1 Namespace erstellen 27](#_Toc511709068)

[8.1.2 Data Type erstellen 27](#_Toc511709069)

[8.1.3 Messagetype erstellen 28](#_Toc511709070)

[8.1.4 Service Interface erstellen 29](#_Toc511709071)

[8.1.5 Message Mapping 30](#_Toc511709072)

[8.1.6 Operation Mapping 31](#_Toc511709073)

[8.2 Integration Builder 31](#_Toc511709074)

[8.2.1 Communication Channel erstellen 31](#_Toc511709075)

[8.2.2 Integrated Configuration erstellen 31](#_Toc511709076)

[8.3 SProxy 31](#_Toc511709077)

[8.3.1 BP\_Search 32](#_Toc511709078)

[8.3.2 BP\_GetDetails 33](#_Toc511709079)

[8.3.3 BP\_QMC\_Products 34](#_Toc511709080)

[8.3.4 BP\_Update 35](#_Toc511709081)

[9 Planung Webanwendung 36](#_Toc511709082)

[9.1 Überlegungen 36](#_Toc511709083)

[9.2 Bibliotheken 36](#_Toc511709084)

[9.3 GUI Entwurf 37](#_Toc511709085)

[9.3.1 GUI Hauptseite 37](#_Toc511709086)

[9.3.2 GUI für die GP Auswahl 38](#_Toc511709087)

[9.3.3 d GUI für den Änderungsmodus 38](#_Toc511709088)

[9.4 Workflows 39](#_Toc511709089)

[9.4.1 GP suche und Informationen ausgeben 39](#_Toc511709090)

[9.4.2 GP ändern 40](#_Toc511709091)

[10 Entwicklung Webanwendung 41](#_Toc511709092)

[11 Testen 41](#_Toc511709093)

[12 Endergebnis 41](#_Toc511709094)

[13 Schlussbericht 41](#_Toc511709095)

[13.1 Ist Soll 41](#_Toc511709096)

[13.2 Persönliche Reflexion 41](#_Toc511709097)

[14 Glossar 41](#_Toc511709098)

[15 Quellenverzeichnis 41](#_Toc511709099)

[16 Abbildungsverzeichnis 41](#_Toc511709100)

[17 Tabellenverzeichnis 41](#_Toc511709101)

[Anhang 43](file:///C:\Users\stadni\Desktop\IPA\IPA%20Bericht.docx#_Toc511709102)

[Anhang 43](file:///C:\Users\stadni\Desktop\IPA\IPA%20Bericht.docx#_Toc511709103)



Teil 1

Obligatorische Kapitel

# Aufgabestellung

Nachfolgend wird die Aufgabenstellung im Originaltext aus PkOrg aufgeführt.

## Ausgangslage

SAP ist in unserem Unternehmen das führende Stammdatensystem. Wir benötigen für die Zukunft eine Möglichkeit auf die Daten ohne "SAP GUI" oder auch mit Mobilen Geräten zuzugreifen. Aus diesem Grund soll ein Prototyp erstellt werden, welcher eine Webseite als UI hat und via Webservice aus dem SAP die Daten darstellen und mutieren kann. Es gibt in SAP die Möglichkeiten Daten auf Web Technologie auszugeben, aber wie immer sind die Technische Voraussetzungen und der Implementierungsaufwand im SAP Umfeld recht hoch. Es wäre unteranderem manchmal auch sehr hilfreich, Daten möglichst schnell und ohne grossen Aufwand aus dem SAP ziehen zu können und diese an den Benutzern zu Verfügungen stellen zu können. Denkbar wäre diese Lösung auch als Entwicklungswerkzeug in einem POC (Proof of Concept) zur Visualisierung späterer Spezifikationen in einem Projekt.

## Detaillierte Aufgabenstellung

### Ursprung des Prototyps

Die WWZ arbeitet mit dem Partnerunternehmen Quickline zusammen. Über Quickline werden Telekomprodukte verkauft. Es gibt eine Schnittstelle zu Quickline, welche die Workflows der Bestellungen, Umzüge und Stammdatenmutationen des Kunden zur WWZ übermitteln. Der Fokus des Prototyps ist der Prozess der Stammdatenmutationen. Für den Fachbereich gibt es im SAP ein einfaches Cockpit. In diesem Cockpit kann der Workflow für einen Kunden ausgewählt werden. Durch diese Selektion werden zwei Aktionen ausgelöst.

1. In einem GUI Container im SAP GUI wird eine Quickline Webseite aufgerufen, welche die Stammdaten des Kunden anzeigt.

2. In einem anderen GUI Container im SAP GUI, wird der SAP WebGUI gestartet und die Transaktion BP (ändern des Geschäftspartners) aufgerufen.

Der Facharbeiter vergleicht die Stammdaten der beiden Systeme. Nach einer Validierung der Daten passt er im SAP WebGUI die Daten des Kunden an. Danach bestätigt der Facharbeiter den Workflow und der Workflow wird bei Quickline abgeschlossen.

Da der SAP WebGUI von SAP veraltet ist und letztens ein Problem mit dem Memory auf unserem SAP System ausgelöst hat, kam die initiale Idee zustande den SAP WebGUI durch eine normale Webseite abzulösen.

Was ist der SAP WebGUI: https://blogs.sap.com/2015/07/04/what-where-when-why-webgui/

Was ist die Transaktion BP: In der Transaktion BP werden die Stammdaten zu einem Geschäftspartner (Kunde) gepflegt.

Zu den Stammdaten gehören die Üblichen Daten wie die Kontaktdaten der Person und deren Anschrift. Die Transaktion ist das Gengenstück zu XD01 bei dem Modul SD für den Kunden, aber speziell für die Versorgungsindustrie.

### Systemarchitektur

Die WWZ hat grundsätzlich im SAP Umfeld eine 3tier Architektur im Einsatz.

Es gibt 3 SAP ERP Systeme:

T01 - Customizing/Entwicklung

Q01 - Test/Stagesystem

P01 - Produktivsystem

Bei den SAP PI Systemen, haben wir nur eine 2tier Architektur:

SAP ERP Q01 -> SAP PI PT1

SAP ERP P01 -> SAP PI PP1

Aus diesem Grund müssen die Entwicklungen die im T01 gemacht werden und eine PI Schnittstelle benötigen, zwingend in das SAP Q01 mit dem SAP SolMan transportiert werden. Der SAP SolMan verwaltet die Änderungen und ist an das Transportmanagement angehängt ist.

Als Webserver für den Prototyp ist ein XAMPP lokal auf dem entwicklungs PC mit Windows 7 installiert. Bei einer eventueller Produktivsetzung nach der IPA wird die Webanwendung auf einem Server innerhalb der DMZ in der WWZ kopiert.

Die komplette Systemarchitektur welche an der IPA benötigt wird, ist bereits vollständig aufgebaut. An den Systemen ist keine Konfiguration mehr notwendig. Ein User für den Zugriff der Webanwendung auf das PI ist angelegt und geprüft.

### Musskriterien

Es ist eine Webanwendung zu realisieren welche Stammdaten vom Geschäftspartner, die im SAP ERP abgelegt sind, ausgibt. Neben der Anzeige sollen auch einige Daten geändert werden können, welche dann im SAP wieder verbucht werden. Die Schnittstelle zum Auslesen der Daten muss über das SAP PI erfolgen. Primäres Ziel dieses Prototypens ist es, die Machbarkeit und Schwierigkeiten einer Webanbindung an das SAP aufzuzeigen. Die Anwendung soll ohne spezielle Konfiguration auf dem Webserver und ohne zusätzlichem Modulen von XAMPP Webserver laufen.

Es soll eine Grundlage/Vorlage geschaffen werden, welche für schnelle Implementation z.b an einem POC hilft für einen Webservice ein GUI zu erstellen. Über dieses GUI kann der Webservice getestet werden und z.b dem Fachbereich oder externen Partner die Funktionen visualisiert werden. Denkbar sind auch kleinere/einfachere Anwendungen wie bei diesem Prototyp z.b Mutationen von Stammdaten.

### Technische Vorgaben

Webanwendung:

- Die Webanwendung soll auf einem Webserver: XAMPP auf WIN7 64Bit laufen.

- Die Daten sollen zwischen der Webanwendung und dem SAP PI über eine SOAP Schnittstelle kommunizieren. Innerhalb des SOAP Request werden die Daten im XML Format abgebildet (SAP PI Standard bei einem SOAP Adapter).

- Als Programmiersprache für die Webanwendung soll JavaScript und HTML verwendet werden.

- Zusätzliche benötigte Bibliotheken für die Webanwendung können frei gewählt werden.

SAP PI:

- Das Design der Schnittstellen (Webservice) muss im "Enterprice Service Builder" erstellt werden

- Die Konfiguration der Schnittstellen muss im "Integration Builder" gemacht werden

- Communications Channel soll vom Typ SOAP sein

- Die Kommunikation soll über HTTP erfolgen (HTTPS ist "out of scope")

SAP ERP:

- Das Coding für die Datenbeschaffung soll im generierten Proxy des Webservice erstellt werden

- Für die Datenbeschaffung sind SAP Funktionsbausteine oder Klassen zu verwenden, ausser es sind keine vorhanden oder deren Funktionalität recht nicht aus.

### Abgrenzungskriterien

Auf die Implementation der Sicherheit der Applikation wird bei dieser Umsetzung bewusst verzichtet, da dies der Rahmen der Umsetzungszeit sprengen würde. Die Angaben der Authentifizierung welche über Basic Authentication gemacht werden soll, darf fix im Code hinterlegt werden.

### Zielgruppen

Der Prototyp dieser Anwendung ist primär für Entwickler und IT Fachleute gedacht.

### Betriebsbedingungen

Die Web Anwendung soll im Internet Explorer Browser 11 auf einem Desktop oder Notebook laufen. Es sollen keine speziellen Voraussetzungen oder Installationen auf einem Web Server nötig sein.

Spezifikation der Webanwendung

Das GUI der Webanwendung soll die Zugehörigkeit zur WWZ Gruppe wiederspiegeln. Ein WWZ Logo und das WWZ Blau sollen Bestandteil des Designs sein. Als Vorlage dient die Webseite der WWZ. Ein spezifischer Style Guide wird nicht vorgegeben und auch nicht erwartet. Der Fokus des Prototyp liegt auf den Funktionen und dem Backendsystem. Die Bedienelemente und der Look and Feel sind der heutigen Zeit angemessen auszulegen. Bootstrap kann verwendet werden, ist aber kein Musskriterium. Rückmeldungen und Fehlerbehandlung nach dem Aufruf eines Service soll dem Benutzer als Feedback angezeigt werden.

Folgende Funktionen sollen in der Anwendung implementiert werden:

Suche von Geschäftspartner nach:

- GP Nummer

- Name

- Vorname

- Firmen-/Gruppenname

- Strasse

-Hausnummer

- Ort

- PLZ

**Funktion:**

Die Werte für die Suche sollen in einzelnen Inputfeldern eingegeben werden können.

Wenn die Suche ausgeführt wird, sollen maximal die ersten 50 Einträge zurückgegeben werden. Zusätzlich soll dem Benutzer eine Meldung angezeigt werden, dass es mehr Suchtreffer gibt als angezeigt werden. Die Suchergebnisse sollen in einer Liste angezeigt werden. Ein Treffer auf der Suchliste soll ausgewählt werden können, damit die Stammdaten des Kunden geladen und angezeigt werden.

**Validierung:**

Mindesten eines dieser Felder muss einen Wert enthalten bevor die Suche ausgeführt wird.

Wenn die GP Nummer ausgewählt wird, darf diese nur Numerische werte enthalten und 7 Zeichen länge haben.

Anzeigen von Geschäftspartner:

* Kundedaten anzeigen (Standartadresse)

- Anrede

- Name 1

- Name 2

- Vorname

-Nachname

- Ort

- Strasse

- PLZ

- Land

- Region

* Kommunikation

- Telefon

- Mobiletelefon

- E-Mail

• Quickline Produkte anzeigen

○ QMC-ID

- Internet Dienstleistungsprodukt

- Fix-Telefon

- DTV

- Verte!

- QLTV

- Mobile Dienstleistungsprodukt

- Kombi Dienstleistungsprodukt

**Funktion:**

Nachdem ein Treffer in der Suche angewählt wurde, werden die Stammdaten und die Produkte von Quickline des Kunden angezeigt. Die Daten sollen auf eine Übersichtliche weise angezeigt werden. Die Stammdaten und Produktdaten sollen Gruppiert sein.

Geschäftspartner ändern:

• Kundedaten ändern

○ Standartadresse

- Anrede

- Name 1

- Name 2

- Vorname

- Nachname

○ Kommunikation

- Telefon

- Mobiletelefon

- E-Mail

Bevor die Kundendaten geändert werden können, muss der Benutzer ein Änderungsmodus aktivieren. Der Modus soll Fehlmanipulationen an den Stammdaten verhindern. Vor dem Speichern soll eine einfache Validierung durchgeführt werden. Nach dem Speichern soll dem Benutzer eine Statusmeldung angezeigt werden. Wenn das Speichern erfolgreich war, soll der Änderungsmodus verlassen werden.

**Validierung Standartadresse bei:**

- Person: Anrede / Nachname / Vorname müssen gefüllt sein.

- Gruppe: Name 1 / Name 2 müssen gefüllt sein.

- Organisation: Name muss gefüllt sein.

**Validierung Kommunikation bei:**

- Telefon darf nur Nummern enthalten.

- Mobiletelefon darf nur Nummern enthalten.

- E-Mail muss ein @ enthalten.

### Testumfang

Für jeden erstellten Webservice im SAP PI soll ein Funktionstest mit dem SoapUI Tool gemacht werden. Mithilfe des WSDL vom SAP PI ist ein Projekt anzulegen. Mit Hilfe des Request soll die Funktionsweise des Webservice geprüft werden. Funktioniert der Aufruf über das PI bis ins ERP? Werden die Stammdaten korrekt ausgelesen oder geändert? Im Testprotokoll soll der Request und der Respons aufgeführt werden. Pro Service sollen mindestens 3 Testfälle durchgeführt werden.

Weiter soll ein Funktionstest der gesamten Anwendung gemacht werden. Besondere Aufmerksamkeit sollte der Mutation im SAP Backend geschenkt werden, es darf nicht zu Verbuchungsabbrüchen kommen! Der Funktionstest soll vom Umfang her pro Geschäftspartner Ausprägung (Person, Gruppe und Organisation) mindestens 2 Testfälle beinhalten. Der Funktionstest soll die Suche, Anzeige und Änderung beinhalten. Im Testprotokoll sollen die Angaben der Maske in der Webanwendung und die Stammdaten vom SAP ERP ersichtlich sein.

### Dokumentation

Da es sich hier primär um eine Umsetzung eines Prototypen handelt, ist der Fokus auf der technischen Dokumentation. Es wird eine kurze (max. 5 Seiten) technische Dokumentation erwartet, welche aufzeigt welche Schritte gemacht wurden um das Ziel zu erreichen. Zusätzlich sollte ein Diagramm für den Datenfluss und ein kurzer Abschnitt über das Fazit der Umsetzung enthalten sein. Auf alle anderen Dokumentationen wird verzichtet. Wenn Bedarf besteht, kann auf die IPA Dokumentation zurückgegriffen werden.

## Mittel und Methoden

Bei der Programmierung im SAP Umfeld ist die Namenskonvention einzuhalten. Im SAP ERP sollen Funktionsbausteine oder Klassen von SAP verwendet um Daten zu lesen und zu verbuchen. Wenn möglich soll der direkte Zugriff auf Tabellen vermieden werden. Die Webanwendung wird bis ins SAP Stage System Q01 transportiert und dort getestet und abgenommen.

Folgende Mittel stehen für die Umsetzung zu Verfügung:

- WWZ Windows 7 Standard PC mit Zugang auf das Internet.

- MS Office Packet

- SoapUI, Webservice Testtool

- Snagit, für Printscreens

- Notepad++, Texteditor mit XML Unterstützung

- Internet Explorer 11

- XAMPP, als Webserver

- atom.io, für die Erstellung der Webanwendung

- MockFlow, für das Erstellen der Mockup's

- HTML und JavaScript, für die Realisierung der Webanwendung

- SAP PI Weboberfläche, für das Anlegen der PI Schnittstellen

- SAP ERP ABAP, für das Programmieren in Proxyklassen

## Vorkenntnisse

Nicole hat Erfahrungen im Bereich der Programmierung im SAP sei ca. 1,5 Jahren. Jedoch hatte sie vom Letzten Herbst bis Februar dieses Jahrs ein Projekt geleitet ohne regelmässige Programmierungen. Im Bereich Webentwicklung bringt sie vor allem die Erfahrungen aus der Schule mit. In der WWZ hat sie im Bereich Webentwicklung nur 2-3 kleinere Aufgaben gemacht.

## Vorarbeiten

Es wurde ein kleiner POC mithilfe einer vorhandenen Webseite gemacht, um zu schauen ob für Nicole eine solche IPA umsetzbar ist.

## Neue Lerninhalte

Alle Systeme/Funktionen und Methoden zur Umsetzung der Anwendung sind bekannt. Funktionsbausteine für das Lesen und Mutieren der Daten sind evtl. noch nicht bekannt oder es müssen Alternativen gewählt werden.

SAP PI und das Anlegen von Webservice ist für Nicole recht neu. Die Behandlung der Datenstrukturen aus dem SAOP Request können evtl. für Schwierigkeiten sorgen.

## Arbeiten in den letzten 6 Monaten

Oktober bis Januar 2017 hat Nicole vor allem an dem Relaunch unserer Intranet Seite gearbeitet. Bei diesem Projekt hat sie die Teilprojektleitung und Koordination mit externen Partner gemacht, Programmierung von ihrer Seite war keine involviert. Februar bis März hat Nicole im SAP Umfeld Übungen, kleinere Programmierungen und Anpassungen an Programmen gemacht. Ebenfalls hat sie Erfahrungen mit Webservice im Zusammenhang mit dem SAP PI gesammelt.

# Projektorganisation

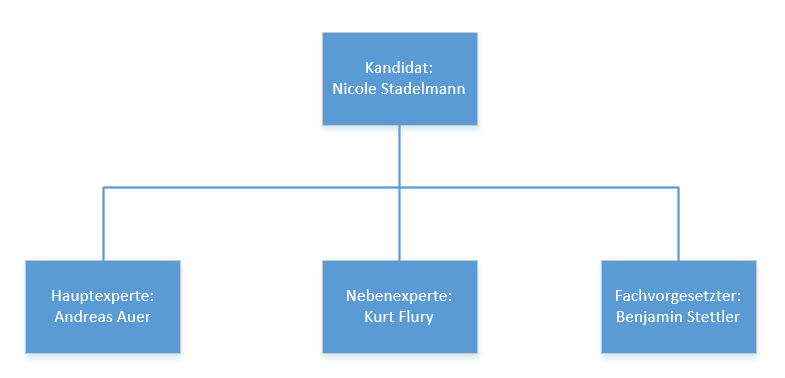


Abbildung 1: Projektorganisation

## Projektrollen

|  |  |
| --- | --- |
| Rolle | Tätigkeit |
| Kandidat | Führt die IPA anhand der Vorgaben der Detailbeschreibung durch. Wird nach der Ausführung anhand der Beurteilungskriterien bewertet. |
| Hauptexperte | Der Hauptexperte Validiert und kontrolliert die Kriterien der IPA. Desweitern besucht er den Kandidaten während der IPA in der Firma. Er bewertet die IPA anhand der Beurteilungskriterien und führt das Fachgespräch. |
| Nebenexperte | Der Nebenexperte unterstütz die Hauptexperte. Er nimmt ebenfalls an der Präsentation und dem Fachgespräch teil. |
| Fachvorgesetzter | Ist während der IPA Zeit anwesend und begleitet den Kandidaten in dieser Zeit. Es beurteilt das Arbeitsverhalten und bewertet am Ende ebenfalls die Arbeit des Kandidaten. |

## Projektmethode

Für die Umsetzung des Projektes wurde die Projektmanagementmethode IPERKA gewählt. IPERKA besteht aus 6 Phasen: Informieren, Planen, Entscheiden, Realisieren, Kontrollieren und Analysieren.

I Informieren

In der ersten Phase informieren sich die entsprechen Personen über den Projektauftrag. Anfällige Fragen können bereits zu Beginn auftauchen, und geklärt werden. Zudem wird dadurch klar, was als Ergebnis erwartet wird.

P Planen

In der zweiten Phase wird beschrieben wie man bei der Arbeit vorgehen möchte, und welche Optionen es gibt. Es wird ein Zeitplan mit den verschiedenen Teilschritten/ Aufgaben erstellt und deren Bearbeitungsdauer geschätzt. Des Weiteren werden verschiedene Lösungsmöglichkeiten geprüft.

E Entscheiden

In der dritten Phase muss das weitere Vorgehen bestimmt werden. Es wird beschrieben welche Lösungsvarianten gewählt wurden und warum.

R Realisieren

In dieser Phase werden die geplanten Arbeitsschritte umgesetzte. Der Zeitplan sollte hierbei so gut wie möglich eingehalten werden. Die effektive Arbeitszeit für die einzelnen Arbeitsschritte müssen festgehalten werden und im Arbeitsjournal notiert werden.

K Kontrollieren

In dieser Phase sollen die erhaltenen Ergebnisse getestet werden. Durch die Kontrolle wird überprüft, ob die umgesetzte Arbeit den Anforderungen entsprechen. Die Resultate der Tests sind schriftlich festzuhalten. Missstände müssen klar aufgezeigt werden.

A Auswerten

In der letzten Phase wird die gesamte Arbeit ausgewertet. Es wird festgehalten, ob alle Ziele und Anforderungen erfüllt wurden. Ebenfalls eine Selbsteinschätzung und Reflexion wird in dieser Phase erfasst.

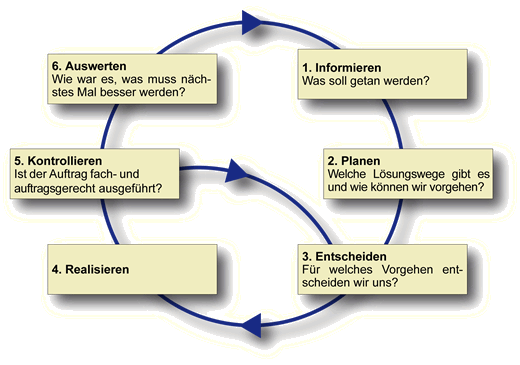


Abbildung 2: Das IPERKA Modell

### Umsetzung der Projektmethode

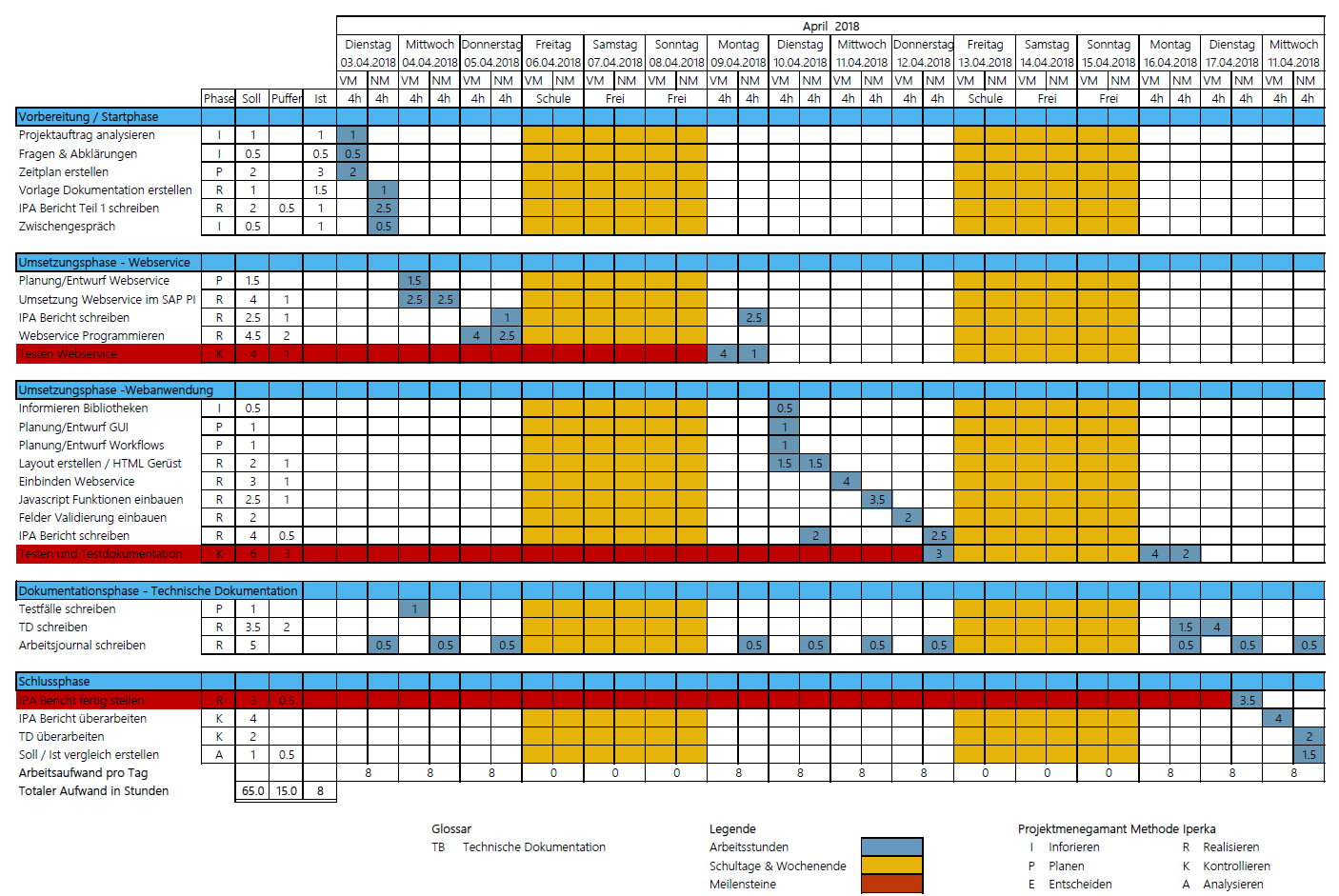
Für dieses Projekt wurde entschieden, die Projektmethode wie üblich auf das gesamte Projekt anzuwenden. Zusätzlich werden aber auch auf einzelne Arbeitspakete die Methode angewendet. So werden für einzelne Arbeitspakete Informationen gesammelt, geplant, Entscheidungen getroffen, realisiert und Kontrolliert.

Dadurch kann eine bessere Planung der einzelnen Arbeitspakete gewährleistet werden. Zusätzlich beeinflussen so Entscheidungen oder mögliche Änderungen in der Umsetzung die nachfolgenden Arbeitspakte nicht/weniger, da diese erst später geplant werden. Somit kann die Planung auf dem abgeschlossen vorgängigen Arbeitspaket aufgebaut werden.

Folgende Arbeitspakete beinhaltet das Projekt:

* Umsetzung des Webservice
* Umsetzung der Webanwendung

# Zeitplan



# Arbeitsjournal

**Legende:**

|  |  |
| --- | --- |
|  | An Arbeitspakt gearbeitet und beendet |
|  | Nicht an diesem Arbeitspaket gearbeitet |
|  | Arbeit an nicht geplantem Arbeitspaket |

|  |  |  |  |
| --- | --- | --- | --- |
| **Dienstag, 03. April 2018** | **Soll** | **Puffer** | **Ist** |
| 1. **Projektauftrag analysieren**   Mit der Analyse des Projektauftrags und der Beurteilungen konnte ich mir einen Überblick über die verlangten Aufgaben und mir die ersten Gedanken zu der Umsetzung des Projektes machen. | **1** |  | **1** |
| 1. **Fragen & Abklärungen**   Waren zu diesem Zeitpunkt noch keine notwendig. | **0.5** |  | **0** |
| 1. **Zeitplan erstellen**   Das Erstellen des Zeitplans beanspruchte mehr Zeit als gedacht. Dies aufgrund der Übersichtlichkeit und der Schätzung der benötigenden Zeiten. Mir wurde hier bewusst, dass 80 Stunden nicht so viel ist, wie man Anfangs meint.   1. **Vorlage Dokumentation erstellen**   Ich habe heute die Dokumentationsvorlage erstellt. Ich habe dies bewusst heute eingeplant, da ich nicht am Ende des Projektes Zeitdruck bekommen wollte, wegen der Gestaltung des Dokuments.   1. **IPA Bericht Teil 1 schreiben**   Später schrieb ich den 1. Teil des IPAs Bericht. Ich hatte dafür nicht so lange wie gedacht, da ich die Detail Beschreibung 1 zu 1 übernehmen konnte. Auch der Zeitplan war bereits erstellt und konnte nur noch eingefügt werden. Die meiste Zeit benötiget ich für dich Erstellung des Diagramms der Projektorganisation und die Rollenbeschreibungen.   1. **Zwischengespräch**   Gegen Ende des ersten Tages kam Herr Auer vorbei. Im Zwischengespräch gingen wir gemeinsam die Checkliste durch, klärten allfällige Fragen. Des Weiteren zeigte ich meinen Zeitplan und was ich bereits erarbeitet habe.   1. **Arbeitsjournal** | **2**  **1**  **2**  **0.5**  **0.5** | **0.5** | **3**  **1.5**  **1.5**  **0.5**  **0.5** |
| **Total:** | **7.5** | **0.5** | **8** |

Der Start heute was leider etwas stressig. Am Morgen, bevor ich mit der Arbeit beginnen konnte, erfuhr ich, dass es Probleme mit dem Intranet gab. Danach konnte ich endlich mit meiner IPA starten. Der erste Tag verlief ohne weitere Probleme. Ich freue mich auf die weitern Tage.

|  |  |  |  |
| --- | --- | --- | --- |
| **Mittwoch, 04. April 2018** | **Soll** | **Puffer** | **Ist** |
| 1. **Planung / Entwurf Webservice**   Ich begann den Tag mit dem Entwerfen der Webservices. Ich hatte etwas länger als gedacht, da das Abbilden der Elemente der Request und Response Nachrichten doch sehr aufwändig war. Jedoch bin ich froh, dass ich dies gemacht habe, da es mir bei der Umsetzung sehr geholfen hat und ich wiederum dort Zeit einsparen konnte. | **1.5** |  | **2.5** |
| 1. **Umsetzung Webservice im SAP PI**   Bei diesem Arbeitsschritt war ich um einiges schneller als geplant. Ohne Zweifel hat mir die Planung im Arbeitsschritt zuvor sehr viel geholfen, da ich mir bei der Struktur keine Gedanken mehr machen musste und genau wusste, was ich will.   1. **Testfälle schreiben**   Inzwischen hatte ich ein Bild davon, wie meine Webanwendung am Ende aussehen sollte. Aufgrund meiner Vorstellungen und den Vorganden des Projektes schrieb ich die Testfälle. Jedoch benötigte ich hierfür mehr Zeit als gedacht, da die Anzahl der Testfälle doch grösser war, als Anfangs angenommen. | **4**  **1** | **1** | **2.5**  **2.5** |
| 1. **Arbeitsjournal**   Am Ende des Tages schrieb ich das Arbeitsjournal | **0.5** |  | **0.5** |
| **Total:** | **7** | **1** | **8** |

Die für heute geplanten Aufgaben konnte ich alle so erfüllen, auch wen die eingeplanten Zeiten meistens falsch geschätzt waren. Das, was ich bei den Testfällen und der Planung an Zeitverloren habe, gewann ich durch die schnelle Fertigstellung der Webservices im PI.

Ich bin mit dem Tag sehr zufrieden. Ich war produktiv und konnte meine geplanten Aufgaben alle erfüllen. Ich hoffe, dass ich auch im weiteren Verlauf des Projektes so gut vorankommen werde wie heute.

|  |  |  |  |
| --- | --- | --- | --- |
| **Donnerstag, 05. April 2018** | **Soll** | **Puffer** | **Ist** |
| 1. **Webservice Programmieren**   Das Programmieren der Webservices verlief ohne grosse Probleme. Allerdings benötigte ich doch etwas mehr Zeit als ursprünglich eingeplant was dank dem Puffer aber keine Probleme machte. | **4.5** | **2** | **5.5** |
| 1. **IPA Bericht schreiben**   Da ich nicht den ganzen Puffer von dem vorherigen Arbeitsschritt benötigte, konnte ich länger an dem IPA Bericht schreiben als geplant. Dies zu meinem Glück, denn ich war mit der Dokumentation im Rückstand. | **1** |  | **2** |
| 1. **Arbeitsjournal**   Am Ende des Tages schrieb ich das Arbeitsjournal. | **0.5** |  | **0.5** |
| **Total:** | **6** | **2** | **8** |

Die Programmierung der Webservices ging gut und schnell voran. Da ich die verwendeten Funktionsbausteine bereits kannte, wusste ich wie ich diese zu verwenden hatte. Einzig das Übergeben der gefundenen Daten war bei jenen schwer, wo mehrere gefunden werden konnte. Aber nach dem ersten Webservice hatte ich verstanden, wie das geht, und konnte auch dies bei den anderen schnell umsetzen.

Das Arbeiten am IPA Bericht war wichtig, den ich hinkte hier stark hinterher. Zum Glück hatte ich am Vortag, bei der Planung der Webservices, alles gut dokumentiert, so, dass ich dies eins zu eins in den IPA Bericht kopieren konnte. So musste ich nur noch die Realisierung der Webservices im IPA Bericht nachtragen.

Ich bin noch immer guter Dinge. Das Programmieren hatte mir Spass gemacht und ich freue mich darauf, die Webservices testen zu können. Der Zeitplan ist bisher gut aufgegangen. Ich hoffe dies bleibt so.

|  |  |  |  |
| --- | --- | --- | --- |
| **Montag, 09. April 2018** | **Soll** | **Puffer** | **Ist** |
| 1. **Webservice Testen**   Das testen der Webservices verlief ohne Probleme. Es traten wenige Fehler auf, welche ich gleich nach dem testen korrigierte. Deswegen auch die grosszügige Soll Zeit. Nachdem ich die Fehler korrigiert hatte, testete ich die fehlerhaften Testfälle gleich nochmal. Zusätzlich musste ich für jeden SOAP UI Test die Request und Response Nachricht kopieren, was noch zusätzlich Zeit kostete. | **4** | **1** | **5** |
| 1. **IPA Bericht schreiben**   Da ich am Vortag bereits mehr Zeit in den IPA Bericht verwenden konnte, musste ich heute nicht mehr so viel machen. Ich schrieb den Abschnitt über die Programmierung der Webservices fertig.   1. **Informieren Bibliotheken**   Da ich an diesem Tag weniger Zeit für fast alle Aufgaben benötigte als geplant, konnte ich bereits mit einer Aufgabe des Folgetages beginnen. Ich verwendete mehr Zeit für die Informationsbeschaffung der Bibliotheken als eigentlich geplant. | **2.5**  **0.5** |  | **1.5**  **1** |
| 1. **Arbeitsjournal**   Am Ende des Tages schrieb ich das Arbeitsjournal. | **0.5** |  | **0.5** |
| **Total:** | **6** | **2** | **8** |

Der heutige Tag verlief gut. Ich konnte alle Webservicetests durchführen und gleich die notwendigen Korrekturen vornehmen. Auch beim IPA Bericht ging es gut voran, so dass ich bereits eine Aufgabe des Folgetages beginnen und abschliessen konnte.

Ich bin zurzeit entspannt und freu mich, dass ich so gut in der Zeit liege.

|  |  |  |  |
| --- | --- | --- | --- |
| **Dienstag, 10. April 2018** | **Soll** | **Puffer** | **Ist** |
| 1. **Informieren Bibliotheken**   Dieser Arbeitsschritt konnte bereits am Vortag ausgeführt werden.   1. **Planung / Entwurf GUI**   Obwohl ich im Kopf schon von Beginn an ein klares Bild der Webanwendung hatte, benötigte das erstellen des Entwurfes ein bisschen mehr Zeit als gedacht. | **0.5**  **1** |  | **-**  **1.5** |
| 1. **Planung / Entwurf Workflows**   Auch bei den Workflows benötigte ich ein wenig mehr Zeit als geplant, da ich mir genau überlegen musst, wie alles abläuft und ineinander spielt.   1. **HTML Grundgerüst erstellen**   Beim Erstellen des HTML Grundgerüstes kam ich gut voran. Ich programmierte alles, was ich bereits ohne JS Logik programmieren konnte. Vor allem die 2 Modals und der Hilfstext beanspruchten eine gewisse Zeit. Ebenso das Formular mit den Inputfeldern für die Suche.   1. **IPA Bericht schreiben**   Beim IPA Bericht konnte ich gleich die Planung der Webanwendung schreiben. Für die Umsetzung des HTML Gerüstes hat die Zeit leider nicht mehr gereicht. | **1**  **2**  **2** | **1** | **1.5**  **2.5**  **2** |
| 1. **Arbeitsjournal**   Am Ende des Tages schrieb ich das Arbeitsjournal. | **0.5** |  | **0.5** |
| **Total:** | **7** | **1** | **8** |

Auch heute kam ich wieder gut voran. Alle geplanten aufgaben konnte ich abschliessen, ausser der IPA Bericht ist noch nicht ganz auf aktuellsten Stand. Die Hälfte der Zeit ist heute vorbei, und ich bin guter Dinge. Der Zeitplan geht bisher gut auf, was hoffentlich auch so bleibt. Morgen beginnt das Einbinden der Webservices und das Programmieren der JS Funktionen. Darauf freue ich mich bereits.

|  |  |  |  |
| --- | --- | --- | --- |
| **Mittwoch, 11. April 2018** | **Soll** | **Puffer** | **Ist** |
| 1. **Einbinden Webservice**   Das Einbinden der Webservices stellte mich vor eine Herausforderung. Da ich die verwendete Bibliothek noch nie verwendet habe, musste ich zuerst herausfinden, wie man diese einsetzt. Obwohl eine kleine Dokumentation vorhanden war, musste ich viele verschiedene Einstellungen versuchen, bis ich den ersten Webservice zum laufen brachte. Beim letzten Webservice musste ich noch eine tiefere Struktur zurückgeben, was nochmals einige Zeit beanspruchte, bis ich herausgefunden hatte, wie ich dies bewerkstelligen konnte. Schlussendlich hatte ich trotz des eingeplanten Puffers zu wenig Zeit.   1. **JS Funktionen einbinden**   Beim programmieren der JS Funktionen ging es gleich weiter wie beim Einbinden der Webservices. Die Response Nachrichten des Webservice konvertierte ich jeweils in ein JSON, da dies einfacher zum auslesen war. Jedoch war es noch immer schwer, die richtigen Pfade anzugeben, um an die gewünschten Daten zu kommen. Nach diesen 2.5 Stunden, wusste ich gerade einmal, wie ich die Daten auslesen konnte. Mehr hatte ich aber noch nicht gemacht. | **3**  **2.5** | **1**  **1** | **5**  **2.5** |
| 1. **Arbeitsjournal**   Am Ende des Tages schrieb ich das Arbeitsjournal. | **0.5** |  | **0.5** |
| **Total:** | **6** | **2** | **8** |

Der heutige Tag war bezüglich der Planung ein Desaster. Obwohl ich wusste, dass ich zuerst herausfinden musste, wie ich die Bibliothek verwende, hätte ich nicht gedacht, dass mich das Einbinden der Webservices ganze 5 Stunden kosten wird. Auch bei den restlichen JS Funktionen lief es nicht besser. Wie ich die JSON Pfade richtig angeben muss, war kompliziert. Erst durch das Debuggen und das analysieren des JSON Objektes während dem Debuggen zeigte mir die Pfade an, welche ich benötigte. Diese schienen recht seltsam. Ich muss gestehen das ich nicht weiss, ob dies die richtigen Pfade sind, oder ob es noch eine bessere Lösung dafür gegeben hätte. Was aber zählt ist, dass ich eine funktionierende Lösung gefunden habe.

Der heutige tag hat mich jedoch Zeitplan mässig nach hinten geworfen. Ich hoffe es gelingt mir, dies wieder aufzuholen.

|  |  |  |  |
| --- | --- | --- | --- |
| **Donnerstag, 12. April 2018** | **Soll** | **Puffer** | **Ist** |
| 1. **JS Funktionen einbinden**   Eigentlich sollte dieser Arbeitsschritt gestern beendet worden sein. Dies war jedoch leider nicht möglich. Auch heute ging es nicht voran. Am Morgen hatte ich es sogleich mit einem Fehler zu tun, welcher erst 3 Stunden später gelöst war. Immerhin hatte ich den Dreh in den letzten langsam raus und konnte die JS Funktionen so gut wie abschliessen. Nur noch die Felder Validierung musste noch erledigt werden.   1. **Felder Validierung einbauen**   Zum glück mussten nicht viele Felder validiert werden. So konnte ich diese in weniger als der geplanten Zeit erarbeiten und wieder ein wenig Zeit aufholen.   1. **IPA Bericht schrieben**   An diesem Arbeitsschritt konnte ich heute leider nicht arbeiten.   1. **Testen und Testdokumentation**   An diesem Arbeitsschritt konnte ich heute leider nicht arbeiten. | **-**  **2**  **2.5**  **3** | **-** | **6**  **1.5**  **-**  **-** |
| 1. **Arbeitsjournal**   Am Ende des Tages schrieb ich das Arbeitsjournal. | **0.5** |  | **0.5** |
| **Total:** | **6** | **2** | **8** |

Der Fehler am Morgen hat mich noch weiter zurückgeworfen. Ich bekam vom Webservice Aufruf einen Fehler zurück, dass keine Response Nachricht erhalten wurde. Jedoch sah ich im PI dass die Request Nachricht erhalten und erfolgreich eine Response Nachricht gesendet wurde. Irgendwo schein diese aber verloren zu gehen. Nach 3 Stunden intensiver Fehlersuche war der Fehler plötzlich verschwunden. Die Ursache ist mir aber noch immer nicht bekannt. Leider weitere 3 Stunden welche ich verloren habe… langsam mache ich mir sorgen, ob ich die verlorene zeit noch aufholen kann. Es bleiben mir nur noch 3 Tage für alle restlichen Aufgaben…

|  |  |  |  |
| --- | --- | --- | --- |
| **Montag, 16. April 2018** | **Soll** | **Puffer** | **Ist** |
| 1. **JS Funktionen einbinden (Arbeit vom Vortag)**   Eigentlich wollte ich bereits am Donnerstag mit diesem Arbeitsschritt beginnen. was leider nicht möglich war. Daher war ich heute den ganzen Tag am Testen. Die gefundenen Fehler begann ich nach dem Testen gleich zu korrigieren. Dafür habe ich bei diesem Arbeitsschritt ebenfalls Zeit eingeplant.   1. **Testdokumentation**   An diesem Arbeitsschritt wurde heute nicht gearbeitet. | **6**  **1.5** | **3** | **7.5**  **0** |
| 1. **Arbeitsjournal**   Am Ende des Tages schrieb ich das Arbeitsjournal. | **0.5** |  | **0.5** |
| **Total:** | **6** | **2** | **8** |

Heute war ich den ganzen Tag am Testen und gefundene Fehler am korrigieren. Im Nachhinein bin ich mir nicht sicher, ob ich besser die Fehler so belassen hätte, damit ich mich mehr der Dokumentation hätte widmen können. Ich bin mir nicht sicher, ob ich mit allem noch fertig werden kann in den nächsten 2 Tagen… langsam mach ich mir Sorgen.



Teil 2

Projektdokumentation

# Kurzfassung des IPA Berichts (Management Summary)

## Kurze Ausgangssituation

In der WWZ AG ist SAP das führende System. Wichtige Informationen zu Geschäftspartner sind über verschiedene Transaktionen und Tabellen verteilt. Dies erschwert das zusammentragen und ändern von den gewünschten Daten, vor allem für noch nicht geübte oder Gelegenheitsanwender.

## Umsetzung

Um zu testen, ob Daten aus dem System ausgelesen und werden können, und auf einer Webseite dargestellt werden können, soll ein Prototyp umgesetzte werden. Des Weiteren soll die Möglichkeit getestet werden, Daten in der Webanwendung zu ändern und ins SAP ERP zu übertagen und zu speichern.

Um dies zu verwirklichen muss zuerst die Herkunft der Daten im SAP ERP abgeklärt werden, und nach einer möglichen Lösung, diese einfach im SAP auslesen/ändern zu können. Des Weiteren müssen Webservices erstellt werden, welche die Kommunikation zwischen Web und SAP System und der Daten Übergabe regeln.ss

Die Seite wird mit HTML und JavaScript erstellt. Sie soll selbsterklärend sein, und die Eingaben überprüfen. Es sollen mehrere Geschäftspartner zurückgegeben werden können, woraufhin einer davon ausgewählt werden kann. Persönliche Informationen und QMC Informationen des Geschäftspartners sollen angezeigt werden. Ein Änderungsmodus soll das Ändern der persönlichen Informationen ermöglichen und gleichzeitig verhindern, dass ausversehen Änderungen durchgeführt werden. Die Eingaben der neuen Werte im Änderungsmodus müssen überprüft werden.

## Ergebnis

Geschäftspartner können über die Webseite gesucht werden. Die Eingaben der Suchkriterien werden überprüft und eine Fehlermeldung zurückgegen, falls die Eingaben nicht erlaubt sind. Im Falle, dass mehrere Treffer gefunden werden, kann der Anwender einen Geschäftspartner aus einer Liste auswählen. Die Persönlichen Informationen des GPs werden angezeigt und können über einen Button, wecher den Änderungsmodus aktiviert, geändert werden. Die Eingaben werden überprüft und Fehlermeldungen ausgegeben, wenn nicht erlaubte eingaben vorhanden sind. Bei mehreren QMC IDs kann der Anwender zwischen allen IDs auswählen und die Produkte anzeigen lassen.

Tritt bei den Webservice selbst einen Fehler auf, wird dies erkannt und eine entsprechende Fehlermeldung ausgegeben.

Die Webanwendung ist selbsterklärend. Für den Fall, dass trotzdem Fragen auftauchen, kann über einen Hilfe-Button ein Hilfs-text eingeblendet werden.

# Vorbereitung

## Infrastruktur und Organisatorisches

Für die IPA wurde ein Arbeitsplatz in einem separaten Raum eingerichtet, damit keine Störungen von Telefonaten, Gesprächen oder anderen Geräuschen verursacht wird.

Auf GitHub wurde ein Account eingerichtet, mit zwei Repositories.

1. Repositorie:

Im ersten Repositorie werden alle benötigten Dokumente, wie der IPA-Bricht, Testfälle, Zeitplan und Notizen abgelegt. Über die GitHub Desktop Anwendung wird das Repositorie auf das persönliche Home-Verzeichnis geklont.

1. Repositorie:

Im zweiten Repositorie wird der Projektordner mit je einem Ordner für JS und CSS Files, Bilder und Fonts erstellt. Dieses Repositorie wird in den htdocs Ordner von XAMPP geklont damit sie lokal über einen Webserver aufgerufen werden können.

Über die GitHub Desktop Anwendung kann ich beide Repositories einfach auf die GitHub hochladen.

## Risikomanagement

Um einen Datenverlust zu verhindern werden folgende Vorkehrungen getroffen:

* Mindestens 2x am Tag werden alle Dokumente und Files auf GitHub hochgeladen. Zusätzlich kann durch das Hochladen auf GitHub das Projekt Versioniert werden und alte Versionen von Files und Dokumenten wiederhergestellt werden.
* Den XAMPP Ordner sowie alle IPA Dokumente und Files werden zusätzlich mindesten 2x am Tag auf dem persönlichem Home Verzeichnis gesichert. Von diesem Verzeichnis erstellt die WWZ standartmässig jede Stunde ein Backup.
* SAP?

## Analyse des Projektauftrags

Als erster Arbeitsschritt wurde der Projektauftrag und die Beurteilungskriterien analysiert. Anhand der erhaltenen Informationen wurden die einzelnen Arbeitsschritte erstellt, auf welchen der Zeitplan aufgebaut wurde.

Der Projektauftrag befindet sich in diesem Dokument unter Punkt 1 und der Zeitplan unter Punkt 3.

# Entwurf Webservice

## Überlegungen

Im SAP werden die Geschäftspartner in verschiedenen GP-Typen unterteilt. Zum erhalten der

Auf der Webseite soll schlussendlich nur die Daten von einem GP angezeigt werden. Um trotzdem eine Suchfunktion zu ermöglichen, welche mehrere Treffer zurückgeben kann, muss je ein Webservice für die GP suche und einen für das erhalten der GP Informationen erstellt werden.

Es sollen zwei verschiedene Arten von GP Informationen angezeigt werden. Die Persönliche Informationen und die QMC Produkte. Um dies zu verwirklichen könnte ein Web Service erstellt werden, welcher beide Informationen auf einmal ausliest oder zwei Web Service, welche die beiden Informationsarten getrennt beschafft.

Um die Informationen eines GPs zu ändern muss ein weiterer Web Service erstellt werden, welcher die neuen daten dem SAP ERP System übergibt und deren Speicherung auslöst.

Für die Verwirklichung aller Funktionen müssen daher 3- 4 Webservice erstellt werden.

Um beim Erstellen der Webservice keine wichtigen Elemente bei den Request und Response Nachrichten zu vergessen, wird für jeden Webservice einen Entwurf erstellt. Die Entwürfe sollen Name und Funktion des Webservice beschriebt und die Struktur der Request und Response Nachrichten aufzeigen. Zusätzlich soll die Struktur die enthaltenen Elemente, ihre Kardinalität und Beziehung zu den anderen Elementen aufzeigen.

Des Weiteren soll jede Response Nachricht eine Message Element enthalten. Dies soll ermöglichen, Fehlermeldungen welche während dem ausführen des Web Service aufgetaucht sind, dem Anwender in der Webanwendung auszugeben.

## Benötigte Funktionsbausteine und Tabellen

Nachfolgend werden die für die Programmierung der Webservice benötigte Tabelle und Funktionsbausteine aufgeführt und ihre Funktionen und Informationen beschrieben.

### Funktionsbauseine

Folgende Funktionsbausteine werden für das Programmieren der Web Service im SAP ERP verwendet:

* BAPI\_BUPA\_SEARCH\_2
* BAPI\_BUPA\_CENTRAL\_GETDETAIL
* BAPI\_BUPA\_ADRESS\_GETDETAIL
* BAPI\_BUPA\_CENTRAL\_CHANGE

BAPI\_BUPA\_SEARCH\_2

Durch diesen Funktionsbaustein kann nach GPs gesucht werden. Als Suchkriterien können zentrale Geschäftspartnerdaten und Adressattribute angegeben werden.

BAPI\_BUPA\_CENTRAL\_GETDETAIL

Dieser Funktionsbaustein gibt zentrale Geschäftspartnerdaten, wie Personen-, Firmen- oder Gruppenname, GP-Typ, Anrede usw., zu einem GP zurück.

BAPI\_BUPA\_ADRESS\_GETDETAIL

Dieser Funktionsbaustein gibt die Adressdaten eines GPs zurück.

BAPI\_BUPA\_CENTRAL\_CHANGE

Mit diesem Funktionsbaustein können zentrale Geschäftspartnerdate, wie Personen-, Firmen- oder Gruppenname, Telefonnummern oder Mailadressen, geändert werden.

### SAP Tabellen

Folgende SAP Tabellen werden für das Programmieren der Web Service im SAP ERP verwendet:

* ZBC\_QMC\_CRM\_DATA
* BUT000

ZBC\_QMC\_CRM\_DATA

In dieser Tabelle befinden sich alle QMC IDs und Informationen zu deren QMC Produkte.

BUT000

In dieser Tabelle befinden sich alle GPs mit zentralen Informationen.

## Entwurf Webservice 1: Kunden suche

**Name:** BP\_Search

**Funktion:** Durch diesen Web Service soll der Anwender über verschiedene Angaben einen GP suchen können. Zurückgeben soll der Web Service eine Liste aller Treffer, auf welche die vom Anwendern angegeben Angaben zutreffen.

**Request Felder**

|  |  |  |  |
| --- | --- | --- | --- |
| Feld Name | Feld Wert | Kardinalität | Kind von: |
| GP\_No | GP Nummer | 0…1 |  |
| Name\_1 | Vorname oder Namenszusatz | 0...1 |  |
| Name\_2 | Firmen, - Gruppen oder Nachname | 0…1 |  |
| GP\_Type | Geschäftspartner Type | 0…1 |  |
| Street | Strasse | 0…1 |  |
| House\_No | Hausnummer | 0…1 |  |
| PLZ | PLZ | 0…1 |  |
| City | Stadt | 0…1 |  |

Tabelle 1: Entwurf BP\_Search Request

Response Felder

|  |  |  |  |
| --- | --- | --- | --- |
| Feld Name | Feld Wert | Kardinalität | Kind von: |
| Result\_Count | Anzahl gefunden GPs | 1 |  |
| Person\_List |  | 0…1 |  |
| Person\_Details |  | 0…unbounded | Person\_List |
| GP\_No | GP Nummer | 0…1 | Person\_Details |
| Name\_2 | Name 2 (Bei Privatpersonen Vorname) | 0…1 | Person\_Details |
| Name\_1 | Name 1 | 1 | Person\_Details |
| GP\_Type | Geschäftspartner Type | 1 | Person\_Details |
| Message | Nachricht | 0…1 |  |
| Message\_Text | Nachrichten Text | 1 | Message |
| Message\_Type | Nachrichten Type | 1 | Message |

Tabelle 2: Entwurf BP\_Search Response

## Entwurf Webservice 2: Kundendaten auslesen

**Name:** BP\_Details

**Funktion:** Der Webservice soll ausgeführt werden, nachdem sich der Anwender einen GP, aus der vom Webservice 1 zurückgegebenen liste, ausgesucht hat.

Der Webservice soll alle relevanten GP daten des ausgewählten GPs suchen und zurückgeben.

**Request Felder**

|  |  |  |  |
| --- | --- | --- | --- |
| Feld Name | Feld Wert | Kardinalität | Kind von: |
| GP\_No | GP Nummer | 1 |  |

Tabelle 3: Entwurf BP\_Details Request

**Response Felder**

|  |  |  |  |
| --- | --- | --- | --- |
| Feld Name | Feld Wert | Kardinalität | Kind von: |
| GP\_Type | Geschäftspartner Type | 1 |  |
| Title | Anrede | 0…1 |  |
| Lastname | Name | 0…1 |  |
| Firstname | Vorname | 0…1 |  |
| Name\_1 | Name 1 | 0…1 |  |
| Name\_2 | Name 2 | 0…1 |  |
| City | Stadt | 0…1 |  |
| Street | Strasse | 0…1 |  |
| PLZ | PLZ | 0…1 |  |
| Country | Land | 0…1 |  |
| Region | Region | 0…1 |  |
| Tel\_List | Telefon Nummern Liste | 0..1 |  |
| Tel\_Count | Anzahl gefundener Telefonnummern | 1 | Tel\_List |
| * Tel\_Detail |  | 0…unbounded | Tel\_List |
| * Tel\_No | Telefon Nummern | 1 | Tel\_Detail |
| * Tel\_ID | Telefon ID | 1 | Tel\_Detail |
| E\_Mail\_List | E-Mail-Adressen Liste | 0..1 |  |
| E\_Mail\_Count | Anzahl der gefunden E-Mail-Adressen | 1 |  |
| * E\_Mail\_Detail |  | 0…unbounded | E\_Mail\_List |
| * E\_Mail\_Adress | E-Mail-Adresse | 1 | E\_Mail\_Detail |
| * E\_Mail\_ID | E-Mail ID | 1 | E\_Mail\_Detail |
| Message | Nachricht | 0…1 |  |
| Message\_Text | Nachrichten Text | 1 | Message |
| Message\_Type | Nachrichten Type | 1 | Message |

Tabelle 4: Entwurf BP\_Details Response

## Entwurf Webservice 3: QMC Produkte auslesen

**Name:** BP\_QMC\_Products

**Funktion:** Der Webservice soll ausgeführt werden, nachdem sich der Anwender einen GP, aus der vom Webservice 1 zurückgegebenen liste, ausgesucht hat.

Der Webservice soll alle relevanten QMC Produkte des GP zurückgeben.

**Request Felder**

|  |  |  |  |
| --- | --- | --- | --- |
| Feld Name | Feld Wert | Kardinalität | Kind von: |
| GP\_No | GP Nummer | 1 |  |

Tabelle 5: Entwurf BP\_QMC\_Products Request

**Response Felder**

|  |  |  |  |
| --- | --- | --- | --- |
| Feld Name | Feld Wert | Kardinalität | Kind von: |
| QMC\_List |  | 0…1 |  |
| Result\_Count | Anzahl gefundener QMC IDs | 1 | QMC\_List |
| QMC\_Details |  | 1… unbounded | QMC\_List |
| QMC\_ID | QMC ID | 0…1 | QMC\_Details |
| Internet\_product | Internet Dienstleistungsprodukt | 0…1 | QMC\_Details |
| Telephone\_product | Telefon Dienstleistungsprodukt | 0…1 | QMC\_Details |
| DTV | DTV | 0…1 | QMC\_Details |
| Verte | Verte! | 0…1 | QMC\_Details |
| QL\_TV | QL TV | 0…1 | QMC\_Details |
| Mobile\_product | Mobile Dienstleistungsprodukt | 0…1 | QMC\_Details |
| Combi\_product | Kombi Dienstleistungsprodukt | 0…1 | QMC\_Details |
| Message | Nachricht | 0…1 |  |
| Message\_Text | Nachrichten Text | 1 | Message |
| Message\_Type | Nachrichten Type | 1 | Message |

Tabelle 6: Entwurf BP\_QMC\_Products Response

## Entwurf Webservice 4: Kundendaten ändern

**Name:** BP\_Update

**Funktion:** Wen der Anwender im Änderungsmodus eine Änderung vornimmt und speichert soll der Web Service ausgeführt werden und die entsprechenden Daten richtig im SAP ERP ändern.

**Request Felder**

|  |  |  |  |
| --- | --- | --- | --- |
| Feld Name | Feld Wert | Kardinalität | Kind von: |
| Title | Anrede | 0…1 |  |
| Name\_1 | Name 1 / Vorname | 0..1 |  |
| Name\_2 | Name 2 / Nachname | 0..1 |  |
| Tel\_Change\_List | Telefon Nummer Liste | 0..1 |  |
| * Tel\_Change\_Detail |  | 0…unbounded | Tel\_Change\_List |
| * Tel\_Number | Telefon Nummer | 1 | Tel\_Change\_Detail |
| * Tel\_ID | Nummer ID | 1 | Tel\_Change\_Detail |
| E\_Mail\_Change\_List | E-Mail Liste | 0..1 |  |
| * Mail\_Change\_Detail |  | 0…unbounded | E\_Mail\_Change\_List |
| * E\_Mail\_Adress | E-Mail-Adresse | 1 | Mail\_Change\_Detail |
| * E\_Mail\_ID | E-Mail ID | 1 | Mail\_Change\_Detail |

Tabelle 7: Entwurf BP\_Update Request

**Response Felder**

|  |  |  |  |
| --- | --- | --- | --- |
| Feld Name | Feld Wert | Kardinalität | Kind von: |
| Message\_List |  | 0…1 |  |
| Message\_Details |  | 0… unbounded | Message\_Details |
| Message \_Text | Nachricht | 1 | Message\_Details |
| Message\_ID | Nachrichten Type | 1 | Message\_Details |

Tabelle 8: Entwurf BP\_Update Response

## Entscheidungen

Wie unter Punkt 8.1 beschrieben, wäre es möglich die GP Informationen und QMC Informationen in einem Webservice auszulesen, oder auf zwei Webservice aufzuteilen.

Zwei getrennte Webservice bieten in der Zukunft bessere Weiterentwicklungsmöglichkeiten im Bereich der Webanwendung. Des Weiteren können bei getrennten Webservice immerhin eine Informationsgruppe angezeigt werden, sofern ein Webservice erfolgreich war.

# Realisierung Webservice

Die WWZ setzt das 3 Komponentenmodel ein. Dies bedeutet, dass ein Service in folgende 3 Teile aufgeteilt ist:

- **Sendendes System (Web / Anfragesteller)**

Name: WEB, 1.0 of wwz.ch

- **SAP PI**

Name: ZSAP\_PI\_WWZ of wwz.ch

- **Empfangendes System (SAP / Antworter)**

Name: ZSAP\_ERP\_WWZ of wwz.ch

Aus diesem Grund müssen fast alle Komponenten 3-mal erstellt werden.

## Enterprise Services Builder

### Namespace erstellen

Zu Beginn muss im Enterprise Services Builder in jeder Komponente ein Namespace erstellt werden. Folgende Namespace wurden für das gesamte Projekt erstellt:

|  |  |  |
| --- | --- | --- |
| **WEB, 1.0 of wwz.ch** | **ZSAP\_ERP\_WWZ of wwz.ch** | **ZSAP\_PI\_WWZ of wwz.ch** |
| http://wwz.ch/S/99980/BP\_Search | http://wwz.ch/R/99980/BP\_Search | http://wwz.ch/X/99980/BP\_Search |
| http://wwz.ch/S/99981/BP\_GetDetails | http://wwz.ch/R/99981/BP\_GetDetails | http://wwz.ch/X/99981/BP\_GetDetails |
| http://wwz.ch/S/99982/BP\_QMCProducts | http://wwz.ch/R/99982/BP\_QMCProducts | http://wwz.ch/X/99982/BP\_QMCProducts |
| http://wwz.ch/S/99983/BP\_Update | http://wwz.ch/R/99983/BP\_Update | http://wwz.ch/X/99983/BP\_Update |

### Data Type erstellen

Die Datentypen definieren die Struktur und die Elemente für die später erstellten Message Types inklusive deren Typen und Kardinalitäten. Als Typ eines Elementes wird in der WWZ standartmässig den xsd:string verwendet.

Die Datentypen werden anhand der Entwürfe erstellt und müssen jeweils in der Web und in der ERP Komponente Erstellt werden.

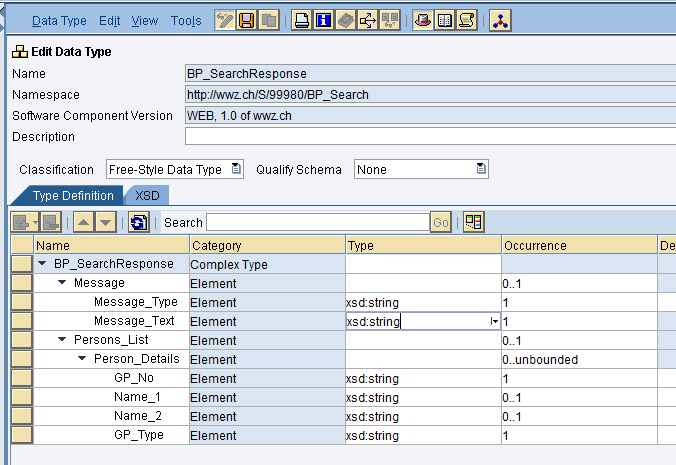


Abbildung 3: Erstellter Datentyp für BP\_Search Response

### Messagetype erstellen

Der Message Type definiert den Aufbau der XML Nachricht für die Request und Response Nachrichten. Ihm wird ein Data Type zugewiesen, welcher die Struktur der XML Nachricht beschreibt. Die Struktur wird automatisch erstellt, sobald ein Data Type zugewiesen wurde.

Des Message Type muss immer für den Request und die Response angelegt werden. Und dies jeweils im in der Web- und ERP- Komponente

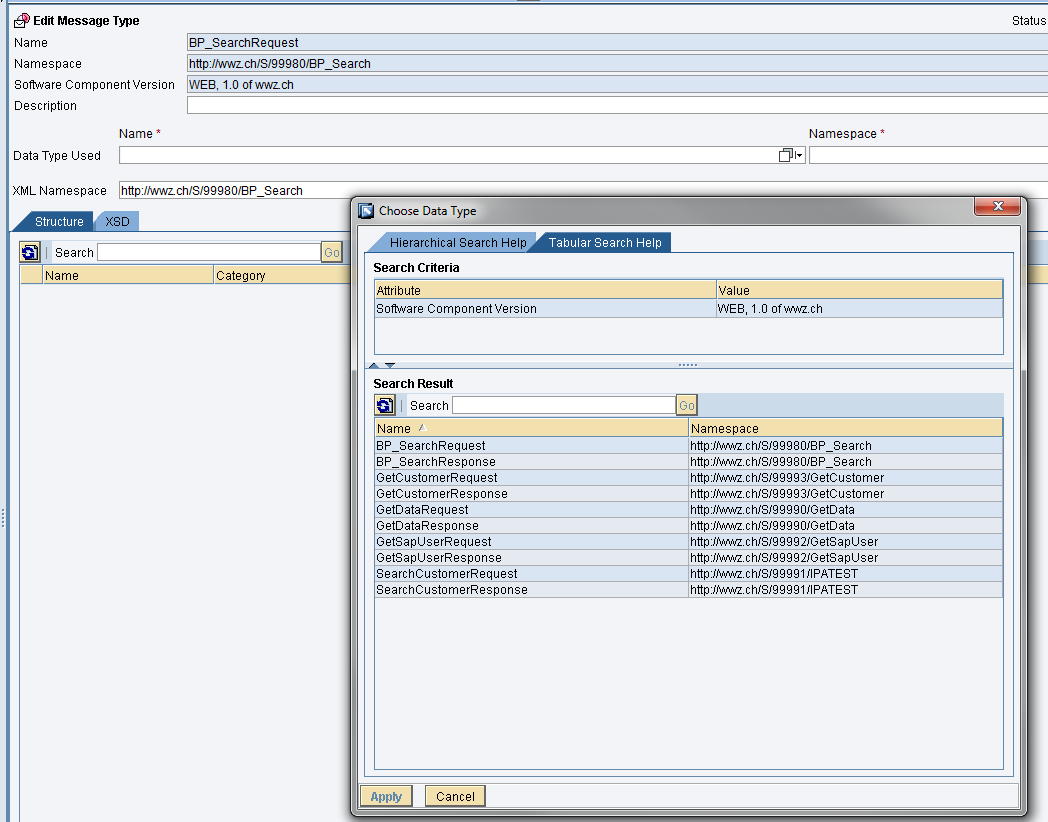


Abbildung 4: Auswählen des Data Types für den Request Message Type von BP\_Search

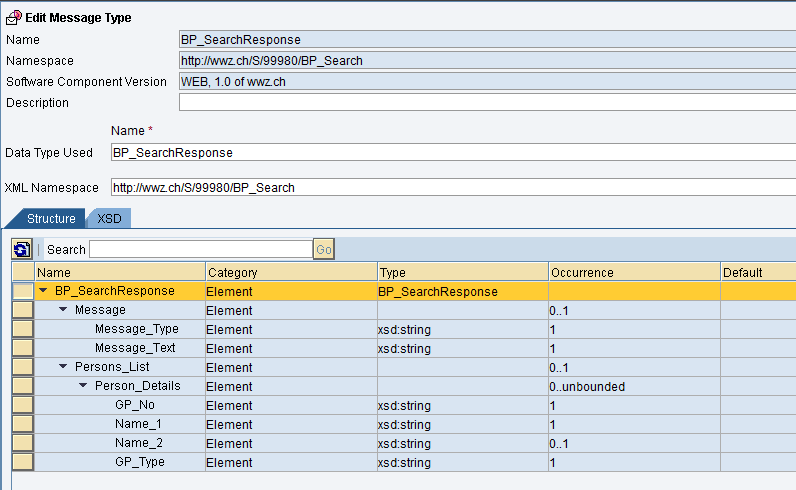


Abbildung 5: Der fertige Message Type mit der automatisch erstellen Struktur

### Service Interface erstellen

Dem Service Interface muss jeweils eine Kategorie zugewiesen werden. Der Service Interface in der ERP Komponente bekommt die Kategorie «Inbound». Damit dieser Service Interface aufgerufen werden kann, benötig er einen passenden Service Interface mit der Kategorie «Outbound». Dieser wird in der Web-Komponente erstellt.

Zusätzlich muss dem Service Interface jeweils die erstellten Message Types für die Request beziehungsweise für die Responsenachricht übergeben werden.

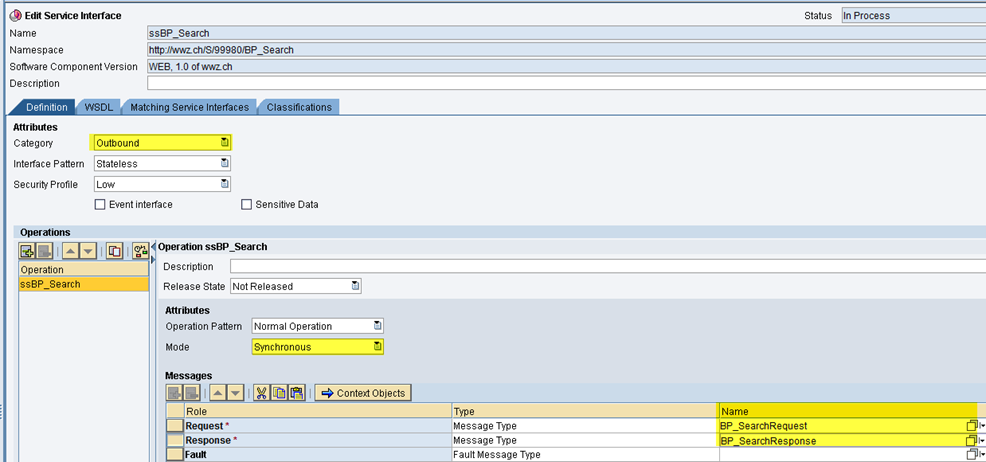


Abbildung 6:Das erstellte Service Interface mit der Outbound Kategorie und den zugewiesenen Message Types

### Message Mapping

Die erstellten Message Types müssen in der PI Komponente miteinander verbunden werden.

Hierbei wird der Message Type für den Request aus der Web-Komponente mit dem Message Type für den Request aus der ERP-Komponente miteinander verbunden. Zusätzlich müssen jeweils zusammengehörenden Elemente der Message Types verbunden werden. Wurden die Elemente in beiden Data Types gleich benannt, kann dieser schritt automatisch durchgeführt werden. Ansonsten muss die Verbindung der Elemente von Hand durchgeführt werden.

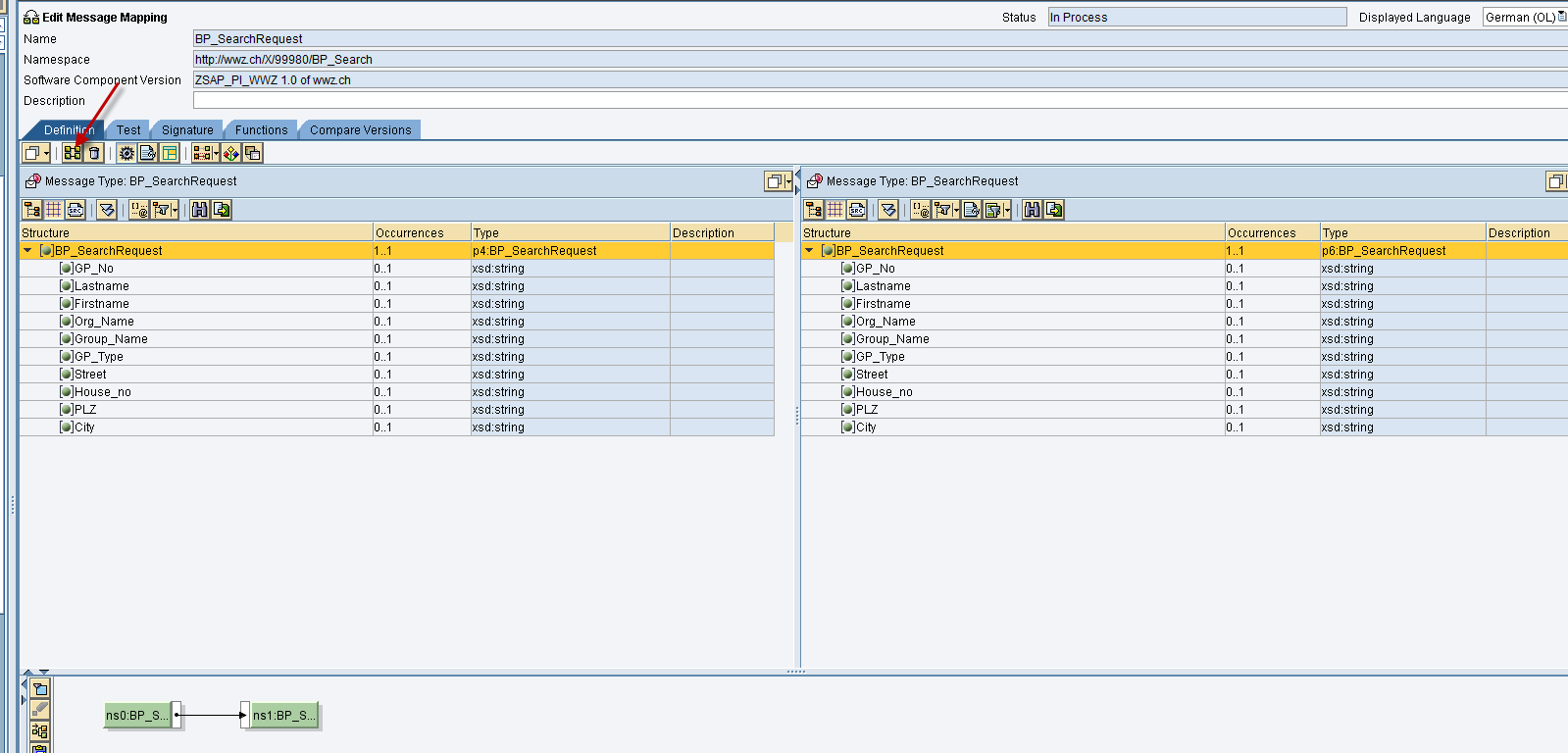


Abbildung 7: Mapping der BP\_Search Request Nachricht

### Operation Mapping

Über das Operation Mapping wird die Beziehung des Inbounds Service Interface mit dem Outbound Service Interface erstellt.

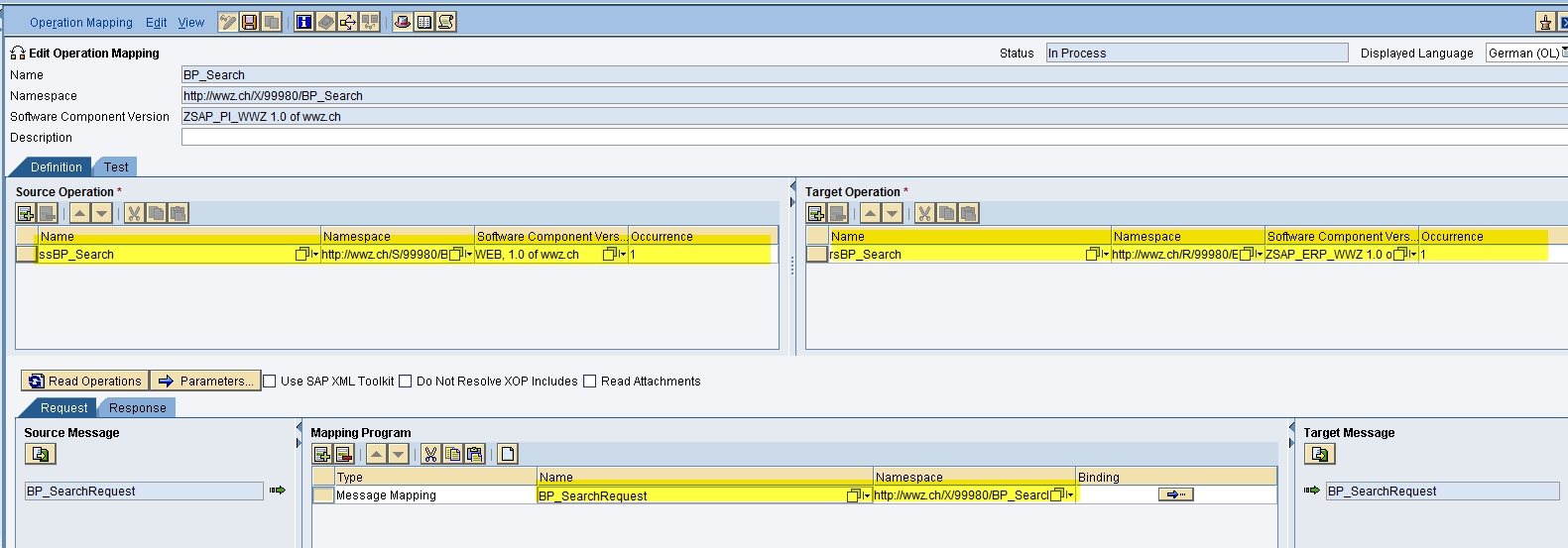


Abbildung 8: Das Operation Mapping für den Webservice BP\_Search

## Integration Builder

### Communication Channel erstellen

Über den Kommunikationskanal werden die Details für die Eingangs- oder Ausgangsverarbeitung einer Nachricht definiert. Es wird je ein Kommunikationskanal als Empfänger- und einen als Senderkanal erstellt. Die Rolle (Sender oder Empfänger) wird über die Kommunikationskomponente zugeordnet.

### Integrated Configuration erstellen

Über die Integrated Configuraton werden alle erstellte Komponente zusammengeführt.

Das Inbound Processing definiert, welcher Typ und Eingangskanal benutz wird. Der Reciever bestimmt, wer die Nachricht erhält. Über das Reciever Interface wird bestimmt welches Mapping verwendet wird. Und der Outbound Processing definiert welcher Typ und Ausgangskanal verwendet wird.

## SProxy

Wurden die Webservice im Enterprise Services Builder und im Integration Builder fertiggestellt, kann mit der Programmierung der Web Service im ERP System vorgefahren werden. Hierzu wir die Transaktion SPROXY verwendet.

Zusätzlich befindet sich im Anhang der ganze Code von allen Webservices.

### BP\_Search

Zu Beginn müssen die Inputparameter (Suchkriterien) einer lokalen Struktur übergeben werden. Diese wird dann beim Aufruf der Funktion BAPI\_BUPA\_SEARCH\_2 übergeben.

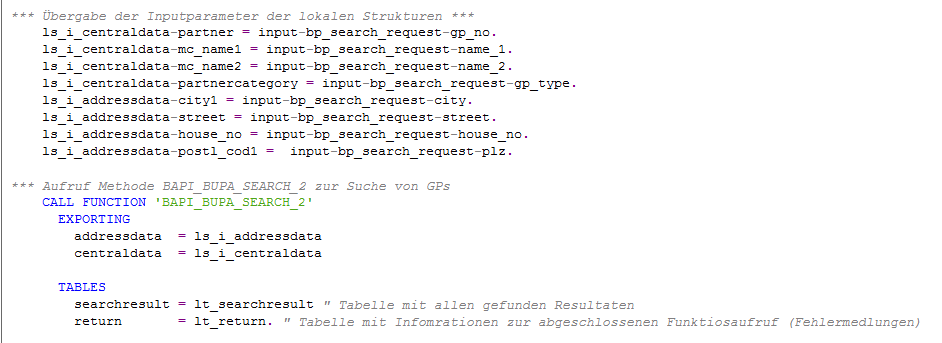


Abbildung 9: Übergabe der Inputparameter und Aufruf der Funktion BAPI\_BUPA\_SEARCH\_2

Wurde bei der Suche GPs gefunden, werden diese einen internen Tabelle übergeben. Für jeden Datensatz in der Tabelle wird die Funktion BAPI\_BUPA\_CENTRAL\_GETDETAIL aufgerufen.

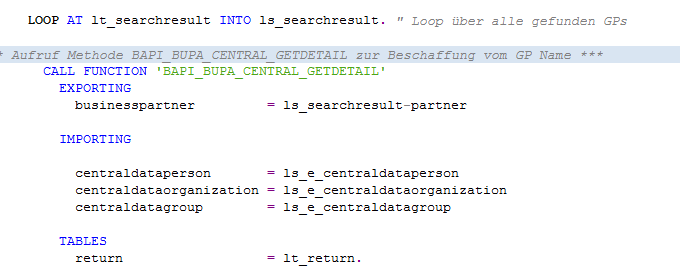


Abbildung 10: Loop über alle Datensätze und Aufruf der Funktion für jeden GP

Zusätzlich muss noch der GP Typ für jeden GP ermittelt werden, falls bei der Suche nach keinen bestimmten GP Typ mitgegeben wurde. Dazu wird eine Select Abfrage auf die Tabelle but000 durchgeführt.

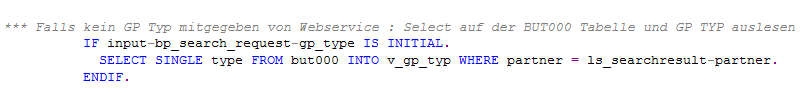


Abbildung 11: Select Abfrage für den GP Typ

### BP\_GetDetails

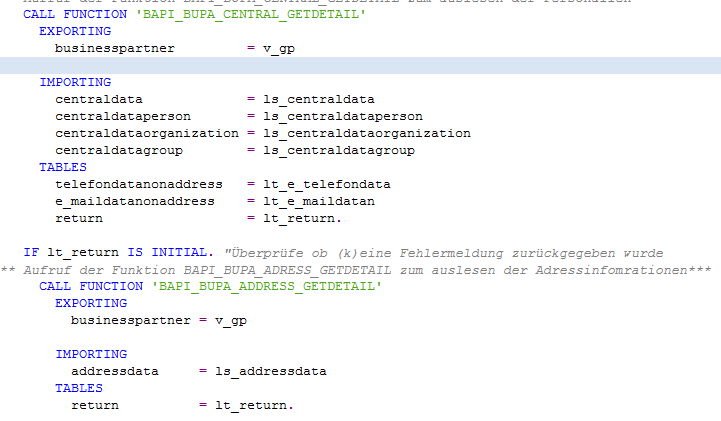
Ausgelesen werden die persönlichen GP Informationen mit dem Funktionsbaustein 'BAPI\_BUPA\_CENTRAL\_GETDETAIL'. Findet dieser Informationen, wird der Funktionsbaustein… aufgerufen zum Auslesen der Adressdaten Bei beiden Funktionsbauseinen muss die GP Nummer als eingabewert übergeben werden.

Abbildung 12: Aufruf der Funktionen 'BAPI\_BUPA\_CENTRAL\_GETDETAIL’ und 'BAPI\_BUPA\_CENTRAL\_GETDETAIL'

Die Anrede muss über den «Title\_key» herausgefunden werden. Dazu wird eine Case Abfrage verwendet. Die Namen werden ebenfalls über eine Case Abfrage anhand des GP Types aus verschiedenen Strukturen ausgelesen.

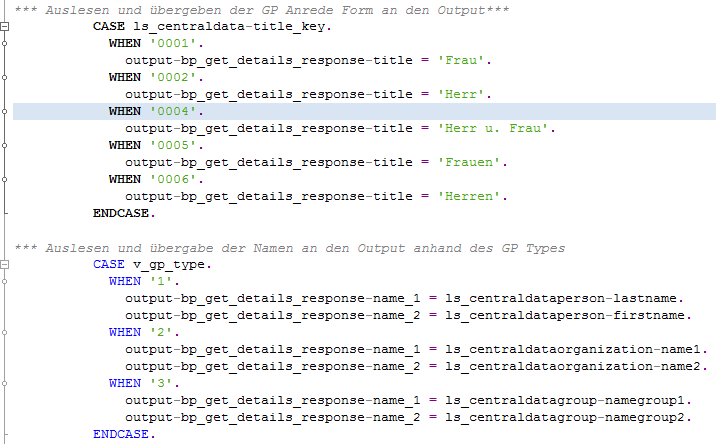


Abbildung 13: Case Abfrage für die Anrede und Namen

### BP\_QMC\_Products

Über eine Select Abfrage werden alle QMC IDs des gewünschten GPs aus der Tabelle zbc\_qmc\_crm\_data gelesen und in eine lokale Tabelle übertragen. Für alle Produkte besteht in der Tabelle ein Feld, welches mit einem X gefüllt ist, falls ein Produkt vorhanden ist und weiteren Feldern mit den Namen den jeweiligen Produkten.

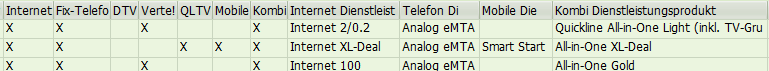


Abbildung 14: Ausschnitt Tabelle ZBC\_QMC\_CRM\_DATA



Abbildung 15: Select-Abfrage zum auslesen aller QMC IDs des ausgewählten GPs

War die Select Abfrage erfolgreich wird eine Loop über die lokale Tabelle ausgeführt und überprüft, welche Produkte vorhanden sind. Alle Informationen werden einer weiteren lokalen Tabelle übergeben. Wurde dies für alle Produkte durchgeführt, werden die Daten dem Output übergeben.



Abbildung 16: Auslesen der QMC Produkte

### BP\_Update

Für jeden möglichen änderbaren Wert muss abgefragt werden, ob ein neuer Wert übergeben wurde. Falls ein neuer Wert vorhanden, muss dieser in eine lokale Struktur übergeben werden. Zusätzlich musste in einer anderen, identischen Struktur bei allen Felder die geändert werdet sollen, ein X gesetzt werden.



Abbildung 17: Überprüfung und anschliessende Übergabe der Werte in die beiden identischen Strukturen

Bei Privatpersonen musste überprüft werden, ob eine neue, gültige Anrede gesetzt wurde. Dies wurde über eine If Abfrage des GP Typs gemacht und anschliessend einer Case Abfrage des neuen Wertes für die Anrede.

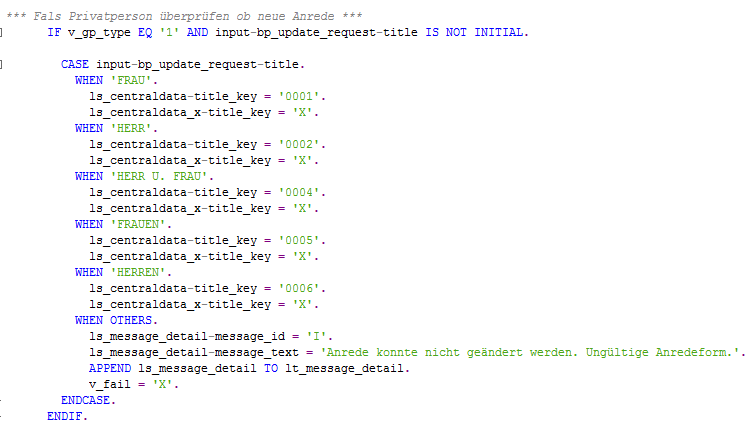


Abbildung 18: Case Abfrage für die Überprüfung der Anrede

Da mehrere Telefonnummern und E-Mail-Adressen auf einmal geändert werden konnte, musste über die Inputtabelle geloopt werden. Für jede geänderte Nummer mussten die Informationen einer lokalen Tabelle übergeben werden und ebenfalls ein «X» in der identischen Tabelle gesetzte werden. Die identische Tabelle mit den X-Werten muss gleich viele Datensätzen beinhalten wie jene mit den neuen Werten. Das selbe musste für die Änderungen an E-Mail-Adressen vorgenommen werden.

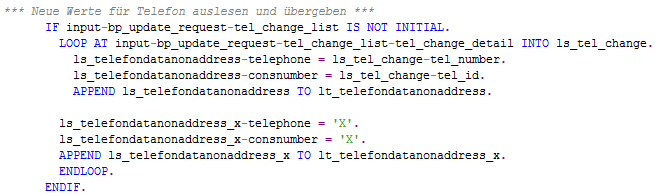


Abbildung 19: Loop für das Auslesen aller neuen Werte für Telefonnummern

# Planung Webanwendung

## Überlegungen

Das Projekt soll eine Single-Page-Webseite werden, da nicht viele Informationen angezeigt werden müssen. Ver verlinken mehrere HTML-Seiten würde die Webanwendung zu sehr verwinkelt und unhandlich machen, was nicht sein sollte.

Wird bei der Suche mehrere GPs gefunden, sollen diese in einer Liste angezeigt werden. Über die Liste kann der Anwender sich ein GP aussuchen. Wird jedoch nur ein GP gefunden, soll direkt der Webservice zum Auslesen der GP Informationen gestartet werden.

Ein GP kann mehrere QMC IDs besitzen. Daher sollte es möglich sein, dass der Anwender zwischen den QMC IDs hin und her wechseln kann und imsmer die Informationen zu einer QMC ID angezeigt werden.

der Änderungsmodus soll klar erkennbar sein. Zudem sollte der Änderungsmodus so umgesetzt werden, dass der Anwender diesen verlassen muss, auch wenn keine Änderungen vorgenommen wurde. Dies soll verhindern, dass ein Anwender im Änderungsmodus bleibt und unbeabsichtigt (weitere) Änderungen vornimmt.

## Bibliotheken und Framework

Folgende Bibliotheken/Framewoks werden für die Programmierung der Webseite verwendet:

* Bootstrap
* Jquery.soap

### Bootstrap

Das Bootstrap Framework wird in diesem Projekt wegen er einfachen Anwendbarkeit eingesetzte. Es bietet viel HTML Funktionen, welche ohne grossen Aufwand implementiert werden können. Dank dem eigenen CSS erscheinen die Bootstrap Elemente ohne weiteres Zutun in einem modernen Style.

Über das Grid-System von Bootstrap kann zusätzlich einfach ein Responsive Design erstellt werden. Dies ist für dieses Projekt nicht relevant, könnte aber bei einer Weiterentwicklung der Webseite hilfreich sein.

### jquery.soap

jquery.soap Bibliothek wird in diesem Projekt für die Kommunikation der Webservices verwendet. Es ermöglicht das eingeben des Request in XML DOM, XML string oder JSON und kann die Response Nachricht ebenfalls in einer von diesen Formen zurückgeben.

### Entscheidungen

Die Entscheidung der beiden obengenannten Bibliotheken / Frameworks wurde auf Grund des Umfangs der vorhandenen Dokumentationen, verglichen mit vergleichbaren Bibliotheken.

## GUI Entwurf

Um bei der Erstellung der Webseite einen Anhaltspunkt zu haben, wie diese am Aussehen soll, wird ein Entwurf für das GUI erstellt. Das GUI soll einfach und selbsterklärend sein, so dass selbst unerfahrene Benutzer die Anwendung leicht bedienen können.

### GUI Hauptseite

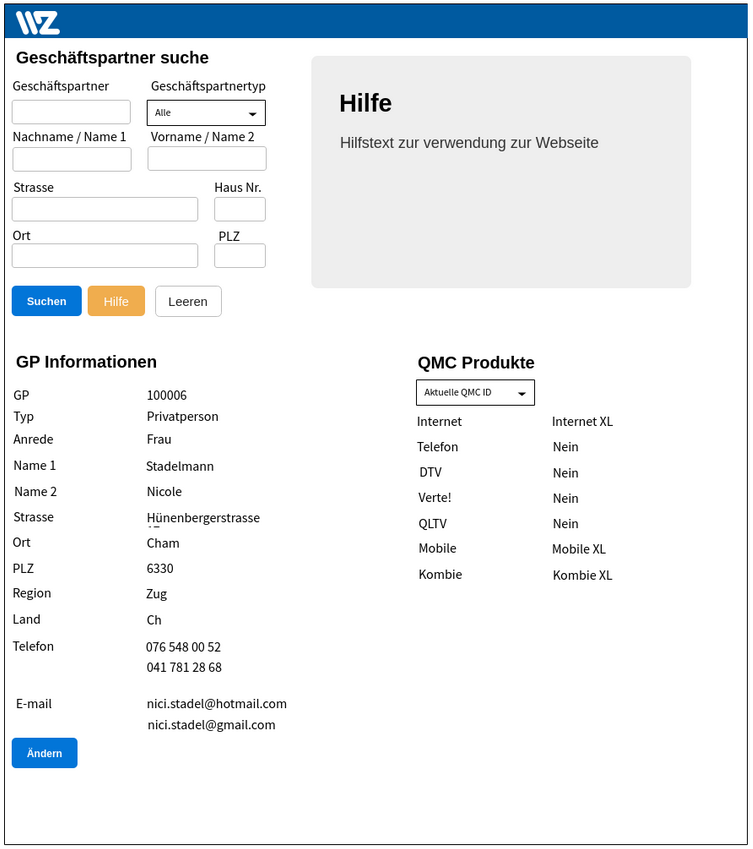


Abbildung 20: GUI für die Webseite

Überlegungen:

Über den Such-Button soll die GP Suche Ausgeführt werden. Werden mehrere Ergebnisse gefunden soll ein Fenster mit allen Gefunden GPs angezeigt werden, aus diesen kann ein GP ausgesucht werden. Sobald ein GP ausgesucht wurde, sollen die Abschnitte für die GP Informationen und die QMC Produkte angezeigt werden. Da ein GP mehrere QMC IDs besitzen kann, soll über eine Selection Box zwischen den QMC IDs auswählen. Bei den GP Informationen befindet sich ein änderungs-button. Wird in einem weiteren Fenster der Änderungsmodus aktiv. Der Hilfstext soll nur Angezeigt werden, wenn auf den Hilfe-Button geklickt wird.

### GUI für die GP Auswahl

Wen bei der Suche mehrere Treffer gefunden wurden, soll ein Fenster geöffnet werden, welches alle Treffer (maximal 50) anzeigt. Das Fenster wird über die Modal Funktion von Bootstrap umgesetzt. Die Gefunden GPs werden in einer Tabelle, angezeigt.

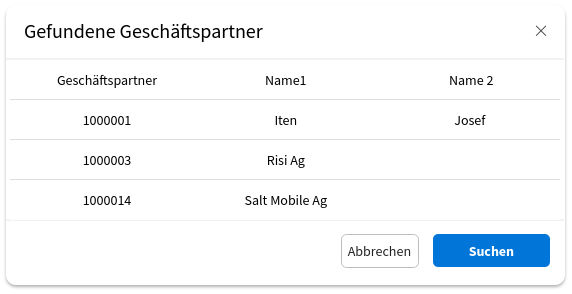


Abbildung 21: Die Auswahl eines GPs über die Modal Funktion von Bootstrap

### GUI für den Änderungsmodus

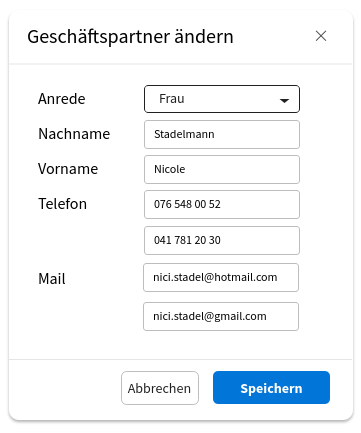
Der Änderungsmodus soll in einem Fenster, mit Hilfe der Bootstrap Modal Funktion, geöffnet werden.

Abbildung 22: Der Änderungsmodus

## Workflows

### GP suche und Informationen ausgeben



### GP ändern



# Entwicklung Webanwendung

Nach der Planung der Webanwendung wird diese anhand der erstellten GUI Entwürfe und Flussdiagrammen erstellt.

## Formular für die Geschäftspartnersuche

Als erstes wird das Formular erstellt, welches die Eingabe der Suchkriterien ermöglicht. Beim Feld für die Eingabe des Geschäftspartners definierte ich mit dem Attribut «maxlenght» eine maximallänge. Damit ist es den Benutzer nicht möglich, einen Wert einzugeben, welcher länger ist als ein Geschäftspartner im SAP ERP.

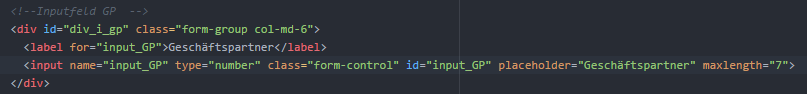


Abbildung 23: Inputfeld für den GP kann nicht länger als 7 sein

Für den GP Typ erstellte ich eine Selektionsbox. Wen ein Geschäftspartnertyp ausgewählt wird, wird der Value Wert des ausgewählten Option Element beim Aufruf des Webservice BP\_Search übergeben. Der Value Wert entspricht jeweils dem Wert, welcher der GP Typ im SAP ERP hat. Wird kein GP Typ ausgewählt, wird ein leerer Wert dem Webservice übergeben.

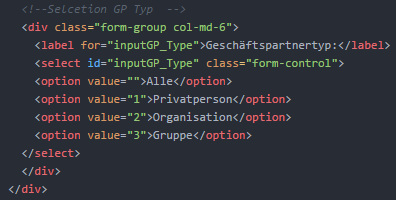


Abbildung 24: Selektionsbox für die Auswahl des GP Typs

Die restlichen Inputfelder wurden normal erstellt. Am Ende des Formulars befinden sich drei Buttons.

Suchen-Button: Der Suchbutten ruft die Funktion checkSearchForm() auf, welche die Eingaben überprüft.

Hilfe-Button: Dieser Button macht ein Hilfetext neben dem Formular sichtbar.

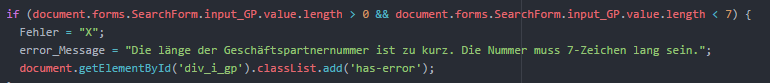
Felder leeren-Button: Dieser Button leert alle Felder des Formulars

## Funktion checkSearchForm()

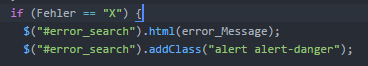
Die Funktion überprüft die eingegeben Werte des Formulars auf folgende Punkte:

* Mindestens ein Feld muss mit einem Wert gefüllt worden sein
* Falls der GP angegeben wurde: der Wert ist genau 7 Zeichen lang

Ist das Wert im GP Feld nicht lang genug, wird dieses nach der Validierung rot und eine Fehlermeldung erscheint. Wurde gar kein Feld ausgefüllt, werden alle Felder rot und eine entsprechende Fehlermeldung erscheint. Um die Felder rot hervorzuheben, wird die Klasse «has-error» von Bootstrap über den Aufruf document.getElementById den Elementen hinzugefügt.



Die Fehlermeldung wird einem, im HTML Gerüst vorbereitetem, leeren div hinzugefügt. Zusätzlich wird dem Div die Klassen «alert» und «alert-danger» übergeben, damit die Fehlermeldung rot ausgegeben wird.



War die Validierung erfolgreich wird die Funktion ssBPSearch() aufgerufen.

## Funktion ssBPSearch()

Am Anfang dieser Funktion wird der Webservice BP\_Search ausgeführt. Dazu wird die Bibliothek jquery.soap verwendet. Folgende Information müssen beim Aufruf des Webservice übergeben werden:

Folgende Optionen müssen zum Senden des Request abgegeben werden:

* URL
* Data

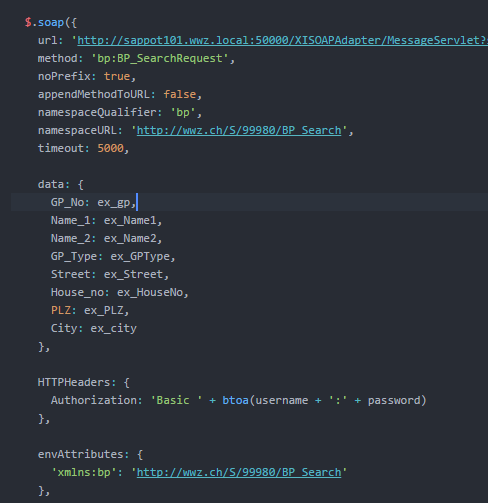
URL:

Die URL definiert den Endpunkt der Webservices.

Data:

Bei dieser Angabe handelt es sich um die Elementwerte welcher dem Request mitgegeben werden. Diese können über JSON definiert werden. Hierzu werden zuerst die Werte der Imputfelder an die dafür vorgesehene Variablen übergeben. Im Data werden die Namen der Elemente, so wie sie im WSDL File des Webservice geschrieben stehen, angegeben. Jedem dieser Elementnamen kann nach einem : die entsprechende Variabel zugewiesen werden.

Der gesammte aufruf des webservices sieht so au:



Kommt es während dem Webservice zu einem Fehler, wird automatisch in dem entsprechenden Error Funktion gesprungen, welche selbst ausprogrammiert werden kann. In diesem Fall wird eine Fehlermeldung ausgegeben.

Ist der Aufruf des Webservice erfolgreich, springt das Programm automatisch in die success Funktion er Bibliothek, welche ebenfalls selbst ausprogrammiert werden kann.

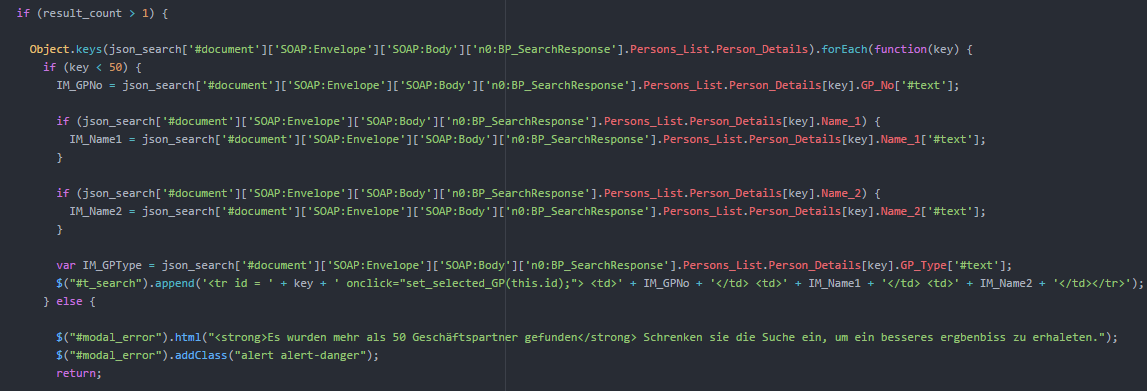
Als erstes wird die soapResponse Nachricht in JSON umgewandelt, welches über einen Befehl der jquery.soap Bibliothek geht.

Durch die erhalten werte wird überprüft, wie viele GPs gefunden wurde.

**Kein Treffer:** wurde kein Treffer gefunden, wird eine entsprechende Fehlermeldung ausgegeben**.**

**1 Treffer:** Wurde genau ein Treffer gefunden wird sofort die Funktion getDetails() ausgeführt.

**Mehrere Treffer:** Wurden mehrere Treffer gefunden, eine forEach schleife ausgeführt, GP für GP ausliest, und in eine HTML Tabelle übertragen wird. Sind mehr als 50 Einträge vorhanden, wird nach dem 50isten GP wird die schleife abgebrochen. Die Tabelle mit den GPs wird in einem Modal geöffnet. Wurden mehr als 50 GPs gefunden, wird in dem Modal zusätzlich eine Information ausgegeben, dass nur die 50 GPs angezeigt werden.



Nachdem ein GP aus der Tabelle ausgesucht wurde, kann über den Such-Button im Modal die Funktion getDetails() aufgerufen werden.

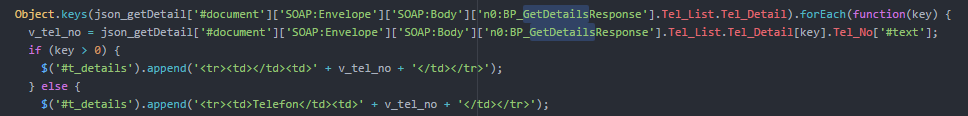
## Funktion getDetails()

Diese Funktion ruft den Webservice BP\_GetDetails auf. Übergeben wird die GP Nummer des ausgewählten GPs.

Der Aufruf des Webservice funktioniert gleich wie in der Funktion ssBPSearch() und wird hier nicht nochmals näher erklärt. Die Error Funktion wurde auch hier so ausprogrammiert, dass eine Fehlermeldung ausgegeben wird.

War der Aufruf des Webservers erfolgreich, werden alle erhaltene Information, wo vorhanden, ausgelesen. Überall, wo ein wert zurückgekommen ist, wird dieser und den Namen für den in eine Tabelle geschrieben. Die Tabelle dient dazu, dass alle erhaltenen Werte und deren Beschriftung einheitlich unter einander angezeigt werden.

Die Telefonnummern und E-Mail-Adressen werden jeweils über eine forEach Schleife ausgelesen, da ein GP von diesen mehrere besitzen kann.



Am Ende der sucsess Funktion wird die Funktion getQMCProducts() aufgerufen.

## getQMCProducts()

Diese Funktion ruft den Webservice BP\_QMCProducts auf. Übergeben wird die GP Nummer des ausgewählten GPs.

Der Aufruf des Webservice funktioniert gleich wie in der Funktion ssBPSearch() und wird hier nicht nochmals näher erklärt. Die Error Funktion wurde auch hier so ausprogrammiert, dass eine Fehlermeldung ausgegeben wird.

War der Aufruf des Webservice erfolgreich wird zuerst die Anzahl der gefunden QMC IDs ausgelesen.

**Kein Treffer:** Wurde keine QMC ID gefunden, wird eine entsprechende Nachricht ausgegeben**.**

**1 Treffer:** Wurde genau ein Treffer gefunden werden alle Informationen der QMC ausgeben. Auch hier werden die Informationen in einer HTML Tabelle ausgegeben.

**Mehrere Treffer:** Wurden mehrere QMC IDs gefunden, werden alle IDs ausgelesen und dynamisch einem Dropdown hinzugefügt. Wird eine neu QMC ID im Dropdown ausgewählt, wird der onchange Event ausgelöst, welcher die Funktion changeQMCData() ausführt.

## changeQMCData()

Diese Funktion wird jedes Mal ausgeführt, wen im QMC Dropdown ein QMC ID ausgewählt wurde.

Die Funktion sucht in der Response Nachricht von Webserver BP\_getQMCProducts nach der entsprechenden QMC ID und liest die Daten aus. Die Daten werden in der entsprechenden HTML Tabelle ausgegeben. Wird wieder eine andere QMC ID ausgesucht, werden die Daten in der Tabelle mit jenen der neu ausgewählten QMC ID überschrieben.

## openChangeModus()

Diese Funktion öffnet den Änderungsmodus. Dafür wird zuerst überprüft, welche daten zu diesem GP vorhanden sind. Anrede, Name 1, Name 2 sowie alle Telefonnummern und E-Mailadressen werden, falls vorhanden, aus der noch bestehende Response Nachrocht vom Webservice BP\_getdetails ausgelesen. Für alle ausgelesenen Elemente wird ein Inputfeld erstellt. Alle erstellten Inputfelder werden einem Modal hinzugefügt.

Der Anwender kann die neuen Werte in dem Modal eingeben. Über den Speicher Button wird die Funktion checkUpdateForm() aufgerufen.

## checkUpdateForm()

Diese Funktion überprüft die eingaben in der Telefon- und E-Mail Feldern. Die E-Mail-Adressen werden drauf überprüft, ob ein @ Zeichen ein Punkt vorhanden sind. Dafür wird eine weitere Funktion verwendet, welche ich aus dem Internet übernommen haben. Der Code ist entsprechen gekennzeichnet. Bei den Telefonnummern wird überprüft, ob nur nummerische werte enthalten sind.

Ist bei der Validierung alles in Ordnung, wird die Funktion ssBP\_Update() aufgerufen. Ansonsten werden entsprechende Fehlermeldungen ausgegeben.

## ssBP\_Update()

Diese Funktion den Webservice BP\_Update auf. Übergeben werden alle Werte, welche im Änderungsmodus eingegeben wurden.

Bei diesem Webservice können mehrere Telefonnummern und E-Mail-Adressen übergeben werden können, kann bei der Data angaben diese nicht so einfach übergeben werden. Aufgrund dessen wird bei diesem Webservice Aufruf eine XML dem DATA übergeben werde.

Die XML Struktur wird dynamisch erstellt indem die einzelnen xml Zeilen über den Befehl xml.pusch() hinzugefügt werden.

Für die Telefonnummern und E-Mail-Adressen wird jeweils eine forEach schleife durchlaufen und die neuen werte für jede Telefonnummer und E-Mail-Adresse aus den Inputfelder ausgelesen. Am ende jedes Durchlaufs wird der xml.push() Befehl durchgeführt und die Zeilen mit dem neuen Werten der XML übergeben.

# Testen

# Endergebnis

# Schlussbericht

## Ist Soll

## Persönliche Reflexion

Mit dem Endprodukt bin ich persönlich zu frieden. Jedoch wurde es gegen Ende sehr stressig. Bis und mit dem Testen der Webservices war ich gut im Zeitplan. Erst als es an das Entwickeln der Webseite selbst ging verlor ich viel Zeit. Ich denke es lag an den Bibliotheken welche ich verwenden musste, so wie an meinen bescheidenen JS Kenntnissen. Gerade bei der Bibliothek für die Kommunikation mit dem Webservice, welche ich das erste Mal verwendete. Daher musste ich natürlich zuerst herausfinden, wie man diese benutzte. Auch wenn es Dokumentationen dazu gab, konnte ich dies in der Praxis nicht so schnell umsetzen. Auch das Auslesen der, in JSON umgewandelte, Response Nachrichten war mir zu Beginn nicht klar, was wieder ein gewisser Zeitverlust bedeutete.

Zu allem hatte ich plötzlich einen Fehler bei der Kommunikation, welcher ich nicht herausfinden konnte. Das SAP PI sagte, dass die Request Nachricht erhalten und eine Response Nachricht gesendet wurde. Die Bibliothek jedoch gab mir die Meldung zurück, dass keine Response Nachricht erhalten wurde. Erst nach 3 Stunden war der Fehler wieder verschwunden. Aber die Ursache des Fehlers ist mir noch immer nicht bewusst.

Alles in Allem hätte ich für die Programmierung aber auch für die Dokumentation mehr Zeit einplanen müssen. Danke den 15 Stunden Puffer, welche ich über das gesamte Projekt verteilt hatte, ging es zum Glück trotzdem knapp auf. Jedoch entstand die Technische Dokumentation während drei Stunden Arbeit zuhause.

Ich würde sagen, dass 80 Stunden für das Projekt fast zu wenig war. Auch wenn es zeitlich mehr oder weniger aufging, war es sehr stressig und für ein paar Kleinigkeiten blieb keine Zeit mehr um diese anzupassen oder zu optimieren.

# Glossar

# Quellenverzeichnis

**Im aktuellen Dokument sind keine Quellen vorhanden.**

# Abbildungsverzeichnis

[Abbildung 1: Projektorganisation 13](#_Toc511652381)

[Abbildung 2: Das IPERKA Modell 14](file:///C:\Users\stadni\Desktop\IPA\IPA%20Bericht.docx#_Toc511652382)

[Abbildung 3: Erstellter Datentyp für BP\_Search Response 28](#_Toc511652383)

[Abbildung 4: Auswählen des Data Types für den Request Message Type von BP\_Search 29](#_Toc511652384)

[Abbildung 5: Der fertige Message Type mit der automatisch erstellen Struktur 29](#_Toc511652385)

[Abbildung 6:Das erstellte Service Interface mit der Outbound Kategorie und den zugewiesenen Message Types 30](#_Toc511652386)

[Abbildung 7: Mapping der BP\_Search Request Nachricht 30](#_Toc511652387)

[Abbildung 8: Das Operation Mapping für den Webservice BP\_Search 31](#_Toc511652388)

[Abbildung 9: Übergabe der Inputparameter und Aufruf der Funktion BAPI\_BUPA\_SEARCH\_2 32](#_Toc511652389)

[Abbildung 10: Loop über alle Datensätze und Aufruf der Funktion für jeden GP 32](#_Toc511652390)

[Abbildung 11: Select Abfrage für den GP Typ 32](#_Toc511652391)

[Abbildung 12: Aufruf der Funktionen 'BAPI\_BUPA\_CENTRAL\_GETDETAIL’ und 'BAPI\_BUPA\_CENTRAL\_GETDETAIL' 33](#_Toc511652392)

[Abbildung 13: Case Abfrage für die Anrede und Namen 33](file:///C:\Users\stadni\Desktop\IPA\IPA%20Bericht.docx#_Toc511652393)

[Abbildung 14: Ausschnitt Tabelle ZBC\_QMC\_CRM\_DATA 34](#_Toc511652394)

[Abbildung 15: Select-Abfrage zum auslesen aller QMC IDs des ausgewählten GPs 34](#_Toc511652395)

[Abbildung 16: Auslesen der QMC Produkte 34](#_Toc511652396)

[Abbildung 17: GUI für die Webseite 37](#_Toc511652397)

[Abbildung 18: Die Auswahl eines GPs über die Modal Funktion von Bootstrap 38](#_Toc511652398)

[Abbildung 19: Der Änderungsmodus 38](file:///C:\Users\stadni\Desktop\IPA\IPA%20Bericht.docx#_Toc511652399)

# Tabellenverzeichnis

[Tabelle 1: Entwurf BP\_Search Request 23](#_Toc511627681)

[Tabelle 2: Entwurf BP\_Search Response 23](#_Toc511627682)

[Tabelle 3: Entwurf BP\_Details Request 24](#_Toc511627683)

[Tabelle 4: Entwurf BP\_Details Response 24](#_Toc511627684)

[Tabelle 5: Entwurf BP\_QMC\_Products Request 25](#_Toc511627685)

[Tabelle 6: Entwurf BP\_QMC\_Products Response 25](#_Toc511627686)

[Tabelle 7: Entwurf BP\_Update Request 26](#_Toc511627687)

[Tabelle 8: Entwurf BP\_Update Response 26](#_Toc511627688)



Anhang

Anhang