Outlook/Exchange Adapter für CGM G3 Clinical Information System MRP

Krainer Philipp

BACHELORARBEIT

Nr. 1310458007-B

eingereicht am
Fachhochschul-Bachelorstudiengang

Medizin- und Bioinformatik

in Hagenberg

im Dezember 2016

Diese Arbeit entstand im Rahmen des Gegenstands

IGS

im

Sommersemester 2016

Betreuer:

Oliver Krauss MSc

Erklärung

Ich erkläre eidesstattlich, dass ich die vorliegende Arbeit selbstständig und ohne fremde Hilfe verfasst, andere als die angegebenen Quellen nicht benutzt und die den benutzten Quellen entnommenen Stellen als solche gekennzeichnet habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Hagenberg, am 1. Dezember 2016

Krainer Philipp

Inhaltsverzeichnis

Kurzfassung

Diese Arbeit setzt sich mit der Implementierung eines Moduls für MRP mit der EWS API auseinander. Das MRP-System generiert mit Hilfe von Ressourcen und Regeln Termine, welche in das Microsoft Outlook übertragen werden. Ziel ist es die Verbindung von MRP und Microsoft Outlook herzustellen. Das Modul ist in JAVA programmiert. Mit Hilfe der EWS API wird die Verbindung zum Server geschaffen. Bei der Implementierung wird geachtet, dass die Richtlinien von der Zielsetzung eingehalten werden. Es werden drei Konzepte vorgestellt (Manuelle Interaktion mit der Microsoft Outlookoberfläche, Ansteuerung von Microsoft Outlook mit Hilfe von EWS, Ansteuerung mittels MRP). In der Implementierung werden die Funktionalitäten sowie Methoden beschrieben, die verwendet worden sind. Weiters werden in der Implementierung die EWS Funktionen angewendet und an das MRP angepasst. Die einzelnen Funktionen wurden mittels der Methode des benutzerinteraktiven Testens überprüft.

Abstract

This thesis deals with the implementation of a module for the MRP System using the EWS API. The MRP System generates Appointments using resources and rules. These Appointments get exported to Microsoft Outlook. The Connection of MRP and Microsoft Outlook presents the goal of this thesis. The programming language JAVA is used to implement the functions and methods of this software. The EWS API handles the requests from the MRP System and communicates with the server. The implementation follows the rules of the objective. There are three concepts for using Microsoft Outlook, MRP and EWS. The implementation shows used methods and functions for the implementation with EWS. The implementation contains the EWS related functions as well as the methods, which are adapted for the MRP System. The functions as described in the implementation are tested with the method called userinteractive testing.

Kapitel 1

Einleitung

Diese Arbeit ist der praktische Teil der Bachelorarbeit. Diese wurde im Rahmen des Semesterpraktikums vom 01.03 bis 30.06.2016 verfasst. Das Praktikum im Umfang von 16 Wochen wurde bei der Firma CGM Clinical Austria am Standort Linz absolviert. CompuGroup Medical (CGM) ist ein österreichischer Softwarehersteller, der umfassende IT-Lösungen zur Optimierung des Gesundheitswesens produziert. Software-Lösungen, welche die Prozesse von niedergelassenen Ärzten und deren Personal sowie von medizinischem, pflegerischem und administrativem Krankenhauspersonal unterstützen. Die CGM beschäftigt in Österreich ca. 250 Mitarbeiter.

1.1 Zielsetzung

Das Modul Multidimensionale Ressourcen Planning (MRP) ermittelt Termine für Operationen in Krankenhäusern unter der Berücksichtigung der beteiligten Ressourcen (Personen, Geräte, Material) und vor und nachgelagerten Untersuchungen inklusive Betten und Zimmern. Ziel dieses Projektes ist es für die ressourcenbezogenen Daten (z.B. Dienste von Ärzten , Operationstermine etc.) eine Möglichkeit zu schaffen, Termine in den persönlichen Kalendern der beteiligten Personen (Teams) anzuzeigen. Damit sollen die Anwender in ihrer gewohnten Umgebung die Ergebnisse des komplexen Planungsprozesses des MRP-Systems vermittelt bekommen.
Es wird davon ausgegangen, dass Termine im Microsoft Outlook eingetragen werden. Dies soll automatisch nach der Generierung von vorgegebenen Terminvorschlägen geschehen. Diese Terminvorschläge werden als Teil des MRP-Systems generiert. Die Aufgabe war es mit Hilfe der Java Exchange Web Services (EWS) Application Programming Interface (API) die Daten an das Microsoft Outlook zu senden bzw. an den Server zu übermitteln. Die EWS API erleichtert das Arbeiten mit Microsoft Outlook.

Bei flexibler API, kann sowohl Java, C# als auch Extensible Markup Lan-

guage (XML) verwendet werden. Die Termine werden dabei in Microsoft Outlook-Kalendern dargestellt. Änderungen von Terminen im Microsoft Outlook haben keine Auswirkung auf das MRP System. Weiters können sogenannte „Appointments“ mit Hilfe von vordefinierten Filtern gesucht und exportiert werden. Dies ermöglicht eine einfachere Handhabung von Abfragen. Die Appointments werden dann in Outlook-Termine umgewandelt und können so exportiert werden.

Kapitel 2

State of the Art

2.1 EWS API

2.1.1 Beschreibung

Die EWS API ist eine Schnittstelle, die von Microsoft bereitgestellt wird, welche die Kommunikation mit einen Microsoft Exchange-Server zulässt. Die API kann mit Java, C# und XML angesteuert werden und beinhaltet einen umfangreichen Funkionskatalog um die verschiedenen Funktionen in Microsoft Outlook anzusteuern. Diese Funktionen dienen zur Verwaltung von Emails, Kontakten, Aufgaben, Terminen, Kalendern sowie Rechten. Die API wurde entwickelt um programmgesteuerte Abfragen und Aufgaben mit Hilfe eines Microsoft Outlookservers zu realisieren. Sie unterstützt synchrone sowie asynchrone Abfragen und Anfragen an den Server, die zur Abfrage von Verfügbarkeiten und Benutzerrechten sowie von Microsoft Outlookelementen und Ordnern dienen. Weiters kann mit Hilfe der API gefiltert, gesucht werden, weiters lässt die EWS API einen Mehrbenutzerbetrieb zu.[5]

2.1.2 Funktionen

HyperText Transfer Protocol und HyperText Transfer Protocol Secure

Das Internetprotokoll HyperText Transfer Protocol (HTTP) beschreibt die zustandslose Übertragung von Internetdaten. Die Übertragung läuft über die Webmethoden GET, PUT und POST. HTTP verwendet die Standardports, welche in der Transmission Control Protocol (TCP)-Richtlinie definiert worden sind. HyperText Transfer Protocol Secure (HTTPS) stellt die sichere Verbindung von HTTP dar, wobei die Verschlüsselung über Secure Socket Layer (SSL) erfolgt. Die Webseiten werden mittels HyperText Markup Language (HTML), JavaScript und (Personal Home Page) PHP angesteuert, wobei die Daten mit XML und JavaScript Object Notation (JSON)

übertragen werden.[5]

Exchange Service

Der Exchange Service ist der zentrale Service, welcher Aufgaben verarbeitet und in Verbindung mit dem HTTP(S) Protokoll den Exchange-Server ansteuert. Dieser Service stellt die Verbindung mit dem Server mittels Benutzername, Passwort, Serveradresse her. Außerdem sind zusätzliche Inforationen zur Serverversion und Übertragungsart gespeichert. Um die Kommunikation mit dem Server zu ermöglichen, hat die EWS Funktion freigeschaltet zu sein.[5]

Items und Folder

Die Funktionen der EWS API funktionieren nach dem Prinzip der Eindeutigkeit. Dies bedeutet, dass jeder Ordner, jedes Mail und jeder Termin seine eigene ID besitzt, welche vom Server festgelegt wird. Dabei gibt es keine Einschränkung, da jedes Element über die gleiche Priorität verfügt. Weiters kann mit Hilfe dieser eindeutigen ID jedes gesonderte Element abgerufen und beliebig verändert werden. Die Ordner über verfügen eine eindeutige ID mit der auf den ganzen Ordner zugegriffen werden kann. Auch besitzen spezielle Ordner wie z.B. der Posteingang sogenannte *WellKnownFolderNames*, welche wie Aliase bzw. Pseudonyme für IDs fungieren.[5]

Email

Emails können mit Hilfe der API gesendet und empfangen werden. Dabei besitzt jedes Email einen zugeordneten Ordner. Die Emails werden automatisch versendet sobald sie am Server eingetroffen sind. Auch können Emails von beliebigen Konten abgefragt und verändert werden. Vordefinierte Kategorien ermöglichen eine organisierte Verwaltung. Es können Dateien aus dem lokalen Dateisystem an Emails angehängt werden. Wichtige Emails können per Filter gesucht und gefunden werden sowie Posteingangsregeln erstellt und abgerufen werden. Emails können bei Bedarf in einen Unterordner verschoben werden. [6]

Termine / Appointments

Termine können erstellt, abgerufen und gelöscht werden. Sie repräsentieren eine Hauptfunktionalität im Outlook. Jeder Termin beinhaltet Datum, Zeit, Ort sowie Teilnehmer. Die Termine werden in den persönlichen Kalendern abgelegt. Weiters können den Terminen optionale Teilnehmer hinzugefügt werden sowie periodisch wiederholende Termine werden als Serientermine hinterlegt. Neben dem Hauptkalender können zusätzliche Kalender erstellt werden. Eine weitere Funktionalität stellt die Abfrage von Verfügbarkeiten

dar, diese liefern in weiterer Folge Terminvorschläge. Dabei wird auch die terminliche sowie räumliche Verfügbarkeit mitgeliefert. [6]

Tasks

Ein weitere Funktionalität der API stellen Tasks dar, welche erstellt und abgerufen werden können. Tasks enthalten eine Liste von Aufgaben, welche Personen zugeordnet werden und von diesen abgearbeitet werden. Einzelne Aufgaben können priorisiert und kategorisiert werden.[5]

Kontakte

Kontakte stellen neben den Terminen eine wichtige Rolle in der Microsoft Outlookumgebung dar. Die Kontakte werden im Adressbuch verwaltet und können angelegt, abgerufen sowie gelöscht werden. Kontakte können aus vordefinierten Dateien importiert und exportiert werden. Das Adressbuch enthält Kontakte sowie Kontaktgruppen. In den Kontakten sind Namen, Telefonnummern, Emailadressen und persönliche Daten hinterlegt. In dem Adressbuch kann gesucht und gefiltert werden.[6]

2.2 Multidimensionales Ressourcen Planning (MRP)

2.2.1 Beschreibung

Das MRP-System ist zur Planung von Krankenhausterminen entwickelt worden. Die Hauptaufgabe von MRP ist es aus gegebenen Ressourcen (Personen, Räume, Geräte, Betten, Materialien) in Verbindung mit Regeln Planungskalender für einzelne Räume zu erstellen. Diese Kalender repräsentieren Operationsplanungen, die eine zentrale Einheit in einem Krankenhausinformationssystem (KIS) darstellen. Ein KIS besteht in erster Linie aus den Zentralkomponenten Organisation, Bettenmanagement, Aufnahme, Entlassung sowie Operationsplanungen. MRP deckt nicht die zentralen Komponenten ab, da sich dieses System nur mit der Planung von Räumen und Terminen mithilfe von Ressourcen beschäftigt. Es übernimmt nicht die Verwaltung des Krankenhauses! [4]

2.2.2 Funktionsweise

Das MRP System generiert aus vorgegebenen Daten geeignete Termine und Pläne für Räume des Krankenhauses. Es werden die benötigten Ressourcen gesammelt und in einer zentralen Datenbank verwaltet. Aus den Datensätzen werden mittels Ressourcenmanager die Ressourcen zusammengetragen

und Appointments generiert. Diese werden dann mit Hilfe des Kapazitätsmanagementtool den verfügbaren Kapazitäten der einzelne Ressourcen zugeordnet und in einer Kalenderansicht dargestellt. [4]

2.2.3 Masterdata

Beschreibung

Jedes Feature im MRP-System folgt immer der selben Struktur. Jedes Feature besteht aus den Teilen API, Component, Datenbank und Service. Dabei wird die Drei-Schichten-Architektur realisiert. Dies bedeutet, dass es drei unterschiedliche Implementierungsweisen gibt, welche als eigenständig fungieren. Die unterste Schicht wird als Datenbank-Schicht bezeichnet und ist zuständig für die Verwaltung von persistenten Daten. Die Mittlere Schicht wird als Business-Logik bzw. Verarbeitungsschicht bezeichnet. Diese ruft die Daten von der untersten Schicht auf und leitet sie gegebenenfalls an die oberste Schicht weiter. Es besteht die Möglichkeit für Brechungen und Änderungen der Daten. Die oberste Schicht stellt die Präsentationsschicht dar und ist zuständig für die visuelle Darstellung der Daten. Die 3-Schichten-Architektur hat den Vorteil, dass die Aufgaben der Schichten klar von einander getrennt sind. Weiters gibt es die Component-Service-Implementation, welche aus den Komponenten, Datenbank, Service, und API bestet.[2]

Struktur

Jedes Feature folgt der Component-Service-Implementation, welche aus folgenden Teilen besteht:

Datenbank: Das Datenbankfeature verwaltet alle Daten in einer vordefinierten Datenbanktechnologie, die variabel auswählbar ist. Die Datenbank kann entweder SQL oder file-basiert sein. Das Speichern und Abrufen von Daten wird über den jeweiligen Datenbanktreiber betrieben, welcher mit einer standardisierten Datenbankansteuerung kommuniziert. Dieser befindet sich innerhalb eines *Data Access Object*. Dieses beinhaltet die relevanten Datenbankzugriffsfunktionen und liefert ein DTO.

Service: Der Service erhält Daten von der Datenbank und schickt sie an die Components weiter. Die Services dienen den Components für eine erleichterte Abfragemöglichkeit von Datenobjekten. Weiters benötigen die Services vordefinierte Filterfunktionen. Die Funktionalität der Services gehört der Business-Logik-Schicht an.

Components: Die Components stellen im Feature die Hauptbestandteile dar. Diese repräsentieren die eigentliche Logik, da in diesen Funktionen die Ver-

arbeitung der Daten erfolgt. Die Components erhalten die Daten aus der Datenbank und wandeln diese in *Data Transfer Objects* um. Die Objekte werden an die API weitergeleitet.

API: Die API ist ein Teil der Businessschicht und stellt die Verbindung zum Frontend da. Die API bekommt die Daten von den Components und wandelt diese in webkonforme Übertragungsobjekte um, welche an das Frontend weitergeleitet werden. Außerdem ist die API zuständig für das Empfangen von Daten auf dem Frontend.

Frontend: Das Frontend repräsentiert die Präsentationsschicht, welche für den jeweiligen Benutzer sichtbar ist. Im Frontend werden die Benutzerinteraktionen verwaltet, Daten zu Transferobjekte umgewandelt und an das Backend gesendet.[1] [2] [3] [4]

2.2.4 Hospital Information System

Das Hospital Information System (HIS) von CGM wird unter der Produktbezeichnung G3 HIS geführt und besteht aus mehre Modulen. Die Zentralkomponenten eines KIS sind in diesem System integriert und bilden eine einheitliche Weboberfläche. Diese ist betriebssystemunabhängig und wird in einem vom G3 HIS unterstützten Browser verwendet. Das G3 HIS Projekt umfasst mehrere Projektgruppen, die in die unterschiedlichen Aufgaben (Organisation, Bettenmanagement, Aufnahme, Entlassung, Operationsplanungen) eines Krankenhauses unterteilt sind. Sie bieten eine umfangreiche Konfiguration und Verwaltung der einzelnen Funktionsgruppen bzw. Module an. Die Software stellt die 3. Generation dar und wird ständig weiterentwickelt.[2]

Kapitel 3

Konzept und Design

In diesem Kapitel werden die möglichen Konzepte zur Übertragung von Daten aus dem MRP ins Microsoft Outlook beschrieben. Diesbezüglich werden drei verschieden mögliche Konzepte zur Lösung des gegenständlichen Themas beschrieben:

3.1 Konzept 1: Manuelle Interaktion mit der Microsoft Outlookoberfläche

Der User verwendet das Microsoft Outlook und hat dabei die Möglichkeit folgende Funktionen zu verwenden:

3.1.1 Microsoft Exchange Server

Der Microsoft Exchange Server dient der zentralen Ablage und Verwaltung von E-Mails, Terminen, Kontakten und Aufgaben. Um als Anwender die Funktionen von dem Exchange Server nutzen zu können, ist eine zusätzliche Client-Software nötig. Microsoft liefert dafür das Programm Microsoft Outlook.

3.1.2 Emails

Emails können gesendet, empfangen und in Ordnern hinterlegt werden. Emails, welche Termineinladungen enthalten, werden nach deren Bestätigung in den persönlichen Kalender übertragen. Weiters ist eine schnelle Abfrage der im persönlichen Ordner gespeicherten Emails durch eine Suchfunktion möglich.

3.1.3 Termine und Kalender

Jeder Benutzer verfügt mindestens über einen Kalender. Dieser Kalender besitzt persönliche Termine sowie die Termine des Arbeitsumfeldes. Weiters

können weitere Termine wie Feiertage, Geburtstage etc. angezeigt werden. Eine Option ist die Übermittlung von Texten bei Termineinladungen. Darüber hinaus ist es möglich ganze Gruppen zu einem gemeinsamen Termin einzuladen. Eine Erinnerungsfunktion wird als eine weitere Servicefunktion angeboten.

3.1.4 Aufgaben

Aufgaben werden als Arbeitsliste hinterlegt. Diese sind den jeweiligen Personen zugeordnet und sind in einer eigenen Kategorie im Microsoft Outlook hinterlegt.

3.1.5 Kontakte / Adressen

Personendaten wie Name, Adresse, Telefonnummern, Emailadressen und weitere Informationen sind in den Kontakten hinterlegt. Es besteht die Möglichkeit Kontakte in einem Globalen Adressbuch abzubilden.

3.1.6 Globales Adressbuch

Die Kontakte sind nach Gruppen geordnet u. eingeteilt. Diesbezüglich sind Such- bzw. Filterfunktionen verfügbar.

3.1.7 Outlook Web App

Mithilfe des Outlook Web Access kann im Browser für das eigene Outlookkonto auf Emails, Kontakte und Aufgaben zugegriffen werden. Die EWS API hat die gleiche Funktionalität wie die Webschnittstelle, da diese nach dem selben Prinzip aufgebaut ist. Es gibt eine mobile Outlookversion, die mithilfe der Webschnittstelle auf den Outlookserver bzw. das Microsoft Outlook zugreift. [6]

3.2 Konzept 2: Ansteuerung von Microsoft Outlook mit Hilfe von EWS

Der Benutzer verfügt über Programmierkenntnisse u. kann die Schnittstelle ansteuern und verwenden. Die EWS API wird von Microsoft zur Verfügung gestellt und dient zur Ansteuerung von Microsoft Outlook über eine geeignete Programmiersprache. Diese API beinhaltet dieselbe Funktionalität wie die Webschnittstelle u. arbeitet nach denselben Prinzip der Webschnittstellen. EWS bietet zusätzliche Funktionen zu der Webschnittstelle wie Abfragen von Verfügbarkeiten und Benachrichtigungen. Die EWS API ermöglicht den Programmierer eine einfachere Ansteuerung von Microsoft Outlookservern. Die Schnittstelle bietet eine ausreichende Funktionalität, sodass im Microsoft Outlookclient keine manuellen Aktionen durchgeführt werden müssen.[5]

3.3 Konzept 3: Ansteuerung mittels MRP

MRP ist eine von CGM entwickelte Software, die Termine für Operationen Krankenhäusern unter der Berücksichtigung aller beteiligter Ressourcen optimiert und berechnet. Das MRP-Modul beinhaltet folgende Funktionalitäten:

1. Zu Beginn werden die verfügbaren Ressourcen (Personen, Räume, Geräte, Betten, Materialien) angelegt und zugeordnet.
2. Zuordnung der ressourcenbezogenen Daten: Jeder Ressource werden Informationen beigegeben
3. Mittels der Ressourcen werden mithilfe von Kapazitäten(terminliche Verfügbarkeiten) Organisationseinheiten (Zeitbereiche mit verknüpften Ressourcen) gebildet.
4. Aus diesen Einheiten werden Vorschläge gebildet. Diese Vorschläge beinhalten lediglich eine Zeitspanne. Um einen Termin festlegen zu können, müssen Ressourcen definiert werden. Dies entsteht aus den bereits hinterlegten Personendaten sowie aus den Verfügbarkeitsabfragen der EWS Schnittstelle. Mit diesen erweiterten Funktionen können konkrete Termine generiert u. nach deren Bestätigung ins Outlook exportiert werden. Die exportierten Termine können im betreffenden Kalender eingesehen werden. Das Anlegen von Ressourcen u. das Selektieren von exportierten Terminen erfolgt manuell.

[1] [2] [4]

Kapitel 4

Implementierung

In Abstimmung mit CGM wurde das Konzept 3 für die Umsetzung gemäß Zielsetzung herangezogen. Dabei wurden die Funktionen der EWS-API verwendet und in den Methoden der nachfolgenden Module aufgerufen:

4.1 Microsoft Outlook

Die nachfolgenden aufgelisteten Packages beziehen sich auf die Klassen: *Packagename + Service* , *I + Packagename + Service* und *Packagename + Dto*
Die jeweilige Klasse des Packages implementiert eine vordefinierte Schnittstelle (Interface), welche einen vordefinierten Benennungsschema folgt. Die Funktionalitäten sind unabhängig vom MRP-System, können aber von diesem verwendet werden. Die Datentypen bzw. Datenklassen sind im Package *Model* definiert. Diese Klassen können gleichnamige Klassen im MRP haben, haben aber keine Beziehung zu diesen Klassen.

4.1.1 Appointments

Mit Hilfe des Appointmentservices können Microsoft Outlook-konforme Appointments angelegt, verwaltet und abgerufen werden. Weiters ist möglich Appointments in einem XML-Dokument zu speichern. Das XML Dokument beinhaltet die Datenfelder aus dem *Model* in einer validen XML-Struktur. Diese XML-Dokumente können zur Erstellung von Appointments verwendet werden. Es besteht die Möglichkeit Konflikte zu betreffenden Appointments vom Server abzufragen. Der *AppointmentService* implementiert das zugehörige Interface *IAppointmentService* und verwendet den Datentyp *AppointmentDto*.

Mit der Methode *createAppointment* kann ein Appointment angelegt werden. Der *ExchangeService* wird als Parameter übergeben. Die Methode *build-*

Appointment erstellt den Datentyp *Appointment*. Die Methode *createAppointments* ermöglicht das gleichzeitige Speichern und Anlegen von Appointments. Eine weitere Funktionalität ist durch die Methode *hasConflictedAppoinments* gegeben. Diese Funktion überprüft mit Hilfe des Exchangeservers Konflikte bei betreffenden Appointments. Die Funktionen *appointmentListToXML*, *xmlToDto* und *xmlToAppointmentList* ermöglichen die Verwaltung von outlookkonformen Appointments mithilfe von XML.

4.1.2 Availability

Durch den Availabilityservice können Verfügbarkeiten von Personen und Räumen abgefragt werden. Der *AvailabilityService* implementiert das zugehörige Interface *IAvailabilityService* und verwendet den Datentyp *AvailabilityDto*.

Die Hauptfunktion *checkAvailability* ruft mit Hilfe des *ExchangeService* Verfügbarkeiten über die EWS-Schnittstelle ab. Die Methode *checkAvailbilityForPerson* überprüft die Verfügbarkeit von Personen anhand deren Email-Adressen. Mit Hilfe der Methode *checkAvailbilityForRooms* können im Gegensatz zur obengenannten Methode anstelle von Personen Räume abgefragt werden. Die beiden Funktionen *getRoom* und *getAllRooms* liefern Räume zurück. Personen und Räume werden durch die jeweiligen Emails repräsentiert.

4.1.3 Kalender

Mittels Kalenderservice können Kalender angelegt und abgerufen werden. Der *CalendarService* implementiert das zugehörige Interface *ICalendarService* und verwendet den Datentyp *CalendarDto*.

Die Mehode *getCalendars* ruft die verfügbaren Kalender einer Emailadresse ab. Die Emailadresse repräsentiert Personen oder Räume. Zusätzliche Kalender werden durch die Methode *createNewCalendar* angelegt. Auch gibt es die Methode *sendWarningToUser*, mit dieser kann ein Email an den betreffenden Benutzer versendet werden. Die Emailnachricht enthält die betreffende Warnung.

4.1.4 Kontakt

Der Kontaktservice ermöglicht das Anlegen, Abrufen und Verwalten von Outlookkontakten in den jeweiligen persönlichen Adressbüchern. Der Kontaktservice arbeitet mit dem Benutzer des Exchangeservices. Der *ContactService* implementiert das zugehörige Interface *IContactService* und verwendet den Datentyp *ContactDto*.

Die Methode *getContacts* ruft persönliche Kontakte des angemeldeten Benutzers aus dem persönlichen globalen Adressbuch ab. Die Kontakte sind unsortiert und können mittels Filter zusätzlich aussortiert werden. Neue Kontakte werden über die Funktion *buildContact* erstellt. Die Funktion *getContactDtos* liefert mithilfe der Funktion *buildContact* eine Liste vom vordefinierten Datentyp *ContactDto*.

4.1.5 Email

Der Emailservice ermöglicht das Senden von bestehenden Emails. Diese beinhalten die relevanten Felder der Emails. Weiters können die Emails aus dem jeweiligen Postfach des Benutzers aufgerufen werden. Der *EmailService* implementiert das zugehörige Interface *IEmailService* und verwendet den Datentyp *EmailDto*.

Durch die Mehode *sendEmail* können Emails versendet werden. Diese werden in der Mehode *buildEmail* erstellt und enthalten alle relevanten Informationen zu Sender und Empfänger sowie Betreff und Emailtext. Die Methode *getEmails* ruft eine vordefinierte Anzahl Emails aus dem Posteingang ab.

4.1.6 Exchangeservice

Der Exchangeservice dient als zentrale Verbindung zum Microsoft Outlook. Die Features bzw. Methoden , welche mit dem Exchange-Server kommunizieren verwenden ExchangeService als Parameter. Die Klasse wird *ExchangeServiceConnector* genannt und ist zuständig für die Verbindung zum Server sowie für die Anfragen der Features. Dieser Service bezieht sich ausschließlich auf das eigene Outlookkonto bzw. auf das Konto, welches mit Passwort und Benutzername angegeben wird. Weiters besteht die Möglichkeit die Exchangeserverversion festzulegen. Diese Klasse implementiert das zugehörige Interface *IExchangeServiceConnector*.

Die Serververbindung wird mit Hilfe der Methode *createServerConnection* initialisiert, wobei der Benutzername und das Passwort als Parameter übergeben werden. Die gleichnamige Methode verwendet zusätzlich den Parameter Server-URL. Die XML konforme Methode *createServerConnection* baut mit Hilfe der Daten eines XML-Dokuments eine Serververbindung auf. Die drei genanten Methoden retournieren eine Instanz des Exchangeservice. Dieser Service existiert nur einmal zur Laufzeit. Damit wird garantiert, dass immer der gleiche Server angesteuert wird. Mit der Methode *saveSettingsToXML* können die Servereinstellungen in einem XML-Dokument gespeichert werden.

4.1.7 Filter

Die Filter bestehen aus Filterkomponenten, welche die eigentlichen Filter repräsentieren. Die Komponenten werden vordefiniert und können frei kombiniert werden. Diese Filterkomponenten folgen dem Prinzip der überpersistenten Speicherung. Dies bedeutet, dass bereits gefilterte Daten trotzdem erhalten bleiben. Grundsätzlich ist jede Filterkomponente aus zwei Teilen aufgebaut: Der erste Teil beinhaltet die Informationen zu den Filtern. Der zweite Teil stellt eine Datensammlung dar. Diese Daten bleiben solange erhalten solange die Filterkomponenten aktiv sind. Die Filterkomponenten werden gleichzeitig bzw. hintereinander angewendet, um die logischen Funktionen AND und OR zu repräsentieren Eine weitere Funktionalität ist der Vergleich von Methoden mit den Variablen. Dies bedeutet, dass der Variablenname nicht bekannt ist.

Die Mehode *apply* beschreibt die Hauptfunktionalität des Filters und beinhaltet die Funktionalitäten, die zur Anwendung eines bestimmten Filters erforderlich sind. Das *FilterDto* beinhaltet sowohl die Filterkomponenten als auch die zu filternden Daten. Mit der Methode *applySimpleFilter* wird ein Filter für eine Sammlung von zu filternden Objekten angewandt. Die Funktion *applyFilter* wird verwendet, wenn die zu filternden Objekte den Typ *AppointmentComponent* darstellen. Die zwei Funktionen *applyHelixFilter* werden verwendet, wenn die Filter vom MRP-System stammen. Die Funktionen *getFiltered* und *getUnfiltered* dienen zum Abrufen von gefilterten und ungefilterten Daten eines *FilterDto*s

4.1.8 Model

Die verwendeten Objekte bzw. Datentypen in den einzelnen Features bzw. Packages werden im Model repräsentiert. Die einzelnen Klassen repräsentieren die jeweiligen Datentypen, welche einer vordefinierten Struktur folgen. Diese folgt der Bean-Definition. Diese bedeutet, dass alle Membervariablen *private* sind und daher wird auf die Variablen mit Getter und Setter zugegriffen. Die einzelnen definierten Datentypen besitzen einen öffentlichen Default-Konstruktor ohne Parameter.

4.2 Masterdata-Feature

Als Masterdata werden Features bezeichnet, welche zentrale Daten im MRP verwalten und in einer Datenbank gespeichert werden.

4.2.1 Filter

In dem MRP-Feature Masterdata wurde ein Feature hinzugefügt, welches zur Verwaltung von Filtern dient. Die Filter werden in der zentralen Daten-

bank von MRP gespeichert und über die Weboberfläche von MRP verwaltet, da diese zu den Masterdata-Features zählen. Die Filter dienen zur Filterung von MRP-Appointments. Jeder Filter beinhaltet die benötigten Filterkomponenten, welche für das Filtern der Daten verantwortlich sind.

4.2.2 Filterkomponenten

Jeder Filter hat Filterkomponenten untergeordnet, welche die Informationen zu Filterart, Filtertyp und Filterwert enthalten. Diese Komponenten werden in einer Relation zu den Filtern in der Datenbank abgelegt. Die Filterkomponenten sind immer für einen bestimmten Filter zuständig und können nicht einzeln verwendet werden. Diese Filterkomponenten dienen zum Filtern von Appointments.

4.2.3 Weboberfläche

In der Weboberfläche von dem MRP-System werden die Filter und Filterkomponenten verwaltet. Die Oberfläche wird von G3 HIS bereitgestellt und bietet die Funktonalität von MRP. Mit Hilfe der Filter werden Appointments gesucht und angezeigt. Zukünftig sollen selektierte Appointments ins Microsoft Outlook exportiert werden können.

Kapitel 5

Evaluierung

5.1 Einleitung

Das Microsoft Outlookconnectormodul ist im MRP-System teil-integriert. Das Modul befand sich im gleichen Repository ohne vollständige Integration. Dieses Modul wurde in einem eigenen Branch des Repositories verwaltet. Das Modul stellte daher keinen Teil der vordefinierten Testumgebung dar.

5.2 Testumgebung

Die Testumgebung wurde mit Hilfe eines *Jenkins-Maven-Testserver* realisiert, welcher selbständiges automatisiertes Testen zuließ. In dem Repository gab es die Branches *master* und *dev*. Diese wurden in die automatisierte Testumgebung integriert und in fixierten Zeitabständen getestet. Da das zu bearbeitende Modul kein Teil der zwei Branches war, konnte es nicht getestet werden. Durch den Umstand des zeitlich verzögerten Testprozesses (Ursache lag bei CGM) musste auf andere Testmittel umgestiegen werden.

5.3 Benutzerinteraktives Testen

Auf Grund des verzögerten Testprozesses musste auf die Methodik benutzerinteraktives Testen zurückgegriffen werden. Die gewählte Methodik wird als Benutzer interaktives Testen bezeichnet, besser bekannt als *Whitebox-Test*. Diese Methodik beruht auf der visuellen Kontrolle des Programmierers. Dabei überprüft der Programmierer die Ergebnisse in der grafischen Oberfläche bzw. Konsole. Diese Methodik kann nicht als Test im Sinne eines Testframeworks angesehen werden, da nicht überprüft werden konnte, ob die Testergebnisse den erwarteten Ergebnissen entsprechen. Dies kann erst mit der zukünftigen Integration des Moduls überprüft werden. Diese Testmethodik dient rein zur visuellen Kontrolle ob die geforderten Datenfelder

richtig in der Konsole angezeigt werden. Trotzdem kann mit Hilfe dieser Methodik die Funktion des Programmes getestet werden. Dies ist auch möglich wenn kein geeignetes Testframework vorhanden ist. Da das Modul mit einem Server kommuniziert, stellt sich automatisiertes Testen mittels Frameworks als aufwendig und komplex dar.

Im Rahmen der Testung wurden folgende Funktionen überprüft:

Getestete Funktionen	Test durchgegangen
Appointments anlegen	ja
Appointments abrufen	ja
Availablilities abfragen	ja
Kalender anlegen	ja
Kalender abfragen	ja
Kontakte abrufen	ja
Adressbuch abfragen	ja
Emails senden	ja
Emails abrufen	ja
Serververbindung initialisieren	ja
Serververbindung speichern	ja
Serververbindung abfragen	ja
Filter verwalten	ja
Filterkomponenten verwalten	ja
Filterkomponenten zuordnen	ja
MRP-Filter verwalten	ja
MRP-Filterkomponenten verwalten	ja
MRP-Appointments exportieren	nein

Tabelle 5.1: Tests

Die obengenannten Funktionen wurden ausführlich getestet.

Kapitel 6

Zusammenfassung

In diesem Kapitel werden die Ergebnisse zusammengefasst und diskutiert.

6.1 Resultate

Bei dieser Arbeit wurde ein Programm implementiert , welches als Modul für das MRP-System der Firma CGM fungiert. Die Implementierung wurde mittels JAVA durchgeführt und daher wurde die Java EWS API verwendet. Die EWS API dient zur Ansteuerung eines Microsoft Outlookservers mithilfe von Programmmethoden. Das Modul verwendete diese API um bereitgestellte Termine aus dem MRP-System an das Microsoft Outlook zu exportieren. Die Daten wurden von weiteren Modulen des MRP-Systems generiert und bereitgestellt. Allerdings fehlte die Integration des Moduls in das gesamtheitliche MRP-System , da dieses zum damaligen Zeitpunkt noch nicht fertiggestellt war. Das Testen erfolgte durch benutzerinteraktives Testen. Dieses beruhte auf einer rein visuellen Betrachtung des Benutzer. Die diesbezüglichen Tests verliefen positiv.

6.2 Diskussion

Das in dieser Arbeit erstellte Programmmodul ist universell einsetzbar, wurde aber geringfügig an das MRP-System angepasst. Die komplexen Datentypen die die EWS verwendet, wurden durch das vereinfachte DTO ersetzt. Die universelle Einsetzbarkeit wurde damit zusätzlich verbessert. Weiters wurde die XML-Konformität implementiert. Dies bedeutet, dass XML-Dokumente gelesen und geschrieben werden können. Die Implementierung der verwendeten Filter wurde generisch gehalten, um eine universelle Einsetzbarkeit zu ermöglichen.

6.3 Ausblick

Da eine vollständige Integration des MRP-Moduls in das Microsoft Outlook noch nicht vollständig möglich war, kann davon ausgegangen werden, dass dies in der Folge von der Firma CGM erfolgt. Das MRP-Modul ist ein Teil der *G3 HIS* Software, die bereits in älteren Versionen am Markt ist. Dies macht die Software insbesonders für jene Krankenhäuser interessant, welche über ältere Versionen bereits verfügen.

Kapitel 7

Anhang

7.1 Literatur

Das Handbuch Microsoft Outlook 2013, Thomas Joos, 2013, O'Reilly Verlag GmbH

JAVA EWS Reference Dokument in GIT, https://github.com/OfficeDev/ews-java-api/wiki/Getting-Started-Guide

CGM G3 HIS Documentation , CGM, Internes Dokument(vertraulich)

G3 Reverse Proxy,

CGM HTML5 Controls Browser

Quellenverzeichnis

Literatur

[1] CGM Clinical. „CGM HTML5 Controls Browser“. Documentation to the UI Elements (siehe S. 7, 11).

[2] CGM Clinical. „G3 HIS Reference Documentation“. Documentation to CGM G3 HIS Sofware (siehe S. 6, 7, 11).

[3] CGM Clinical. „G3 Reverse Proxy“. Technical Documentation for the G3 HIS Project (siehe S. 7).

[4] CGM Clinical. „MRP Technical Documentation“. Technical Documentation for the MRP Project (siehe S. 5–7, 11).

[5] *Java EWS API Getting Started Guide*. https://github.com/OfficeDev/ews-java-api/wiki/Getting-Started-Guide, 2016 (siehe S. 3–5, 10).

[6] T. Joos. *Microsoft Outlook 2013 - Das Handbuch*. O'Rielly Verlag Gmbh, 2013 (siehe S. 4, 5, 9).