Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Bachelorarbeit

Alexander Vette

Ein Kalender zur Unterstützung kontextberücksichtigender Tagesplanung

Fakultät Technik und Informatik
Department Informatik

Faculty of Engineering and Computer Science
Department of Computer Science

Alexander Vette

Ein Kalender zur Unterstützung kontextberücksichtigender Tagesplanung

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang Technische Informatik
am Department Informatik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. rer. nat. Kai von Luck
Zweitgutachter : Prof. Dr. rer. nat. Gunter Klemke

Abgegeben am 5. Juli 2012

Alexander Vette

Thema der Bachelorarbeit

Ein Kalender zur Unterstützung kontextberücksichtigender Tagesplanung

Stichworte

Kalender, Tagesplaner, Digital Social Media, Context Awareness, Semantic Web

Kurzzusammenfassung

Kalender und Terminplaner begleiten den Menschen seit langer Zeit bei der Planung und Koordination von Terminen aus den verschiedenen kontextuellen Umfeldern, in denen er sich bewegt. Bedingt durch den digitalen Wandel der Gesellschaft und der Verschiebung des klassischen Arbeitsumfelds durch eine immer stärkere Vernetzung der Menschen untereinander enstehen völlig neue Anforderungen an die Gestaltung des Tagesablaufs. Im Rahmen dieser Bachelorarbeit erfolgt die Entwicklung einer Anwendung zur kontextberücksichtigenden Tagesplanung, die den Nutzer bei der Planung und Vorbereitung des Alltags begleitet und dabei vergleichbar zu einem menschlichen Assistenten agiert. Die Anwendung soll zeigen, dass sich Anhand zusätzlicher Daten, vor allem aus Quellen des *Digital Social Media*, ein solches Verhalten in Grundzügen nachbilden lässt.

Alexander Vette

Title of the paper

A calendar in support of context-aware day planning

Keywords

Calendar, Day Planner, Digital Social Media, Context Awareness, Semantic Web

Abstract

Calendars and day planners have been used by people for planning and coordinating events of different contextual settings for a long time. Due to social changes based on digital media and the accompanying shift of the classic work environment conditioned by global networking new ways of scheduling a person's daily tasks are required. As part of this bachelor thesis an application in support of context-aware day planning is developed, supporting the user in structuring and preparing his daily schedule like a human assistant would do. This application shall demonstrate that under the involvement of additional data gathered by sources of *Digital Social Media* the basics of such a behaviour can be realised.

Inhaltsverzeichnis

1 Einführung

Unsere Gesellschaft unterliegt einem stetigen Wandel in allen Bereichen des alltäglichen Lebens, dazu gehört auch der Wandel unseres Arbeitsumfelds. Computer und Smartphones als Teil ubiquitärer Systeme, wie sie bereits 1991 von Marc Weiser (Weiser (1991)) beschrieben wurden, begleiten den Menschen allgegenwärtig und ermöglichen ihm eine ständige Vernetzung mit anderen. Er hat die Möglichkeit, von jeden Punkt der Erde, der über eine Kommunikations- und Datenanbindung verfügt, zu agieren.

Viele Berufe erfordern heutzutage zur Durchführung der Tätigkeit keine physische Anwesenheit an der Arbeitsstelle und können „out of office“, zum Beispiel unterwegs, bei einem Kunden oder aus dem *Home-Office* heraus wahrgenommen werden. Dabei können neben dem Ort auch die Arbeitszeiten immer flexibler gehalten werden, der klassische *nine to five* Arbeitstag tritt somit immer weiter in den Hintergrund.

Die Kommunikation mit anderen Menschen per Videochat-Software erlaubt die Zusammenarbeit von Teams über die ganze Welt verteilt. Hierdurch wird die Planung des Alltags erschwert, da die Menschen ihre Termine aus unterschiedlichen Zeitzonen heraus koordinieren müssen. Dies erschwert zusätzlich die Einhaltung der sogenannten *Work-Life-Balance* (siehe u.a. Wiese (2007)), da die Welten aus Privatleben und Arbeitsalltag durch diese zeitlichen Randbedingungen und die immer engmaschigere Vernetzung kontinuierlich weiter miteinander verschmelzen.

Diesen Alltag aus Arbeits- und Privatleben gilt es zu koordinieren, das Zeitmanagement ist hierbei ein zentrales Stichwort. Um die Planung von Tagesabläufen festzuhalten und visualisieren zu können, nutzt der Mensch Kalender und Tagesplaner in verschiedensten Varianten. Digitale Kalender erlauben die Distribution und Koordination von Terminen in vernetzten, kollaborativen Gruppen. Somit ist es möglich, Termine an andere, sogar ohne vorherige Absprache mit diesen, zu delegieren.

Ein Problem, das aus dieser „Fremdverplanung“ durch andere resultiert, ist der fehlende Überblick darüber, wann und wo Termine im Tagesablauf anstehen, welche Ziele sie verfolgen und welcher Bedarf zur Vorbereitung besteht. Hierzu gibt es verschiedene Möglichkeiten: Von Erinnerungen durch *Personal Information Management* Tools bis hin zu einem persönlichen Assistenten, der mit der Zeit ein immer umfassenderes Wissen über die Person, für die er arbeitet, ansammelt.

Das Ziel dieser Arbeit ist es, eine Anwendung zu entwickeln, die in ihrer unterstützenden und begleitenden Funktionsweise bei der alltäglichen Terminplanung der eines menschlichen persönlichen Assistenen in Grundzügen ähnelt. Dazu ist es erforderlich, der Anwendung eine „Idee“ davon zu vermitteln, in welchem Kontext sich ein Mensch über den Tag hinweg befindet und welche Ziele die geplanten Termine verfolgen. Hierzu soll sich die Software zusätzlicher, externer Informationsquellen bedienen. Diese sollen es ermöglichen, die digitalen Äquivalente der Personen, die mit den einzelnen Terminen in Verbindung stehen, zu beschreiben sowie die Orte, an denen die Termine stattfinden, widerzuspiegeln.

Gliederung der Arbeit

Im Grundlagenkapitel (2) werden zunächst nötige Begrifflichkeiten, Formate sowie Technologien vorgestellt, die zum Verständnis der behandelten Inhalte dieser Arbeit erforderlich sind.

Die anschließende Analyse (3) beschäftigt sich genauer mit der Nutzung von Kalendern und den philosophischen und psychologischen Aspekten des Begriffs „Event“. Ein realistisch konstruiertes Bild der Nutzung einer Anwendung zur kontextberücksichtigenden Tagesplanung wird anhand von Szenarien genauer beschrieben. Zusammen mit einer Auswahl an vergleichbaren Arbeiten wird ein Anforderungskatalog an die hier zu entwickelnde Applikation erarbeitet.

Eine Übersicht über die Eigenschaften der einzelnen Programmbestandteile zur Erfüllung der entwickelten Anforderungen liefert das anschließende Designkapitel (4). Es beschreibt die Komponenten der Software und geeignete Architekturmuster, die für eine Realisierung verwendet werden. Es stellt ebenfalls die Anforderungen vor, die an externe Dienste zur Anreicherung des vorhandenen Datenbestandes bestehen, um der Anwendung eine Kontextbasiertheit zu ermöglichen.

Die Evaluierung und konkrete Auswahl dieser Dienste wird in der Realisierung (5) beschrieben. Unter Berücksichtigung der gewählten Architekturmuster wird die Implementierung der Anwendung vorgestellt und ihr Funktionsumfang durch eine Evaluation genauer betrachtet.

Im Schlussteil (6) findet eine Zusammenfassung der Ergebnisse dieser Arbeit sowie ein Ausblick für mögliche Weiterentwicklungen und Möglichkeiten der Anwendung statt.

Alle Bezeichnungen in dieser Arbeit, die sich auf Personen beziehen, gelten gleichermaßen für das männliche als auch das weibliche Geschlecht, unabhängig von der in der konkreten Formulierung verwendeten geschlechtsspezifischen Bezeichnung.

2 Grundlagen

Dieses Kapitel soll dazu dienen, einen kurzen Überblick über die in dieser Arbeit behandelten Technologien und Begrifflichkeiten, soweit diese zum Verständnis erforderlich sind, zu geben. Für einen detailierteren Überblick sei an dieser Stelle auf die weiterführende Fachliteratur verwiesen.

2.1 Digital Social Media

Unter dem Begriff *Digital Social Media* (im Folgenden DSM) werden alle digitalen Medien bzw. Plattformen vereint, die dem Menschen dazu dienen, miteinander zu kommunizieren oder Informationen auszutauschen. Der Fokus der einzelnen Plattformen liegt dabei entweder primär auf der Kommunikation miteinander oder auf der Publikation, Bearbeitung und dem Austausch von konkreten Inhalten, die von den Personen generiert werden (auch bezeichnet als *User Generated Content*). Der Begriff ist eng angelehnt an den des *Web 2.0*, da die Plattformen, die DSM ausmachen, auf den Grundlagen dieser Generation des *World Wide Web* (im Folgenden WWW) aufbauen. DSM umfasst soziale Netzwerke, Content Communities (z.B. *Youtube*[1] oder *Last FM*[2]), Blogs, Bewertungsportale (z.B. *Epinions*[3]), Social Bookmarking Sites (z.B. *Delicious*[4]), Wikis sowie soziale Spiele (z.B. *Second Life*[5]) (Bry u. a. (2010)).

2.2 Semantic Web

Das WWW ist bis heute eine Ansammlung von Dokumenten, die Informationen enthalten und miteinander vernetzt werden können. Der Mensch kann auf diese Infomationen zugreifen, indem er mit Hilfe einer Maschine nach einem entsprechenden Dokument sucht oder

[1] http://www.youtube.com
[2] http://www.lastfm.de
[3] http://www.epinions.com
[4] http://delicious.com
[5] http://secondlife.com

dieses direkt anfordert. Die Maschine „versteht“ dabei die Bedeutung des angeforderten Dokumenteninhalts allerdings nicht, auch bei Suchanfragen wird das passende Ergebnis nur durch den Abgleich mit bestimmten Schlüsselwörtern gefunden.
Die Notwendigkeit des Verstehens von Dokumenteninhalten durch Maschinen besteht allerdings, angetrieben vor allem durch den Wandel weg vom *Personal Computer* hin zu ubiquitären Systemen, die für den Anwender nicht wahrnehmbar im Hintergrund agieren. Diese Systeme führen immer komplexere Aufgaben aus und verknüpfen sich dabei miteinander in einem „Internet der Dinge“ (vgl. Ashton (2009)), um so die Lücke zwischen realer und virtueller Welt zu schließen (Fleisch u. a. (2005)).

Die Inhalte von Dokumenten für Maschinen verständlich zu machen ist das Ziel des *Semantic Web*. Vorgestellt im Jahr 2001 durch Tim Berners-Lee, den Erfinder des WWW und Direktor des W3C Konsortiums[6] beschreibt es ein Netz, in dem Informationen durch zusätzliche Semantik angereichert für Maschinen interpretier- und manipulierbar gemacht werden können (Lee u. a. (2001)). Somit ist es möglich, Daten nicht nur auf der Dokumentenebene miteinander zu verlinken, stattdessen können sie auf der Ebene ihrer Bedeutung miteinander verknüpft werden.

Das *Semantic Web* baut dabei auf das bestehende WWW auf und erweitert die bereits für Maschinen lesbaren Formate wie beispielsweise HTML um zusätzliche Metadaten, welche die Eigenschaften des Dokumenteninhalts beschreiben. Die Beispiele 2.1 und 2.2 beschreiben eine solche Annotation. Dem Wort „Hamburg“ wird ein Link auf eine Quelle angehängt, unter der das Wort näher beschrieben wird.

```
1 <item>Hamburg</item>
```

Listing 2.1: Tag in einer standard HTML Seite

```
1 <item rdf:about="http://de.dbpedia.org/page/Hamburg">Hamburg</item>
```

Listing 2.2: Tag in einer semantisch erweiterten HTML Seite

Die zusätzlichen Metadaten werden durch Ontologien beschrieben, welche die komplexen Beziehungen von Dingen miteinander verknüpfen. Eine solche Aussage besteht aus dem Tripel Subjekt, Prädikat und Objekt. Das Prädikat erweitert dabei die bisherige Speicherung von Daten im Schlüssel-Wert Format. Somit können Informationen untereinander in Beziehung gesetzt werden.

Die in dieser Arbeit verwendeten Ontologien des *Semantic Web* sowie *SPARQL* als Abfragesprache zur Verarbeitung der Annotationen werden im Folgenden kurz vorgestellt.

[6] http://www.w3c.org

2.2.1 RDF

Das *Resource Description Framework* (im Folgenden RDF) ist eine der Hauptkomponenten des *Semantic Web* und eine Empfehlung des W3C Konsortiums (W3C (2004)). Es beschreibt Informationen über eine Ressource durch die beschriebene Subjekt-Prädikat-Objekt Struktur:

Subjekt: Die Ressource, die beschrieben werden soll. Diese wird durch einen eindeutigen *Uniform Resource Identifier* (kurz: *URI*), meistens eine URL, identifiziert.

Prädikat: Eine Eigenschaft der Ressource. Eine Ressource kann beliebig viele dieser Prädikate haben.

Objekt: Der Wert der Eigenschaft, dieses kann ebenfalls eine Ressource oder ein Literal zur Darstellung von z.B. Zahlen, Daten, oder Wahrheitswerten sein.

Der Aufbau dieser Tripel-Struktur stellt einen gerichteten Graphen dar, eine einfache RDF Beschreibung im XML Format wird in Beispiel 2.3 dargestellt.

```
<?xml version="1.0" encoding="UTF-8" ?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:dc="http://purl.org/dc/elements/1.1/">
  <rdf:Description rdf:about="http://de.wikipedia.org/wiki/Resource_Description_Framework">
    <dc:title>Resource Description Framework</dc:title>
    <dc:publisher>Wikipedia - Die freie Enzyklopädie</dc:publisher>
  </rdf:Description>
</rdf:RDF>
```

Listing 2.3: RDF Beispiel als XML Notation (Quelle: Wikipedia (2012a))

Die RDF Syntax wird durch eine XML Struktur ausgedrückt. Andere Notationen sind ebenfalls möglich, finden in der Praxis allerdings nur bedingt Verwendung. Durch `xmlns:rdf` wird ein Link auf den Namensraum gesetzt, der das verwendete Vokabular eindeutig identifiziert und beschreibt, in diesem Fall `RDF` und `DC`, die Verwendung mehrerer Namensräume und Ontologien ist möglich. Die Ressource mit dem zugehörigen *URI* wird durch `rdf:about` beschrieben. Das angeführte Beispiel hat zwei Eigenschaften: einen Titel `dc:title` sowie einen Herausgeber `dc:publisher`, die beide auf Literale verweisen.

2.2.2 FOAF

Die bereits vorgestellte Beschreibung im Tripel-Format kann neben der Charakterisierung von Dingen auch zur Beschreibung von Personen benutzt werden. Das Ziel des *Friend Of A Friend* (im Folgenden FOAF) Projekts (FOAF Projekt (2000)), vorgestellt im Jahr 2000, ist die maschinenlesbare Beschreibung des eigenen Personenprofils, der Aktivitäten sowie der Beziehungen zu anderen Personen und Objekten. Durch die Verknüpfung von Personen

untereinander können Graphen über das soziale Gefüge, in dem sie sich befinden, aufgespannt werden.
Anhang 1 beschreibt den Aufbau einer FOAF Profildatei, diese können durch Tools wie dem *FOAF-a-Matic*[7] manuell generiert werden. Wie sich dem Beispiel entnehmen lässt, hat der Anwender die Möglichkeit, statische Informationen zu seiner Person anzugeben (z.B. Namen, Spitznamen, Kontaktdaten, Webpräsenzen). Zusätzlich können durch die Eigenschaft `foaf:knows` Referenzen auf die Profildateien der mit dem Nutzer befreundeten Personen gemacht werden.

2.2.3 SIOC

Das *Semantically-Interlinked Online Communities Project* (im Folgenden SIOC) aus dem Jahr 2004 ist ein weiteres Projekt, das sich mit der Verlinkung von sozialen Daten durch eine Ontologie befasst. Der Schwerpunkt liegt hierbei auf dem *User Generated Content*, der durch Personen publiziert wird. Bei den Netzen kann es sich unter anderem um Blogs, Foren, Mailinglisten oder auch Wikis handeln. Anhang 2 beschreibt den Aufbau einer XML Datei mit SIOC Inhalten. Wie auch das FOAF Vokabular basiert SIOC auf RDF. Die Referenz auf die Person, die den jeweiligen Beitrag erstellt hat, kann neben `sioc:has_creator` mit dem zugehörigen `sioc:UserAccount`, welche den Bezug zur Identität der Person auf der jeweiligen Plattform herstellen, mittels `foaf:maker` auf die im Netz abrufbare persönliche Profildatei der Person erfolgen. Dadurch ist es FOAF nicht nur möglich, Personen miteinander zu verlinken, sondern auch die Inhalte, die von ihnen erstellt werden.

2.2.4 SPARQL

SPARQL Protocol and RDF Query Language ist eine Abfragesprache für Inhalte, die mit RDF beschrieben werden und ist seit 2008 eine offizielle Empfehlung des W3C Konsortiums (W3C (2008)). Unter Angabe des verwendeten Vokabulars können die hinterlegten Datentripel abgefragt und als Graph dargestellt werden. Eine einfache Abfrage wird in den Beispielen 2.4 bis 2.6 dargestellt (Quelle: W3C (2008)). Die Namen der Mitarbeiter, die in einer eigenen Ontologie festgelegt sind, werden mittels SPARQL abgefragt und in FOAF-konforme Namen transformiert. Die in Beispiel 2.4 verwendete Turtle-Notation[8] ist eine alternatives Format zur XML Schreibweise.

```
@prefix org:    <http://example.com/ns#> .
2
_:a  org:employeeName   "Alice" .
4 _:a  org:employeeId     12345 .
```

[7] http://www.ldodds.com/foaf/foaf-a-matic
[8] http://www.w3.org/TeamSubmission/turtle/

```
_:b  org:employeeName   "Bob" .
_:b  org:employeeId     67890 .
```

Listing 2.4: RDF Beispiel in alternativer Turtle-Notation

```
PREFIX foaf:   <http://xmlns.com/foaf/0.1/>
PREFIX org:    <http://example.com/ns#>

CONSTRUCT { ?x foaf:name ?name }
WHERE  { ?x org:employeeName ?name }
```

Listing 2.5: SPARQL Abfrage

```
@prefix org: <http://example.com/ns#> .

_:x foaf:name "Alice" .
_:y foaf:name "Bob" .
```

Listing 2.6: Ergebnis der SPARQL Abfrage

Der Aufbau der Abfragen erfolgt ebenfalls als Tripel Notation. Bei der Suche werden zu bestimmende Variablen durch ein „?“ markiert (in Bsp. 2.5 `?x` und `?name`). Dadurch werden alle Tripel ermittelt, bei denen die Anordnung der zu bestimmenden Subjekte, Prädikate oder Objekte übereinstimmt (im Beispiel `org:employeeName`). Der vollständige Funktionsumfang wird auf den Seiten der W3C (2008) näher beschrieben.

2.3 Context Awareness

Unter *Context Awareness* ist zu verstehen, dass eine Anwendung Informationen und Services unter Einbeziehung des momentanen Kontextes des Benutzers bereitstellen kann. Dey definiert Kontext als:

> *„A system is context-aware if it uses context to provide relevant information and/or services to the user, where relevancy depends on the user's task.“*
>
> - Dey u. a. (1999)

Mit Kontext wird dabei jede relevante Information bezeichnet, die dazu benutzt werden kann, die Situation einer Person, eines Ortes oder eines Objektes zu charakterisieren. Die Schwierigkeit hierbei liegt in der Bestimmung dieses Kontextes, da die Applikation im Gegensatz zum Menschen über keinen „gesunden Menschenverstand“ und keinen Erfahrungsschatz verfügt, anhand dessen sie auf äußere Umstände oder implizit formulierte Einzelheiten reagieren kann.
Es besteht daher die generelle Anforderung an *Context-Aware* Software, so viele Informationen wie möglich über die betreffende Situation zu sammeln und auszuwerten. In ihrer

Arbeit *Towards a Better Understanding of Context and Context-Awareness* stellen Dey u. a. (1999) *Identität*, *Ort*, *Zeit* und *Aktivität* als die vier primären Faktoren vor, die eine Situation charakterisieren. Diese können dazu verwendet werden können, Rückschlüsse auf weitere Informationen zu ziehen. Die Identität beschreibt die beteiligten Personen, der Ort beschreibt die nähere Umgebung des Anwenders oder der Applikation, in der sie zum Einsatz kommt. Durch die Erfassung aller verfügbaren Daten soll kontextberücksichtigende Software abhängig von der Situation drei Eigenschaften erfüllen können:

- Darstellung von Informationen und Services für den Anwender.
- Automatische Ausführung eines Services.
- Kennzeichnung von Kontextelementen zur späteren Verwendung.

2.4 iCal und vCard

Das iCalender Format (im Folgenden iCal) ist ein standardisiertes Format zum Austausch von Kalenderdaten, dessen genauer Umfang durch den vorgeschlagenen Standard RFC5546 (2009) beschrieben wird. Das Format kann dazu genutzt werden, Eventdaten, To-do-Objekte oder Journal-Einträge unabhängig von der jeweiligen Kalenderapplikation auszutauschen. Der iCal Standard wird von einer Vielzahl bekannter Hersteller von Kalendersoftware in seiner aktuellen Version 2.0 unterstützt. Der Aufbau eines einfachen Termins in einer iCal Datei wird in Listing 2.7 beschrieben.

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//Apple Inc.//iCal 4.0.4//EN
BEGIN:VEVENT
UID:uid5@haw-hamburg.com
DTSTAMP:20120604T160000Z
ORGANIZER;CN=Alexander Vette:MAILTO:alexander.vette@haw-hamburg.de
ATTENDEE;CN=Kai v. Luck:MAILTO:luck@informatik.haw-hamburg.de
DTSTART:20120604T160000Z
DTEND:20120604T180000Z
SUMMARY:Meeting zum Thema iCalendar
LOCATION:HAW Hamburg
END:VEVENT
END:VCALENDAR
```

Listing 2.7: iCal Eintrag

Ein weiteres, durch eine RFC festgelegtes, Format zum Austausch persönlicher Informationen (einer „Visitenkarte“), ist das vCard Format. Spezifiziert durch RFC6350 (2011) ähnelt es dem iCal Format in seinem Aufbau aus Eigenschaften und Attributen. Listing 2.8 beschreibt einen beispielhaften vCard Eintrag.

```
BEGIN:VCARD
VERSION:3.0
N:Vette;Alexander
FN:Alexander Vette
ORG:HAW Hamburg
URL:http://www.haw-hamburg.de
EMAIL;TYPE=INTERNET:alexander.vette@haw-hamburg.de
TEL;TYPE=voice,work,pref:+49 1234 56788
ADR;TYPE=intl,work,postal,parcel:;;Berliner Tor 7;Hamburg;;20099;Germany
END:VCARD
```

Listing 2.8: Aufbau einer vCard Datei

Wie das Beispiel zeigt, bietet das Format die Möglichkeit, eine Vielzahl von Attributen zu verwalten, etwa Adressen, Kontaktinformationen und zugehörige Internetpräsenzen. Diese lassen sich in weitere Unterkategorien, z.B. privat oder beruflich, aufteilen.

2.5 Gaußsche Osterformel

Die Gaußsche Osterformel, benannt nach Carl Friedrich Gauß, bezeichnet einen Satz von Gleichungen, mit denen das Osterdatum (Ostersonntag) eines beliebigen Jahres berechnet werden kann. Der Ostersonntag gehört zu den beweglichen Feiertagen, hat also kein festgelegtes Datum. Stattdessen richtet er sich, festgelegt durch das erste *Konzil von Nicäa* und den Berechnungen des Begründers der Christlichen Zeitrechnung, Dionysius Exiguus, nach dem ersten Vollmond im Frühling, an dessen darauffolgenden Sonntag er fällt. Frühlingsanfang ist am 21. März, ein Vollmond an diesem Tag zählt als Frühlingsvollmond, daher ist der frühstmögliche Termin für Ostern der 22. März, der späteste, bedingt durch den Julianischen Kalender, der 25. April. Die Berechnung des Ostersonntags ist interessant für Kalender, da sich alle beweglichen christlichen Feiertage in ihrem Datum nach Ostern richten (z.B. Himmelfahrt 39 Tage oder Pfingsten 49 Tage nach Ostersonntag).
In seinen Gleichungen vereinfacht Gauß die bereits bestehende Berechnung des Osterdatums, genannt *Computus* (v.lat. computus = „Berechnung“, der Vorläufer des Wortes Computer). Listing 2.9 beschreibt des Algorithmus in seiner Version aus dem Jahr 1816.

```
Julianischer Kalender                |       Gregorianischer Kalender
     a = Jahr mod 19                 |
     b = Jahr mod 4                  |
     c = Jahr mod 7                  |
     k = Jahr div 100                |
     p = (8k + 13) div 25            |
     q = k div 4                     |
     d = (19a + M) mod 30            |      M = 15 ? M = (15 + k ? p ? q) mod 30
     e = (2b + 4c + 6d + N) mod 7    |      N =  6 ? N = (4 + k ? q) mod 7
```

Listing 2.9: Gaußscher Osteralgorithmus von 1816 für Julianische und Greogorianische Kalender

Der resultierende Ostersonntag fällt auf den „(22 + d + e)ten März“, wobei der 32. März dem 1. April entspricht. Der Algorithmus ist bis heute, abgesehen von gerigfügigen Änderungen durch die Verschiebung der Mondphasen des Gregorianischen im Gegensatz zum Julianischen Kalender, gültig. In Deutschland wird der Ostersonntag von der Physikalisch-Technische Bundesanstalt mit Hilfe der erweiterten Gaußsche Osterformel unverbindlich berechnet.

3 Analyse

In diesem Kapitel werden die Anforderungen an eine kontextberücksichtigende Tagesplanerapplikation erarbeitet. Hierzu wird zunächst der psychologische und philosophische Aspekt des Begriffs „Event“ genauer betrachtet. Im Anschluss wird das Nutzungsverhalten von Kalendern untersucht und die Eigenschaften unterschiedlicher Kalendervarianten genauer betrachtet.
Die möglichen Funktionen einer kontextberücksichtigenden Anwendung zur Tagesplanung werden durch die Formulierung verschiedener Szenarien vorgestellt. Eine Übersicht über unterschiedliche Ansätze zur Entwicklung von Anwendungen, die den gewohnten Funktionsumfang herkömmlicher Kalendersysteme erweitern, dient bei der anschließenden Entwicklung eines Anforderungskatalogs als Quelle möglicher zu implementierender Funktionen.

3.1 Kalender

Das folgende Kapitel soll einen Überblick darüber geben, unter welchem historischen Hintergrund Kalender entstanden sind und welchen Zweck sie verfolgen.
Die Grundlage eines Kalenders bilden die einzelnen Termine, deren Begriffsdefinition eng mit der des Begriffs „Event“ verbunden ist. Zur Beschreibung eines Events gibt es in der analytischen Philosophie verschiedene Ansätze, die ein Event als Gebilde aus Objekten, Eigenschaften und einem Zeitpunkt bzw. Zeitraum oder als Zeitpunkt der Änderung eines Zustands (Kim (1998)) beschreiben.
Einen ähnlichen, jedoch stärker auf Kalendereereignisse bezogenen, Ansatz der Definition bietet das philosophische Gebilde der *prospektiven Erinnerung*, welche in Kapitel 3.1.1 näher beschrieben wird. Zusätzlich wird eine mögliche Definition zum genaueren Verständnis des Kontexts eines Termins gegeben.
In den darauffolgenden Abschnitten 3.1.3 und 3.1.4 werden die unterschiedlichen Nutzungsarten und Eigenheiten von Kalendern näher beschrieben. Zum Abschluss werden in Kapitel 3.1.5 die Unterschiede bei der Benutzung von Kalendern in Papier- und digitaler Form erläutert.

3.1.1 Prospektive Erinnerung

Das alltägliche Leben ist bestimmt von Dingen, die zu einem festgelegten Zeitpunkt erledigt oder bedacht werden müssen, zum Beispiel das Zahlen von Rechnungen innerhalb einer bestimmten Frist oder ein Arzttermin. Der Vorgang des Erinnerns an Aufgaben, die in der Zukunft ausgeführt werden müssen, wird als *prospektive Erinnerung* bezeichnet Kvavilashvili u. Fisher (2007). Im Gegensatz dazu steht die *retrospektive Erinnerung*, welche den Prozess beschreibt, sich an Ereignisse in der Vergangenheit zu erinnern. Die Literatur unterscheidet zwischen zwei Motiven, die eine prospektive Erinnerung auslösen (Francis-Smythe u. a. (2006)):

Event-basiert Die Erinnerung an die Notwendigkeit zur Erledingung einer Aufgabe wird durch Assoziation mit einem auftretenden Event ausgelöst, etwa sich daran zu erinnern, einen Brief zu versenden, weil man einen Briefkasten gesehen hat.

Zeit-basiert Die Erinnerung an etwas ist verbunden mit der Assoziation durch einen Zeitpunkt oder eine Zeitspanne, wie zum Beispiel die Teilnahme an einem Meeting, welches zu einer bestimmten Uhrzeit beginnt, oder das Versenden einer E-Mail am selben Tag.

Der Ablauf einer Erinnerung von ihrer Erzeugung bis zur Ausführung wird in Abb. 3.1 dargestellt. Der Vorgang beginnt zunächst mit der Formulierung und Kodierung der Absicht, in der Zukunft etwas bestimmtes auszuführen (a). Dabei werden die Schritte auf dem Weg von diesem Ist-Zustand bis zum Event (f) nicht weiter berücksichtigt. Vielmehr wird der Zeitpunkt des Events als Ist-Zustand angenommen, vom dem aus eine Rückerinnerung möglich ist. Die betreffende Person hat eine Vorstellung davon (eine mentale Projektion), wie das Event bei seinem Eintreten aussehen und ablaufen wird (b). Für gewöhnlich ist die Ausführung der projizierten Aufgabe mit einem festen Zeitpunkt oder einer Zeitspanne verbunden, daher wird die Aufgabe geplant und terminlich festgehalten (c).

Mit der Erinnerungsphase (d) beginnt der *Test-Wait-Test-Exit* Zyklus (Kvavilashvili u. Fisher (2007)). Durch die vorher angesprochenen Ursachen, die eine prospektive Erinnerung auslösen können, wird die betreffende Aufgabe zu bestimmten Anlässen im Gedächtnis der Person in den Vordergrund gerückt. Entscheidend ist hierbei die richtige Erinnerung zur richtigen Zeit. Erinnert sich eine Person zu einem Zeitpunkt an die zu erledigende Aufgabe (= „Test“), werden die äußeren Umstände auf den richtigen Zeitpunkt zur Ausführung der Aufgabe geprüft (e). Ist dieser erreicht, wird die Tätigkeit durchgeführt (= „Exit“) (f), falls nicht, verfällt das Gedächtnis wieder in einen Zustand des Wartens (= „Wait“), bis der nächste Moment der Erinnerung erreicht wird (= „Test“) (d).

Die Stadien des prospektiven Erinnerns lassen sich auf Kalender übertragen, daher werden diese als Unterstützung im prospektiven Prozess angesehen (Payne (1993)). Durch Termine,

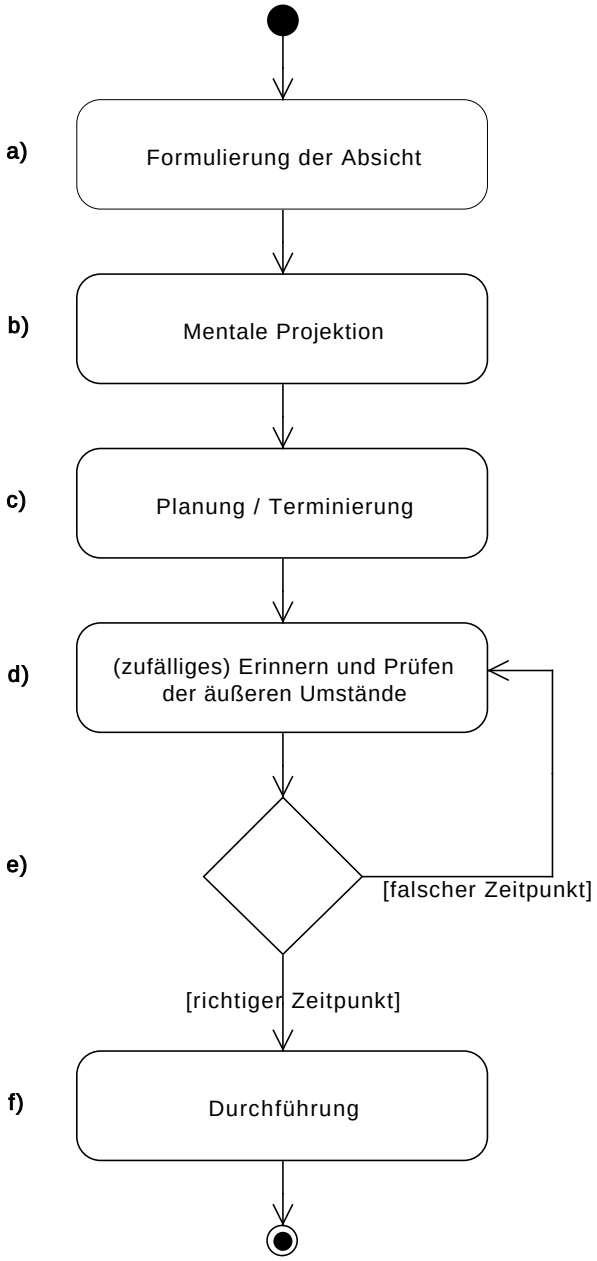

Abbildung 3.1: Darstellung der prospektiven Erinnerung

die bereits in einem Kalender festgehalten wurden, können Abhängigkeiten erkannt werden, die bei der Entstehung neuer Anliegen einfließen (ausgelöst durch Termin X erinnert sich die Person an Termin Y). Die Encodierung des Anliegens erfolgt durch das Eintragen in den entsprechenden Kalender, ebenso die Terminierung durch Eintragen einer Zeit.

3.1.2 Kontext und situativer Hintergrund von Events

Jeder Termin sowie jedes Event im generellen Sinn sind einzigartig, da die Zusammenstellung der Faktoren, die ein Event ausmachen, nie völlig identisch ist (Casati u. Varzi (2007)). Neben den statischen Kontextelementen wie Zeit und Ort (ggf. auch Anlass und Teilnehmer), die bei der Terminplanung überwiegend im Vorfeld festgelegt werden, spielen dynamische Faktoren während des Termins eine große Rolle. So können sich Umwelteinflüsse wie z.B. das Wetter oder die persönliche Verfassung auf den Verlauf des Events auswirken. Des Weiteren ist ein Event immer eingebettet in einen größeren Rahmen, der das Ziel vorgibt (z.B. eine Person anzurufen, um eine Konferenz zu planen).

Einen möglichen Ansatz, den Kontext eines Events zu erfassen bietet die *5W1H* Methode, da sich die einzelnen Komponenten dieser Methode auf die Elemente eines Termins abbilden lassen: Eine Anzahl Teilnehmer (*Wer?*) trifft sich zu einer bestimmten Zeit (*Wann?*) an einem festgelegten Ort (*Wo?*), um ein vorgegebenes Thema zu behandeln (*Was?*). Der Grund des Termins ist das Erreichen eines höheren Ziels (*Warum?*). Der Weg zur Realisierung dieses Ziels ist der Gegenstand des Termins (*Wie?*). Jeder Teilaspekt dieser Methode stellt einen situativen „Schnappschuss“ dar. Aus der Gesamtheit der Teilaspekte lassen sich somit Rückschlüsse auf die Situation, in der ein Event stattfindet, schließen (Seiie u. Woontack (2005)).

3.1.3 Kategorisierung von Events

Die Klassifizierung und Kategorisierung von Objekten gehört zu den zentralen kognitiven Kompetenzen des Menschen. Einen Sachverhalt kategorisieren zu können bedeutet, ihn von anderen Dingen abgrenzbar zu machen (Waldmann (2008)). Für den Menschen ist es somit möglich, das Wissen, das er bereits über die betreffende Kategorie verfügt, auf diesen Sachverhalt anzuwenden.
Kategorien unterstützen den Menschen im Lernprozess, indem sie ihm „ermöglichen, neue Erfahrungen mit bereits vorhandenem Wissen in Kontakt zu bringen“ (Waldmann (2008)).

Im Bezug auf Kalender haben Kategorien zwei Hauptfunktionen. Die erste besteht in der Abgrenzung unterschiedlicher sozialer Welten. Ein klassisches Beispiel hierfür ist die Unterscheidung von *privat* und *beruflich* kategorisierten Events. Der zweite Faktor, der sich an

ersterer Intention anlehnt, ist die Spezifizierung der Erwartungshaltung an einen Termin; von einer Veranstaltung, die als *privat* eingestuft ist, wird eine Person etwas anderes erwarten, als von einer Veranstaltung, die als *beruflich* gekennzeichnet ist.

3.1.4 Die Benutzung von Kalendern

Kalender begleiten den Menschen bereits seit der Antike bei der Erfassung und Verwaltung von wichtigen Daten zu einem bestimmten Zeitpunkt. Die Formulierung von Terminen zählt zu den Phasen des *Personal Information Management* (im Folgenden PIM), das sich mit der Beschaffung, Organisation und Verwaltung von Informationen zur späteren Verwendung beschäftigt. Dabei zählt das Erstellen von Kalendereinträgen zur Verwaltungsphase. Diese verfolgt das Ziel, zur richtigen Zeit die richtige Information präsent zu haben (Jones (2007)). Die im Kalender festgehaltenen Daten sind das Bindeglied (*Mapping*) zwischen dem Bedürfnis (*Need*) und der dazu benötigten *Information*. Ein Beispiel hierzu ist die Notwendigkeit, eine Person zu einer bestimmten Uhrzeit anrufen zu müssen. Zur Erfüllung des Bedürfnisses (des Anrufs) ist eine Information über die Telefonnummer der Person sowie über die Uhrzeit des Anrufs erforderlich. Beides wird im Kalender hinterlegt.

Grundsätzlich unterscheidet sich die Benutzung von Kalendern im privaten Umfeld von der im beruflichen Kontext, da im jeweiligen Fall unterschiedliche Anforderungen gelten (Crabtree u. a. (2003)). In der Literatur überwiegt der Anteil an Studien über den Gebrauch von Kalendern und Groupware-Systemen im Firmenkontext, da diese eng miteinander verknüpft sind. Die Aussagen über die Nutzung von Kalendern in den folgenden Abschnitten beziehen sich daher auf die Nutzung im beruflichen Umfeld.

Die Hauptfunktion eines Kalenders besteht in der Planung und Koordination von Events, entweder für den Benutzer selbst, oder im Falle einer gemeinschaftlichen Nutzung, zur Koordination der ganzen Gruppe. Weitere Ziele werden von Palen (1999) beschrieben:

Zeitliche Orientierung: Kalender bieten nicht nur die Möglichkeit zur absoluten zeitlichen Festlegung von Terminen, sondern auch eine relative Positionierung von Events zueinander, wodurch Abhängigkeiten besser erkennbar werden.

Verfolgung von Ereignissen in der Gegenwart: Dies geschieht vor allem zur späteren Verwendung der aufgezeichneten Informationen in der Zukunft (z.B. das Aufzeichnen von medizinischen Daten). Der Kalender erfüllt somit eine tagebuchähnliche Funktion.

Erinnerung: Eine Funktionalität, die vor allem bei elektronischen Kalendern an großer Bedeutung gewonnen hat. Hierbei kann sowohl eine retrospektive als auch prospektive Erinnerung an bestimmte Events erfolgen.

Festhalten von Notizen: Informationen und Anmerkungen, beispielsweise aus Meetings, werden als Terminobjekt festgehalten, um eine spätere Assoziation mit dem genauen Zeitpunkt zu ermöglichen.

Einen weiteren Aspekt bildet die Verwaltung von To-do-Listen, wobei der Kalender zur Fixierung von Informationen über das To-do-Objekt und zur Festlegung von Fristen dient.

PIM in Papier- als auch elektronischer Form bieten in den meisten Fällen eine vordefinierte Maske für Termineinträge mit Eintragungsmöglichkeiten für Ort, Datum und Uhrzeit, Teilnehmer, Betreff und Notizen. Von diesen Feldern werden im Allgemeinen jedoch nur die für Datum, Zeit und den Zweck benutzt. Der Ort spielt vor allem bei sich nicht wiederholenden Terminen eine Rolle[1] (Tungare u. a. (2008)).

3.1.5 Kalendersoftware und Papierkalender im Vergleich

Das Zeitalter elektronischer Kalender begann im Jahr 1979 mit der Patentierung des *Electronic Calendar and Diary* durch Alfred B. Levine (Levine (1979)). Der Stellenwert von Kalendersoftware wurde durch die Einführung von Groupwaresystemen wie z.B. *WordPerfect Library*[2], dem ersten Groupwaresystem aus dem Jahr 1986, interessant.

Das Ziel von Groupwaresystemen ist es, das produktive Zusammenarbeiten einer Gruppe zu steigern und den Beteiligten eine Hilfe im geimeinschaftlichen Arbeitsumfeld zu sein (Beaudouin-Lafon u. a. (1999)). Zum Funktionsumfang solcher Systeme gehört neben Adressbüchern und E-Mail Postfächern auch ein Kalender, der von einem einzelnen Anwender, als auch gemeinschaftlich von einer Arbeitsgruppe, genutzt werden kann.
Neben Groupware-Kalendern existeren weitere Kalenderanwendungen, die vor allem im privaten, aber auch im beruflichen Kontext Verwendung finden. Diese sind zumeist in den Funktionsumfang von Betriebssystemen als Stand-Alone Software oder als Bestandteil von E-Mail Programmen realisiert.
Eine weitere Form des elektronischen Kalenders, der in den letzten Jahren immer weiter an Bedeutung gewonnen hat, ist die Umsetzung als Online-Kalender. Diese sind im WWW erreichbar und bieten den gleichen Funktionsumfang wie ihr Desktop Pendant.
Alle genannten Kalendersysteme bieten die im vorhergehenden Abschnitt angeführten Felder zur Eintragung von Termindetails.

Ein in der Literatur oft diskutiertes Problem ist die Frage, in welchem Verhältnis elektronische Kalender und der „klassische“ Papierkalender heutzutage zueinander stehen. Studien über die Nutzung von Kalendern beschreiben für beide Varianten unterschiedliche Vor- und

[1] Das Problem der nur teilweise ausgefüllten Termine wird bei den Überlegungen zu den Anforderungen an ein späteres Gesamtsystem wieder aufgegriffen
[2] http://en.wikipedia.org/wiki/WordPerfect

Nachteile. So beschreiben Tungare u. a. (2008) vier Vorteile von Papierkalendern gegenüber ihrem elektronischen Pendant:

„Paper Trails“: Abgesagte Termine werden in Papierkalendern einfach durchgestrichen, so bleibt eine Spur erhalten, dass dieser Termin zwar geplant war, aber nicht stattgefunden hat. Ein Event, welches nicht geplant wird, wird auch nicht eingetragen. Im Gegensatz hierzu verschwinden in elektronischen Kalendern gelöschte Termine ohne jede Spur.

Opportunistisches Wiederholen: Wandkalender erlauben bereits durch einen flüchtigen Blick eine Übersicht über vermerkte Termine. Wird ein neuer Termin eingetragen, werden dabei die bereits festgehaltenen Termine überflogen (dieses steht in Verbindung mit der prospektiven Erinnerung).

Annotationen: Ein Papierkalender erlaubt zusätzliche Kennzeichnungen im Text und gestattet eine „freie Gestaltung“ der zur Verfügung stehenden Fläche. So können wichtige Punkte unterstrichen oder umrandet, Symbole als Assoziation für etwas hinzugefügt, oder zusätzliche *Micronotes* (Lin u. a. (2004)) angefügt werden. Diese können, z.B. angehängt an einem gemeinsamen Kalender in Papierform, als Erinnerung dienen, was die jeweilige Person zu diesem Termin beitragen oder vorbereiten muss.

Vorausgefüllte Events: In vielen kommerziell erhältlichen Papierkalendern sind bereits besondere Ereignisse, beispielsweise gesetzliche Feiertage und Ferien, eingetragen. Speziell für Firmen oder Universitäten erstellte Kalender beinhalten zusätzlich Termine für hauseigene Fristen oder Events.

Eine weitere Form der Annotation besteht in der Niederschrift von Events oder Termindetails auf Notizzetteln. Dieses ist häufig dann der Fall, wenn der Anwender seinen elektronischen Kalender nicht zur Verfügung hat oder auf das Eintragen über seinen PIM verzichten möchte. Die Notizen werden dann entweder im elektronischen System nachgetragen oder bei der Papierversion einfach auf der entsprechenden Seite angefügt.

Einige Menschen sehen in Papierkalendern einen „emotionalen Gegenstand“, der über ihr eigenes Leben oder auch das von verstorbenen Angehörigen als eine Art Tagebuch dient. Diese werden somit auch nicht entsorgt, sondern über die Jahre aufbewahrt und teilweise sogar an nachfolgende Generationen weitergegeben (Tungare u. a. (2008)).

Besonders im privaten Umfeld werden Kalender in Papierform bevorzugt. Speziell bei der Koordination von Terminen innerhalb der Familie ist diese Variante vorteilhaft, da sie einfach zu benutzen und für alle Familienmitglieder (auch Kinder) an einer zentralen Stelle zugänglich ist (Neustaedter u. a. (2009)).

Seit der Einführung elektronischer Kalender und den ersten Vergleichsstudien zwischen beiden Varianten (u.a. Kincaid u. a. (1985), Payne (1993)) hat sich der Funktionsumfang von

Kalendersoftware stark weiterentwickelt. Während die ersten Studien noch immense Nachteile gegenüber der klassischen Variante gesehen haben, haben sich diese Systeme bis heute so weit entwickelt, dass sie die Möglicheiten von Papierkalendern weit übertreffen. Als essentielle Vorteile lassen sich folgende Elemente formulieren:

Reminder- und Alarm-Funktionalität: Als größter Vorteil von Kalendersoftware wird die Einführung von Erinnerungs- und Alarmfunktionen beschrieben. Diese können akustisch oder visuell und in verschiedenen zeitlichen Abstufungen erfolgen (Tungare u. a. (2008)):

Unterbrechend: Kurz vor dem Beginn des Termins als Aufforderung, die momentane Tätigkeit zu unterbrechen.

Vorbereitend: Als Erinnerung, dass für den Termin noch etwas vorbereitet werden muss oder zur prospektiven Erinnerung (vgl. 3.1.1).

Langfristig: Monate oder Jahre im Voraus, als Gedankenstütze.

Vernetzung mit anderer Software: Kalenderapplikationen erlauben eine Vernetzung mit anderen Informationsquellen, z.B. einem elektronischen Adressbuch oder E-Mails, mit dem Termin in Verbindung stehenden Dokumenten sowie die Verbindung von Ortskarten mit der Terminlokalität.

Mobile Nutzung: Auf elektronische Kalender kann von verschiedenen Orten und Geräten zugegriffen werden. Die immer stärkere Ausbreitung mobiler PIM ermöglicht einen Zugriff auf den Online-Kalender wie auf einen Papierkalender, den die Person mit sich führen würde.

Übersichtlichkeit: Vordefinierte Felder sorgen für eine klare Struktur des Terminaufbaus. Einträge aus Kalendern für unterschiedliche Zwecke (privat, beruflich, von Dritten, etc.) können durch verschiedene Farben kenntlich gemacht werden.

Wiederverwertbarkeit: Daten können einfacher ausgewertet werden, etwa um den Stundenaufwand für Tätigkeiten abzurechnen oder einen Tätigkeitsbericht für einen größeren Zeitraum zu verfassen.

Die Vernetzung mit Adressbüchern und E-Mail Postfächern, das Teilen von Kalendern mit Anderen sowie Alarmfunktionen machen elektronische Kalendersysteme unverzichtbar. Auf der anderen Seite sind Papierkalender durch ihre Simplizität und die Möglichkeit der „freien“ Gestaltung des Termineintrags unentbehrlich. Jedes System hat seine eigenen Vorteile, daher gibt es bis heute keine Tendenz dahingehend, dass sich eine der beiden Kalendervarianten komplett durchsetzen und die andere verdrängen wird. Vielmehr werden die Synergieeffekte beider genutzt, indem für unterschiedliche Anforderungen das jeweils bessere Modell genutzt wird (Blandford u. Green (2001)).

3.2 Szenarien

In den folgenden Abschnitten soll anhand mehrerer Szenarien ein realistisch konstruiertes Bild der Verwendung einer kontextbezogenen Kalenderanwendung erzeugt werden. Hierzu wird, wie in Mark Weisers Artikel *The computer for the 21st century* (Weiser (1991)) ein Morgen im Leben von Sal beschrieben. Sie verwendet eine kontextbezogene Kalenderanwendung, anhand derer die einzelnen Teilaspekte der Applikation vorgestellt werden.

3.2.1 Übersicht über die Termine des Tages

> *Nach dem Frühstück legt Sal ihr Smartphone auf den Surface[3] Tisch in ihrer Küche, um mit der Planung ihres Arbeitstages zu beginnen. Auf dem Surface wird eine Karte mit Markierungen der Orte, an denen sie heute Termine hat, angezeigt. Neben den markierten Orten wird ihr eine textuelle Kurzbeschreibung der Termine dargestellt. Zum einen ein Meeting um 10 Uhr in ihrer Firma sowie ein Kundengespräch um 14 Uhr in Hamburg-Rahlstedt.*

Dieses Szenario beschreibt die Basisfunktionalität einer Kalenderanwendung, nämlich die einfache Darstellung der Termine des Tages. Diese wird hier zusätzlich durch eine geografische Darstellung in einer Karte angereichert. Wie bei einem „gewöhnlichen“ Kalendereintrag bekommt der Nutzer eine Übersicht zu Zeit, Ort, Betreff und Teilnehmern des jeweiligen Termins dargestellt.

3.2.2 Detailansicht eines Termins

Die folgenden Szenarien beschreiben eine informationelle „Zoom“-Funktionalität für die einzelnen Termine. Der Anwender kann den Fokus bei der Betrachtung eines Events auf die verschiedenen Teilaspekte wie z.B. Ort, Zeit oder Teilnehmer legen. Zur Anreicherung des Informationsgehalts werden Daten aus verschiedenen Informationsquellen hinzugezogen.

Nutzung von Daten aus sozialen Netzwerken

> *Zunächst interessiert Sal das Meeting in ihrer Firma, da es ein Folgetermin ihrer Projektgruppe ist und sie den Vorgängertermin bedingt durch ihren Urlaub verpasst hat. Sie zoomt in das Event hinein und betrachtet einen Zeitstrahl mit*

[3] http://www.microsoft.com/surface/en/us/default.aspx

den Terminen der vorherigen Treffen zum gleichen Thema. Neben dem heutigen Termin sind noch vier weitere Treffen bis zum Projektabschluss geplant, außerdem wird ihr das Protokoll mit den getroffenen Beschlüssen des letzten Meetings angezeigt. Sie legt den Fokus auf die Teilnehmer des Meetings. Eine Person ist ihr unbekannt, daher klickt sie auf seinen Namen, um sich weitere Informationen anzeigen zu lassen. Der Mitarbeiter hat seine Tätigkeit in ihrer Firma begonnen, während sie abwesend war. Sie meint sich jetzt doch zu erinnern, den Namen schon einmal gehört zu haben. Um an weitere Informationen über den Mitarbeiter zu gelangen, zoomt sie noch eine Stufe weiter in das Profil und sieht einen Graphen, der ihn mit ihren Bürokollegen und Freunden Sören und Alex verbindet. Alle drei haben an der gleichen Hochschule studiert sowie den gleichen Studentenjob während ihres Studiums ausgeführt.

Anhand der teilnehmenden Personen wird beschrieben, wie Daten aus sozialen Netzwerken benutzt werden können, um die Verbindungen der involvierten Personen untereinander zu verdeutlichen und fehlendes Wissen über diese zu vervollständigen.

Nutzung von Ortsdaten und Präferenzen

Jetzt interessiert Sal der zweite Termin, denn sie hat den Kunden bislang nur bei der Kickoff-Veranstaltung in ihrer Firma getroffen. Normalerweise präferiert sie es, alle Strecken innerhalb Hamburgs mit dem Fahrrad zurückzulegen. Jedoch ist die Distanz nach Rahlstedt größer als zehn Kilometer und für den heutigen Tag ist Regen vorhergesagt. Der Kalender blendet ihr einige mögliche Bahnverbindungen ein. Die Abfahrtszeit ist dabei so berechnet, dass sie noch die Möglichkeit bekommt, ihre Unterlagen vor Ort zu sortieren und ihre Präsentation vorzubereiten. Allerdings ist die Zeit zwischen dem vorausgesagten Ende des ersten Meetings und der Abfahrt zum zweiten zu knapp, um noch in der Kantine Mittag essen zu können. Der Kalender weist sie daher darauf hin, lieber etwas Essen mitzunehmen.

In diesem Szenario werden zusätzlich Daten über Zeiträume, Streckenplanung und das Wetter hinzugezogen. Aus den Informationen über die Distanz zweier zu besuchender Orte sowie den Wetter- und Zeitinformationen macht die Software automatisiert Vorschläge zu einer sinnvolleren Planung der Wege. Ein weiterer Aspekt, der in dieses Szenario hineinspielt, ist die Berücksichtigung von Präferenzen. Die Anwendung hat die Information, dass Sal es bevorzugt, mit dem Fahrrad zu fahren und mittags lieber in der firmeninternen Kantine isst. Können diese „Rituale“ nicht eingehalten werden, wird sie von der Software informiert.

3.2.3 To-do-Listen und Reminder

Reminder sind eine der zentralen Funktionen elektronischer Kalender. Sie können sowohl für Events als auch für selbst definierte To-do-Objekte verwendet werden. Die folgenden Szenarien beschreiben die Verwendung solcher Reminder für unterschiedliche Anwendungsfälle.

To-do-Listen

> *Sal hat es vor ihrem Urlaub nicht mehr geschafft, ihren Blazer in die Reinigung zu bringen. Als sie sich Termine des morgigen Tages anzeigen lässt, bemerkt sie, dass sie am morgigen Tag eine Besprechung mit einem möglichen neuen Kunden hat. Aus den ihr dargestellten Informationen zur Firma des Kunden geht hervor, dass es sich um ein traditionsbewusstes Familienunternehmen handelt und daher formelle Kleidung für das Meeting geboten ist. Sie öffnet ein Feld, um ein neues To-do-Objekt anzulegen, trägt „Reinigung Blazer“ ein und wählt die höchste Prioritätsstufe aus. Kurz darauf blendet die Anwendung ein Popup-Feld mit der Adresse einer Reinigung ein, die auf ihrem Weg zur Arbeit liegt und einen Express-Service anbietet.*

Dieses Szenario beschreibt die Erstellung eines To-do-Objektes. Hierbei genügt die einfache Angabe eines Betreffs und einer Priorität für den zeitlichen Rahmen zur Erledigung der Aufgabe.
Durch die Analyse des To-do-Betreffs können über Ortsinformationsdienste die entsprechenden Geschäfte o.ä. zur Erledigung der Aufgabe bestimmt oder eventuell direkt beauftragt werden. Zusätzlich zum Ort kann die Anwendung abhängig von der Priorität einen passenden Zeitraum zwischen Terminen suchen, um für die Durchführung keine zusätzlichen Umwege zu erzeugen.

Reminder

> *Sal zoomt zurück auf die Übersichtskarte und bemerkt ein rotes Warnfeld mit dem Hinweis, ihren Firmenlaptop nicht zu vergessen, den sie während ihres Urlaubs zu Hause benutzt hat, da im Verlauf des Tages noch eine Präsentation ansteht für die sie ihr Notebook benötigt.*

Wie bereits im vorhergehenden Szenario erkennt die Anwendung über den Kontext eines Termins, in diesem Fall einer Präsentation mit dem eigenen Rechner, die Notwendigkeit, den Anwender auf bestimmte Dinge hinzuweisen. Dass die Applikation Informationen darüber

hat, dass der beschriebene Laptop in Sals Wohnung ist, stellt die Möglichkeit der Verbindung mit anderen kontextsensitiven Anwendungen dar, wie etwa die Nutzung von räumlicher Suche in einer intelligenten Wohnumgebung.

3.2.4 Automatisierung von Aufgaben

Die Erledigung von „lästigen“ Aufgaben durch andere, beispielsweise durch einen persönlichen Assistenten, ist eine Funktionalität, die ebenfalls durch eine Tagesplaneranwendung erfüllt werden kann. Die folgenden zwei Szenarien beschreiben diese Funktion.

Erkennen von Vorhaben

> *Sal glaubt sich zu erinnern, dass jemand aus ihrem Bekanntenkreis zuletzt nach Hamburg-Rahlstedt umgezogen ist. Sie wechselt auf das Ortsfeld des Kalenders und wählt die Option „Meine Freunde anzeigen“ aus. Anhand der neu eingeblendeten Markierungen auf der Karte sieht sie, dass sogar zwei ihrer alten Schulfreunde mittlerweile dort wohnen. Sie verschickt über den Kalender an beide eine Nachricht mit einer Einladung für ein gemeinsames Essen um 17 Uhr. Die Anwendung erkennt Sals Vorhaben und zeigt ein Fenster mit drei möglichen Vorschlägen für eine Lokalität, sortiert nach Kundenbewertungen. Sie wählt „italienisch“, das Kalendersystem schickt daraufhin eine Reservierungsanfrage an das Restaurant und blendet den Ort auf der Übersichtskarte ein.*

Dieses Szenario beschreibt die automatisierte Erstellung von Terminen mit den damit verbundenen Aufgaben. Aus der Kommunikation des Anwenders mit anderen Personen erkennt die Anwendung die Absicht und stellt ähnlich wie bei den To-do-Listen mögliche Vorschläge bereit. Aus den verschiedenen Informationen, die der Anwender über den Termin bereitstellt (in diesem Fall Teilnehmer und die Absicht, gemeinsam essen zu gehen), generiert die Applikation ein neues Event und kümmert sich um die damit verbundenen Aufgaben (z.B. das Reservieren eines Tisches).

Automatisierung bei der Termineintragung

> *Sals Kinder kommen in die Küche, sie würden gerne einen Film im Kino sehen der am heutigen Abend anläuft. Sal fügt ihrem Kalender einen neuen Termin hinzu und gibt als Teilnehmer sie und ihre beiden Kinder an. Sie trägt dabei keine explizite Uhrzeit ein, sondern setzt das Terminfeld in die Abendstunden und trägt als Betreff „Kino“ und den Titel des Films ein. Ein Popup öffnet sich und*

Sal wählt das entsprechende Kino. Da Sals Kinder noch relativ jung sind hebt die Applikation die um 18:30 Uhr beginnende Vorstellung hervor. Die anschließende Reservierung der Karten erfolgt anhand der eingetragenen Teilnehmer: Ein Erwachsener und zwei Kinder.

In diesem Beispiel erfolgt ein konkreter Terminvorschlag durch die Anwendung. Durch die Informationen aus Betreff und Teilnehmern sowie einem Zeitfenster, in dem der Termin stattfinden soll, wird eine mögliche Zeit festgemacht. Wie schon im zuvor vorgestellten Szenario übernimmt die Anwendung mit der Reservierung die Aufgaben eines persönlichen Assistenten.

3.2.5 Mobile Nutzung

Die vorhergehenden Szenarien gehen ausschließlich von einem lokalen Betrieb der Kalenderanwendung aus. Mögliche Funktionen bei der Nutzung einer mobilen Variante werden im folgenden Szenario beschrieben.

Sal ist auf dem Weg von ihrem Büro zu ihrem zweiten Termin in Hamburg-Rahlstedt. Das schlechte Wetter, das ihr durch den Kalender bereits morgens angekündigt wurde, hat sich in einen orkanartigen Sturm verwandelt und den Zugverkehr kurzzeitig zum Erliegen gebracht. Das Smartphone in ihrer Jackentasche vibriert. Ihr Kalender meldet, dass aufgrund der Entfernung zum Ziel und der Zeit, die noch bis zum Beginn des Termins bleibt, ein pünktliches Erscheinen unwahrscheinlich wird. Die Anwendung blendet ihr eine Telefonnummer und eine E-Mail Adresse ein, beide von ihrem Gesprächspartner. Zusätzlich wird ihr eine neue mögliche Ankunftszeit angezeigt.

In diesem Szenario bedient sich die Kalenderanwendung Smartphone-spezifischer Möglichkeiten, etwa der Ortsbestimmung durch GPS oder Triangulation, um auf Störungen im Tagesablauf hinzuweisen. Kann, wie beschrieben, ein Termin nicht mehr pünktlich wahrgenommen werden, stellt die Software mögliche Kommunikationswege zur Verfügung, über die Terminpartner erreicht werden können.

3.3 Vergleichbare Ansätze

In den folgenden Abschnitten werden fünf Arbeiten vorgestellt, die sich mit Kalenderanwendungen befassen, deren Funktionsumfang über den bekannter, am Markt präsenter, Applikationen hinausgeht.
Die erste Arbeit befasst sich mit einem Kalender, der den Anwender unter Einbeziehung von

Alltagswissen bei der Terminplanung unterstützen soll. Im Anschluss wird eine Feldstudie im Zusammenhang mit einer Anwendung präsentiert, deren Ziel es ist, Aussagen darüber zu treffen, ob ein Termin als solches wirklich stattgefunden hat, um so mit dem Kalender eine qualitativ höherwertige Quelle für Sensordaten anbieten zu können. Weiterhin wird eine Anwendung für mobile Endgeräte vorgestellt, die mit Hilfe von Orts- und Zeitinformationen den Anwender in seiner Tagesplanung unterstützen soll. Die folgende Arbeit befasst sich mit der Bestimmung von Vorhaben und Zielen, die ein Termin verfolgt. Abschließend wird eine weitere Anwendung für mobile Geräte vorgestellt, die unter Berücksichtigung von Kontakten, Textnachrichten und weiterer PIM-spezifischer Komponenten den Kontext für eine mobile Verwendung aufbereitet.
Die Erkenntnisse dieser Ansätze sollen in den Anforderungskatalog der in dieser Arbeit vorgestellten Applikation einfließen.

3.3.1 A calendar with common sense

Die Arbeit von Mueller (2000) befasst sich mit dem Problem der Kontexterfassung durch Software im Bezug auf die Inhalte eines Termineintrags. Durch falsche Kontextinterpretation seitens der Software kann der Anwender falsche Information dargestellt bekommen, wodurch es zu Fehlplanungen oder unangemessenen Reaktionen kommen kann.

Die vorgestellte Applikation *SensiCal* stellt einen möglichen Lösungsansatz dar, aus den einzelnen Informationen, die von den Terminbestandteilen abgeleitet werden können, auf den Kontext eines Termins zu schließen, um so auf mögliche Probleme reagieren zu können.
Die Basis von *SensiCal* bildet *ThoughtTreasure*[4] (Mueller (2003)), eine Datenbank bestehend aus Informationen, die auf Grundlage von Aussagen und deren Schlussfolgerungen eine *Knowledgebase* bilden. Aufgebaut sind diese als Tripel bestehend aus Subjekt, Prädikat und Objekt.
Einige Beispiele sind:

- People have fingernails.
- The evening extends from about 5 pm to 9 pm.
- A hotel room has a bed, night table, minibar, ...

Neben *ThoughtTreasure* wird programmintern eine Wissensdatenbank mit durch den Anwender verwalteten Fakten verwendet, beispielsweise über die Kontakte des Adressbuchs. Die Anwendung läuft als Erweiterung zu bestehenden Hostapplikationen.
Wird in dieser Hostapplikation ein Termin eingetragen, erfolgt eine Analyse des Termins durch die Software in drei Schritten:

[4] Das Projekt ist mittlerweile inaktiv.

1. **Extrahieren der relevanten Informationen:** Diese können der Typ des Termins, die Teilnehmer, der Ort (Name, Typ, zugehörige Koordinaten) und weitere, zusätzliche Informationen, die auf Basis der in *ThoughtTreasure* abgelegten Knowledgebase aufbereitet werden, sein.
2. **Ergänzen von fehlenden Informationen:** Nicht vorhandene Informationen werden anhand anderer bekannter Informationselemente vervollständigt, z.B. die Dauer eines Termins, die anhand der Informationen über die Aktivität ermittelt werden kann.
3. **Untersuchen des Termins auf mögliche Probleme:** Dieser Aspekt spiegelt den eigentlichen Verwendungszweck von *SensiCal* wider, da an dieser Stelle auf die eigentlichen Kontextelemente reagiert wird. Zu jedem Teilaspekt eines Kalendereintrags wird anhand von Aussagen des „gesunden Menschenverstands“ auf mögliche Probleme hingewiesen. Für die Prüfung werden Fragestellungen wie z.B. „Ist die Reisezeit realistisch geplant?“, „Ist der Ort verfügbar (Öffnungszeiten)?“, „Ist der Startzeitpunkt des Termins dem Anlass angemessen?“ eingesetzt.

3.3.2 The Calendar as a Sensor: Analysis and Improvement Using Data Fusion with Social Networks and Location

Im heutigen Arbeitsalltag bilden gemeinsam genutzte Kalendersysteme einen de facto Standard zur Kommunikation und Koordination von Terminen. Diese umfassende und verbreitete Nutzung solcher Systeme erhöht gleichzeitig den Grad an verfügbaren Informationen über Termine sowie deren Teilnehmer und bietet anderen Systemen die Möglichkeit, den Kalender als Sensor für weitere Anwendungen zu benutzen.
Die Arbeit von Lovett u. a. (2010) beschäftigt sich mit dem Problem der fehlenden Genauigkeit der im System abgebildeten Events. So kann ein Termin trotz Eintragung nicht stattfinden, zusätzliche, spontane Meetings können sich ohne vorherige Eintragung ereignen, die Teilnehmer können sich ändern o.ä.. Einen möglichen Ansatz zur Steigerung der Genauigkeit bietet die Einbeziehung zusätzlicher Informationen über Ort und Daten aus sozialen Netzen. Die Arbeit stellt hierzu zwei Prozesse vor, zum einen den *Data Fusion Process*, der Daten aus bestehenden Kalenderinformationen mit Ortsinformationen und Daten aus sozialen Netzen[5] kombiniert sowie einen *Event Management Process*, der aus den aggregierten Daten Termine erstellt, aktualisiert oder löscht.
Ausgangspunkt der vorgestellten Arbeit ist eine Feldstudie, in der Mitarbeiter eines Unternehmens ihre *Microsoft Outlook*[6] Kalender- und E-Maildaten über einen Zeitraum von sechs Wochen zur Verfügung gestellt haben. Zeitgleich wurden alle Mitarbeiter mit Bluetooth-fähigen

[5] In diesem Fall E-Mail Daten
[6] http://office.microsoft.com/de-de/outlook/

Smartphones ausgestattet, die eine Lokalisierung innerhalb des Unternehmens zum Zeitpunkt ihrer Termine ermöglichten.

Im Ergebnis der Feldstudie spiegelt sich die mangelnde Qualität des Kalenders als Sensor wider. Von knapp 500 eingetragenen Events haben nur 8% den Status eines *Genuine Event*, wären also wiederverwertbar als Sensordaten für andere Anwendungen.
Also Konsequenz kommen die Autoren zu folgender Hypothese:

> *„The shared calendar is a potentially valuable source of context but its usefulness as a virtual sensor is limited if its content is not a good representation of reality. The similarity between calendar event data and reality can be improved through fusion of the calendar with other sources of context.“*
>
> - Lovett u. a. (2010)

Um die Gültigkeit dieser Hypothese zu untersuchen, führen die Autoren einen *Data Fusion Process* ein, dessen Arbeitsweise in Abb. 3.2 dargestellt wird: Ein eingehender Satz Daten[7] wird darauf untersucht, ob sich Teilnehmer an der gleichen lokalen Position im Unternehmen befinden.

Sollte dies der Fall sein, wird anhand von Sensordaten aus E-Mails und dem Kalender geprüft, ob eine soziale Interaktion zwischen diesen Personen stattgefunden hat. Ist das der Fall wird geprüft, ob bereits ein Termin eingetragen ist. Sind alle drei genannten *Enabler* zutreffend, wird ein Auftrag zur Erstellung eines Events an den *Event Management Process* übergeben, welcher ein Event einträgt, falls es noch nicht existiert, es aktualisiert, falls es bereits vorhanden ist und beendet, wenn über einen Zeitraum *t* keine Updates vorkommen.

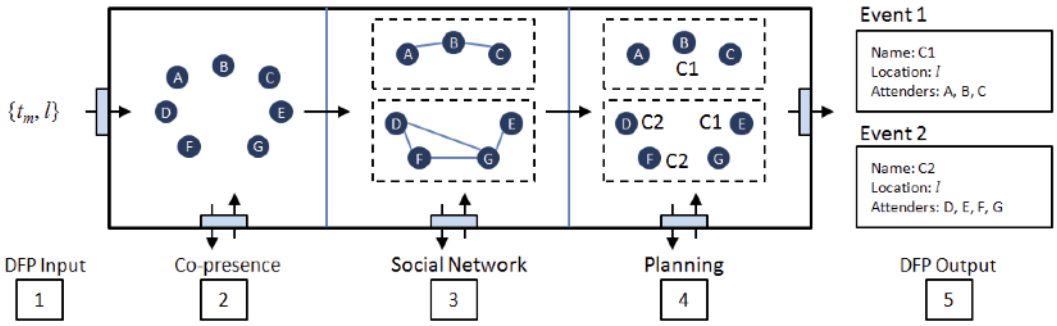

Abbildung 3.2: Arbeitsweise des *Data Fusion Process* (aus Lovett u. a. (2010))

Im Ergebnis zeigt sich eine deutliche Verbesserung der Qualität der im Kalendersystem abgebildeten *Genuine Events* durch Reduzierung der „falschen“ Events auf ca. 16% des Ursprungswertes.

[7] Hier werden die Daten der Feldstudie benutzt, das Stattfinden eines Termins soll in Echtzeit ermittelt werden

3.3.3 A Smart Calendar Application for Mobile Environments

In der Arbeit von Gkekas u. a. (2007) wird eine Kalenderanwendung für mobile Endgeräte vorgestellt, die neben bekannter Funktionen von Kalendersystemen weitere Features beinhaltet, die den Anwender proaktiv in seiner Tagesplanung unterstützen sollen. Der Fokus liegt hierbei auf der zusätzlichen Nutzung von Informationen über die Zeit und den Ort, an dem sich der Nutzer zu einem bestimmten Zeitpunkt befindet, bzw. ein Termin stattfindet. Unter Berücksichtigung persönlicher Voreinstellungen werden drei Ziele verfolgt:

1. Organisation des Terminplans anhand von freien und blockierten Zeitslots.
2. Empfehlungen von Aktivitäten, die zu bestimmten Zeiten wahrgenommen werden können.
3. Unterstützung in der zeitgerechten Bearbeitung von Aufgaben.

Durch Funktionen wie dem automatisierten Taggen von Multimediainhalten soll das System dem Anwender störende Aufgaben abnehmen. Die vorgestellte Applikation basiert auf der von Demiris u. a. (2005) entwickelten Plattform *INTGUIDE*, die ubiquitäre Services für personalisierte und ortsbasierte Informationen unter Einbeziehung von Werbung zur Verfügung stellt. Neben den durch den Anwender explizit angegebenen Voreinstellungen nutzt die Software Lernalgorithmen, um Regeln über die Vorlieben des Benutzers zu erstellen und die Granularität der vorhandenen Informationen zu verfeinern.

Bei der Realisierung wird eine Baumstruktur zur Anordnung der einzelnen Komponenten gewählt, um so eine bessere Modularisierung zu ermöglichen. Weitere Anforderungen an die Architektur sind u.a. Skalierbarkeit, Sicherheit, offene Schnittstellen, um Informationen durch Dritte zu ermöglichen sowie eine Client-Server Architektur mit Thin-Client, um die Rechenlast auf den mobilen Endgeräten gering zu halten. Welchen Grad an Informationen das zentrale Serversystem über die im Kalender eingetragenen Termine erhält, liegt beim Anwender.

3.3.4 EventMinder: A Personal Calendar Assistant That Understands Events

Kalendersoftware fehlt im Gegensatz zu persönlichen Assistenten die Möglichkeit des Verstehens von Termininhalten und Zielen. Gründe hierfür sind das Fehlen von Alltagswissen, Zielen und persönlichen Vorlieben. In Smith (2007) wird ein Ansatz vorgestellt, der dieses Problem behebt. Hierzu werden zwei Modelle, ROMULUS und JULIUS präsentiert, die in Kombination als EVENTMINDER die Ziele des Anwenders erkennen, mögliche Alternativen liefern und nützliche Zusatzinformationen bereitstellen sollen.

ROMULUS legt den Fokus auf die Extraktion der relevanten Daten des Beschreibungstextes sowie deren Übertragung auf ein generisches Event Modell. Hierzu bedient sich das Modell des *Semantic Role Labeling*. Die so gewonnenen Daten können dadurch den entsprechenden Feldern Teilnehmer, Ort, Zeit, etc. zugeordnet werden. Als besonderes Problem hierbei wird die Unterspezifizierung von Termininhalten angesprochen. Die Ursache besteht nach Meinung des Autors darin, dass alltägliches und als allgemein bekannt vorausgesetztes Wissen keine genauere Formulierung findet.
Um fehlende Inhalte zu ergänzen, stützt sich das Modell auf ähnliche Events, die der Benutzer bereits abgeschlossen hat. Zusätzlich werden fix definierte Fragen nach dem *5W1H* Prinzip auf die Terminbeschreibung angewendet, um so die richtigen Slots der einzelnen Komponenten zuweisen zu können.
Das zweite Modell, JULIUS, konzentriert sich auf die Extrahierung des Vorhabens, das ein Event verfolgt. Aus diesem Vorhaben wird auf das Ziel des Termins geschlossen. Hierzu werden Bibliotheken mit vordefinierten Absichten und Zielen hinzugezogen.

3.3.5 Exploiting Context for Mobile User Experience

In Lee (2010) wird eine Kalenderapplikation für Smartphones vorgestellt, die das Ziel verfolgt, dem Anwender neben bekannten Kalenderfunktionalitäten zusätzliche Informationen zur Verfügung zu stellen, die eine Arbeit im mobilen Kontext unterstützen sollen. Die Notwendigkeit hierfür wird in der immer stärkeren Verbreitung mobiler Endgeräte gesehen, die zwar über genügend Rechenleistung verfügen, um Mobile Computing zu ermöglichen, deren Formfaktor und Eingabemöglichkeiten jedoch eine Nutzung wie bei klassischen Desktop PCs nur bedingt zulassen.
Die Kombination aus Personen, Ort, Datum und Zeit, Aktivität und Thema eines Events stellen als *mini-context* die Grundlage für eine mobile, kontextsensitive Kalenderanwendung dar. Zusätzliche Daten werden durch Komponenten, die ein Smartphone von sich aus liefern kann, etwa Adressbucheinträge, Positionsdaten sowie Textnachrichten, bereitgestellt. Durch Einbeziehung weiterer Daten aus Webdiensten wird ein Daten-Mashup erzeugt, das dem Anwender angepasst an den Formfaktor des jeweiligen Gerätes dargestellt wird.
Als besondere Schwierigkeit bei der Datenaufbereitung wird der *menschliche Faktor* bei der Terminerstellung gesehen. So ist es für eine Geocoding-Anwendung beispielsweise nicht möglich, den richtigen Ort zu ermitteln, wenn statt einer kompletten Adresse nur die Bezeichnung (beispielsweise die eines Konferenzraumes) einer Lokalität angegeben wird. Als möglicher Lösungsansatz wird GPS Tagging zur Zeit des Events vorgestellt. Die Sicherung der Koordinaten zusammen mit der Bezeichnung des Ortes erfolgt in einer Datenbank, die für alle User offen ist. Ein weiteres Problem besteht laut Meinung des Autors in den freien Textfeldern (Betreffzeile und Beschreibungstext) sowie Texten aus externen E-Mails und

Textnachrichten, die in ihrer Formulierung nicht eindeutig sind. Als Lösungsansatz wird die Einbeziehung von *Natural Language Processing* angesprochen.

3.3.6 Zusammenfassung

Die vorgestellten Arbeiten verfolgen unterschiedliche Ansätze zur Erweiterung von Kalenderprogrammen durch zusätzliche, meistens kontextbasierte Features. Jede Arbeit bietet für sich interessante Teilaspekte, die bei der Entwicklung einer eigenen Software berücksichtigt werden können. Ein umfassender Ansatz, so wie er in dieser Arbeit realisiert werden soll, findet sich dabei jedoch nicht. Vor allem die Tagesplanung bleibt durch fast alle Ansätze unberücksichtigt.
So verfolgt die Arbeit von Mueller (2000) den Ansatz, die Kalendersoftware möglichst „menschlich“ auf etwaige Probleme reagieren zu lassen. Dabei wird jedoch nicht auf den genauen Kontext des Termins Rücksicht genommen, sondern nur auf eine Datenbank mit vordefinierten Fakten zugegriffen, aus der durch Erkennen bestimmter Schlüsselwörter eine passende „Reaktion“ herausgesucht wird. Diese Datenbank und das Kalenderprojekt existieren zum heutigen Zeitpunkt nicht mehr.
Lovett u. a. (2010) verfolgen in 3.3.2 einen Ansatz, der mit kontextberücksichtigender Tagesplanung nicht direkt in Verbindung steht. Vielmehr beschäftigen sie sich mit der Steigerung der Qualität der im Kalender vorhandenen Daten für eine spätere Weiterverwendung. Die vorgestellte Studie, welche nur in einem Gebäude per Bluetooth Lokalisierung stattgefunden hat, wäre durchaus auch auf größere Gebiete übertragbar. Dazu müsste eine Ortsbestimmung per GPS oder WLAN Positionierung erfolgen. Die beschriebene Qualitätssteigerung der Termindaten hätte beispielsweise auf die in 3.3.4 beschriebene Verwertung zurückliegender Eventdaten positive Synergieeffekte.
Die in 3.3.3 vorgestellte Anwendung bedient sich einiger Kontextelemente, die durch mobile Endgeräte zur Verfügung gestellt werden können, in diesem Fall Ort und Zeit. Interessant hierbei ist das proaktive Handeln der Applikation, welches dem Anwender „lästige“ Aufgaben abnehmen soll. Ausserdem kann die Effektivität des Tagesablaufs durch bessere Organisation von Zeitslots zur Bearbeitung von To-do-Objekten gesteigert werden. Die Anwendung kann aus vordefinierten Einstellungen und Lernalgorithmen bessere Ergebnisse erzielen. Der Kontext von Terminen wird jedoch nicht berücksichtigt.
EventMinder von Smith (2007) setzt sich vor allem mit der Problematik des Verstehens von Terminen durch Software auseinander. Die beiden vorgestellen Modelle extrahieren unter Verwendung von *Semantic Role Labeling*, einer Technik der maschinellen Sprachverarbeitung, relevante Teildaten aus Kalendereinträgen (Beschreibungstexten) und versuchen so auf Vorhaben und Ziele eines Termins zu schließen. In puncto Terminkontext ist dieser Ansatz sehr ähnlich zu dem der hier zu entwickelnden Software, allerdings bleibt hierbei die Tagesplanung komplett unbeachtet.

Eine weitere Anwendung für mobile Endgeräte wird abschließend in 3.3.5 vorgestellt. Wie bereits in 3.3.3 werden auch hier die Vorteile von Smartphones genutzt, auf einem Gerät genaue Ortsdaten, Zeit und Daten sozialer Netze nutzen zu können, um daraus ein Mashup für die mobile Nutzung zu erstellen. Die Autoren betrachten viele der kritischen Faktoren, die bei der Entwicklung einer kontextbasierten Kalenderanwendung beachtet werden müssen. Hierbei heben sie besonders das Problem des „menschlichen Faktors“ hervor. Jedoch fehlt, wie bereits bei den anderen Ansätzen, die Einbeziehung der Tagesplanung des Anwenders.

3.4 Anforderungsanalyse

Die in 3.2 vorgestellten Szenarien beschreiben einige der Anforderungen, die an einen kontextberücksichtigenden Tagesplaner gestellt werden. Im Folgenden wird auf Basis dieser Szenarien sowie inspiriert durch die Ansätze der in 3.3 präsentierten vergleichbaren Arbeiten, ein Anforderungskatalog an die zu entwickelnde Applikation erstellt.

3.4.1 Systemidee

Die zu entwickelnde Anwendung soll, vergleichbar mit einem persönlichen Assistenten, den Anwender proaktiv in seiner Tagesplanung unterstützen und ihm einfache organisatorische Aufgaben abnehmen. Die Applikation ersetzt dabei nicht die bestehende Kalender-Infrastruktur des Anwenders, stattdessen ergänzt sie den bestehenden Funktionsumfang als zusätzliche, zum Host synchronisierte, Anwendung.

Für die Anwendung ist der Tagesablauf des Anwenders durch elektronische Kalendereinträge vorgegeben. Diese liegen in der externen Kalender-Hostapplikation vor. Eine grundsätzliche Anforderung an das System ist die Darstellung der einzelnen Termine des Tages durch die Basisinformationen über Teilnehmer, Ort, Zeit und Betreff. Neben der einfachen Darstellung soll die Software die Möglichkeit bieten, diese Termine zu editieren und die Änderungen an die Hostanwendung zu übermitteln.

Die eigentliche Hauptfunktion der Anwendung bildet die kontextuelle Erweiterung der extrahierten, statischen Eventdaten durch geeignete Zusatzinformationen. Hierdurch findet eine Ergänzung der einzelnen Teilaspekte, die einen Termin ausmachen, statt. Es sind also zusätzliche Informationen zu Personen, Ort, Zeit, Anlass und eventuell verknüpften Dokumenten des jeweiligen Events. Anhand dieser Zusatzinformationen soll es der Anwendung ermöglicht werden, Rückschlüsse auf den Kontext im Tagesablauf und die jeweilige Situation, in der ein Termin steht (s. Kap. 3.1.2), zu ziehen.
Zusätzlich sollen diese Informationen dazu dienen, dem Anwender ein genaueres Bild darüber zu vermitteln, wie der jeweilige Termin einzuordnen ist, ob dieser z.B. zu einer Serie

von Terminen gehört, die auf ein finales Ziel hinauslaufen. In diesem Fall wären die Informationen und verknüpften Dokumente der bisherigen Termine zur gleichen Thematik für den Anwender interessant. Neben den intern verknüpften Dokumenten können zusätzliche externe Informationen, beispielsweise aus Online-Communities oder Informationen der online verfügbaren Tagespresse zur Anreicherung der jeweiligen Thematik genutzt werden. Eine nähere Analyse dieser kontextuellen Erweiterung findet in Kapitel 3.4.2 statt.

Mit diesen Termindaten und deren Erweiterung soll eine Funktionalität zur unterstützenden Tagesplanung des Benutzers als zweiter Anforderungskomplex an die Anwendung einhergehen.
Zu dieser Planung gehören beispielsweise die Berechnung und Darstellung der Routen des Benutzers zwischen den einzelnen Terminen unter Berücksichtigung der verfügbaren Zeit sowie der Wege von und nach Hause. Ein weiterer Aspekt ist die Einbeziehung von „alltäglichen“ Dingen, wie zum Beispiel der Planung des Mittagessens. Der Benutzer soll die Möglickeit bekommen, persönliche Präferenzen anzugeben, die bei der Planung des Tagesablaufs berücksichtigt werden. Solche Voreinstellungen können beispielsweise die Art des bevorzugten Fortbewegungsmittels oder die Angabe von Pausenzeiten sein.

Der in 3.1.5 beschriebene Vorteil elektronischer Kalender, Reminder für ein Event einrichten zu können, soll ebenfalls berücksichtigt werden. Diese können hierbei je nach Präferenz in einem festen Zeitintervall vor dem nächsten Event oder dynamisch unter Berücksichtung von weiteren Faktoren, z.B. der Fahrtzeit, erfolgen. Diese Alarmfunktion soll neben der beschriebenen Anwendung als Erinnerung für einen kommenden Termin ebenfalls eine hinweisende Funktion übernehmen. Ändert sich ein Terminparameter in der Hostapplikation oder wird ein Termin sogar vollständig gelöscht, was bei gemeinsam genutzten Kalendern durch Dritte, also ohne die Kontrolle des Anwenders, erfolgen kann, soll die Software den Benutzer auf diese Änderungen hinweisen und ggf. den so entstandenen Zeitraum, je nach Vorgabe des Benutzers, neu verplanen.

Eine weitere Funktion, die von der Software unterstützt werden soll, ist die Realisierung eines To-do-Listen Moduls. Der Benutzer soll die Möglichkeit bekommen, ein neues To-do-Objekt anzulegen und dessen Dringlichkeit zu bestimmen. Die Aufgabe der Anwendung besteht darin, anhand gegebener Parameter dieses Objektes einen geeigneten Ort zu finden, an dem die formulierte Aufgabe erledigt werden kann. Dieser Ort muss geografisch so gewählt werden, dass er auf der zu absolvierenden Strecke zwischen zwei Terminen, oder im Umkreis eines Terminortes, unter geringem Aufwand erreichbar ist. Bei der Suche nach dem nächstmöglichen Zeitraum zur Erledigung der Aufgabe spielt die vom Nutzer vorgegebene Priorisierung eine vorrangige Rolle.

3.4.2 Kontextanalyse

Persönliche Assistenten haben die Möglichkeit, angemessen auf die Bedürfnisse der Person, für die sie arbeiten, zu reagieren. Sie sind in der Lage, Situationen zu verstehen und aus ihnen zu lernen. Sie können, auch ohne dass ihnen ein „kompletter Satz“ an Informationen zu einem Event bereitgestellt wird, handeln, da sie anhand vorhergehender vergleichbarer Situationen ihre Schlüsse auf die Präferenzen und Absichten einer Person ziehen können.

Das Verhalten eines persönlichen Assistenten auf eine Anwendung zur Tagesplanung in Grundzügen zu übertragen ist das Ziel dieser Arbeit. Hierzu ist es zunächst erforderlich, die unterschiedlichen Kontexte zu analysieren, die eine solche Applikation erfassen und verwerten soll. In Reaktion auf die erkannten Situationen kann die Anwendung entsprechende Informationen bereitstellen oder Services ausführen.

Für eine möglichst genaue Bestimmung des Kontextes ist es erforderlich, alle verfügbaren Informationen zu sammeln und auszuwerten, die zur Charakterisierung einer Situation beitragen. Dies formuliert Dey in seiner Definition von Kontext:

> *„Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves.“*
>
> - Dey (2001)

Hierbei kann der Fokus des zu erfassenden Kontexts unterschiedlich sein. Das Szenario *Nutzung von Daten aus sozialen Netzwerken* beschreibt den situativen Hintergrund eines Termins, wie in 3.1.2 vorgestellt. In diesem Fall handelt es sich um ein Folgemeeting mit bestimmten Teilnehmern zu einem festgelegten Thema. Die Absicht ist hier das Erreichen eines Ziels durch eine Reihe von Events.
Eine andere Betrachtungsweise ist die Kontexterfassung des Nutzers, wie sie in *Mobile Nutzung* dargestellt wird. Hierbei wird die Situation des Anwenders zum aktuellen Zeitpunkt erfasst. Mittels des erkannten Kontexts soll die Anwendung Informationen bereitstellen oder bestimmte Services ausführen.
In beiden Fällen ist zur Erfassung des jeweiligen Kontexts die Ansammlung und Aufbereitung aller verfügbaren Daten erforderlich.

Digital Social Media als Quelle personenbezogener Informationen

Die Identität der beteiligten Personen, also im Fall von Events alle verfügbaren Informationen über die Teilnehmer oder im Verlauf der Tagesplanung Informationen über den Anwender an sich, gehören zu den von Dey beschriebenen primären Kontextinformationen, anhand derer

weitere Daten abgeleitet werden können. Ausgangspunkt für die verfügbaren Daten soll das persönliche Adressbuch des Benutzers sein, welches einen grundlegenden Datenstamm liefert. Anhand der Informationen des Adressbuchs kann ein erster Schluss auf das Verhältnis des Anwenders zu einer anderen Person gezogen werden. Die Daten, die ein Adressbucheintrag liefern kann sind jedoch, bedingt durch die Formalisierung des vCard Standards (s. 2.8), auf relativ statische Informationen limitiert. Daher ist es erforderlich, die Profile der Personen durch weitere Informationsquellen zu erweitern, um so den Teilnehmern ein dynamischeres Profil zu verleihen und ihr „digitales Äquivalent“ adäquater zu beschreiben.

Eine nützliche Informationsquelle für dynamischere und persönlichere Nutzerdaten bieten die Plattformen des DSM. Obwohl die ersten Vertreter erst um die Jahrtausendwende herum entstanden sind, hat sich der Anteil der Nutzer, die sich in solchen Communities bewegen, stark weiterentwickelt (in Deutschland knapp 50%, China ca. 30%, in den USA über 70% der Bevölkerung, Stand 2009) (Bry u. a. (2010)). Dieses wird auch durch die von den Rundfunkanstalten ARD und ZDF seit mittlerweile 15 Jahren jährlich durchgeführte Onlinestudie belegt (ARD und ZDF (2012)). Diese beschreibt das Nutzungsverhalten des Internets durch Personen, aufgelistet nach verschiedenen Medientypen und Alterskategorien. Die Studie belegt, dass der Anteil von Personen, die Mitglied in einer Online Community sind, in den letzten Jahren immer weiter angestiegen ist. Dieses betrifft vor allem die Altersgruppe der 14- bis 29-jährigen.

Die Plattformen, die von den Menschen verwendet werden, führen zu einem Wandel im Kommunikationsverhalten, weg von klassischen Telefonkontakten und teilweise sogar E-Mails hin zu den Kommunikationswegen, die von den DSM Plattformen angeboten werden.
Ein weiterer Aspekt bei der Nutzung ist die Repräsentation des eigenen Personenprofils. Angepasst an den jeweiligen Hintergrund einer Plattform (z.B. beruflich oder privat) stellen Menschen eine immer größere Fülle von persönlichen Informationen für andere bereit.

Die Erkenntnisse dieser Studien führen zu dem Rückschluss, dass die Quellen des DSM eine geeignete Voraussetzung zur Anreicherung der Personenprofile bieten und die zu implementierende Anwendung Schnittstellen zu ausgewählten Plattformen implementieren soll.

Ortsinformationen

Informationen über den Ort sind ein weiterer Teil der primären Kontextinformation, daher soll die Anwendung so viele Daten wie möglich über den Ort erfassen und verwerten können. Die Quellen für den Ort sind entweder das Ortsfeld des Events, wie es beispielsweise im Szenario *Übersicht über die Termine des Tages* der Fall ist, oder die aktuelle Position des Benutzers, wie in *Mobile Nutzung* beschrieben. Die Verwendung dieser gewonnenen Daten verfolgt unterschiedliche Intentionen.

Routenplanung: Die Anwendung soll, wie im vorgestellten Szenario *Nutzung von Ortsdaten und Präferenzen*, die Orte der Termine dazu nutzen, Routeninformationen und Fahrpläne bereitzustellen. Zusätzlich zu den Terminen besteht hier noch die Möglichkeit, aus dem persönlichen Profil des Anwenders die privaten Adressinformationen zu ermitteln und Weginformationen von und nach Hause zu ermitteln.

Umkreissuche: Im Rahmen der hier vorgestellten Anwendung soll die Umkreissuche für verschiedene Zwecke genutzt werden. Das Szenario *Erkennen von Vorhaben* beschreibt einen klassischen Anwendungsfall: Durch den bekannten Ort soll für einen bestimmten Anlass die passende Lokalität im Umfeld ermittelt werden. Die vorgestellten Vorschläge sind sortiert nach ihrer Bewertung. Eine weitere Anforderung wird im Szenario *To-do-Listen* vorgestellt, in dem nicht die Umkreissuche im Vordergrund steht, sondern die Suche nach einem *Point of Interest* (im Folgenden POI) zur Erledigung eines To-do-Objekts.

Wetter: Die Applikation soll Wetterdaten für einen bestimmten Ort erfassen und auswerten können. Diese sind notwendig, um dem Nutzer, wie in *Nutzung von Ortsdaten und Präferenzen* dargestellt, auf bestimmte Sachverhalte hinzuweisen.

Verifizierung von Terminen: Dieser Aspekt der Nutzung von Ortsdaten ist vor allem für die spätere Nutzung von vergangenen Events notwendig. Die in 3.3.2 vorgestellte Arbeit von Lovett u. a. (2010) beschreibt die Nutzung der Ortsbestimmung der Meeting-Teilnehmer zur Zeit des Termins, um so zu verifizieren, ob ein Termin wirklich stattgefunden hat. Diese genaueren Daten über Events können der Anwendung dabei helfen, präzisere Aussagen über zukünftige Termine zu machen, die beispielsweise ähnlich, aber nicht vollständig, ausgefüllt sind.

Zeitbezogene Informationen

Die Informationen, die die Anwendung über die Zeit erfassen kann, spielen eine weitere Hauptrolle bei der Kontexterfassung, sowohl für ein Event, als auch im Tagesablauf des Anwenders.
Für die Anwendung werden verschiedene Komponenten der Zeiterfassung benötigt. Dieses beginnt in einem größeren Kontext durch die „Positionierung“ des Tages im laufenden Jahr, woraus Rückschlüsse über die aktuelle Jahreszeit und damit verbundene äußere Umstände gezogen werden können (z.B. im Winter nicht mit dem Fahrrad zu fahren). Der nächte Schritt besteht in einer Kategorisierung der Art des Tages. Dieses kann ein Werktag, ein Tag am Wochenende oder ein Feiertag sein. Hieraus können weitere Informationen über Öffnungszeiten und Erreichbarkeiten anderer Menschen abgeleitet werden. Eine weitere Verfeinerung wird durch Einbeziehung der aktuellen Uhrzeit, bzw. der Terminzeiten, ermöglicht. Hierdurch können ebenfalls Informationen über Erreichbarkeiten oder Öffnungszeiten gewonnen werden.

Feld	Inhalt
Wer?	Sal, Alex, Sören
Wann?	Samstag, 05.05.2012 Beginn: 9:00 Uhr Ende: 12:00 Uhr
Wo?	HAW Hamburg
Was?	Sondermeeting Projektfinanzierung

Tabelle 3.1: Beispieltermin

In Verbindung mit der Planung von Routen können so Aussagen darüber getroffen werden, welche Art der Fortbewegung die sinnvollste ist und welche Fortbewegungsmittel verfügbar sind.

Aktivitätsbezogene Informationen

Die letzte primäre Kontextkategorie besteht im Erkennen der Aktivität. Dieses kann zum einen das *Was?* eines Termins, zum anderen das Erkennen der Absicht des Anwenders im Tagesablauf sein.
Durch eine Analyse von Betreff und des Beschreibungstext eines Termins können Aussagen über Absicht und thematischen Rahmen dieses Event getroffen werden. Wie bereits in 3.1.2 beschrieben, kann ein Termin in eine Serie von vorhergehenden und nachfolgenden Terminen eingebettet sein, die zusammen zur Erreichung eines Gesamtziels dienen.
Durch die Begutachtung des eigentlichen Termininhaltes können dem Anwender weitere relevante Informationen zum gleichen Thema verfügbar gemacht werden. Diese können entweder von externen Quellen abgefragt oder durch die mit den Events verknüpften Dokumente bereitgestellt werden. Neben der Analyse des Termininhalts spielt die Aktivität des Anwenders im Tagesablauf eine weitere Rolle. Hierbei geht es darum, welches „Ziel“ der Benutzer zum aktuellen Zeitpunkt verfolgt.

Aggregation der Kontextelemente

Mit Hilfe der vorgestellten primären Kontextelemente und dem damit möglichen Rückschluss auf weitere Faktoren ist es möglich, ein Abbild einer Situation zu erstellen. Wichtig hierbei ist, dass das Zusammenspiel der einzelnen Teilaspekte beachtet wird, da es genügt nicht, sich auf einen einzelnen Aspekt zu stützen. Ein fiktiver Beispieltermin (3.4.2) soll dieses verdeutlichen:

Würde man versuchen, den Kontext des Beispieltermins nur anhand der Zeit zu bestimmen, könnte dieses zu einem falschen Ergebnis führen. Falls der Anwender in den Einstellungen seiner Software angegeben hat, dass er von Montag bis Freitag arbeitet und somit das Wochenende als freie Zeit festgelegt ist, würde die Anwendung den Termin als privat einstufen. Ebenso würde es sich verhalten, wenn der Fokus nur auf den teilnehmenden Personen liegt[8]. Zieht man jedoch zusätzlich die beiden anderen Teilaspekte Ort und Aktivität hinzu, verlagert sich die Kategorie des Events wieder in das berufliche Umfeld, da der Ort und das Thema des Termins aus dem Tätgkeitsfeld des Anwenders stammen. Hierdurch werden die Teilnehmer eher in der Beziehung *Arbeitskollege* als in der Relation *Freund* betrachtet.
Dieses einfache Beispiel demonstriert, dass es stets erforderlich ist, alle Terminaspekte gleichermaßen zu berücksichtigen. Nur so kann eine korrekte Einordnung der Terminintention erfolgen.

3.4.3 Kalender-Hostanwendung

Zentraler Ausgangspunkt der Anwendung sind die Kalendereinträge des Anwenders. Von ihnen hängt die Gestaltung der Tagesplanung ab. Die zu entwickelnde Anwendung soll in die bestehende Kalender-Infrastruktur, die der Nutzer bereits verwendet, integriert werden. Die dort verwalteten Termine werden durch die Applikation um zusätzliche Informationen erweitert, um sie dann grafisch zu präsentieren. Die Anwendung soll demnach nicht als eigenständige Umsetzung einer vollständigen Kalendersoftware dienen.
Eine solche Integration stellt die Anforderung an eine Kalender-Hostanwendung, eine Schnittstelle für externe Applikationen zur Verfügung zu stellen. Über diese soll die Software lesend als auch schreibend mit dem Kalenderhost interagieren. Dadurch können Termine ausgelesen, erstellt, editiert und gelöscht werden.

3.4.4 Eine Anwendung für mehrere Zwecke

Das zu entwickelnde System soll dem Anwender als Ausgangspunkt seiner Tagesplanung dienen. Neben der eigentlichen Nutzung des Kalenders und der Verwaltung von To-do-Listen spielt die Einbeziehung weiterer Funktionalitäten eine relevante Rolle. Lee (2010) beschreibt in seiner in 3.3.5 vorgestellten Arbeit die Berücksichtigung weiterer Software in die Kalenderapplikation, um so den Wechsel zwischen verschiedenen Unterprogrammen überflüssig zu machen und dem Anwender eine komfortablere Nutzung zu ermöglichen.

So kann es beispielsweise interessant sein, die Routenplanung durch die Darstellung auf einer Übersichtskarte zu unterstützen, ohne diese in einem zusätzlichen Browserfenster öff-

[8] In den Szenarien werden Alex und Sören als Freunde von Sal vorgestellt.

nen zu müssen. An Events angehängte Dokumente sollten direkt in der Anwendung abruf- und darstellbar sein. Zur Kommunikation mit anderen Personen sollte eine E-Mail Funktion integriert werden. Im Zusammenhang mit der mobilen Nutzung wird die Verwendung weiterer Programme interessant, die durch Smartphones bereitgestellt werden. So kann das direkte Versenden von Kurzmitteilungen oder die Verwendung der Telefonfunktion aus der Applikation heraus wichtig für den Anwender sein, wie es das Szenario *Mobile Nutzung* beschreibt. Des Weiteren schlägt Lee die Verwendung der GPS Funktion vor, um Inhalte mit Orten zu verknüpfen. Dieses sollte neben der automatischen Verwendung durch die Anwendung ebenfalls manuell durch den Nutzer möglich sein, um so Orte mit bestimmten Dingen verknüpfen zu können.

3.4.5 Evaluation und Archivierung vergangener Events

Die Anwendung soll die Möglichkeit besitzen, auf zurückliegende Events und abgeschlossene To-do-Objekte zugreifen zu können, um sie für neue Termine und Aufgaben zu nutzen. Die Verwendung kann dabei verschiedene Motive haben:

Kontextbestimmung: Die „Erfahrungswerte“ aus beendeten Terminen können bei der Bestimmung der Situation neuer Events dienlich sein. In Dey (2001) wird das Verknüpfen von Kontext mit Informationen zur späteren Verwendung als eine der drei zentralen Funktionalitäten von *context-aware* Software vorgestellt.

Zeitliche Abschätzungen: Daten von beendeten Events und Aufgaben können dazu genutzt werden, genauere Aussagen über die Dauer vergleichbarer, zukünftiger Termine zu machen. Mit Hilfe dieser Informationen können dann Termine, die mit der Applikation erstellt werden, bereits mit einer entsprechenden Dauer versehen werden, etwa die Dauer eines Meetings oder die Zeit, die zur Absolvierung einer Aufgabe wahrscheinlich benötigt wird.

3.4.6 Nichtfunktionale Anforderungen

In den vorhergehenden Abschnitten wurden die Aufgaben der Anwendung festgelegt. Neben diesen funktionalen Anforderungen ist es für den Entwickler bei der Realisierung neuer Software erforderlich, diese unter den Gesichtspunkten nichtfunktionaler Qualitätsmerkmale zu entwickeln, um so ein wiederverwertbares und hochwertiges Produkt bereitzustellen. Die für die hier zu realisierende Anwendung relevanten nichtfunktionalen Anforderungen werden im Folgenden kurz beschrieben.

Zuverlässigkeit

Der Begriff Zuverlässigkeit ist laut ISO 9126 (1991) definiert als die Fähigkeit eines Systems, unter bestimmten Bedingungen ein bestimmtes Level an Performanz innerhalb eines festgelegten Zeitfensters aufrechtzuerhalten. Zu den Attributen von Zuverlässigkeit zählen die *Korrektheit*, *Fehlertoleranz* und die *Wiederherstellbarkeit*.
Vor allem die Korrektheit der Ergebnisse, die das System liefert ist für eine Kalenderanwendung eminent wichtig. Die fehlerhafte Darstellung von Terminen, zum Beispiel durch eine falsche Uhrzeit oder die falsche Zuordnung des Terminortes kann für den Anwender unangenehme Folgen haben und machen die Software somit unbrauchbar. Fehlerhafte Eingaben durch den Benutzer müssen von der Software erkannt werden. Ebenso darf der Import von fehlerbehafteten Daten aus der Hostapplikation nicht zu Inkonsistenzen oder Programmabstürzen führen, sondern muss durch entsprechende Fehlerroutinen abgefangen werden, um einen stabilen Programmfluss zu gewährleisten. Die *Wiederherstellbarkeit* bezeichnet dabei die Fähigkeit des Systems, die betroffenen Daten wiederherzustellen, um auf das volle Leistungsniveau zurückzukehren.

Portierbarkeit

Die in Kapitel 3.2 vorgestellten Szenarien beschreiben die Nutzung der Software auf verschiedenen Geräten mit unterschiedlichen Betriebssystemen, andersartiger Hardware und verschiedenen Bildschirmauflösungen. Das Design der Anwendung sollte deswegen dahingehend ausgelegt sein, dass es mit möglichst geringem Aufwand an unterschiedliche Umgebungen anpassbar ist und plattformunabhängig arbeitet. Der Umfang der dargestellten Informationen sollte an die jeweiligen Möglichkeiten der genutzten Hardware angepasst sein.

Sicherheitsanforderungen

Sicherheit spielt bei der Nutzung von Kalendersoftware eine wichtige Rolle. Die in der Analyse beschriebenen Teilaspekte der Kontextidentifizierung bedienen sich vertraulicher Benutzerdaten aus sozialen Netzen sowie seines Adressbuchs. Die im Kalender festgehaltenen Termine können ebenfalls private Daten und Informationen, die nicht für Dritte zugänglich sein sollen, enthalten. Die Software sollte daher einen Authentifizierungsmechanismus unterstützen, der die Daten vor unbefugtem Zugriff schützt.
Einen weiterer Aspekt bei den Überlegungen zur Sicherheit ist die Verwendung der ortsbasierten Daten. Die Anfragen der Applikation an die standortbezogenen Dienste könnten dazu verwendet werden, Bewegungsprofile des Anwenders zu erstellen. Die Software sollte daher nur die zum Betrieb erforderlichen Anfragen an die entsprechenden Dienste senden.

Erweiterbarkeit

Die Anwendung soll ihre Daten aus verschiedenen Quellen beziehen, deren Art und Anzahl sich im Laufe der Zeit immer wieder verändern kann. Daher ist es erforderlich, bei der Entwicklung Schnittstellen zu schaffen, die einen nahtlosen Wechsel bestehender Datenquellen erlauben. Die Einbindung weiterer Sensoren sollte mit geringem Aufwand realisierbar sein. Diese Schnittstellen müssen von den nicht beliebig änderbaren Kernfunktionen der Software getrennt werden.
Neben den Datenquellen kann sich der Funktionsumfang der Software verändern, alte Programmteile können durch neue ersetzt werden und neue Features können hinzukommen. Die Änderungen des Funktionsumfangs erfordert daher eine lose Kopplung der einzelnen Programmteile, um einen einfachen Austausch zu gewährleisten.

3.5 Zusammenfassung

In diesem Kapitel wurden die Anforderungen an eine kontextbasierte Tagesplanerapplikation erarbeitet. Hierzu wurde der geisteswisschenschatfliche Aspekt des Begriffs *Event* näher betrachtet. In diesem Zusammenhang spielt der Vorgang der prospektiven Erinnerung eine hervorzuhebende Rolle. Die Analyse der Benutzung unterschiedlicher Kalendervarianten in Papierform und als Software führte zu den ersten Anforderungen, welche an die zu entwickelnde Software bestehen.
Die Beschreibung möglicher Funktionen, die eine Kalendersoftware unter Einbeziehung von Kontextinformationen leisten kann, wurde anhand verschiedener Szenarien eines fiktiven Tagesablaufs demonstriert.
Durch die Vorstellung verschiedener Ansätze, die sich bereits mit Anwendungen zur kontextuellen Erweiterung von Terminen beschäftigt haben, wurde deutlich, dass bislang noch keine umfassende Realisierung einer solchen Applikation exisitiert.
Die Systemidee einer solchen Anwendung wurde im Anschluss präsentiert. Die Informationen sowie die benötigten Daten, die zur Erfüllung der Szenarien benötigt werden, wurden in der Anforderungsanalyse herausgearbeitet. Das zu entwickelnde System soll sich in die bestehende Kalender-Infrastruktur des Anwenders eingliedern und als zentrale Komponente ein Modul zur Erweitung der Kontextinformationen beinhalten. Als primäre Bestandteile zur Anreicherung der bestehenden Daten wurden die Teilaspekte Ort, Zeit, Person sowie Aktivität festgelegt. Eine weitere Komponente zur Verwaltung von To-do-Listen, deren Bearbeitung eingebettet in den Tagesablauf des Nutzers erfolgen soll, wurde hier ebenfalls vorgestellt.

4 Design

Die in den vorhergehenden Abschnitten vorgestellten Anforderungen an eine kontextbasierende Kalenderanwendung sollen in diesem Kapitel dazu genutzt werden, das Gesamtdesign eines solchen Systems zu entwickeln.
Hierzu werden zunächst die Hauptkomponenten der Anwendung vorgestellt und ihre Aufgaben näher beschrieben. Im Anschluss wird die Integration dieser Bestandteile in eine geeignete Architektur unter Berücksichtigung adäquater Designpattern und Techniken des objektorientierten Softwaredesigns näher erläutert.

4.1 Komponentenübersicht

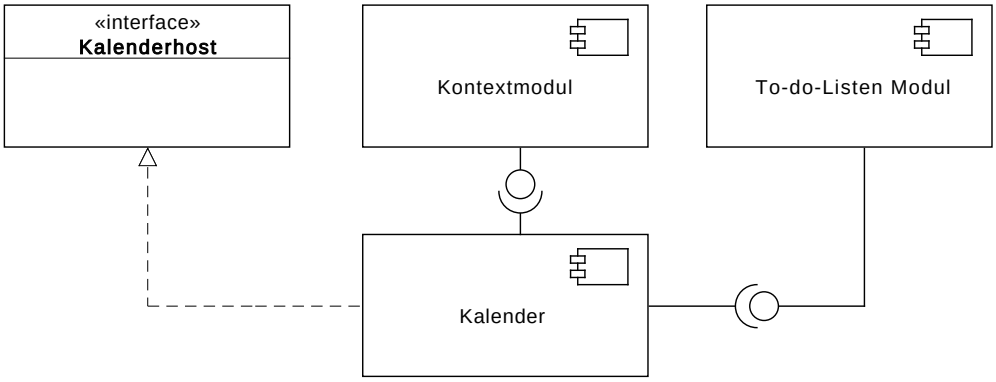

Abbildung 4.1: Komponentenübersicht

Zur Erfüllung der in Kapitel 3.4 beschriebenen Anforderungen ist eine grobe Gliederung des Gesamtsystems in drei Teilbereiche erforderlich. Diese werden in Abb. 4.1 dargestellt.
Ausgangspunkt ist hierbei die Schnittstelle zu einer Kalender-Hostapplikation, über welche die Anwendung die Termine des Benutzers abrufen und verwalten kann. Diese Daten dienen als Basis und liefern erste Details über anstehende Events und den Tagesablauf.
Die extrahierten Informationen werden dann mit Hilfe eines Kontextmoduls um weitere Inhalte angereichert. Diese liegen dem Kontext des Anwenders und dem der Termine zu Grunde.

Als dritte Hauptkomponente kommt ein Modul zur Verwaltung von To-do-Listen zum Einsatz. Die Verwaltung dieser Listen findet losgelöst von der Hostanwendung des Benutzers statt. Die Erledigung der einzelnen Aufgaben soll allerdings, ebenfalls angereichert durch weitere Daten des Kontextmoduls, in den Tagesablauf der Person integriert werden.

Die folgenden Abschnitte sollen dazu dienen, die technischen Anforderungen an die einzelnen Komponenten des Systems näher zu beschreiben.

4.1.1 Kalender-Hostapplikation

Die Kalendereinträge, die in der Hostanwendung des Benutzers verwaltet werden, sind der Ausgangspunkt für alle weiteren Funktionalitäten, die in der zu entwickelnden Software implementiert werden sollen. Eine zentrale Anforderung an den Adapter, über den die Schnittstelle zur Hostanwendung realisiert wird, liegt in der Verarbeitung von standardisierten Austauschformaten, um eine Ankopplung aller Systeme, die diese unterstützen, zu ermöglichen.

Ein standardisiertes Format, das zum Austausch von Kalenderinformationen benutzt wird, ist das in den Grundlagen vorgestellte iCal Format (s. Kap. 2.4). Das iCal Format muss durch die Hostanwendung unterstützt werden, damit ein Adapter, über den die Kommunikation zu dieser Applikation abläuft, genaue Informationen über den Aufbau der Terminelemente hat. Diese werden für die zu entwickelnde Anwendung entsprechend aufbereitet.

Neben dem standardisierten Format zum Datenaustausch ist es erforderlich, dass die Hostapplikation eine geeignete Kommunikationsschnittstelle unterstützt. Mit Hilfe der hier zu entwickelnden Software legt der Anwender Termine an oder bearbeitet diese. Hierzu ist es erforderlich, dass die Kommunikationsschnittelle der Hostanwendung sowohl einen lesenden als auch schreibenden Zugriff auf die Termine des Anwenders ermöglicht.

Eine weitere wünschenswerte, aber nicht zwingend erforderliche, Eigenschaft der Hostanwendung ist die direkte Verbindung mit dem Adressbuch des Benutzers. Hierdurch können einem Termin zugeordnete Personen eindeutig mit der Referenz aus dem Adressbuch in Verbindung gebracht werden.

4.1.2 Kontextmodul

Die Realisierung des Kontextmoduls ist einer der wichtigsten Punkte bei der Entwicklung der in dieser Arbeit vorgestellten Anwendung. Durch die zusätzlichen Informationen, die diese Komponente zu den bestehenden Terminen des Anwenders liefert, wird eine Kontextbezogenheit ermöglicht. Die Umsetzung dieses Moduls basiert auf den in Kapitel 2.3 vorgestellten

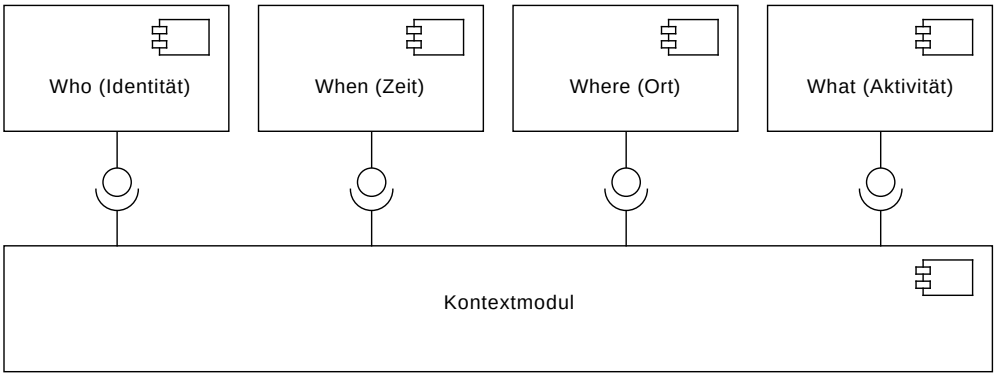

Abbildung 4.2: Übersicht des Kontextmoduls

primären Faktoren zur Erfassung von Kontext. Für das Design dieser Komponente bedeutet dies eine weitere Gliederung in Submodule für Identität (Personen), Ort, Zeit und Aktivität (Anlass). Diese Aufteilung wird durch Abb. 4.2 dargestellt.

Who - Identität

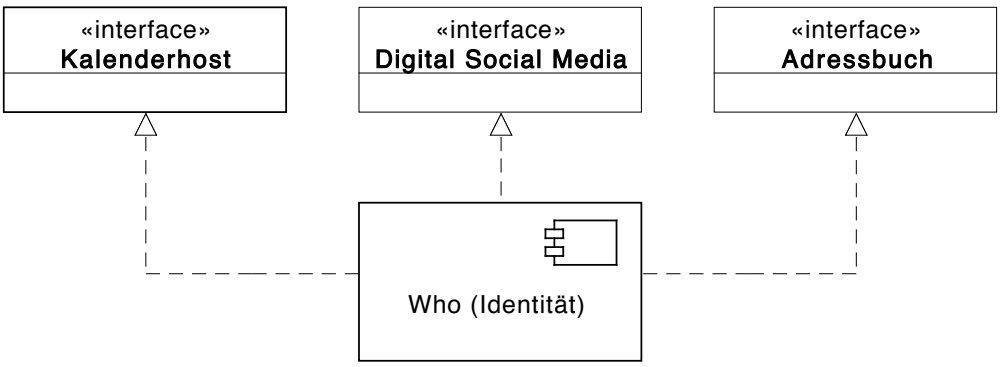

Abbildung 4.3: Datenquellen des Personenmoduls

Für die Möglichkeit der Kontextbestimmung durch die Anwendung ist das Wissen über die an Terminen beteiligten Personen sowie vor allem Wissen über den Benutzer selbst, essentiell. Die grundlegenden Informationen über die Teilnehmer eines Events erhält das Personenmodul über die Kalender-Hostanwendung des Benutzers. Der erste Schritt zur Erweiterung dieser Daten ist die Einbeziehung der verfügbaren Informationen aus dem persönlichen elektronischen Adressbuch des Anwenders. Im Idealfall erhält die Software dadurch umfassendere Informationen zu den unterschiedlichen Adressen des jeweiligen Teilnehmers, darüber

hinaus verschiedene Kontaktinformationen wie Telefonnummern, Instant-Messaging Benutzernamen sowie E-Mail Adressen.

Für die Integration in das zu entwickelnde System ist es hierbei, wie auch bei den Kalendereinträgen, erforderlich, dass der Aufbau der Adressbucheinträge für eine Auswertung durch entsprechende Adapter durch ein standardisiertes Format erfolgt. Dadurch ist es möglich, jedes Adressbuch, dass dieses Format unterstützt, zu verwenden. Ein Standard, der sich hierfür eignet, ist das in Kap. 2.4 vorgestellte vCard Format. Dieses wird durch den überwiegenden Teil der heute verbreiteten elektronischen Adressbücher unterstützt. Ein lesender Zugriff auf das Adressbuch ist hierbei ausreichend.

Die Anreicherung der statischen Adressbuchdaten erfolgt durch Quellen des DSM. Bedingt durch die geringe Verbreitung von semantisch angereicherten Personendaten im Internet ist es im Allgemeinen nicht möglich, diese Daten über einfache semantische Suchanfragen an die unterschiedlichen Plattformen zu erhalten. Es ist daher erforderlich, dass der Anwender auf diesen direkt mit den Kontakten seines Adressbuchs vernetzt ist. Die Schnittstelle zu einer solchen Plattform muss es der Anwendung ermöglichen, alle dem Benutzer zugänglichen Daten abrufen zu können. Des Weiteren muss die Kommunikation mit einer solchen Plattform über ein standardisiertes Datenformat erfolgen. Der personenbezogenen Komponente ist es unter Realisierung plattformspezifischer Wrapper-Klassen möglich, verschiedene Quellen des DSM zu verwenden.

Where - Ort

Ausgangspunkt der ortsspezifischen Komponente ist wieder die Kalender-Hostapplikation, aus der Informationen darüber abgerufen werden, an welchem Ort ein Termin stattfindet. Damit diese Ortsinformationen eindeutig werden ist es erforderlich, diese in ein Schlüssel-Wert-Paar umzuwandeln. Ein verbeiteter Ansatz hierzu ist die Umwandlung einer Adresse in ein eindeutiges Koordinatenpaar aus Längen- und Breitengrad. Hierzu soll eine Komponente für Geocoding-Dienste zum Einsatz kommen, die als Antwort der Übermittlung einer Adresse ein eindeutiges Koordinatenpaar liefert. Für alle weiteren Dienste, die zur Bearbeitung von geospezifischen Anfragen zum Einsatz kommen, ist es daher erforderlich, diese Koordinaten als Eingabeparameter zu akzeptieren.

Wie sich Abbildung 4.4 entnehmen lässt, erfüllt das Modul für Geoinformationsdienste neben der Zuordnung von Koordinaten weitere Zwecke. Ein Dienst zur Abfrage von Wetterinformationen dient dazu, Wetterangaben für den aktuellen Tag sowie zu den einzelnen Terminorten zu machen. Ein Wetterdienst muss dazu mindestens die Wetterdaten für den aktuellen Tag liefern. Da sich ein Termin aber auch über mehrere Tage erstrecken kann, sollten ebenfalls Informationen zu den naheliegenden Folgetagen verfügbar sein.

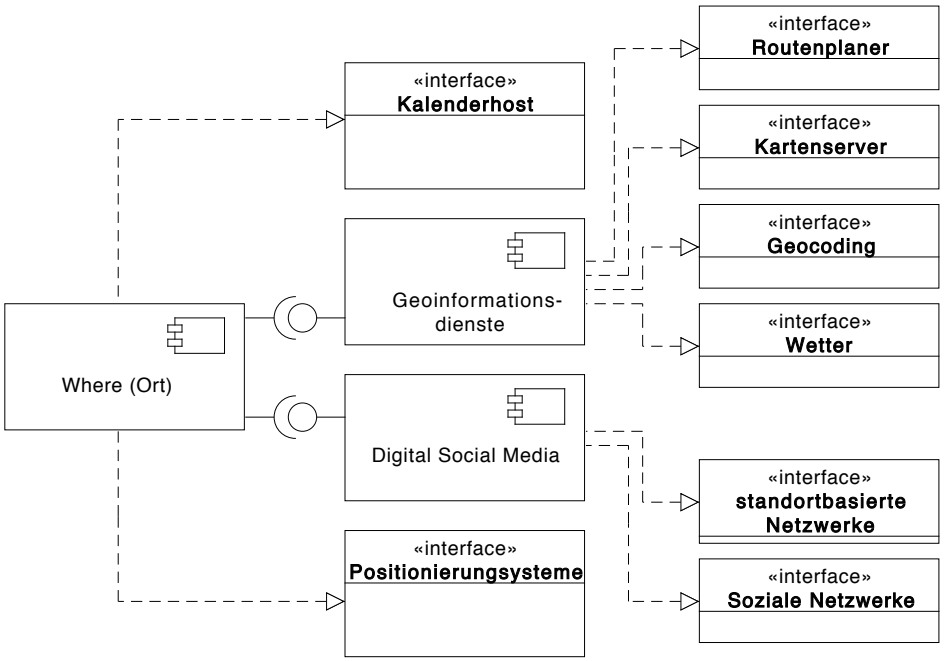

Abbildung 4.4: Datenquellen des Ortsmoduls

Ein Dienst zur Routenplanung kommt für die Berechnung der Dauer und Route der Wege, die der Anwender zu absolvieren hat, zum Einsatz. Hierbei handelt es sich um die Berechnung von Routen im „gewohnten“, regionalen Umfeld des Anwenders, da sich bei größeren Strecken die Wahl des Transportmittels sowie die Dauer der Reise unterscheiden kann (beispielsweise Zug und Flugzeug) und von den regulären Transportmitteln des Benutzers abweicht. Ein Dienst zur Routenplanung muss neben der Übermittlung von Koordinaten als Start- und Zielpunkt persönliche Voreinstellungen des Anwenders zur Art der Fortbewegung (z.B. zu Fuß oder mit dem Auto) verarbeiten können. Ein Kartenserverdienst dient dazu, die abgefragten Daten zu visualisieren. Dieser muss die Ergebnisse der Routenplanung interpretieren und darstellen können.

Auch bei den Ortsinformationen spielen die Daten des DSM eine wichtige Rolle. Hier geht es vor allem um die Informationen aus standortbasierten sozialen Netzwerken. Anhand der gewonnenen Daten werden Informationen über spezielle POIs bereitgestellt, die über ein eindeutiges Koordinaten identifiziert werden. Zusätzlich werden die standortbasierenden Dienste zu der in 3.4.2 beschriebenen Umkreissuche genutzt. Hierzu besteht die Anforderung, dass bei der Anfrage an einen solchen Dienst neben Ortskoordinaten ein Betreff übergeben werden kann, angelehnt an das in 3.2.4 vorgestellte Szenario beispielsweise „Italienisches Restaurant“. Das Ergebnis einer solchen Anfrage muss verschiedene Daten zu den einzelnen POIs enthalten. Hierzu zählen Name, Adresse (im Ideafall auch die exakten Ko-

ordinaten), Kontaktmöglichkeiten und, falls benötigt, Bewertungen und Kommentare anderer Nutzer.

When - Zeit

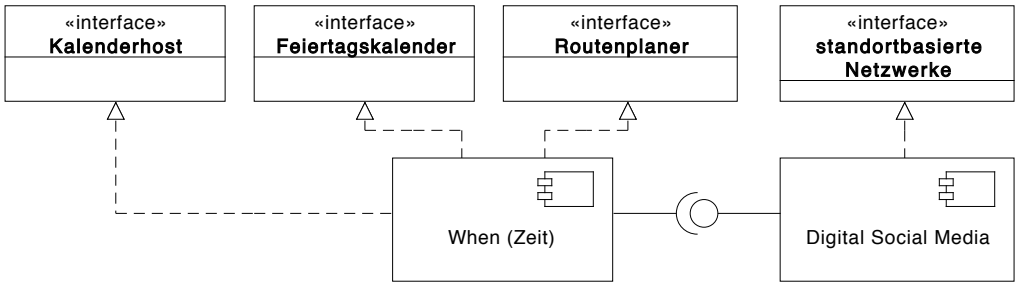

Abbildung 4.5: Datenquellen des Zeitmoduls

Das Zeitmodul (s. Abb. 4.5) ist als weitere Komponente des Kontextmoduls der dritte Bestandteil zur Ermittlung und Bereitstellung primärer Kontextinformationen. Es greift ebenfalls auf die Basisinformationen der Termine aus der Hostapplikation zu. Die zeitlichen Informationen sind vor allem wichtig zur Festlegung des Tagesablaufs. Hierzu gehört ebenfalls die Einplanung von To-do-Objekten. Für die zeitliche Erfassung der Wege, die der Anwender zu absolvieren hat, bezieht die Zeitkomponente, wie bereits das Ortsmodul, die Fahrtzeiten über den bereits vorgestellten Dienst zur Routenplanung. Informationen aus standortbasierten Netzwerken sollen dazu dienen, Öffnungszeiten oder weitere zeitliche Besonderheiten eines POIs zu ermitteln. Eine weitere Komponente zur Abfrage von Feiertagsdaten und den daraus resultierenden Besonderheiten bei der Terminplanung ist ebenfalls Bestandteil des Zeitmoduls.

What - Aktivität

Die Komponente zur Bestimmung näherer Informationen über den eigentlichen Anlass eines Termins, bzw. die mit dem Termin verbundene Aktivität ist das letzte Teilmodul zur Erfassung von Kontextinformationen. Die Aufgabe dieser Komponente besteht in der Analyse des Beschreibungstextes eines Events, den sie aus der Hostanwendung abruft. Ein Dienst zur Textanalyse untersucht den Beschreibungstext auf bestimmte Schlagwörter, um so eine Verbindung zwischen eventuell vor- und nachfolgenden Terminen der gleichen Thematik herzustellen oder auf Details, die zur Vorbereitung des Termins notwendig sind, hinzuweisen. Die Aktivitätskomponente stellt außerdem eine Verbindung zu verlinkten Dokumenten her. Diese

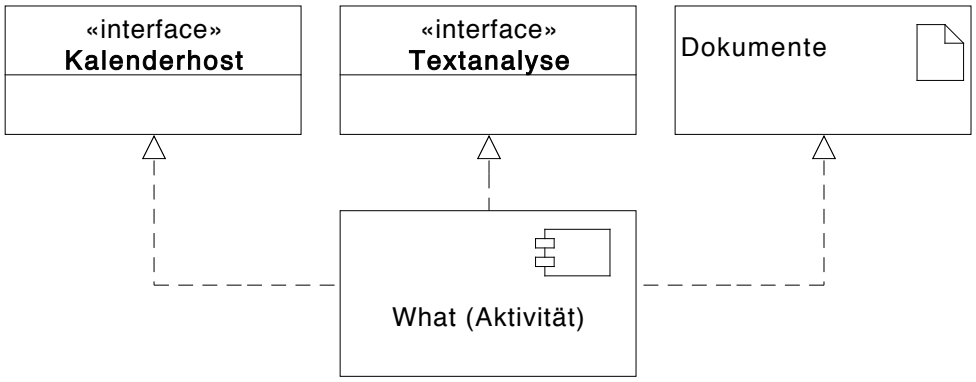

Abbildung 4.6: Datenquellen des Aktivitätsmoduls

müssen durch die Software abrufbar sein, damit sie lokal vom Benutzer verwendet werden können.

4.1.3 To-do-Listen Modul

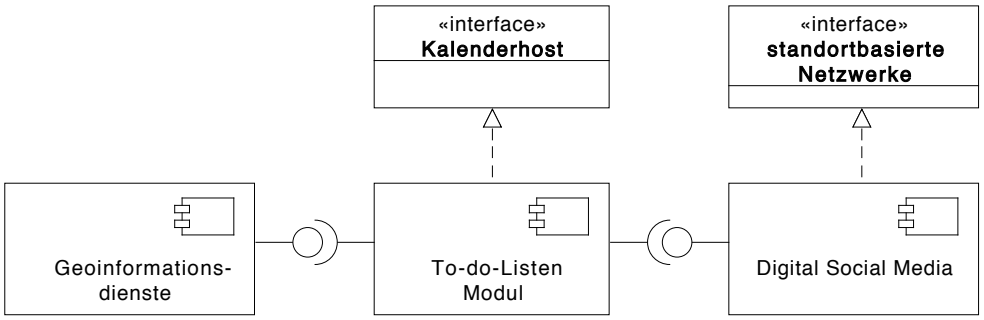

Abbildung 4.7: Komponenten des To-do-Listen Moduls

Als weiteres Modul ist eine Komponente zur Verwaltung von To-do-Listen vorgesehen. Diese werden von der Software und somit unabhängig vom Kalendersystem verwaltet. Diese Komponente bezieht ihre Daten vom Kalenderhost, standortbasierten Netzwerken sowie über Geoinformationsdienste. Die Daten aus dem Kalender sind notwendig, um Informationen über die Orte, an denen sich der Benutzer tagsüber befinden wird sowie die damit verbundenen Dauern und freie Zeiträume zu ermitteln. Mit Hilfe der standortbasierten sozialen Netzwerke wird ein geeigneter POI zur Bearbeitung eines To-do-Objekts ermittelt. Für die Planung der An- und Abreise zu diesem benutzt das Modul Routeninformationen, die sie über Geoinformationsdienste abfragen kann.

4.2 Architektur

Die folgenden Abschnitte sollen dazu dienen, einen Überblick über die Architektur der zu entwickelnden Tagesplaneranwendung zu geben. Getroffene Designentscheidungen lassen sich im Betrieb meistens nur mit hohem Aufwand oder gar nicht verändern, daher spielen die Überlegungen zur Architektur als Teil des Softwareentwurfs eine grundlegende Rolle bei der Entwicklung neuer Software. Perry u. Wolf (1992) beschreiben die Eigenschaften der Architektur als Erfüllung der in einer Anforderungsanalyse gestellten Requirements, der technischen Basis für ein weiteres Design und als wirtschaftliche Grundlage zur Bemessung der entstehenden Kosten. Eine strukturierte Architektur ermöglicht eine Wiederverwendbarkeit der entwickelten Komponenten und die Möglichkeit zur Anaylse von Konsistenz und Abhängigkeiten.

4.2.1 Client-Server-Modell

Die hier vorgestellte Anwendung kommt auf unterschiedlichen Geräten zur Anwendung. Dieses bedeutet auch, dass ein Benutzer mehrere Geräte zur Darstellung seines persönlichen Tagesablaufs verwenden kann. Die Informationen, die sich der Anwender anzeigen lässt, müssen auf allen Geräten konsistent sein. Ebenso müssen mögliche Terminänderungen oder Neuerstellungen durch eine lokale Bearbeitung der Termine an die anderen Geräte kommuniziert werden.
Die Koordination, die Verwaltung sowie die Aufbereitung und Zusammenstellung der relevanten Daten sollte daher sinnvollerweise von einer zentralen Serverinstanz übernommen werden. Dieses impliziert ebenfalls eine Verlagerung der Rechenlast auf die Serverkomponente. Mögliche mobile Clients werden dadurch entlastet, ein Vorteil, der auch in der von Gkekas u. a. (2007) in 3.3.3 vorgestellten Arbeit betont wird.
Ein Client meldet sich bei der zentralen Instanz an und synchronisiert sich mit dem aktuellsten Informationsstand und überträgt lokal festgehaltene Änderungen des Benutzers. Im Falle eines mobilen Clients mit der Möglichkeit zur Ortsbestimmung werden diese Daten ebenfalls übermittelt.
Fällt die Designentscheidung auf eine Client-Server-Architektur, ist zu klären, wie die Kompetenzen der beiden Systeme verteilt werden sollen. Tanenbaum u. Steen (2007) beschreiben eine mögliche Verteilung dieser in fünf Kategorien (s. Abb. 4.8):

(a) In diesem Modell dient der Client ausschließlich zur Darstellung von Informationen, die durch den Server bereitgestellt werden. Zusätzlich erfolgt über den Client die Eingabe von Daten durch den Benutzer. Sämtliche Daten und die Programmlogik werden vom Server verwaltet.

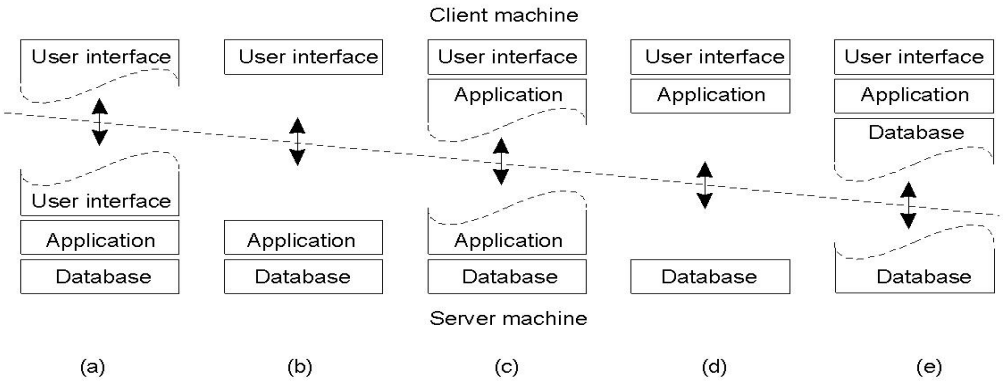

Abbildung 4.8: Aufgabenverteilung im Client-Server-Modell (nach Tanenbaum u. Steen (2007))

(b) Diese Form der Kompetenzverteilung erweitert die Funktionen von (a) um weitere Interaktionsmöglichkeiten. Der Client erhält zusätzlich eine grafische Oberfläche (z.B. ein X-Window System[1]). Die Programmlogik und die persistente Datenschicht liegen weiterhin auf der Serverseite.

(c) Hierbei erhält der Client einen Teil der Anwendung und hat somit die Möglichkeit, einfache Aufgaben ohne einen Server durchzuführen.

(d) Die Programmlogik gehört komplett zum Client. Die einzige Aufgabe des Servers ist die Verwaltung der persistenten Daten.

(e) In dieser Variante liegen fast alle Komponenten auf der Client-Seite (= *Fat Client*). Dieser beinhaltet die Benutzeroberfläche, die Programmlogik und einen Teil der persistenten Daten. Der Server verwaltet nur noch einen Teil der Datenbankeinträge.

Zur Auswahl der passenden Client-Server Architektur ist eine genauere Betrachtung der Anforderungen an Server und Client erforderlich.

Server

Die zentrale Aufgabe des Servers besteht im Sammeln und Aufbereiten der Daten, die zur Realisierung des in 4.1 vorgestellten Kontext- und To-do-Listen-Moduls erforderlich sind. Anhand dieser verarbeiteten Informationen schließt die Anwendung auf den Kontext eines Termins bzw. die momentane Situation des Anwenders und stellt eine geeignete Auswahl an zusätzlichen Daten zur Erweiterung der Termininformationen bereit. Ausgangspunkt hierbei

[1] http://de.wikipedia.org/wiki/X_Window_System

sind die Daten der Kalender-Hostapplikation, die das Fundament für weitere Informationsbeschaffung stellen.
Neben den Online-Quellen spielt die persistente Sicherung der Daten durch den Server eine weitere Rolle. Diese dienen der Serverapplikation im Betrieb für unterschiedliche Zwecke. Die Daten vergangener Events können dazu benutzt werden, kommende, unvollständig ausgefüllte Termine zu vervollständigen oder Entscheidungen darüber zu treffen, in welchem Kontext ein Termin einzuordnen ist. Einmal abgerufene Daten der unterschiedlichen Dienste werden offline verfügbar gemacht und wiederverwendet. Hierdurch wird unnötiger Traffic vermieden und die Performanz der Aufbereitung gesteigert.

Client

Die primäre Aufgabe des Clients besteht in der Darstellung der durch den Server aufbereiteten Informationen. Anhand dieser Daten bekommt der Anwender eine Übersicht über seinen Tagesablauf und erhält die Möglichkeit, einzelne Events und deren Teilaspekte genauer zu betrachten.

Neben der Darstellung von Informationen soll der Nutzer zusätzlich die Möglichkeit bekommen, neue To-do-Objekte anlegen sowie Events neu erstellen oder editieren zu können. Hat das Client-Gerät zusätzliche Features, die für den Betrieb der Anwendung von Nutzen sind, beispielsweise Funktionen zur Positionsbestimmung durch einen GPS Empfänger, werden die Daten dieser Quellen an den Server zur weiteren Verwendung übermittelt (vgl. das Szenario zur mobilen Nutzung 3.2.5).

Für die Auswahl einer geeigneten Aufgabenverteilung im Client-Server-Modell bedeutet das Folgendes: Der zentrale Speicherort für die persistenten Daten liegt auf der Serverseite, ebenso die Programmlogik und die Anbindung an externe Quellen zur weiteren Informationsbeschaffung.
Zur Fokussierung auf die eigentliche Funktionalität der Kalendersoftware wird in dieser Arbeit davon ausgegangen, dass der Client über eine ständige Datenverbindung verfügt und somit den Server jederzeit erreichen kann. Eine lokale Sicherung sämtlicher Daten zur ständigen Verfügbarkeit ist nicht erforderlich. Die Aufgaben des Clients bestehen in der Visualisierung der Daten für den Benutzer, die Möglichkeit zur Interaktion mit dem System und die Einbeziehung optionaler, gerätespezifischer Dienste.

Eine mögliche Realisierung des Clients besteht in der Umsetzung als *Rich Internet Application*[2], in welcher der Benutzer eine Bedienoberfläche dargestellt bekommt. Zusätzlich benötigte Programmteile werden durch die dauerhaft bestehende Internetverbindung dynamisch vom Server angefordert und geladen. In dieser Arbeit erfolgt die Realisierung des Clients jedoch als eigenständige Java-Applikation. Im Bezug auf das von Tanenbaum u. Steen (2007)

[2] http://de.wikipedia.org/wiki/Rich_Internet_Application

vorgestellte Modell stellt daher Variante (c) für eine Realisierung den passenden Ansatz dar.

Delegation Pattern

Das Delegation Pattern ist ein Design Pattern aus der objektorientieren Programmierung. Eine Klasse (der *delegator*) verwendet dabei die Dienste einer anderen Klasse (des *delegate*) zur Bearbeitung einer Aufgabe. Dieses wird als „Steuerungsumkehr“ (engl. *Inversion of Control*) bezeichnet, da die Verantwortlichkeit nicht mehr bei der Hauptklasse sondern auf der Seite des *delegate* liegt.

Bei der Entwicklung der hier vorgestellten Anwendung kann das Delegation Pattern für die Realisierung eines zentralen Zugriffpunktes des Servers verwendet werden, von dem aus alle Bearbeitungsanfragen an die einzelnen Komponenten delegiert werden. Ein Client benötigt dann nur noch Informationen über die Schnittstellen dieses zentralen Zugriffpunktes, um alle gewünschten Daten zu erhalten.

Singleton Pattern

Das Singleton Pattern zählt zur Klasse der Erzeugungsmuster und dient dazu, die mehrmalige Instanziierung eines Objekts zu verhindern. Die Instanziierung des Sinlgeton-Objekts erfolgt über eine Klassenmethode, die statisch gebunden und synchronisiert ist. Die Verwendung des Singleton Patterns kann neben der Kontrolle über den Zugriff eine Steigerung der Performance zur Folge haben, da nur die benötigten Klassen instanziiert werden.

Die Verwendung dieses Erzeugungsmusters in der Applikation ist sowohl für eine Zugriffskontrolle als auch zur Steigerung der Performance sinnvoll.
Zum einen ist es erforderlich, dass für die Datenoperationen nur genau ein Objekt zum Einsatz kommt, um die Konsistenz der Daten zu wahren, zum anderen werden durch die Verwendung des Singleton Patterns nur die benötigten Klassen instanziiert. Dieses erhöht die Performance und vermeidet unnötige Verbindungen sowie Traffic zu externen Diensten im Web.

4.2.2 Model-View-Controller

Das Model-View-Controller Prinzip (im Folgenden MVC) beschreibt ein Architekturmuster in der Softwareentwicklung, das die Aufteilung einer Anwendung in die drei Komponenten Datenmodell (= *Model*), Präsentation (= *View*) und Programmsteuerung (= *controller*) vorsieht

(vgl. hierzu Abb. 4.9). Ziel bei der Verwendung dieses Musters ist die Trennung der Verantwortlichkeiten, um einen flexibleren Programmentwurf zu ermöglichen und jede Komponente für sich wiederverwertbar zu machen. So erlaubt die Strukturierung der Anwendung nach dem MVC-Muster beispielsweise den Austausch der View-Komponente gegen eine andere, wodurch unterschiedliche Oberflächen für verschiedene Systeme unter geringem Aufwand realisierbar werden.

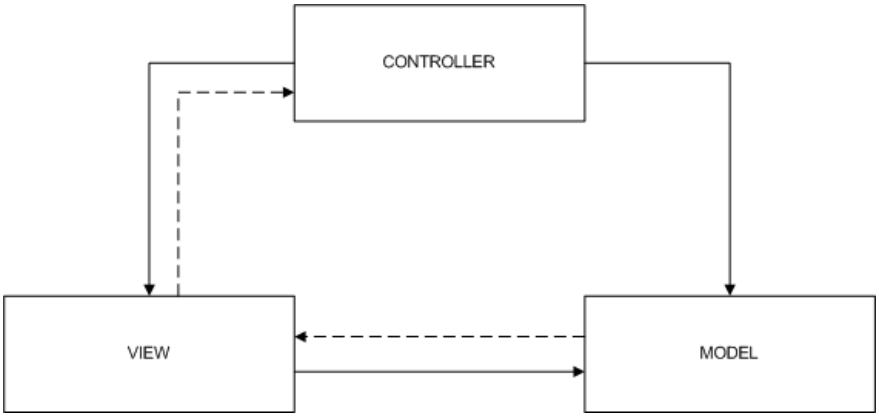

Abbildung 4.9: Model-View-Controller

Der Überblick über den Aufbau der MVC Architektur zeigt, dass die Komponenten nicht völlig voneinander trennbar sind. Die View benötigt Kenntnisse über das Model um Benutzereingaben direkt weiterleiten zu können. Im Hinblick auf die in 4.2.1 vorgestellte Client-Server Architektur ist es jedoch erforderlich, die grafische Oberfläche weitestgehend von jeglicher Logik zu trennen. Daher kommt bei der Realisierung der Anwendung mit dem *Passive-View* Prinzip eine Variante des *Model-View-Presenter* Architekturmusters zum Einsatz, das auf dem ursprünglichen MVC Ansatz unter Berücksichtigung einer noch stärkeren Trennung der einzelnen Komponenten, besonders der von Model und View, basiert.

4.10 beschreibt den Aufbau des Passive View Architekturmusters. Ein Presenter übernimmt die komplette Interaktion mit der Datenkomponente. Finden in dieser Änderungen statt, wird ein Update der View ebenfalls durch diesen vorgenommen. Die Verbindung zwischen Model und View entfällt. Die einzige Aufgabe der View besteht nur noch die Darstellung der Informationen und Weitergabe von Benutzereingaben an den Presenter.
Die Aufgaben der einzelnen Komponenten sowie ihre konkrete Verwendung in der hier vorgestellten Anwendung werden im Folgenden beschrieben.

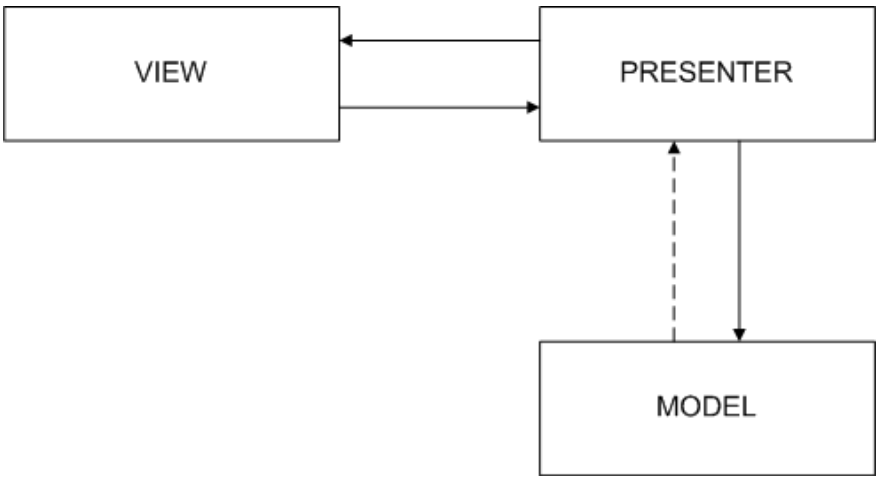

Abbildung 4.10: Passive-View

Model

Allgemein betrachtet beinhaltet das Model den Kern der Anwendung. Es enthält die relevanten Daten sowie die Geschäftslogik des Programms und ist von der Controller- und View-Komponente unabhängig. Findet eine Änderung der Daten im Model statt, werden die anderen Komponenten mit Hilfe des *Observer Pattern* (vgl. hierzu Kapitel 4.2.2) über diese benachrichtigt.

In der hier zu entwickelnden Anwendung umfasst das Model alle zentralen Logikelemente der Anwendung. Dieses bedeutet zum einen die Steuerung sämtlicher Abläufe sowie die Verwaltung der Kommunikationskanäle. Den zentralen Zugriffspunkt des Models für die eigentlichen Programmfunktionen bildet eine durch das Delegation Pattern (s. Kap. 4.2.1) realisierte Zugriffsklasse. Alle Anfragen durch den Presenter werden über diese Klasse empfangen und zur Bearbeitung weitergeleitet. Maskiert hinter dieser Klasse befinden sich die einzelnen Komponenten mit ihren Zugriffspunkten zu den externen Anwendungen und Diensten, die zur Bearbeitung benötigt werden. Durch das Singleton Pattern (s. Kap. 4.2.1) wird dabei sichergestellt, dass von dem jeweiligen Modul nur eine Instanz aktiv ist.

View

Die Visualisierung der im Model vorgehaltenen Daten sowie die Möglichkeit zur Benutzerinteraktion ist die Aufgabe der View-Komponente des Systems. Diese enthält dabei keine eigene fachliche Logik, da es die durch den Benutzer eingegebenen Daten nicht weiter bearbeiten, sondern nur weiterleiten soll. Die View hat im klassischen MVC Aufbau Kenntnis über das Model und kann somit direkte Nachrichten an dieses schicken.

Im Falle der hier angestrebten Umsetzung als Model-View-Presenter Architektur bildet der Presenter den einzigen Zugriffspunkt für die View-Komponente. Alle Benutzereingaben werden an diesen durchgegeben, darzustellende Daten werden über den Presenter angereicht.

Presenter

Der Presenter stellt die Schnittstelle zwischen Datenmodell und der grafischen Benutzerschnittstelle dar. Die über die View eingegebenen Daten werden an den Presenter weitergereicht, der dadurch entstehende Aufträge an das Model weiterleitet. Der Presenter kennt View und Model über definierte Schnittstellen. Dieses ermöglicht eine stärkere Trennung und Austauschbarkeit dieser umgebenden Komponenten.

In der Anwendung übernimmt der Presenter die Verbindung der Komponenten. Bezogen auf eine spätere Einbettung des Architekturmusters in eine Drei-Schichten-Architektur (s. Kap. 4.2.3) wird der Presenter in der Interaktionsschicht angesiedelt. Ein Client umfasst ihn sowie die grafische Oberfläche.

Observer Pattern

Das Observer Pattern gehört zur Klasse der Verhaltensmuster. Es beschreibt die Weitergabe von Änderungen die an einem Objekt auftreten an die von diesem Objekt abhängigen Komponenten, damit sich diese automatisch aktualisieren können. Eine zu enge Bindung der beteiligten Objekte untereinander würde die Wiederverwendbarkeit stark einschränken. Das Observer Pattern verfolgt das Ziel, eine lose Kopplung dieser einzelnen Elemente zu ermöglichen.

Die betreffenden Kommunikationspartner werden als *Subject* und *Observer* bezeichnet. Das Subject stellt für beliebig viele Observer Methoden zur An- und Abmeldung bereit. Des Weiteren kann es durch eine Benachrichtigungsmethode alle angemeldeten Objekte über seine Änderungen informieren. Die Observer implementieren eine Schnittstelle, über die sie bei Änderungen durch das Subject informiert werden. Der aktuelle Zustand des Subjects wird zum Zeitpunkt der Änderung zur Wahrung der Datenkonsistenz gesichert.

In der hier vorgestellten Anwendung kann das Observer Pattern bei der Realisierung unterschiedlicher Aufgaben zum Einsatz kommen. Ein „klassisches“ Einsatzgebiet ist die Verwendung des Patterns bei der Realisierung einer lose gekoppelten MVC Architektur (s. 4.2.2) und dessen Derivaten.

Im Falle des Passive-View Musters dient das Pattern zur Realisierung der asynchronen Kommunikation zwischen Model und Presenter. Das Model übernimmt hierbei die Funktion des Subjects. Tritt eine Änderung auf, informiert es den Presenter, der als Observer fungiert.

Ein weitere Anwendungsmöglichkeit stellt die Verwendung bei der Reminder-Funktionalität des Kalenders dar. Hierbei ist der Thread, der auf das Eintreten eines Reminder-Events prüft, in der Rolle des Subjects. Bei einem auftretenden Event informiert dieser die registrierte Observer Komponente im Model, die wiederum ihren Beobachter (den Presenter) benachrichtigen kann.

4.2.3 Drei-Schichten-Architektur

Schichtenmodelle - oder auch Schichtenarchitekturen genannt - tragen während der Entwicklung von Software zur besseren Strukturierung der Gesamtarchitektur bei. Ziel einer solchen Architektur ist das Zusammenfassen lose gekoppelter, kohäsiver Funktionsblöcke in einer Ebene. Die einzelnen Schichten sind dabei voneinander getrennt und hierarchisch aufgebaut. Die Elemente einer Schicht haben die Möglichkeit, auf die Funktionen der unter ihr liegenden Schicht zuzugreifen. Dabei ist es vom Grad der Trennung abhängig, ob nur ein Zugriff auf die direkt unterliegende Schicht oder auch der Zugriff auf die in der Hierarchie tiefer angesiedelten Schichten erlaubt ist. Ein Zugriff auf die höher liegende Schicht ist nicht möglich, somit reduziert sich der Grad der Abhängigkeiten der Systemkomponenten, was eine Austauschbarkeit ermöglicht und eine einfachere Wartung zur Folge hat.

Für den generellen Aufbau der hier vorgestellten Software und der Integration des vorgestellten Passive-View Architekturmusters bedeutet dieses eine Aufteilung in mehrere Komponenten. Für den Benutzer sichtbar steht im Vordergrund eine grafische Komponente, auf der alle Informationen dargestellt werden und über die eine Möglichkeit zur Interaktion mit dem System besteht.

Die Daten dieses grafischen Frontends werden durch die Komponenten der Logik-Schicht bereitgestellt. Diese ist für die Aggregation und Aufbereitung der Daten aus den verschiedenen Quellen zuständig. Im Hintergrund der Anwendung stehen die verschiedenen Datenquellen. Diese sind einerseits lokal und persistent, um To-do-Listen und einmal gesammelte Daten speichern zu können. Andererseits sind sie als externe Quellen, mit denen durch entsprechende Adapter kommuniziert wird, realisiert.

Für den Aufbau der Software bietet sich die Umsetzung einer Drei-Schichten-Architektur (engl. *Three-Tier-Architecture*) an, da sich das Design der Anwendung in Darstellung bzw. Interaktion, Logik und Datenhaltung aufgliedert. Die Inhalte der einzelnen Ebenen werden im Folgenden näher beschrieben. Abb. 4.11 beschreibt die Integration der einzelnen Komponenten in die Gesamtarchitektur.

Interaktionsschicht

In der Interaktionsschicht (engl. *client tier*) werden alle für den Anwender sichtbaren Elemente zusammengefasst. Sie entspricht der Umsetzung des Clients. Dieser umfasst neben der grafischen Oberfläche zur Darstellung der Kalenderinformationen und weiterer Daten ebenfalls die Interaktions- und Eingabemöglichkeiten des Systems. Durch die Presenter-Komponente beinhaltet die Interaktionsebene einen minimalen Anteil Logik. Diese dient neben der Steuerung der View zur Verarbeitung der Benutzereingaben und eventueller Sensorinformationen. Diese werden an das in der unterliegenden Schicht verankerte Model weitergereicht, das die entsprechende Geschäftslogik zur Verarbeitung der Informationen bereitstellt (vgl. Schatten u. a. (2010), S. 214).

Anwendungslogik

In der Logikschicht (engl. *application-server tier*) werden alle Verarbeitungsmechanismen der Software vereint. Sie beinhaltet das vorgestellte Model der Model-View-Presenter Architektur, das die Komponenten, die zur Realisierung der fachspezifischen Funktionalität der Kalendersoftware erforderlich sind, umfasst.
Neben Kontext- und To-do-Listen-Modul beinhaltet die Logikschicht eine Komponente zur Ansteuerung und Bereitstellung der Infrastruktur. Eine weitere Komponente, die zusätzliche Utility-Klassen für die Bearbeitung nicht-fachlicher Aufgaben enthält, ist ebenfalls Bestandteil.

Datenhaltungsschicht

Die Datenhaltungsschicht (engl. *back end*) bildet die unterste Schicht der Drei-Schichten-Architektur und ist für die Verwaltung der Inhalte zuständig. Neben den Daten, die persistent abgelegt werden, bietet die Datenhaltungsschicht durch entsprechende Adapter Zugriff auf externe Dienste, über die benötigte Daten angefordert werden können.
Bei den persistent abgelegten Daten handelt es sich um die archivierten Kalendereinträge, die angelegten To-do-Objekte, eventuelle Reminder-Daten sowie Daten über POIs, zu denen bereits eine Suchanfrage stattgefunden hat. Die Speicherung dieser Daten erfolgt durch Serialisierungsmechanismen der Programmiersprache. Neben den genannten Daten werden die aus den Quellen des DSM akkumulierten Personenprofile ebenfalls in dieser Schicht abgelegt. Die Speicherung dieser Daten findet nicht durch Serialisierung statt, sondern durch Ablage der Daten in XML Dateien, aufgebaut nach den Vorgaben der in 2.2 vorgestellten Ontologien. Hierdurch wird eine Weiterverwendung der aggregierten Informationen für weitere Anwendungen ermöglicht. Diese können auf diese Daten beispielsweise über das Internet zugreifen.

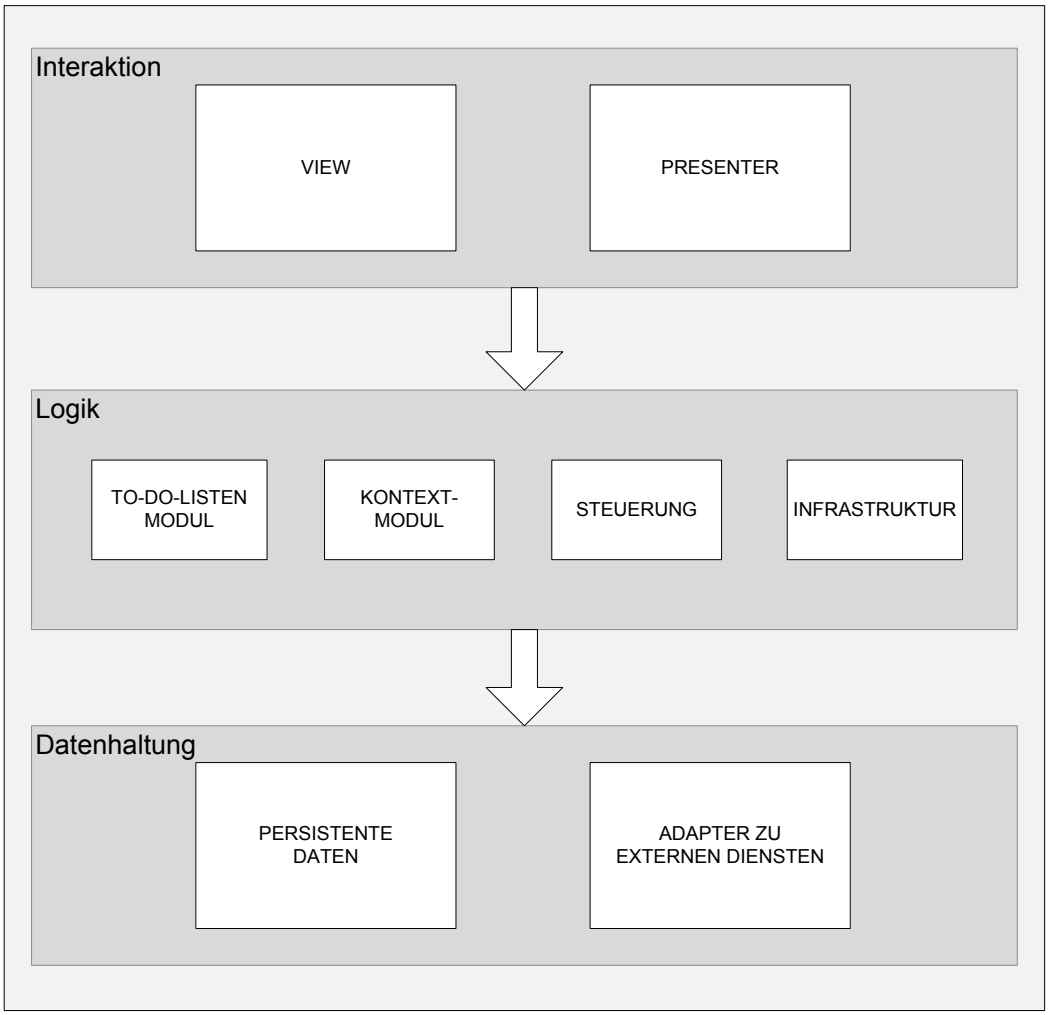

Abbildung 4.11: Drei-Schichten-Architektur mit Integration des Model-View-Presenter Musters

Mögliche Erweiterungen

Eine mögliche Erweiterung dieser Drei-Schichten-Architektur wäre durch Einfügen einer weiteren Ebene zwischen der Datenhaltungsschicht und der Anwendungslogik realisierbar. In dieser Datenzugriffsschicht könnte ein zusätzliches Modul prüfen, ob durch die Logik angeforderte Daten bereits als persistent gespeicherte Daten vorliegen oder diese über die externen Dienste angefordert werden müssen. Durch diese Vorschaltung könnte eine noch stärkere Modularisierung ermöglicht werden, da die Anwendungslogik keine Informationen darüber benötigt, woher die angeforderten Daten aus der Datenhaltungsschicht angefordert werden müssen und somit nur einen Funktionsaufruf für beide Fälle benötigt wird. Ein Nachteil einer zusätzlichen Schicht kann bei strikter Trennung der Ebenen zusätzlicher Programmcode sein, der nur zur Weiterleitung von Aufrufen an die unteren Schichten dient.

5 Realisierung

In diesem Kapitel wird die konkrete Umsetzung der zu entwickelnden Applikation für eine kontextberücksichtigende Tagesplanung anhand der in den vorhergehenden Kapiteln erarbeiteten Anforderungen aus der Analyse (3) sowie den Entscheidungen für das Design des Gesamtsystems (4) beschrieben. Ziel ist die Entwicklung einer Client-Server basierten Anwendung, die den Funktionsumfang eines bestehenden Kalendersystems als zusätzliche Anwendung erweitert. Hierzu findet zunächst die Evaluation und Auswahl der Kalender-Hostapplikation und der geeigneten Webdienste statt, mit denen die zusätzlichen Informationen für das in 4.1.2 vorgestellte Kontextmodul gesammelt werden. In den darauffolgenden Abschnitten wird die Integration dieser Komponenten sowie die Realisierung der zu implementierenden Drei-Schichten-Architektur näher erläutert.

Die Entwicklung des Systems erfolgt, im Hinblick auf eine einfache Portierbarkeit, in der Programmiersprache Java. Hierdurch besteht die Möglichkeit, das ebenfalls in Java implementierte Framework *Jena*[1] zur Verarbeitung von semantischen Daten zu benutzen (s. Kap. 5.4).

5.1 Evaluation geeigneter Dienste

5.1.1 Wahl der Kalender-Hostapplikation

Grundlage für die Wahl eines geeigneten Kalendersystems für die zu implementierende Anwendung bilden die in Kapitel 3.4.3 formulierten Anforderungen. Als mögliche Programme kommen die webbasierten Services von Google sowie die Verwendung einer Groupware in Frage:

Groupware: Eine wünschenswerte Eigenschaft des Hostsystems besteht darin, so viele Daten wie möglich „aus einer Hand“ zu liefern. Ein Groupware-System bietet hierfür eine geeignete Möglichkeit, da diese neben der Kalenderfunktionalität unter anderem ein Adressbuch, E-Mail Konten und die Möglichkeit der Dokumentenverwaltung bieten. Somit sind die in einem Kalendereintrag vermerkten Teilnehmer im Idealfall bereits als

[1] http://jena.apache.org

Kontakt im Adressbuch hinterlegt. Zusätzliche Dokumente oder E-Mails können ebenfalls direkt verlinkt werden.
Groupware-Systeme sind allerdings vorwiegend im beruflichen und nicht im privaten Kontext zu finden. Da der Mensch zu einer Trennung dieser beiden Bereiche neigt (s. 3.1.3), würden sich die Termine vorwiegend auf den Berufsalltag und nicht auf das Privatleben des Anwenders beziehen. Die zuvor vorgestellten Szenarien beschreiben eine Verwendung der Applikation sowohl für berufliche als auch private Termine, weshalb der Einsatz im Verbund mit einer Groupware-Applikation nur bedingt sinnvoll wäre.

Google: Eine Alternative zu einem Groupware-System als Hostanwendung sind die Services, die Google anbietet. Der Anwender erhält dort die Möglichkeit, unter seinem Benutzeraccount einen großen Teil der geforderten Datenquellen an einer Stelle zu vereinigen. Neben einem, über diverse API-Schnittstellen zugänglichen, Kalender werden unter anderem ein Adressbuch, ein E-Mail Postfach und eine Aufgabenverwaltung angeboten. Diese sind ebenfalls über Schnittstellen für externe Anwendungen zugänglich. Als weitere, für Entwickler interessante, Features bietet Google mit seinem *Maps*[2] Dienst einen der bekanntesten Kartendienste im Web an. An diesen können Anfragen für Routenplanungen und Positionsinformationen geschickt werden. Eingebettet in diesen Kartenserver ist ebenfalls ein Bewertungssystem für POIs. Zusätzlich existiert eine frei zugängliche Schnittstelle zur Abfrage von Wetterdaten.

Unter Berücksichtigung einer Weiterverwendung der Anwendung fällt die Wahl des Systems auf die Groupware *Zimbra*[3]. Diese ist als Teil der Infrastruktur im Livingplace Hamburg[4] an der Hochschule für Angewandte Wissenschaften Hamburg bereits vorzufinden und dort schon in anderen Projekten zur Anwendung gekommen. Zimbra bietet neben der Möglichkeit zur Administration von Kalendereinträgen ein Adressbuch zur Verwaltung von Kontakten, ein eigenes E-Mail System sowie einen Speicher für Dokumente, welche unter anderem in Kalendereinträgen verlinkt werden können. Bedingt durch die bereits bestehende Infrastruktur wird Zimbra den von Google angebotenen Kalenderservices vorgezogen. Ein Teil der Dienste von Google findet bei der Realisierung allerdings Verwendung.

Für die Kommunikation einer Anwendung mit Zimbra bestehen mehrere Möglichkeiten. Zum einen bietet Zimbra eine eigene *Representational State Transfer* (im Folgenden REST) - API an, über die via HTTP-Requests verschiedene Anfragen an die einzelnen Komponenten des Servers gestellt werden können. Die andere Möglichkeit ist die Verwendung der frei erhältlichen Java-Clientbibliotheken, die sich zur Implementierung eigener Clientanwendungen benutzen lassen. Im Rahmen der Verwendung von Zimbra im Livingplace Hamburg wurde

[2] https://maps.google.de
[3] http://www.zimbra.com
[4] http://www.livingplace.org

die Realisierung einer solchen Anwendung bereits vorgenommen (s. Barnkow (2010)) und findet an dieser Stelle Verwendung.
Die Kommunikation mit dem Zimbra Server findet über *JavaScript Object Notation*[5] (im Folgenden JSON) Nachrichten statt. Der Nachrichtenaustausch erfolgt, unter Angabe eines festgelegten Topics, über den Message-Broker ActiveMQ der Apache Foundation[6]. Dieser sog. *Kalender-Agent* bietet die Möglichkeit, Events des aktuellen oder des folgenden Tages sowie Termine eines bestimmten Datums abzufragen.

Durch die Festlegung auf Zimbra als Kalendersystem besteht gleichzeitig die Möglichkeit das Adressbuch dieser Groupware zu verwenden. Vorteil hierbei ist, dass die den Kalendereinträgen zugeordneten Teilnehmer mit den Kontakten des Adressbuchs übereinstimmen (bedingt dadurch, dass im Zimbra-Kalender nur Personen als Teilnehmer eingetragen werden können, von denen ein Adressbucheintrag vorliegt). Die Kontakte des Adressbuchs lassen sich über die Zimbra-eigene *REST*-API abfragen und werden, formatiert nach den Spezifikationen des vCard Standards (vgl. 2.4), im JSON Format übertragen.

5.1.2 Quellen des *Digital Social Media*

Soziale Netze bieten dem Anwender die Möglichkeit, Informationen über sich zu präsentieren und Verbindungen mit anderen Nutzern einzugehen. Ellison u. a. (2007) beschreiben die unterschiedlichen Intentionen, die solche Netze verfolgen. So können sich Menschen in Netzen bewegen, die einen berufsorientierten Hintergrund haben, sei es zur Verknüpfung von Fachwissen oder zur Beschreibung der eigenen Berufsqualifikation. Bekannte Plattformen hierfür sind *LinkedIn*[7] und *Xing*[8]. Der Großteil der sozialen Netze dient jedoch persönlicheren Themen, wie etwa der Verbindung mit anderen Menschen durch gemeinsame Interessen oder die Möglichkeit, mit Freunden in Kontakt zu treten. Bekanntester Vertreter hierfür ist die im Jahr 2004 veröffentlichte Plattform *Facebook*[9], die mit mehr als 900 Millionen Nutzern weltweit die Größte ihrer Art darstellt. Der Nutzer erhält dort die Möglichkeit, durch ein persönliches Profil Daten über sich bereitzustellen und sich mit anderen Teilnehmern zu „befreunden“. Neben der Möglichkeit zur Interaktion hat er außerdem hat er die Möglichkeit, gemeinsame Aktivitäten und Vorlieben mit Anderen zu teilen.

Standortbasierte soziale Netzwerke unterstützen den Anwender bei der Suche nach interessanten POIs an einem bestimmten geoprafischen Punkt. Diese Portale bieten dem Nutzer die Möglichkeit, Orte sortiert nach ihrer Kategorie (z.B. Bars, Restaurants,

[5] http://www.json.org
[6] http://activemq.apache.org
[7] http://de.linkedin.com
[8] http://www.xing.com
[9] http://www.facebook.com/

Dienstleister) in einem bestimmten Umgebungsradius zu suchen. Außerdem besteht die Möglichkeit, diese POIs zu bewerten und somit Empfehlungen für andere abzugeben. Zu den bekanntesten Vertretern dieser Netze gehören *Qype*[10] und *Foursquare*[11], Google und Facebook bieten ebenfalls solche Funktionalitäten.

Der Funktionsumfang der vorgestellten Dienste ist realitiv ähnlich. Bis auf Facebook bieten alle Anbieter eine frei zugängliche API, über die Anfragen gesendet werden können. Bei der Realisierung der Anwendung werden daher Qype um Foursquare verwendet. Es kommen zwei Dienste zum Einsatz, um die Anwendung auf eine einfache Austauschbarkeit der Informationsquellen zu testen.

Blogs stellen eine weitere Möglichkeit zur Veröffentlichung von fachspezifischem Wissen oder persönlichen Inhalten dar. Blogging Dienste wie *Blogger*[12] oder *Wordpress*[13] bieten Nutzern die Möglichkeit, ihre Beiträge unter geringem Aufwand in tagebuchähnlicher Form zu verfassen. Die Einträge werden chronologisch angeordnet und können durch *Tags* nach bestimmten Themenfeldern kategorisiert werden. Der Leser solcher Einträge hat zumeist die Möglichkeit, Einträge zu kommentieren oder mit anderen in Diskussion zu treten. Eine spezielle Form von Blogging ist das sog. *Microblogging*. Hierbei kann der Nutzer kurze Beiträge, meist mit einer Länge von unter 200 Zeichen, verfassen und für andere bereitstellen. Der bekannteste Vertreter von *Microblogging* ist *Twitter*[14]. Der Anwender kann dort kurze Einträge publizieren und den Beiträgen anderer User folgen. Durch spezielle Zeichen können Beiträge durch Schlagwörter mit Themen assoziiert und Stellung zu Beiträgen anderer bezogen werden.

Verknüpfung der Informationen

Die vorgestellten Dienste bieten die Möglichkeit, alle, dem Anwender über andere Personen zugänglichen, Informationen durch Programmschnittstellen abzufragen. Ein Problem, welches hiermit einhergeht, ist das fehlende Wissen darüber, welche Dienste eine Person verwendet. Die wenigsten Plattformen im Netz machen es für den Nutzer erforderlich, den richtigen, bzw. vollständigen Namen zu verwenden und öffentlich anzugeben. Stattdessen werden sog. *Nicknames*, also Spitznamen verwendet. Bei einer einfachen Suchmaschinenanfrage kann es somit passieren, dass nicht alle möglichen Informationsquellen gefunden werden. Dieses kann nur gewährleistet werden, falls die Anwendung Informationen über alle Synonyme der anderen Person hat.

[10] http://www.qype.com
[11] https://foursquare.com
[12] http://www.blogger.com
[13] http://wpde.org
[14] https://twitter.com

Eine Möglichkeit der Lösung dieses Problems ist die im Rahmen des *Semantic Web* entwickelte FOAF Ontologie (s. Kap. 2.2.2). Diese verfolgt das Ziel, soziale Netzwerke maschinenlesbar zu modellieren.

Einige kleinere soziale Netzwerke haben die automatische Erstellung von FOAF Links anhand der angegebenen Profildaten in ihre Seiten integriert[15]. Die zuvor genannten größeren Vertreter verzichten jedoch auf eine solche Implementierung zugunsten ihrer eigenen Entwicklungen, wie beispielsweise im Fall von Facebook der Verbreitung des *Open Graph*[16]. Dieser basiert ebenfalls auf der Subjekt-Prädikat-Objekt Tripel-Struktur wie die vorgestellten Ontologien, ist jedoch nicht öffentlich, sondern nur über plattforminterne Authentifizierungsmechanismen, zugänglich. Die Verwendung einer semantischen Suche und der Verarbeitung durch Abfragesprachen wie *SPARQL* (s. 2.2.4) ist trotz Authentifizierung nicht möglich.

5.1.3 Geoinformationsdienste

Zur kontextuellen Erweiterung der verfügbaren Ortsdaten eines Kalenderevents ist die Einbeziehung weiterer geografischer Informationsdienste erforderlich. Bei den Tests verschiedener Anbieter frei zugänglicher Dienste für geospezifische Informationen (*Microsoft Bing*[17], Google Maps oder *Yahoo! Maps*[18]) hat sich gezeigt, dass diese unterschiedliche Informationen zu einem Koordinatenpaar liefern (was gegen die Eindeutigkeit eines Koordinatenpaares spricht), daher ist es ein Hauptauswahlkriterium, dass ein Anbieter so viele der geforderten Dienste wie möglich bereitstellt. Hierbei hat sich das Angebot von Google durchgesetzt, das seit der Einführung seiner *Maps API*[19] im Jahr 2005 ein immer umfassenderes und frei zugängliches Angebot an Geoinformationsdiensten anbietet. Diese bedienen alle geforderten Requirements und sind über eine einfache API-Schnittstelle via HTTP-Request ansprechbar.

Geocoding

Zur Anreicherung der Ortsdaten ist es zunächst erforderlich, den in einem Termin festgelegten Ort in ein Koordinatenpaar von Längen- und Breitengrad umzuwandeln. Somit ist dieser eindeutig und für alle weiteren ortsbasierten Anwendungen verwendbar, da nicht alle Dienste mit normalen Adressen arbeiten können. Diese Übersetzung von Adressen in eindeutige Koordinatenpaare von Längen- und Breitengrad wird als *Geocoding* bezeichnet.

[15] Eine Liste dieser Netzwerke befindet sich unter http://www.w3.org/wiki/FoafSites
[16] https://developers.facebook.com/docs/opengraph/
[17] http://www.bing.com/maps
[18] http://de.maps.yahoo.com
[19] https://developers.google.com/maps/documentation/webservices/index?hl=de

Routenplanung

Ein weiterer Dienst und Bestandteil der Google Maps API ist die sog. *Diections* API, die zur Planung von Routen genutzt werden kann. Mit ihr können die Strecken zwischen zwei Koordinatenpaaren oder Adressen bestimmt werden. Weiterhin lassen sich zusätzliche Parameter übergeben, etwa ob die Strecke mit dem Auto, dem Fahrrad oder zu Fuß zurückgelegt werden soll. Die Antworten können ebenfalls im JSON Format übertragen werden. Als Inhalt wird eine detailierte Beschreibung der Strecke mit allen richtungsrelevanten Punkten (genannt: *Steps*) geliefert, die sich zur Darstellung in eine Karte der Maps API eintragen lassen.

Wetter

Im Internet exisitert eine Vielzahl an Diensten, die Wetterinformationen frei zugänglich zur Verfügung stellen. Dazu zählen für den deutschsprachigen Raum beispielsweise der Deutsche Wetterdienst[20], wetter.de[21] sowie eine undokumentierte API von Google, die an dieser Stelle, bedingt durch das beschriebene Angebot aller Dienste aus einer Hand, Verwendung findet. Eine Anfrage liefert Wetterergebnisse für den aktuellen und die vier darauffolgenden Tage mit Tageshöchst- und Tagestiefsttemperatur sowie einer Kurzbeschreibung der Wetterbedingungen.

5.1.4 Zeitinformationen

Die Anwendung bezieht die zusätzlichen Informationen, die sie für Zeitangaben benötigt, aus verschiedenen Quellen. Mit Hilfe der vorgestellten standortbezogenen Netzwerke und der Geoinformationsdienste kann bereits ein Großteil der benötigten Informationen in Erfahrung gebracht, bzw. dazu genutzt werden, weitere Daten abzuleiten. Hierzu zählen Fahrzeiten sowie die Öffnungszeiten von POIs.
Eine weitere, für die Anwendung wichtige, Informationsquelle ist ein, wie bereits in Kapitel 4.1.2 beschriebener, Feiertagskalender, der alle Feiertagsdaten beinhaltet. Bei der Realisierung kann entweder auf eine externe Quelle zurückgegriffen werden, welche die Feiertagsdaten beispielsweise aufgebaut im iCal Format bereitstellt. Eine andere Möglichkeit bietet eine eigene Realisierung nach der in Kapitel 2.5 vorgestellten Gaußschen Osterformel.
In der hier vorgestellten Anwendung wird die Berechnung der Feiertage nach dieser Formel

[20] http://www.dwd.de
[21] http://www.wetter.de

durchgeführt. Dieses spart zusätzliche Schnittstellen, über die eine geeignete Quelle angebunden werden müsste. Des Weiteren kann die Berechnung dynamisch unter geringem Rechenaufwand erfolgen, somit müssen keine Daten persistent gespeichert werden.

5.1.5 Textanalyse

Die maschinelle Analyse und Auswertung von Texten zur weiteren Verwendung stellt als Computerlinguistik ein Teilgebiet der Künstlichen Intelligenz dar. Die Verarbeitung des Beschreibungstexts eines Kalendereintrags liefert weitere Informationen zu Anlass und Inhalt des jeweiligen Termins (vgl. Kap. 3.3.4). Bei der Realisierung der hier vorgestellten Anwendung wird allerdings auf die Implementierung umfassender Textanalyse-Tools verzichtet, um einen ausgeglichenen Fokus auf alle Teilbereiche der Anwendung zu erhalten. Stattdessen bedient sich die Anwendung zur Erkennung bestimmter Thematiken vorgegebener Schlüsselwörter. Für einen detailierteren Überblick zur umfassenden Textanalyse sei hier auf die weiterführende Fachliteratur verwiesen.

5.1.6 Kommunikation der Komponenten

Die Kommunikation der vorgestellten Komponenten findet sowohl über Schicht- als auch über physikalische Grenzen hinweg statt. Bei der Wahl geeigneter Kommunikationswege ist daher darauf zu achten, die daraus resultierenden Restriktionen einzuhalten.

Die Wahl einer Client-Server-Architektur macht die Auswahl eines geeigneten Kommunikationsmusters der beteiligten Komponenten erforderlich. Sowohl Client als auch Server beinhalten Informationen, die für den anderen relevant sind. Bei der Bereitstellung der Informationen lassen sich zwei Kategorien unterscheiden: Ein Kommunikationspartner hat die Möglichkeit, seine Daten direkt an sein Gegenüber zu senden. Dieses entspricht einer *push*-Kommunikation. Das genaue Gegenteil beschreibt die *pull*-Kommunikation, hierbei „fragt“ ein Kommunikationspartner aktiv sein Gegenüber nach neuen Informationen.

Die Wahl eines geeingeten Kommunikationsmusters für die zu entwickelnde Tagesplaneranwendung ist inspiriert durch die Kommunikation der Komponenten des Living Place. Der Austausch der einzelnen Teilnehmer findet hier über einen *publisher-subscriber*-Mechanismus statt, bei dem die Nachrichten über eine *Message Queue* gepuffert werden. Die Bereitstellung der Daten kann per *push* als auch per *pull* erfolgen.

Im Fall der Kalenderapplikation bedeutet dies, dass der Server seine Mitteilungen an die Clients in einer Warteschlange bereitstellt (*push*) und kein weiteres Interesse daran hat, wer diese Nachrichten konsumiert. Ein Client kann bei der Warteschlange periodisch erfragen, ob neue Daten bereitstehen (*pull*) und diese abrufen.

Ebenso kann ein Client seine Daten, also die Benutzereingaben und zusätzliche, maschinell erfasste Informationen, an eine zentrale Warteschlange *pushen*, in der sie durch den Server abgefragt werden können.

5.2 Präsentationsschicht

Die Präsentationsschicht der Anwendung umfasst die grafische Darstellung der Inhalte für den Benutzer sowie die Möglichkeit zur Interaktion mit dem System. Die konkrete Realisierung des Clients erfolgt als Java Desktop-Anwendung, die neben einer Übersichtsseite des aktuellen Tages alle Teilbereiche eines Termins für detailiertere Informationen bietet. Das Interface der Client-Komponenten ist in Abb. 5.1 dargestellt.

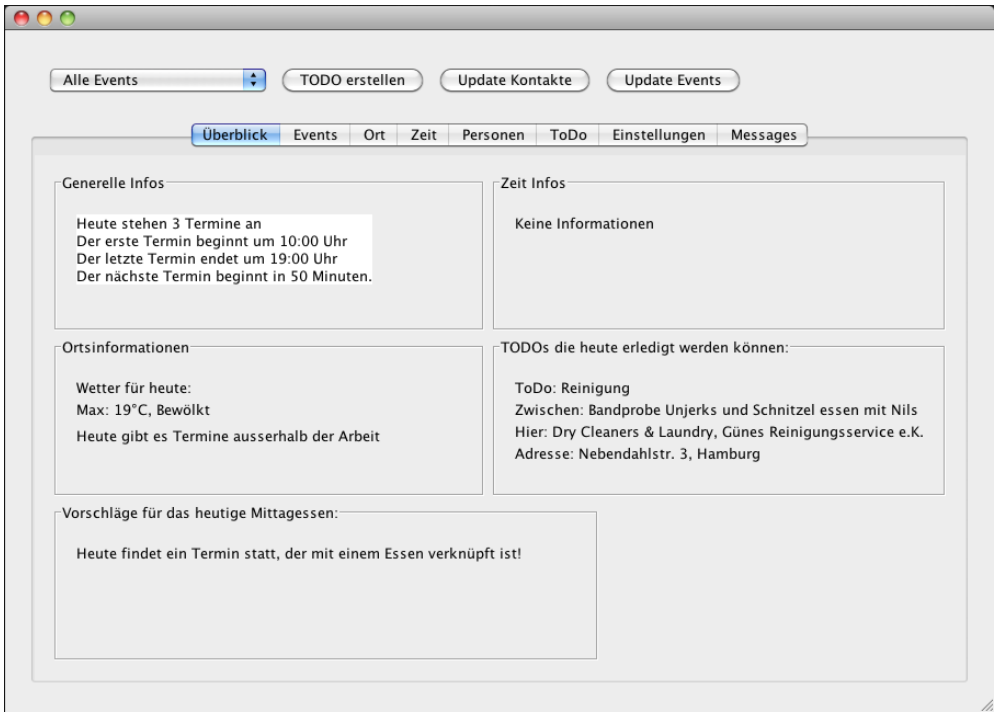

Abbildung 5.1: Übersichtsseite des Tagesplaners

Der Anwender erhält auf der Übersichtsseite einen generelle Beschreibung des Tageslaufs. Zeitliche Besonderheiten, wie etwa ein Feiertag und dadurch resultierende Einschränkungen, werden gesondert hervorgehoben. Neben aktuellen Wetterinformationen bekommt der Benutzer angezeigt, welche Aufgaben eingebettet in den Tagesablauf durchführbar sind. Anhand von Voreinstellungen, die festlegen, ob der Nutzer im Normalfall lieber innerhalb oder

außerhalb seiner Arbeitsstätte isst, werden wechselnde Lokalitäten für ein mögliches Mittagessen vorgeschlagen. Ist ein Termin wie in Abbildung 5.1 bereits mit einem Essen verknüpft, wird dieses kenntlich gemacht.

Neben den verschiedenen Übersichtsseiten hat der Anwender zusätzlichlich die Möglichkeit zur Dateneingabe. Er kann unter den Terminen des Tages ein „aktives“ Event auswählen, zu dem dann in den einzelnen Tabs weitere Informationen angezeigt werden. Außerdem kann der Anwender seine persönlichen Voreinstellungen ändern, ein neues To-do-Objekt anlegen oder die Daten des Servers manuell zu aktualisieren.

Abbildung 5.2: To-do-Objekt anlegen

Das Anlegen eines neuen To-do-Objekts (Abb. 5.2) erfolgt mittels Angabe eines Titels für die Task, einer Beschreibung und einem Ort, an dem die Aufgabe ausgeführt werden kann. Die Priorisierung der Task dient dazu, die Abarbeitung in einem zeitlich kurz- bis langfristigen Zeitrahmen vorzusehen.

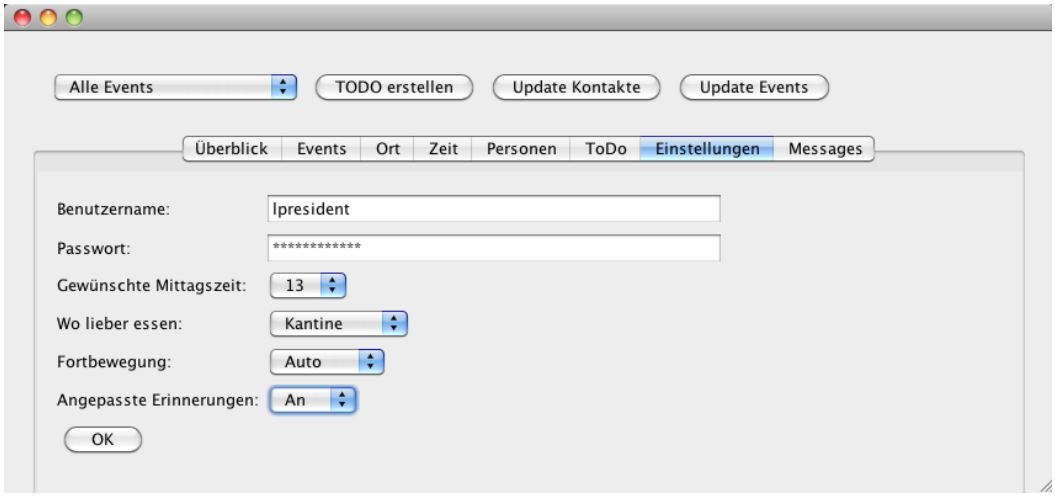

Abbildung 5.3: Einstellungsfenster der Anwendung

In den persönlichen Voreinstellungen kann der Anwender seinen Benutzernamen und sein Passwort für die Authentifizierung gegenüber der Kalender-Hostanwendung angeben, um Zugriff auf seinen Benutzeraccount zu erhalten. Neben den Einstellungen zum Mittagessen kann der Nutzer seine präferierte Art der Fortbewegung angeben (Auto, Fahrrad, zu Fuß). Diese wird dann bei der Berechnung der Fahrtwege berücksichtigt. Ebenso wird dem Benutzer mitgeteilt, seine Präferenz für den Tag zu ändern, falls es die äußeren Umstände erforderlich machen (etwa an einem Regentag mit dem Auto als mit dem Fahrrad zu fahren).

5.3 Logikschicht

5.3.1 Verarbeitung der Daten des *Digital Social Media*

Zur Verarbeitung der Daten aus sozialen Netzen kommt das auf Java basierende Framework Jena zum Einsatz. Der eigentliche Fokus von Jena liegt auf der Verarbeitung von semantisch angereicherten Daten. Es bietet die Möglichkeit, Datentripel über eine an SPARQL (s. Kap. 2.2.4) angelehnte Abfrage-Engine zu extrahieren. Auf diese Ergebnis-Tripel (Subjekt, Prädikat, Objekt) kann mittels „gewöhnlicher“ Java-Methoden zugegriffen werden. Die einmal abgefragten Daten können als lokales Modell in einer Datenbank oder lesbar für semantische Anwendungen in XML- oder N3-Notation[22] gesichert werden. Das komplette Modell aller Personen wird zur Laufzeit als Graph vorgehalten, an den die Anfragen oder Änderungen übermittelt werden können.
Über einen integrierten Schema-Generator kann jede eigene RDF-konforme Ontologie in das Framework integriert werden.

Im Rahmen dieser Arbeit wird das Jena Framework dazu benutzt, dass jeweilige Profil einer Person anhand der extrahierten Personendaten des DSM zu generieren. Diese liegen, bedingt durch die geringe Ausbreitung semantischer Annotationen im Netz, größtenteils nicht als semantisch angereicherte Daten vor. Die Verwaltung sämtlicher Personendaten in einem Modell hat im Gegensatz zur Verwaltung in einzelnen Java Objekten den Vorteil, das Anfragen zu bestimmten Ressourcen (also Subjekten oder Objekten) ein performanteres Ergebnis liefern, da keine Iterationen über jedes einzelne Personenobjekt erfolgen müssen.

5.3.2 To-do-Listen Verwaltung

Zur Realisierung des To-do-Listen Moduls wird auf die Verwendung externer Anwendungen zur Verwaltung der einzelnen To-do-Objekte verzichtet. Stattdessen werden die einzelnen

[22] http://www.w3.org/DesignIssues/Notation3.html

Aufgaben durch den implementierten Server verwaltet und in der servereigenen Persistenzschicht gesichert.

5.3.3 Alarm- und Reminder-Funktionalität

Die Darstellung visueller Erinnerungen an einen kommenden Termin erfolgt in Abhängigkeit der persönlichen Voreinstellungen des Anwenders. Zum einen besteht die Möglichkeit, den Alarm, wie es von bestehenden Kalendersystemen bekannt ist, zu einem festen Zeitpunkt vor dem Termin auszulösen. Die andere Möglichkeit besteht in der zeitlichen Anpassung des Reminders an den mit dem kommenden Termin verbundenem Fahrtweg. Beginnt ein Termin beispielsweise um 15:00 Uhr und der Fahrtweg vom Ort des vorhergehenden Events beträgt 30 Minuten, erfolgt eine Erinnerung an den nächsten Termin angepasst an diese halbe Stunde plus einen Puffer von 15 Minuten.

Eine zusätzliche Aufgabe, die durch dieses Modul behandelt wird, ist die Prüfung auf Änderungen von Terminen des Tages in der Kalender-Hostanwendung. Wird ein Termin verschoben oder entfällt dieser komplett, wird dieses dem Benutzer mitgeteilt. Die Anwendung prüft daraufhin ob der neu entstandene Zeitraum für andere Aufgaben genutzt werden kann, etwa die Bearbeitung eines To-do-Objekts.

5.4 Datenhaltungsschicht

5.4.1 Persistente Daten

In der persistenten Datenschicht werden die Profile der bekannten Personen, bereits abgefragte Ergebnisse zu bestimmten POIs sowie die erstellten To-do-Objekte und persönlichen Voreinstellungen des Benutzers abgelegt.

Die in der Logikschicht durch Jena erstellten Profildateien werden im XML Format abgelegt. Dieses erlaubt, losgelöst von der hier implementierten Anwendung, eine Weiterverwendung der Personenprofile durch weitere Anwendungen, die auf diese Dateien per Web zugreifen können.

Die Sicherung der POIs erfolgt mittels Java-Serialisierung der jeweiligen Objekte. Es besteht die Möglichkeit, diese Daten ebenfalls in semantischer Tripel-Notation abzulegen, da geeignete Ontologien für Ortsdaten existieren (beispielsweise eine Ontologie die das vCard Format widerspiegelt). Die abgefragten Daten stellen keine Zusammenstellung mehrerer Quellen dar und die Aktualität der Ergebnisse spielt eine wichtige Rolle. Daher werden diese von den entsprechenden standortbasierten Netzen erneut abgefragt. Für die Realisierung der

Anwendung ist die Sicherung der Ortsdaten wichtig, da die Logik Kenntnisse darüber benötigt, welche POIs dem Anwender bereits bekannt sind. Bereits in der Datenbank vorhandene POIs werden nach einem bestimmten Zeitraum mit aktuellen Daten überschrieben.
Die Sicherung der To-do-Objekte erfolgt ebenfalls per Java-Serialisierung.

5.4.2 Schnittstellen zu Webdiensten

Neben den persistent abgelegten Daten beinhaltet die Datenhaltungsschicht die Schnittstellen zu den externen Webdiensten, über die zusätzliche Informationen eingeholt werden. Für die Logikschicht ist es dabei irrelevant, ob ein Datensatz persistent vorliegt oder ob dieser erst angefordert werden muss. In beiden Fällen wird die gleiche Anfrage gestellt, worauf in der Datenhaltungsschicht eine Prüfung auf die Existenz des Datensatzes stattfindet. Die Antworten der Webdienste erfolgen im JSON Format. Zur Umwandlung in Java Objekte, die dann von der Software verwendet werden können, wird Google Gson[23] verwendet.

5.5 Evaluation

In diesem Kapitel soll die Funktionsweise der realisierten Anwendung anhand zweier typischer Anwendungsfälle evaluiert werden. Der erste Use-Case beschreibt die Suche nach einem POI für das Mittagessen, während sich der zweite mit der Erstellung eines neuen To-do-Objekts befasst.

5.5.1 Suche nach einem POI für das Mittagessen

Die Planung einer Lokalität für das Mittagessen richtet sich zunächst nach den persönlichen Voreinstellungen des Benutzers. Falls dieser „unterwegs“ als Option ausgewählt hat, sucht die Anwendung nach einem entsprechenden Lokal in der Nähe des vorhergehenden Terminortes.
Zunächst werden die Termine des Tages über den Kalenderadapter von Zimbra abgerufen. Im nächsten Schritt wird nach einem freien Zeitraum um die Mittagszeit gesucht. Kann kein passender Zeitraum gefunden werden bricht die Suche an dieser Stelle ab und der Anwender erhält einen Hinweis. Im regulären Fall, der auch durch Abb. 5.4 beschrieben wird, schließt sich an die Suche nach einem freien Zeitslot die Suche nach einem entsprechenden POI an. Existiert bereits ein passender Ort in der lokalen Datenbank wird dieser, falls er nicht bereits zu häufig in den vorhergehenden Wochen vorgeschlagen wurde, verwendet. Im Beispiel gibt

[23] http://code.google.com/p/google-gson/

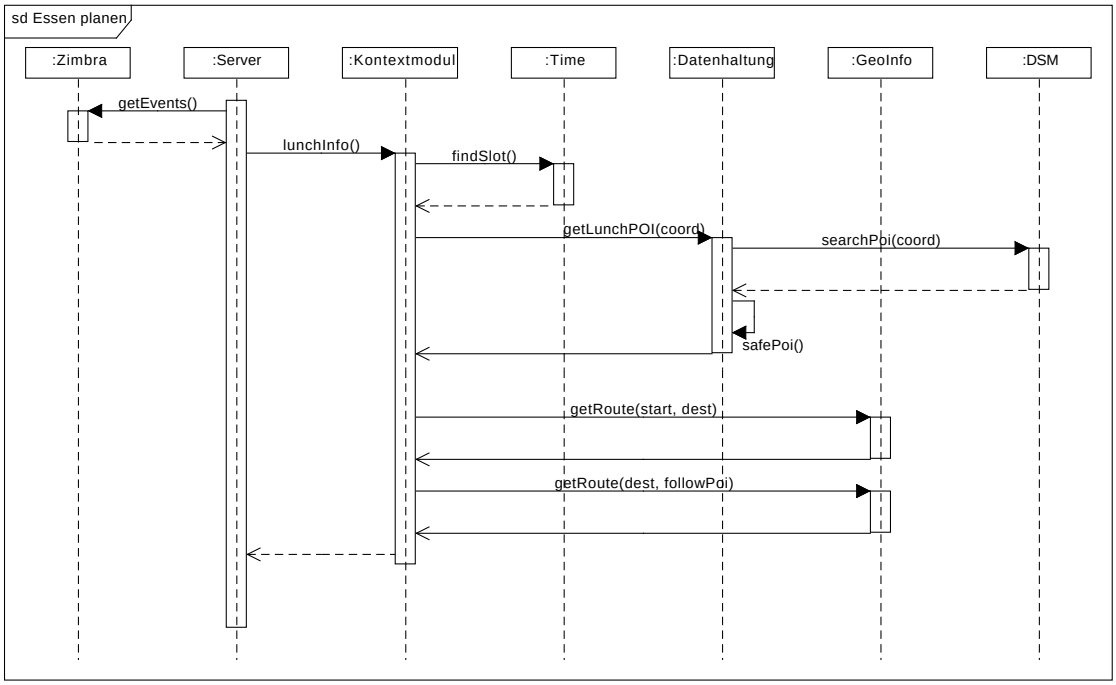

Abbildung 5.4: Suche nach einem POI für das Mittagessen

es noch keine Informationen, daher wird über die DSM Komponente eine Anfrage an Qype oder Foursquare gesendet. Die Resultate werden nach ihrer Bewertung sortiert und die drei bestbewertesten lokal gesichert.
Zur Planung der Anfahrt wird, unter Berücksichtigung des gewünschten Fortbewegungsmittels, über die Schnittstelle der Geoinformationsdienste eine Anfrage an Google Maps gestellt. Die Umkreissuche ist auf einen Radius beschränkt und so ausgelegt, dass mit Fahrtwegen und Essenszeit nicht mehr als eine Stunde zusätzlicher Zeitaufwand entsteht.

5.5.2 Erstellung eines To-do-Objekts

Der interne Programmablauf bei der Erstellung eines To-do-Objekts wird in Abb. 5.5 dargestellt. Die durch den Anwender eingegebenen Daten bestehend aus Betreff, gewünschtem Ort zur Bearbeitung und Priorität der Aufgabe werden zusammengefasst durch den Client zum Server mit dem Auftrag, ein To-do zu erstellen, übertragen.
Im ersten Schritt werden die Termine eines Tages abgerufen. Das Startdatum richtet sich dabei nach der Priorität der Aufgabe (ab sofort bis hin zu drei Wochen Vorlauf). Der nächste Schritt besteht in der Suche nach einem geeigneten Ort zur Bearbeitung der Aufgabe in der Nähe der eingetragenen Termine. Ist an einem gewählten Tag kein Termin vermerkt, werden

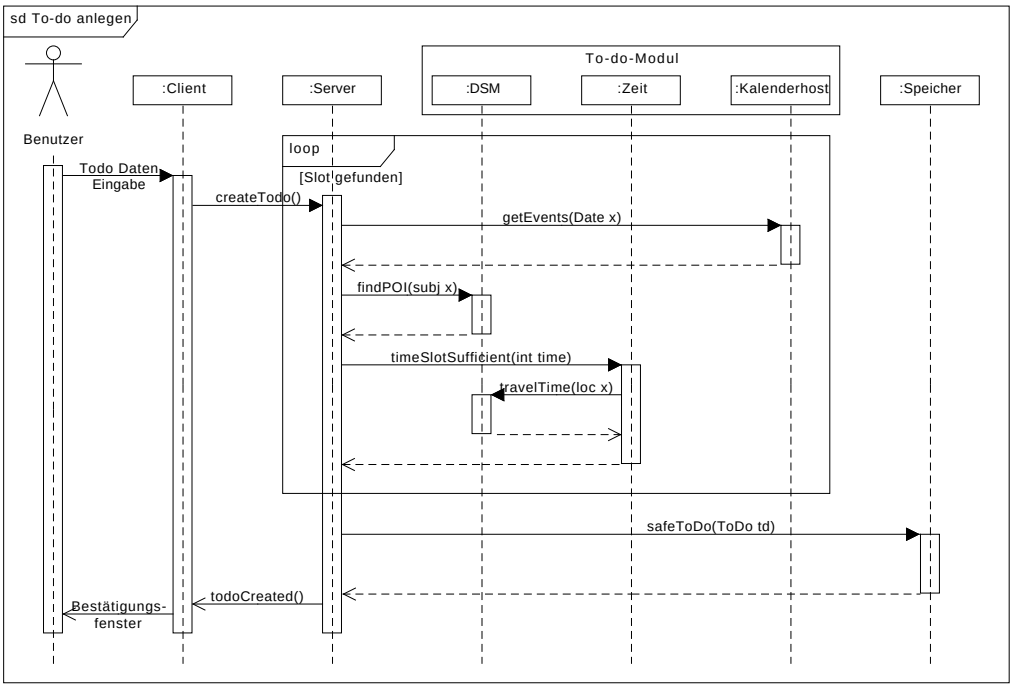

Abbildung 5.5: To-do-Objekt anlegen

die Orte zu Grunde gelegt, an denen sich der Anwender gewöhnlicherweise aufhält (unter der Woche die Arbeitsstätte, sowie über die ganze Woche an seiner Privatadresse, abhängig von der Uhrzeit).
Über die Anfrage an eine Instanz der Zeitklasse wird anhand von Öffnungszeiten und Anfahrtswegen geprüft, ob das mögliche To-do in den gewählten Tag integrierbar ist. Dieser Ablauf wird so lange wiederholt, bis ein passender Tag gefunden ist.
Das erstellte To-do-Objekt wird anschließend in der Persistenzsschicht gesichert. Sind alle Schritte erfolgreich durchlaufen erhält der Client eine Rückmeldung, die dem Benutzer dargestellt wird.

5.6 Fazit

Die Realisierung der in dieser Arbeit vorgestellten Anwendung erfolgte durch eine Serveranwendung sowie einen daran angebundenen Desktop-Client. Die im Design vorgestellten Komponenten konnten als solche implementiert werden. Die lose Kopplung der einzelnen Programmbestandteile durch das *Model-View-Presenter* Architekturmuster konnte ebenso umgesetzt werden.

Mit der Anwendung besteht die Möglichkeit, die Kalenderdaten des Zimbra Groupwaresystems auszulesen und angereichert durch weitere Daten zu präsentieren. Das System bedient sich hierbei des Adressbuchs von Zimbra, um die verknüpften Kontakte eines Events genauer darstellen zu können. Durch Verwendung verschiedener Quellen des DSM und weiterer geobasierter Dienste, die zu Beginn dieses Kapitels evaluiert wurden, erfolgen unter Angabe persönlicher Voreinstellungen die Planung des Tagesablaufs mit Berechnung der einzelnen Routen sowie die Darstellung von Empfehlungen für bestimmte POIs.
Durch eine Analyse bestimmter Keywords kann die Anwendung den Betreff eines Termins einordnen, diesen mit anderen verknüpfen und eventuell verlinkte Dokumente zusammengefasst bereitstellen.
Mit Hilfe einer Zeitkomponente erfolgt die Organisation aller wichtigen Zeitpunkte des Tages sowie die des größeren zeitlichen Rahmens aus Art des Tages hin zu jahreszeitlichen Randbedingungen. Zusätzlich können personalisierte oder statische Reminder den Anwender an kommende Termine erinnern.

Neben der Kalenderfunktion wurde ein Modul zur Verwaltung von To-do-Listen umgesetzt. Hier erfolgt ebenfalls, durch Prüfung auf bestimmte Schlüsselwörter, eine geobasierte Suche nach einem geeigneten POI zur Durchführung der Aufgabe. Anhand der eingestellten Priorisierung sucht die Anwendung in einer dadurch festgelegten Zeitspanne nach möglichen Zeiträumen, innerhalb derer die Aufgabe erledigt werden kann.

Bei der Realisierung der Anforderungen aus der Analyse haben sich vor allem Zimbra als Kalendersystem und die Anbindung der sozialen Netzwerke als problematisch erwiesen. Die Integration des Kalenderagenten zur Kommunikation mit dem Zimbra-Server erfolgte zwar ohne Probleme, ein Zugriff auf die vorliegenden Kalenderdaten kann jedoch zunächst nur lesend erfolgen. Der verwendete Agent bietet die Möglichkeit zur Erweiterung, wodurch ein Schreibzugriff durch eine erweiterte Version realisiert werden könnte. Eine Umsetzung dieses Funktionsumfangs konnte jedoch unter Berücksichtigung des zeitlichen Rahmens dieser Arbeit nicht mehr erfolgen.

Die Anbindung sozialer Netze enthält zwei wesentliche Herausforderungen, die es weiterhin zu bewältigen gilt. Zum einen das fehlende Wissen über mögliche Benutzernamen anderer Personen, zum anderen in der Menge der verfügbaren Daten. Die Verknüpfung von Benutzer- und realem Namen ist eines der Features, zu dessen Lösung die vorgestellte FOAF Ontologie herangezogen werden könnte. Die Verbreitung von entsprechend verfügbaren FOAF Profildateien im Internet und die automatische Generierung solcher Verbindungen ist jedoch derart gering, dass es kaum Sinn macht, diesen Ansatz zu verfolgen. Die Verknüpfung von Usernamen mit den Personen, die sich dahinter verbergen, musste daher bei der Umsetzung dieser Arbeit manuell erfolgen.
Mit der Anbindung des hier entwickelten Systems an die API von Facebook konnten nur die Namen und Facebook-internen IDs der Personen abgefragt werden, mit denen der Anwender in direkter Verbindung steht (seine „Freunde“). Es ist jedoch nicht möglich, Zugriff auf

die Freunde weiterer, an einem Termin beteiligten, Personen zu erhalten. Damit dieses realisiert werden kann, müsste die Integration eines Authentifizierungs-Plugins auf Facebook erfolgen. Hierdurch besteht die Möglichkeit, Dritten einen erweiterten Zugriff auf die eigenen Daten zu gewähren.

6 Schluss

6.1 Zusammenfassung

Den Kontext im Tagesablauf eines Menschen basierend auf seinen Kalendereinträgen zu erkennen um ihm so durch daraus ableitbare Informationen in seiner Planung unterstützend zur Seite zu stehen war das Ziel der in dieser Arbeit realisierten Anwendung zur Unterstützung einer kontextberücksichtigenden Tagesplanung.

Zur Entwicklung der vorgestellten Anwendung wurden anfänglich durch das Grundlagenkapitel (2) wichtige Begriffe und Technologien vorgestellt, die für das weitere Verständis erforderlich waren.

Das Analysekapitel (3) beschäftigte sich zunächst mit einer genaueren Betrachtung des Begriffs „Event“, um zu verdeutlichen, welche geisteswissenschaftlichen Ansätze sich hinter diesen simpel anmutenden Begriffen verbergen. Durch eine Analyse des Nutzungsverhaltens von Kalendern wurde ein Überblick darüber gegeben, warum Terminplanung überhaupt stattfindet und welche Eigenschaften die verschiedenen Kalendertypen ausmachen. Als hervorzuhebende Aspekte dieser Analyse haben sich dabei zum einen der Begriff der *prospektiven Erinnerung*, die den eigentlichen Vorgang der kognitiven Terminwahrnehmung und Planung widerspiegelt, zum anderen die Möglichkeit zur Kontextanalyse eines Termins nach bekannten Methoden, in diesem Fall der *5W1H* Mehode, herausgestellt.
Um ein realistisch konstruiertes Bild einer Tagesplanerapplikation zu erzeugen, wurde die Verwendung einer solchen Anwendung im Anschluss durch verschiedene Szenarien beschrieben. Ein Überblick über vergleichbare Arbeiten, die sich mit verschiedenen Aspekten im Umgang mit Kalendern und Kontextbezogenheit beschäftigt haben, lieferte eine weitere Informationsquelle aus denen im Folgenden ein Katalog an Anforderungen für ein kontextbasierendes Tagesplanersystem entwickelt wurde.
Die zentrale Komponente dieser Anwendung, welche sich in die bestehende Infrastruktur des Benutzers integrieren soll, ist ein Modul zur Analyse des Kontextes einer Person in seinem Tagesablauf. Die Anforderung an diese Komponente besteht darin, die Termininformationen sowie alle weiteren Daten des Tagesablaufs durch geeignete Zusatzinformationen über Personen, Ort, Zeit und Anlass der Events und den umgebenden Tagesablauf zu erweitern, um dem Anwender so eine unterstützende Funktion zu bieten. Eine weitere Komponente zur

Verwaltung von To-do-Listen, deren Bearbeitung eingebettet in den Tagesablauf des Nutzers erfolgen soll, ist ein weiterer Aspekt der Anforderungen.

Diese Anforderungen in ein entsprechendes Design einzubetten war Bestandteil des Designkapitels (4). Hier wurden die einzelnen Funktionsblöcke und ihre Aufgaben definiert. Die zu realisierende Anwendung gliedert sich in drei Hauptkomponenten: Ein Modul zur Kontextverarbeitung sammelt über externe Quellen zusätzliche Informationen zur Kontextidentifizierung, ein Modul zur Bearbeitung von To-do-Objekten nimmt die Planung und Verwaltung von Aufgabenlisten, eine Schnittstelle zur zentralen Kalenderanwendung des Benutzers kommuniziert mit dem Kalenderhost. Über Adapter zu externen Quellen des DSM und der Geobasiertheit werden zusätzliche Informationen abgerufen.
Als geeignete Architektur zur Einbettung dieser Komponenten wurde eine Drei-Schichten-Architektur gewählt, in der ein *Model-View-Presenter* Muster die einzelnen Bereiche repräsentiert.

Die zuvor definierten Anforderungen an die einzelnen Dienste, welche die Anwendung benötigt, wurden im anschließenden Kapitel Realisierung (5) in Form konkreter externer Services umgesetzt. Mit ihnen erfolgte die Umsetzung des Systems zur Erfüllung der entwickelten Requirements. Durch eine abschließende Evaluation anhand zweier Anwendungsfälle wurde die Anwendung auf Tragfähigkeit des umgesetzten Designs geprüft.

Die vergleichbaren Arbeiten haben bereits gezeigt, dass sich verschiedene Entwickler mit der Thematik kontextbezogener Kalender befasst haben. Es gibt allerdings keinen Ansatz, der sich mit allen kontextuellen Aspekten von Tagesplanung und einzelnen Terminen gleichermaßen auseinandersetzt. Die in dieser Arbeit gewonnenen Erkentnisse für ein Gesamtdesign und den möglichen Funktionsumfang sowie die als Ergebnis realisierte Anwendung zur kontextberücksichtigenden Tagesplanung zeigen als ein erster Schritt zur Umsetzung eines umfassenden Designs ein großes Potental zur Weiterentwicklung auf.

Dem möglichen Funktionsumfang der Anwendung sind durch die immer breitere Auswahl an Diensten im Web kaum Grenzen gesetzt. Eine Realisierung aller in 3.2 vorgestellten Szenarien wäre bei Integration des Systems in eine intelligente Wohnumgebung wie dem Living Place der Hochschule für Angewandte Wissenschaften Hamburg und Integration weiterer externer Komponenten vollständig umsetzbar.

Der nächste Schritt zur Erweiterung des Funktionsumfangs sollte die Behandlung der in Kap. 5.6 angeführten Probleme sein. Vor allem ein schreibender Zugriff auf den Kalender des Anwenders wäre interessant um Termine in der Hostanwendung durch den Client ändern zu können. Zusätzlich könnten die Termine zur geplanten Erledigung der To-do-Objekte in den Kalender eingetragen werden, wodurch sie auch unabhängig von der Tagesplaneranwendung wahrgenommen werden können.

Bei den Problemen, die im Zusammenhang mit der Einbindung sozialer Netzwerke aufgetreten sind, gilt es zum einen, die momentanen Entwicklungstrends des *Semantic Web* weiter zu beobachten, zum anderen könnte durch Adapter zu weiteren Quellen das Informationsspektrum verbreitert werden.

Ein weiterer Aspekt bei der Weiterentwicklung sollte die Realisierung eines Clients sein, der auf einem mobilen Endgerät zum Einsatz kommt. Hierdurch können andere Facetten der Tagesplanung näher berachtet werden, besonders die Nutzung von Ortungsdiensten (vgl. 3.2.5 der Szenarien) als zusätzliche Datenquelle.
In der Analyse hat sich herausgestellt, dass Menschen trotz eines elektronischen Kalenders immer noch Papierkalender und Notizzettel für ihre Termine benutzen, da diese Papiervarianten eine freie Gestaltung der Inhalte, die sog. *Paper-Trails* (vgl. hierzu Kap. 3.1.5) auf dem vorhandenen Kalenderblatt oder Zettel erlauben. Multitouchfähige Oberflächen von Smartphones erlauben die Realisierung einfacher Zeichenfunktionen. Die Umsetzung einer frei gestaltbaren Fläche im Kalendereintrag wäre hier sicherlich ein interessantes Feature.

6.2 Fazit

> *„I express my network in a FOAF file, and that is a start of the revolution.“*
> - Berners-Lee (2007)

Mit diesem Zitat beschreibt Tim Berners-Lee, Begründer des World Wide Web, die revolutionären Möglichkeiten, die durch das *Semantic Web* vor allem im Bereich der Vernetzung von Personen möglich sind. Wie bereits in dieser Arbeit gezeigt existieren mit FOAF zur Beschreibung von Personenprofilen und SIOC zur Beschreibung von *user generated content* zwei umfassende und von Seiten des W3C Konsortiums anerkannte Ontologien für Plattformen aus dem DSM bereits seit ungefähr einer Dekade. Allerdings finden sie in den nutzerstärksten Netzwerken keine Verwendung. Stattdessen setzen diese auf eigene Entwicklungen für deren Implementierung in eigene Anwendungen die Realisierung zusätzlicher Software erforderlich ist.
Dabei besteht seitens kommerzieller Plattformen durchaus Interesse an der öffentlichen Annotierung von semantischen Metainformationen, vor allem in der Werbe- und Konsumartikelbranche. So benutzt die amerikanische Handelskette BestBuy[1] bereits seit mehreren Jahren eigene Ontologien, um Suchmaschinen eine effizientere Suche über Produkte, deren Attribute sowie Preise zu ermöglichen. Die Entwicklungstrends im Bereich des *Semantic Web* sollten daher weiter beobachtet werden, da dessen Möglichkeiten einen beträchtlichen Beitrag zur Automatisierung bei der Sammlung von Personendaten und verknüpften Inhalten liefern können.

[1] http://www.bestbuy.com/site/index.jsp

Es ist jedoch ebenso erforderlich, sich mit den möglichen Auswirkungen dieser Vernetzung auseinanderzusetzen. Werden persönliche Informationen „einfach so“ maschinenlesbar veröffentlicht, verliert der Mensch die Kontrolle darüber, welche Informationen andere über ihn erhalten. Die Verfügbarkeit dieser Daten sollte daher weiterhin durch den Nutzer steuerbar sein.
Ein weiterer kritisch zu betrachtender Punkt sind die Folgen der automatisierten Tageplanung:

> *„I claim not to have controlled events, but confess plainly that they have controlled me.“*
>
> - Abraham Lincoln

Das Zitat von Abraham Lincoln, der sich so bereits in einer Zeit lange vor der globalen Vernetzung über eine „Fremdverplanung“ durch andere beklagt hat, wirft die Frage auf, in wiefern sich der Mensch seinen Tagesablauf vorschreiben lassen will, sei es durch andere Menschen oder durch Computer. Damit geht ebenso die Frage einher, was eigentlich ein Gegenüber ausmacht, dass einem die eigene Terminplanung abnimmt.
Einem anderen Menschen die Planung des eigenen Tages in die Hand zu geben erfordert vor allem Faktoren wie genaue Personenkenntnis und eine Vertauensbasis zwischen den beteiligten Personen. Ob diese Anforderungen durch eine Maschine genauso erfüllt werden können bleibt offen. Ebenso die Fragestellung, ob eine Software eine genauso private Sicht auf die eigene Persönlichkeit haben kann, um unseren Tagesablauf nach unseren Vorlieben und unserer Tagesform entsprechend zu planen, wie ein Mensch.
Es existieren kontextsensitive Systeme mit der Intention, die emotionalen Befindlichkeiten eines Menschen anhand der über ihn verfügbaren Informationen zu erfassen und zu verwenden. Biundo u. Wendemuth (2010) stellen diese als sog. *Companion-Systeme* vor, zu deren Entwicklung Eigenschaften wie Vertrauenswürdigkeit und Individualität in den Mittelpunkt gerückt werden. Die realisierten Systeme sollen dem Anwender als persönliche Assistenten dienen und auf seine Bedürfnisse eingehen können.
Trotzdem bleibt die Frage bestehen, ob die Planung des eigenen Alltags durch Computer durchgeführt werden sollte. In der Literatur zur prospektiven Erinnerung und zum Nutzverhalten von Kalendern wird vermehrt diskutiert, welche Auswirkungen proaktive Software auf die Erinnerungsgabe des Menschen hat. Versuchspersonen haben dabei die Vermutung geschildert, sich nur noch schlecht oder gar nicht mehr an Dinge erinnern zu können, seit sie Software oder andere Hilfsmittel benutzen, die ihnen die Aufgabe der Erinnerung abnehmen.

Unsere Gesellschaft befindet sich im stetigen Wandel und die einst klar definierte Grenze zwischen Arbeits- und Privatleben verwischt. Neben der bereits in der Einleitung erwähnten Koordination der so entstehenden verschiedenartigen Termine verändern sich zusätzlich die Randbedingungen innerhalb des Berufsalltags. Arbeitsgruppen befinden sich nicht mehr an

einem Ort, sondern arbeiten als *virtuelle Teams* verteilt über den Globus in verschiedenen Zeitzonen. *Freelancer* ersetzen den „klassischen“ Arbeitnehmer und bekommen Aufträge zugeteilt, die sie ausserhalb von Firmenstandort und Unternehmensgefüge bearbeiten.
Die Anforderungen an die Terminplanung verändern sich hierdurch zusätzlich und führen zu der Diskussion, ob die gewohnte Planung durch die Kernelemente wie Ort und Zeit in Zukunft überhaupt noch den richtigen Ansatz darstellt.

Wie und wodurch der Mensch seinen Tagesablauf plant, bleibt ihm bislang zumeist selber überlassen. Bei der weiteren Entwicklung von kontextsensitiver Kalendersoftware sollte stets darauf geachtet werden, dem Anwender zum einen genug Freiheiten und Eingriffsmöglichkeiten in das System zu bieten, zum anderen sollte für den Nutzer klar ersichtlich sein, welche Dinge ihm von der Software abgenommen werden.

Literaturverzeichnis

[ARD und ZDF 2012] ARD UND ZDF: *ARD und ZDF Onlinestudie, Zugriffsdatum: 18.06.2012*. http://www.ard-zdf-onlinestudie.de/, 2012

[Ashton 2009] ASHTON, K.: That 'Internet of Things' Thing. In: *RFID Journal* 22 (2009), S. 97–114

[Barnkow 2010] BARNKOW, Lorenz: Eine Multitouch-faehige Kuechentheke: Vorbereitende Arbeiten fuer den Tagesplaner / Hochschule fuer Angewandte Wissenschaften Hamburg. Version: 2010. http://users.informatik.haw-hamburg.de/~ubicomp/projekte/master2010-proj1/barnkow.pdf. 2010. – Forschungsbericht

[Beaudouin-Lafon u. a. 1999] BEAUDOUIN-LAFON, M. ; DEWA, Prasun ; EHRLICH, Kate ; DOURISH, Paul ; ELLI, Clarence ; GREENBERG, Saul ; JOHNSON, Chris ; PRAKAS, Atul ; ROSEMAN, Mark ; MACKAY, Wendy E. ; ISHII, Hiroshi ; BEAUDOUIN-LAFON, Michel (Hrsg.): *Computer supported co-operative work*. Wiley, 1999

[Berners-Lee 2007] BERNERS-LEE, Tim: *Giant Global Graph, Zugriffsdatum: 04.06.2012*. http://dig.csail.mit.edu/breadcrumbs/node/215, 11 2007

[Biundo u. Wendemuth 2010] BIUNDO, S. ; WENDEMUTH, A.: Von kognitiven technischen Systemen zu Companion-Systemen. In: *KI-Künstliche Intelligenz* 24 (2010), Nr. 4, S. 335–339

[Blandford u. Green 2001] BLANDFORD, A. E. ; GREEN, T. R. G.: Group and Individual Time Management Tools: What You Get is Not What You Need. In: *Personal Ubiquitous Comput.* 5 (2001), Januar, Nr. 4, 213–230. http://dx.doi.org/10.1007/PL00000020. – DOI 10.1007/PL00000020. – ISSN 1617–4909

[Blogger] BLOGGER: *Blogger*. http://www.blogger.com, . – [Online; Zugriffsdatum: 10. Mai 2012]

[Bry u. a. 2010] BRY, Francois ; CAP, Clemens ; DAHM, Ingo ; MAINTZ, Julia ; SCHAFFERT, Sebastian: Dagstuhl Manifesto. In: *Informatik Spektrum* 33 (2010), Nr. 4, S. 404–416

[Casati u. Varzi 2007] CASATI, Roberto ; VARZI, Achille: Event Concepts. (2007). HAL: http://jeannicod.ccsd.cnrs.fr/ijn_00168510/en/

[Crabtree u. a. 2003] CRABTREE, Andy ; HEMMINGS, Terry ; RODDEN, Tom ; MARIANI, John: Informing the development of calendar systems for domestic use. In: *Proceedings of the eighth conference on European Conference on Computer Supported Cooperative Work*. Norwell, MA, USA : Kluwer Academic Publishers, 2003 (ECSCW'03), 119–138

[Delicious] DELICIOUS: *Delicious*. http://delicious.com/

[Demiris u. a. 2005] DEMIRIS, A ; VLAHAKIS, V ; MAKRI, A ; PAPAIOANNOU, M ; IOANNIDIS, N: intGuide: A platform for context-aware services featuring augmented-reality, based on the outcome of European Research Projects. In: *Signal Processing Image Communication* 20 (2005), Nr. 9-10, 927–946. http://linkinghub.elsevier.com/retrieve/pii/S092359650500072X

[Dey u. a. 1999] DEY, Anind ; ABOWD, Gregory ; BROWN, Peter ; DAVIES, Nigel ; SMITH, Mark ; STEGGLES, Pete: Towards a Better Understanding of Context and Context-Awareness. Version: 1999. http://dx.doi.org/10.1007/3-540-48157-5_29. In: GELLERSEN, Hans-W. (Hrsg.): *Handheld and Ubiquitous Computing* Bd. 1707. Springer Berlin / Heidelberg, 1999. – ISBN 978–3–540–66550–2, 304-307. – 10.1007/3-540-48157-5_29

[Dey 2001] DEY, Anind K.: Understanding and Using Context. In: *Personal and Ubiquitous Computing* 5 (2001), 4-7. http://dx.doi.org/10.1007/s007790170019. – ISSN 1617–4909. – 10.1007/s007790170019

[Directions API] DIRECTIONS API: *Directions API*. https://developers.google.com/maps/documentation/directions/, . – [Online; Zugriffsdatum: 10. Mai 2012]

[Ellison u. a. 2007] ELLISON, Nicole B. ; STEINFIELD, Charles ; LAMPE, Cliff: The Benefits of Facebook ÒFriends:Ó Social Capital and College StudentsÕ Use of Online Social Network Sites. In: *Journal of Computer-Mediated Communication* 12 (2007), Nr. 4, 1143–1168. http://dx.doi.org/10.1111/j.1083-6101.2007.00367.x. – DOI 10.1111/j.1083–6101.2007.00367.x. – ISSN 1083–6101

[Epinions] EPINIONS: *Epinions*. http://www.epinions.com/

[Facebook] FACEBOOK: *Facebook*. http://www.facebook.com/

[Fleisch u. a. 2005] FLEISCH, E. ; FLEISCH, E. ; FLEISCH, E.: *Das Internet der Dinge*. Bd. 1. Springer, 2005

[FOAF-a-Matic] FOAF-A-MATIC: *FOAF-a-Matic*. FOAF-a-Matic

[FOAF Projekt 2000] FOAF PROJEKT: *The Friend of a Friend Projekt, Zugriffsdatum: 04.06.2012*. http://www.foaf-project.org/, 2000

[Foursquare] FOURSQUARE: *Foursquare*. https://foursquare.com, . – [Online; Zugriffsdatum: 10. Mai 2012]

[Francis-Smythe u. a. 2006] FRANCIS-SMYTHE, J.A. ; GLICKSOHN, J.E. ; MYSLOBODSKY, M.S. ; MICHAEL, S. (Hrsg.): *Timing the future: The case for a time-based prospective memory.* World Scientific Publishing, 2006

[Gkekas u. a. 2007] GKEKAS, Georgios ; KYRIKOU, Anna ; IOANNIDIS, Nikos: A smart calendar application for mobile environments. In: *Proceedings of the 3rd international conference on Mobile multimedia communications*. ICST, Brussels, Belgium, Belgium : ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007 (MobiMedia '07). – ISBN 978–963–06–2670–5, 70:1–70:5

[Google Maps] GOOGLE MAPS: *Google Maps API-Webdienste, Zugriffsdatum: 16.06.2012*. https://developers.google.com/maps/documentation/webservices/index?hl=de,

[Google Maps API] GOOGLE MAPS API: *Google Maps API*. https://developers.google.com/maps/, . – [Online; Zugriffsdatum: 10. Mai 2012]

[ISO 9126 1991] *Information technology - Software Product Evaluation - Quality characteristics and guidelines for their use*. 1991

[Jones 2007] JONES, William: Personal Information Management. In: *Annual Rev. Info. Sci & Technol.* 41 (2007), Dezember, Nr. 1, 453–504. http://dx.doi.org/10.1002/aris.144.v41:1. – DOI 10.1002/aris.144.v41:1. – ISSN 0066–4200

[Kim 1998] KIM, J.: Events as property exemplifications. In: *Contemporary Readings in the Foundations of Metaphysics* (1998), S. 310–326

[Kincaid u. a. 1985] KINCAID, Christine M. ; DUPONT, Pierre B. ; KAYE, A. R.: Electronic calendars in the office: an assessment of user needs and current technology. In: *ACM Trans. Inf. Syst.* 3 (1985), Januar, Nr. 1, 89–102. http://dx.doi.org/10.1145/3864.3868. – DOI 10.1145/3864.3868. – ISSN 1046–8188

[Kvavilashvili u. Fisher 2007] KVAVILASHVILI, L. ; FISHER, L.: Is time-based prospective remembering mediated by self-initiated rehearsals? Role of incidental cues, ongoing activity, age, and motivation. In: *Journal of Experimental Psychology: General* 136 (2007), Nr. 1, S. 112

[last.fm] LAST.FM: *last.fm*. http://www.last.fm

[Lee 2010] LEE, Alison: Exploiting context for mobile user experience. In: *Proceedings of the 1st Workshop on Semantic Models for Adaptive Interactive Systems (SEMAIS)*, 2010

[Lee u. a. 2001] LEE, T.B. ; HENDLER, J. ; LASSILA, O. u. a.: The semantic web. In: *Scientific American* 284 (2001), Nr. 5, S. 34–43

[Levine 1979] LEVINE, Alfred B.: *Electronic Calendar and Diary*. http://www.google.de/patents/US4162610. Version: 1979

[Lin u. a. 2004] LIN, Min ; LUTTERS, Wayne G. ; KIM, Tina S.: Understanding the micronote lifecycle: improving mobile support for informal note taking. In: *Proceedings of the SIGCHI conference on Human factors in computing systems*. New York, NY, USA : ACM, 2004 (CHI '04). – ISBN 1–58113–702–8, 687–694

[LinkedIn] LINKEDIN: *LinkedIn*. http://de.linkedin.com/

[Lovett u. a. 2010] LOVETT, Tom ; O'NEILL, Eamonn ; IRWIN, James ; POLLINGTON, David: The calendar as a sensor: analysis and improvement using data fusion with social networks and location. In: *Proceedings of the 12th ACM international conference on Ubiquitous computing*. New York, NY, USA : ACM, 2010 (Ubicomp '10). – ISBN 978–1–60558–843–8, 3–12

[Microsoft Outlook] MICROSOFT OUTLOOK: *Microsoft Outlook*. http://office.microsoft.com/de-de/outlook/

[Mueller 2000] MUELLER, Erik T.: A calendar with common sense. In: *Proceedings of the 5th international conference on Intelligent user interfaces*. New York, NY, USA : ACM, 2000 (IUI '00). – ISBN 1–58113–134–8, 198–201

[Mueller 2003] MUELLER, Erik T.: *ThoughtTreasure: A natural language/commonsense platform*. http://xenia.media.mit.edu/~mueller/papers/tt.html. Version: August 2003

[Neustaedter u. a. 2009] NEUSTAEDTER, Carman ; BRUSH, A. J. B. ; GREENBERG, Saul: The calendar is crucial: Coordination and awareness through the family calendar. In: *ACM Trans. Comput.-Hum. Interact.* 16 (2009), April, Nr. 1, 6:1–6:48. http://dx.doi.org/10.1145/1502800.1502806. – DOI 10.1145/1502800.1502806. – ISSN 1073–0516

[Palen 1999] PALEN, Leysia: Social, individual and technological issues for groupware calendar systems. In: *Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit*. New York, NY, USA : ACM, 1999 (CHI '99). – ISBN 0–201–48559–1, 17–24

[Payne 1993] PAYNE, Stephen J.: Understanding calendar use. In: *Hum.-Comput. Interact.* 8 (1993), Juni, Nr. 2, 83–100. http://dx.doi.org/10.1207/s15327051hci0802_1. – DOI 10.1207/s15327051hci0802_1. – ISSN 0737–0024

[Perry u. Wolf 1992] PERRY, Dewayne E. ; WOLF, Alexander L.: Foundations for the study of software architecture. In: *SIGSOFT Softw. Eng. Notes* 17 (1992), Oktober, Nr. 4, 40–52. http://dx.doi.org/10.1145/141874.141884. – DOI 10.1145/141874.141884. – ISSN 0163–5948

[Physikalisch-Technische Bundesanstalt] PHYSIKALISCH-TECHNISCHE BUNDESANSTALT: *Physikalisch-Technische Bundesanstalt, Zugriffsdatum: 27.06.2012*. http://www.ptb.de/cms/fachabteilungen/abt4/fb-44/ag-441/darstellung-der-gesetzlichen-zeit/wann-ist-ostern.html

[Qype] QYPE: *Qype*. http://www.qype.com/, . – [Online; Zugriffsdatum: 10. Mai 2012]

[RFC5546 2009] RFC5546: Internet Calendaring and Scheduling Core Object Specification (iCalendar) / Internet Engineering Task Force. Version: September 2009. http://tools.ietf.org/rfc/rfc5545.txt. 2009 (5546). – RFC. – 169 S.

[RFC6350 2011] RFC6350: vCard Format Specification / Internet Engineering Task Force. Version: August 2011. http://tools.ietf.org/rfc/rfc6350.txt. 2011 (6350). – RFC. – 73 S.

[Schatten u. a. 2010] SCHATTEN, A. ; BIFFL, S. ; DEMOLSKY, M. ; GOSTISCHA-FRANTA, E. ; ÖSTREICHER, T. ; WINKLER, D.: Best Practice Software-Engineering. In: *Eine praxiserprobte Zusammenstellung von komponentenorientierten Konzepten, Methoden und Werkzeugen. Heidelberg: Spektrum*, 2010

[Second Life] SECOND LIFE: *Second Life*. http://secondlife.com/

[Seiie u. Woontack 2005] SEIIE, Jang ; WOONTACK, Woo: Unified Context Describing User-Centric Situation:Who Where When What How and Why / GIST U-VR Lab., GIST U-VR Lab. 2005 (60(2005-UBI-008)). – Forschungsbericht

[Smith 2007] SMITH, D.A.: *EventMinder: A Personal Calendar Assistant That Understands Events*, Massachusetts Institute of Technology, Diplomarbeit, 2007. http://web.mit.edu/~dsmit/Public/dsmith-ms.pdf

[Tanenbaum u. Steen 2007] TANENBAUM, A.S. ; STEEN, M.: *Verteilte Systeme. 2., Aufl.* 2007

[Tungare u. a. 2008] TUNGARE, Manas ; PÉREZ-QUIÑONES, Manuel A. ; SAMS, Alyssa: An Exploratory Study of Calendar Use. In: *CoRR* abs/0809.3447 (2008)

[Twitter] TWITTER: *Twitter*. https://twitter.com/, . – [Online; Zugriffsdatum: 10. Mai 2012]

[W3C 2004] W3C: *Recource Description Framework, Zugriffsdatum: 04.06.2012*. 2 2004

[W3C 2008] W3C: *SPARQL Query Language for RDF*. http://www.w3.org/TR/rdf-sparql-query/, 1 2008

[Waldmann 2008] WALDMANN, Michael R.: Kategorisierung und Wissenserwerb. (2008). http://www.pedocs.de/volltexte/2009/744/

[Weiser 1991] WEISER, Mark: The computer for the 21st century. In: *SIGMOBILE Mob. Comput. Commun. Rev.* 3 (1991), Juli, Nr. 3, 3–11. http://dx.doi.org/10.1145/329124.329126. – DOI 10.1145/329124.329126. – ISSN 1559–1662

[Wiese 2007] WIESE, Bettina S.: Work-Life-Balance. Version: 2007. http://dx.doi.org/10.1007/978-3-540-71637-2_13. In: MOSER, Klaus (Hrsg.): *Wirtschaftspsychologie*. Springer Berlin Heidelberg, 2007. – ISBN 978–3–540–71637–2, 245-263. – 10.1007/978-3-540-71637-2_13

[Wikipedia 2012a] WIKIPEDIA: *Resource Description Framework, Zugriffsdatum: 04.06.2012*. 6 2012

[Wikipedia 2012b] WIKIPEDIA: *Semantically-Interlinked Online Communities, Zugriffsdatum: 04.06.2012*. http://en.wikipedia.org/wiki/Semantically-Interlinked_Online_Communities, 2012

[Wordpress] WORDPRESS: *Wordpress*. http://wpde.org/, . – [Online; Zugriffsdatum: 10. Mai 2012]

[Xing] XING: *Xing*. http://www.xing.com/, . – [Online; Zugriffsdatum: 10. Mai 2012]

[Youtube] YOUTUBE: *Youtube*. http://www.youtube.com/

Anhang A

FOAF

```
<rdf:RDF
  xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
  xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  xmlns:foaf="http://xmlns.com/foaf/0.1/">

<foaf:PersonalProfileDocument rdf:about="">
  <foaf:maker rdf:resource="#me"/>
  <foaf:primaryTopic rdf:resource="#me"/>
</foaf:PersonalProfileDocument>

<foaf:Person rdf:ID="me">
  <foaf:name>Max Mustermann</foaf:name>
  <foaf:givenname>Max</foaf:givenname>
  <foaf:family_name>Mustermann</foaf:family_name>
  <foaf:nick>Maxu</foaf:nick>
  <foaf:mbox_sha1sum>bf73ae18b9d235e5f3ea1661299d2abe620124af</foaf:mbox_sha1sum>
  <foaf:homepage rdf:resource="http://www.mustermann.de"/>
  <foaf:phone rdf:resource="tel:+494012345678"/>
  <foaf:workplaceHomepage rdf:resource="http://www.haw-hamburg.de"/>
  <foaf:workInfoHomepage rdf:resource="Student"/>
  <foaf:schoolHomepage rdf:resource="http://www.musterschule.de"/>

  <foaf:knows>
    <foaf:Person>
      <foaf:name>Helmine Musterfrau</foaf:name>
      <foaf:mbox_sha1sum>241e72ae1994dfa6db55c585b50493a1e00ba179</foaf:mbox_sha1sum>
      <rdfs:seeAlso rdf:resource="http://www.haw-hamburg.de/foaf/#alexvette"/>
    </foaf:Person>
  </foaf:knows>
  <foaf:knows>
    <foaf:Person>
      <foaf:name>Hermann Muster</foaf:name>
      <foaf:mbox_sha1sum>e4ff536f3f099d638b004d6987d9c787299cf1df</foaf:mbox_sha1sum>
    </foaf:Person>
  </foaf:knows>
</foaf:Person>
</rdf:RDF>
```

Listing 1: FOAF Beispiel als XML Notation

SIOC

```
<sioc:Post rdf:about="http://johnbreslin.com/blog/2006/09/07/creating-connections-between-
    discussion-clouds-with-sioc/">
    <dc:title>Creating connections between discussion clouds with SIOC</dc:title>
    <dcterms:created>2006-09-07T09:33:30Z</dcterms:created>
    <sioc:has_container rdf:resource="http://johnbreslin.com/blog/index.php?sioc_type=site#
        weblog"/>

    <sioc:has_creator>
        <sioc:UserAccount rdf:about="http://johnbreslin.com/blog/author/cloud/" rdfs:label=
            "Cloud">
            <rdfs:seeAlso rdf:resource="http://johnbreslin.com/blog/index.php?sioc_type=
                user&sioc_id=1"/>
        </sioc:UserAccount>
    </sioc:has_creator>

    <foaf:maker rdf:resource="http://johnbreslin.com/blog/author/cloud/#foaf"/>

    <sioc:content>SIOC provides a unified vocabulary for content and interaction
        description</sioc:content>
    <sioc:topic rdfs:label="Semantic Web" rdf:resource="http://johnbreslin.com/blog/
        category/semantic-web/"/>
    <sioc:topic rdfs:label="Blogs" rdf:resource="http://johnbreslin.com/blog/category/blogs
        /"/>
    <sioc:has_reply>
        <sioc:Post rdf:about="http://johnbreslin.com/blog/2006/09/07/creating-connections-
            between-discussion-clouds-with-sioc/#comment-123928">
            <rdfs:seeAlso rdf:resource="http://johnbreslin.com/blog/index.php?sioc_type=
                comment&sioc_id=123928"/>
        </sioc:Post>
    </sioc:has_reply>
</sioc:Post>
```

Listing 2: SIOC Beispiel als XML Notation (Quelle: Wikipedia (2012b))

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 5. Juli 2012	
Ort, Datum	Unterschrift