Technische Universität Kaiserslautern
Fachbereich Informatik
Arbeitsgruppe Wissensbasierte Systeme
Prof. Dr. Prof. h.c. Andreas Dengel

PIMO Diary: Tagebuch-Generierung aus persönlichen Informationsmodellen

Diplomarbeit

in Kooperation mit dem
Deutschen Forschungszentrum für
Künstliche Intelligenz (DFKI) GmbH

Sommersemester 2014

Christian Jilek

Studiengang:	Wirtschaftsingenieurwesen Fachrichtung Informatik
Erstprüfer:	Prof. Dr. Prof. h.c. Andreas Dengel
Zweitprüfer:	Dr. Heiko Maus
Betreuer:	Dr. Heiko Maus

29.09.2014

Kaiserslautern University of Technology
Department of Computer Science
Knowledge-based Systems Group
Prof. Dr. Prof. h.c. Andreas Dengel

PIMO Diary: Diary-Generation from Personal Information Models

Diploma Thesis

in Cooperation with the
German Research Center for
Artificial Intelligence (DFKI) GmbH

Summer Term 2014

Christian Jilek

Course of Studies:	Business Management and Engineering Subject Area Computer Science
First Examiner:	Prof. Dr. Prof. h.c. Andreas Dengel
Second Examiner:	Dr. Heiko Maus
Advisor:	Dr. Heiko Maus

09/29/2014

Eidesstattliche Erklärung / Declaration of Originality

Ich versichere an Eides Statt, dass ich die vorliegende Arbeit selbstständig, ohne fremde Hilfe und ohne Benutzung anderer als die angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen (einschließlich elektronischer Quellen) direkt oder indirekt übernommenen Gedanken sind ausnahmslos als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form oder auszugsweise im Rahmen einer anderen Prüfung noch nicht vorgelegt worden.

I hereby declare that this thesis represents my original work and that I have used no other sources except as noted by citations. All data, tables, figures and text citations which have been reproduced from any other source, including the internet, have been explicitly acknowledged as such. I also declare that I have not submitted this thesis either in terms of this current version or another version to any other faculty.

Kaiserslautern, September 2014

(Christian Jilek)

Contents

0. Introduction, Outline and Summary in German

Dieses Kapitel enhält eine deutschsprachige Einleitung, Gliederung und Zusammenfassung der vorliegenden Arbeit[1].

0.1. Einleitung

Motivation In einer Welt, die sich sprichwörtlich immer schneller und schneller dreht (in Anspielung auf die sich ständig und schnell ändernden Rahmenbedingungen und Anforderungen, mit denen sich Menschen heute konfrontiert sehen) können auch zunehmend Gegenmaßnahmen beobachtet werden: verbesserte Work-Life-Balance, Wellness-Einrichtungen oder Abenteuerurlaube sind nur ein paar Schlagworte, die davon einen Eindruck vermitteln sollen. Insbesondere wenn das Leben sehr hektisch ist, könnten Menschen von Zeit zu Zeit das Verlangen haben, einen Moment inne zu halten um in Erinnerungen zu schwelgen, gedanklich einen glücklichen Moment wiederaufleben zu lassen oder über Herausforderungen, welche sie angenommen und ggf. gerade bewältigt haben, nachzudenken. Da der Mensch ein geselliges Wesen ist, liegt es in seiner Natur solche Erinnerungen mit anderen zu teilen. Wer kennt nicht die Situationen, in denen man unerwartet Freunde oder Kollegen in der Stadt trifft, die man lange nicht mehr gesehen hat? Es wird nach gemeinsamen Freunden, deren Gesundheit, beruflicher Situation oder Kindern gefragt. Vielleicht spricht man über einen gemeinsam verbrachten Urlaub oder zeigt schnell ein paar Fotos auf dem Smartphone oder Tablet. Wieder zu Hause angekommen fragt man sich eventuell: „mein Freund sprach vom Frühjahr vor drei Jahren, was habe ich eigentlich in dieser Zeit gemacht?“

In einer vom Autor durchgeführten Befragung, welche von Maria Wolters (Universität Edinburgh), einer Forscherin im ForgetIT-Projekt (siehe Kapitel 3.1.4), unterstützt wurde, zeigte sich, dass Menschen zum einen ein Interesse daran haben, ihre Gedanken und Erlebnisse (auch physisch) festzuhalten, um sich später daran zu erinnern oder sie mit anderen zu teilen. Auf der anderen Seite sollte dieser Aufbewahrungsprozess nicht allzu zeitaufwändig sein. Außerdem gibt es Bedenken hinsichtlich Sicherheit und Privatsphäre, falls digitale Medien involviert sind (was zunehmend der Fall ist). Im Einzelnen wurden 21 Teilnehmer zum Thema persönliche Reminiszenz und Nutzung sozialer Medien befragt. Die vier bemerkenswertesten Einsichten sind folgende (Details befinden sich in Appendix A):

1. Abgesehen von ihrem Gedächtnis nutzen die Teilnehmer primär **Fotos oder Bilder, Videos, Notizen, (Arbeits-)Zeugnisse und ihren Kalender, um auf ihre Vergangenheit zurückzublicken**.

2. Teilnehmer, die angaben, *selten* oder *nie* **Einträge** in sozialen Netzwerken, Blogs oder Tagebüchern zu **verfassen**, begründeten dies primär mit mangelndem Interesse (61%),

[1] gemäß §21 Abs. 7 Satz 2 der Diplomprüfungsordnung Wirtschaftsingenieurwesen in aktuell gültiger Fassung

aber auch aufgrund von Bedenken hinsichtlich Privatsphäre oder Sicherheit (19%) oder weil sie es als **zu zeitaufwändig** ansehen (11%).

3. 81% der Teilnehmer gab an, für einen vorgegebenen, beliebig gewählten Zeitraum ihres Lebens bei kurzer Bedenkzeit **keine fünf Dinge nennen zu können, welche sie in diesem Zeitraum am meisten beschäftigt haben**.

4. 77% der Teilnehmer *sind* oder *sind möglicherweise* **interessiert an einer Anwendung, mit der sich leicht zurückblicken ließe**.

Das Entwickeln einer solchen Anwendung erfordert die Bewältigung diverser Herausforderungen, welche im nächsten Abschnitt vorgestellt werden.

Ziele Um eine sinnvolle und aussagekräftige Rückschau auf das persönliche und/oder berufliche Leben einer Person (oder Teile davon) zu ermöglichen, werden große Datenmengen benötigt. Zwei sehr wichtige Fragen in dieser Hinsicht sind, **wie diese Daten erlangt werden können** und **wie sie** auf angenehme Weise **dem Benutzer präsentiert werden können**, ohne ihn mit ihrer Fülle zu überwältigen.

Hinsichtlich der zweiten Frage, haben wir uns entschlossen das Konzept eines **Tagebuchs** zu nutzen, insbesondere weil wir die Aspekte einer *Zeitleiste* und eine Art der *redaktionellen Aufbereitung von Text* einbeziehen möchten. Obwohl dies Aspekte sind, welche man auch in Blogs findet, bevorzugen wir dennoch das Tagebuch, weil wir es als das generellere Medium erachten. Der Begriff Blog, der eine Kurzform für *Web*-Log ist, impliziert eine (teil-)öffentliche Verfügbarkeit, so dass Anwendungsfälle mit sehr privaten und sensiblen Daten ungewöhnlich erschienen. Wir werden in Abschnitt 2.1 sehen, dass es neben dem „Teenager-Tagebuch“, in dem Heranwachsende über ihr Leben und ihre Gefühle schreiben (welche auch bereits nicht unbedingt für die Öffentlichkeit bestimmt sind), sehr viele weitere Tagebuchformen gibt, wie etwa das Wissenschafts- oder Pflege-Tagebuch. Insbesondere letzteres würde aus den zuvor genannten Gründen wahrscheinlich nicht mit Blogs in Verbindung gebracht. Da die ersten Tagebücher vor Hunderten von Jahren geschrieben wurden (oder Tausenden – abhängig von der Enge der begrifflichen Definition), ist es für die meisten Menschen ein wohlbekanntes Konzept. Zudem ist es auch kein „zu technisches“ Umfeld für unser Konzept, was wiederum das Ansprechen größerer Zielgruppen erleichtert.

Ein weiter Aspekt unseres Projekts ist, dass unser Tagebuch mehr als eine große sequentielle Ansammlung von Material sein soll. Wir wollen darüber hinaus dem Nutzer ermöglichen, einen tatsächlichen Überblick über seine Vergangenheit zu erlangen – auch für große Zeiträume. Um dies zu erreichen ist es notwendig, Beziehungen zwischen möglicherweise Tausenden von einzelnen Informationselementen (Einträgen) zu identifizieren und angemessene Abstraktionen dafür zu bilden. Wenn ein Nutzer beispielsweise auf das letzte Jahrzehnt zurückblickt, sollte er nicht mit einer Flut von Einzelereignissen überwältigt werden. Stattdessen sollten ihm

kurze und prägnante Angaben wie Projektnamen, Lebensabschnitte oder -situationen, usw. angezeigt werden. Beispiele hierfür sind Begriffe wie Schulzeit, Studium, Hochzeit oder der Name eines Ortes, an dem ein Urlaub oder längerer Auslandsaufenthalt stattfand. Der Nutzer *zoomt* buchstäblich aus einer überwältigenden Masse von Details *heraus*. Falls gewünscht können diese Abstraktionen leicht wieder aufgelöst werden, indem ein bestimmter Zeitabschnitt, zum Beispiel fünf Jahre einer Dekade, zur Konkretisierung ausgewählt wird (*hinein zoomen*). Konkretisierungen (Jahre, Monate, Wochen, Tage) können solange vorgenommen werden, bis der Benutzer beim eigentlichen (nicht weiter aufbereiteten) Grundmaterial, also den konkreten Informationselementen wie Notizen, Fotos, Dokumenten, etc., angelangt ist.

Im Hinblick auf die erste der zuvor gestellten Fragen (jene nach der Erlangung der Daten), wollen wir mit unserer Anwendung auf das Konzept des **Semantic Desktop** aufbauen. Vor ungefähr zehn Jahren führten Wissenschaftler diesen Begriff ein, welcher ein gemeinsames Verständnis für verschiedene, ähnliche Ideen lieferte (Sauermann et al., 2005, S. 3), deren Kern es war, *Semantic Web*-Technologien auf die (Desktop-)Rechner der Nutzer zu bringen. Da Semantic Desktop-Standards basierend auf Ontologien es erlauben, Daten über Anwendungsgrenzen hinaus zu repräsentieren und zu organisieren (Schwarz et al., 2012, S. 331), war es fortan möglich, (große Teile) des persönlichen mentalen Modells eines Nutzers explizit abzubilden und es in all seinen Anwendungen zu nutzen (oder wenigstens in jenen, die sich in das persönliche Semantic Web des Nutzers integrieren). Im Einzelnen bedeutet dies, dass es Anwendungen unter Verwendung des persönlichen Informationsmodells (PIMO) eines Nutzers, welches als Basis für die Wissensrepräsentation auf dem Semantic Desktop dient (Schwarz et al., 2012, S. 319), beispielsweise möglich ist, festzustellen, dass eine bestimmte Entität, die in einer E-Mail erwähnt wurde, tatsächlich eine Person ist. Insbesondere wird auch erkannt, dass es sich um dieselbe Person handelt, welche bereits in einem vom Anwender verfassten Brief erwähnt wurde und welche er zu einem für nächste Woche angesetzten Meeting eingeladen hat. Durch Ausnutzung solchen Wissens kann die tägliche Arbeit der Anwender an ihrem Computer besser durch das System unterstützt werden: die Organisation, insbesondere die Vernetzung, ebenso wie das Teilen von Informationen wird erheblich vereinfacht. Eine detailliertere Einführung dieser Konzepte befindet sich in Kapitel 2.

Während der letzten Dekade wurden mehr und mehr Semantic Desktop-Anwendungen erstellt, welche entweder „nativ“ sind oder Plug-Ins für „traditionelle“ Anwendungen darstellen. Insbesondere das *Deutsche Forschungszentrum für Künstliche Intelligenz (DFKI)* und seine Partner haben Plug-Ins für mehrere Office-Anwendungen wie E-Mail-Clients, Web-Browser und sogar den Windows Datei-Explorer entwickelt (Maus et al., 2013b, S. 2). Da das PIMO eines Nutzers dadurch auf verschiedenste Arten (mit semantisch annotierten Daten) gespeist werden kann, hat sich der Semantic Desktop-Prototyp des DFKIs – hiernach lediglich *der* Semantic Desktop genannt – bereits zu einem relativ ausgereiften System mit reichen persönlichen Informationsmodellen entwickelt (eine entsprechende Nutzung des Systems vor-

ausgesetzt).

Vision Zusammenfassend lässt sich unsere Vision als **Erstellung eines PIMO-basierten Tagebuchs, das sich selbst schreibt**, ausdrücken. Weniger plakativ formuliert möchten wir eine Anwendung entwickeln, welche Tagebücher on-demand aus persönlichen Informationsmodellen generiert, um so den Nutzern ohne großen Aufwand eine Rückschau auf ihr bisheriges (privates und/oder berufliches) Leben zu ermöglichen. Dadurch greifen wir auch die Ergebnisse unserer zuvor angesprochenen Befragung auf. Der Semantic Desktop kann mit allen in Punkt 1 genannten Medien bereits gut umgehen. Da die Tagebücher vom System (automatisch) generiert werden, müssen die Nutzer nicht viel Zeit investieren (Punkt 2). Die Punkte 3 und 4 sind bereits durch das Erstellen der Anwendung erfüllt.

Ein solches Tagebuch erfordert verschiedenste KI[2]-basierte Verfahren, deren Konzeption (oder Adaption) und Implementierung die Kernherausforderungen dieser Diplomarbeit darstellen.

Eine Gliederung der Arbeit folgt im nächsten Abschnitt.

0.2. Gliederung

Kapitel 1 entspricht dem deutschsprachigen Kapitel 0.1 und enthält eine kurze Einführung in das Thema.

Der konzeptionelle und technische Hintergrund wird in **Kapitel 2** vorgestellt.

In **Kapitel 3** werden ähnliche bzw. zugehörige Anwendungen und Arbeiten im wissenschaftlichen bzw. industriellen Bereich evaluiert.

Kapitel 4 enthält die Konzeption bzw. das grundlegende Design der Anwendung. Insbesondere sind dies eine Liste von Anforderungen, Anwendungsszenarien, Use Cases und die grundlegende Struktur der Benutzeroberfläche.

Der generelle Entwurf von Systemarchitektur und Komponenten wird in **Kapitel 5** thematisiert.

Auf Basis dieser Entwürfe wurde eine Proof-of-Concept-Implementierung erstellt, welche zusammen mit Beschreibungen, wie einzelne Teilprobleme gelöst wurden, in **Kapitel 6** zu

[2] KI: Künstliche Intelligenz

finden ist.

Kapitel 7 enthält Ergebnisse einer Evaluation, in der eine Reihe von Nutzern das Programm testen konnten und ihre Einschätzungen dazu abgaben.

Zuletzt folgt in **Kapitel 8** eine kurze Zusammenfassung, sowie ein Ausblick auf etwaige weiterführende Arbeiten zu diesem Thema. Eine Zusammenfassung ist zudem im deutschsprachigen Kapitel 0.3 zu finden.

Folgende Tabelle bietet nochmals eine Übersicht der einzelnen Kapitel:

Kapitel	**Inhalt**
0	Deutschsprachige Einleitung, Gliederung und Zusammenfassung
1	Einleitung
2	Konzeptioneller und technischer Hintergrund
3	Ähnliche bzw. zugehörige Arbeiten aus Wissenschaft und Industrie
4	Konzeption bzw. grundlegendes Design der Anwendung
5	Genereller Entwurf von Systemarchitektur und Komponenten
6	Proof-of-Concept-Implementierung und Beschreibung von Detaillösungen
7	Evaluation des Systems durch Testnutzer
8	Zusammenfassung, Schlussfolgerungen und Ausblick
Anhang	Literatur-, Abbildungs-, Tabellen- und Abkürzungsverzeichnis
A	Umfrage zur Nutzung sozialer Medien und persönlicher Reminiszenz
B	Genauere Evaluationsergebnisse der Testnutzer
C	Mock-ups der Benutzeroberfläche
D	Task and Object-oriented Requirements Engineering (TORE) Framework
E	Digitale Dokumente

Tabelle 0.1: Gliederung / Outline in German

Zum Abschluss des deutschsprachigen Kapitels folgt im nächsten Abschnitt noch eine Zusammenfassung der vorliegenden Arbeit.

0.3. Zusammenfassung

Eine vom Autor zu Beginn dieses Projekts durchgeführte Umfrage zeigte, dass es ein generelles Interesse an einer Anwendung gäbe, mit der sich leicht auf das eigene Leben zurückblicken ließe. Zudem sollte diese Anwendung mit verschiedensten Medien umgehen können und Nutzern auf möglichst zeitsparende Art ermöglichen, ihre Erinnerungen und Erfahrungen festzuhalten (Kapitel 1). Diese Ergebnisse aufgreifend entwickelten wir die Idee, eine Anwendung zu schaffen, welche on-demand Tagebücher aus persönlichen Informationsmodellen von Nutzern generiert. Nach Vorstellung des konzeptionellen und technischen Hintergrunds (Kapitel 2), wurden verschiedene verwandte oder ähnliche Arbeiten und Anwendungen aus Wissenschaft und Industrie evaluiert (Kapitel 3). Nach unseren Recherchen gab es in der Vergangenheit nur wenige Ansätze zur automatischen Generierung von Tagebüchern aus den Datenspuren von Nutzern. Alle von uns präsentierten Tagebuch-Projekte basierten auf dem Auslesen von bspw. Sensordaten oder Activitäts-Logs von Mobilgeräten (primär Smartphones). Mit Mitteln der künstlichen Intelligenz mussten diese Anwendungen die Daten zunächst mit einer Semantik versehen. Da wir den Semantic Desktop nutzen, haben wir den Vorteil, bereits direkt mit semantisch annotierten Daten versorgt zu werden. Es lag daher an uns, diesen Vorteil optimal zu nutzen, um so eine App zu entwickeln, welche einige Defizite früherer Anwendungen überwindet.

Eines der Hauptprobleme dieser Anwendungen war es, Nutzern einen tatsächlichen Überblick über deren Vergangenheit zu bieten. In den meisten Fällen sind Nutzer einer überwältigenden Masse von einzelnen Informationselementen wie Dokumenten, Notizen, Fotos, Kalendereinträgen usw., ausgesetzt, wenn sie auf ausgewählte Zeiträume ihres Lebens zurückblicken. Demzufolge ist es ihnen nicht ohne Weiteres möglich, nachzuvollziehen was tatsächlich in einem Zeitraum passiert ist. Obwohl in verschiedenen, von uns vorgestellten Timeline-Projekten versucht wurde, das Problem mangelnder Übersichtlichkeit anzugehen, wurden – unserer Meinung nach – bisher keine so prägnanten und aussagekräftigen Verdichtungen oder Abstraktionen präsentiert, die eine zufriedenstellende Rückschau möglich machen. Wir versuchten dieses Problem mittels eines Features zum *Hinein- und Herauszoomen in bzw. aus Zeiträumen* zu lösen, welche Konkretisierungs- bzw. Verdichtungsprozesse in Gang setzen (Kapitel 4). Blickt der Nutzer beispielsweise auf ein Jahr zurück, so werden ihm von unserer App, anstelle von Hunderten oder Tausenden einzelner Informationselemente, Abstraktionen wie Projektnamen, Lebenssituationen, Ereignisse, etc. bereitgestellt. Diese Daten werden in Form eines modernen Blogs präsentiert. Um eine adäquate Anzahl von Tagebucheinträgen zu erhalten, ist die Verschmelzung (*Clusterbildung*) und Filterung (*Wichtigkeitsevaluation*) von Informationselementen nötig. Insbesondere ersteres sorgt zudem für eine *hohe Diversität innerhalb des Tagebuchs* und macht es so interessant, betrachtet bzw. gelesen zu werden. Des Weiteren enthalten Tagebucheinträge neben einer textuellen Zusammenfassung aller Informationselemente, aus denen sie bestehen, Icons, welche annotierte Konzepte repräsentieren, und Fotos,

die mit dem Eintrag verknüpft sind. Zusätzlich haben wir eine weiteres Feature namens *Konzeptkontext* entwickelt, welches einen Überblick über die wichtigsten Dinge eines ausgewählten Zeitraums bietet. Durch Betrachten dieses Kontexts können sich Nutzer schnell einen Eindruck davon verschaffen, welche Dinge sie (augenscheinlich) in einer bestimmten Zeitspanne am meisten beschäftigt haben.

Wir haben unsere App als eine verteilte Client/Server-Anwendung konzipiert (Kapitel 5) und eine Proof-of-Concept-Implementierung entwickelt, deren Clientkomponente eine in den sog. *PIMO5-Client* des DFKIs integrierte *HTML5-App* ist. Die Serverkomponente ist ein *JAVA Servlet* (Kapitel 6). Insgesamt umfasst die in diesem Projekt entwickelte Software 7200 Codezeilen (inkl. ungefähr 1500 Zeilen experimentellen Codes, der nur temporär verwendet wurde).

In einer Evaluation durch eine vierköpfige Gruppe DFKI-externer Tester erzielte unsere Anwendung sehr gute Ergebnisse (Kapitel 7). Die Tester fanden die App sehr innovativ und es bereitete Ihnen Spaß, sie zu nutzen. Sie waren zudem von der leichten Bedienbarkeit und der hohen Qualität der Ergebnisse überrascht. Insgesamt gesehen bestätigten sie, dass unsere Anwendung einen leichten und zufriedenstellenden Rückblick auf das Leben einer Person ermöglicht und zudem einen guten Überblick über die Dinge bietet, mit der sich eine Person in einem bestimmten Zeitraum am meisten beschäftigt hat. Es wurden zudem verschiedene Verbesserungsvorschläge gegeben, welche wir (neben weiteren) detailliert in Kapitel 8.2 vorstellen. Primär sind dies Verbesserungen hinsichtlich der Antwortzeit des Systems und der Benutzeroberfläche, sowie die Einbindung sozialer Netzwerke und allgemeine Optimierung bzw. Feintuning verwendeter Algorithmen.

1. Introduction

1.1. Motivation

In a world that proverbially turns faster and faster (referring to fast and constantly changing environments and demands people are facing today), an increasing amount of opposing movements and counter or compensation measures can be found: improved work-life balance, wellness facilities or adventure holidays are only a few buzz words that should give you an impression. Especially if life is very frantic, people might from time to time have the desire to pause for a little while and reminisce about past experiences, relive a happy moment or think about challenges they accepted and maybe just got over. Being a social animal it is part of human nature to also share these memories with others. Who does not know the situations, in which you are in town and unexpectedly meet friends or colleagues you have not seen for quite some time? Questions about common friends and their health, jobs or children are asked, maybe a former holiday spent together is recalled and broached or some photos are quickly presented on a smart phone or tablet. Staying in this example, one might later also ask himself: "my friends talked about spring three years ago, what have I been doing during that time?".

A survey carried out by the author and supported by Maria Wolters (University of Edinburgh), a researcher in the ForgetIT project (see Section 3.1.4), showed that people are on the one hand generally interested in preserving their memories (physically) for later reminiscence or sharing with others. But on the other hand, this preservation process must not be too time-consuming and there are privacy or security concerns if digital media are involved (which is increasingly the case). In particular, 21 participants were asked about their social media usage and personal reminiscence. The four most remarkable insights are as follows (for details please see Appendix A):

1. Besides their memory, the participants primarily use **photos or images, videos, notes, certificates (of employment) and their calendar to retrospect on their past**.

2. Participants who do *not* or *not often* **write entries** in social networks, diaries or web logs (or *blogs* for short) primarily justified this with their lack of interest (61%), but also with privacy or security concerns (19%), or since they consider it to be **too time-consuming** (11%).

3. 81% of the participants said that they are **not able to name five things they were concerned with the most** for a given, arbitrarily chosen period of their lives and a short thinking time.

4. 77% of the participants stated that they *are* or *possibly are* **interested in an application that would ease retrospection**.

In order to develop such an application, several challenges need to be tackled, which we address in the next section.

1.2. Goals

To enable a meaningful retrospection on (parts of) one's private and/or professional life, possibly very large amounts of data are needed. Two very important questions to be answered are **how this data can be acquired** and **how it can be presented to the user** in a comfortable, not overwhelming way.

Addressing the second question, we would like to use the concept of a **diary**, especially since we wanted to take the aspects of a *timeline* and some kind of *editorial preparation of text* into account. These aspects can also be found in a web log. Nevertheless, we still prefer a diary, since we consider it to be the more general medium. The term *web* log implies a certain (semi-)public availability, thus use cases having very private or confidential data would seem unusual. We will see in Section 2.1 that besides a *"teenager diary"*, in which adolescents write about their lives and especially their feelings (which may already not be intended for publishing), there are several other kinds of diaries like scientific or care diaries. Especially the latter would probably not be associated with a web log for the reasons mentioned before. Since the first diaries were written hundreds of years ago (or even thousands, depending on the narrowness of the terminological definition), it is a well-known concept to most people and thus a not "too technical" setting for our concept. This also eases addressing larger target groups.

Another aspect of this project is that our diary should be more than a large, sequential collection of material. Additionally, we would like to enable the user to easily get an actual overview of his past, even for large periods of time. To achieve this, it is necessary to identify relationships among possibly several thousands of individual information items (entries) and create suitable abstractions from them. If a user, for example, looks back on the last decade, he should not be overwhelmed with a view showing plenty of individual events, but compact statements like project names, stages of life, life situations, etc. Examples for those are terms like school years, studies, marriage or the name of a place where a vacation or longer stay abroad has been spent. The user literally *zooms out* of the overwhelming mass of details. If desired, these abstractions can be easily be resolved by selecting a (sub-)period of time for concretization (*zooming in*), e.g. five years of a decade. Concretizations (years, months, weeks, days) can be performed until the user reaches the actual (non-rehashed) basic material, which are concrete information items like notes, photos, documents, etc.

Coming back to the first question of this section's beginning, which was how to obtain the data necessary for diary creation, we would like to base our application on the concept of the

Semantic Desktop. About ten years ago, researchers established this term, which provided a mutual understanding for several similar ideas (Sauermann et al., 2005, p. 3), whose core was bringing *Semantic Web* technologies to the user's desktop. Since Semantic Desktop standards based on ontologies allow representing and organizing data across application borders (Schwarz et al., 2012, p. 331), it was henceforth possible to explicitly express (major parts of) a user's personal mental model and make use of it in all his applications (or at least in those that integrate into his personal semantic web). In particular this means that by applying a user's **personal information model (PIMO)**, which serves as the basis for knowledge representation on the Semantic Desktop (Schwarz et al., 2012, p. 319), applications are, for example, able to detect that a specific entity addressed in an e-mail is actually a person. Furthermore, the system detects that it is the same person mentioned in a letter previously written by the user and also an invitee of a meeting scheduled by the user for next week. By utilizing such knowledge a user's daily work on his computer can be supported much better by the system. The organization, and especially the interconnection, as well as the sharing of information is considerably eased. For a more detailed introduction to these concepts please see Chapter 2.

Over the last decade more and more Semantic Desktop applications have been created, which are either "native" or plug-ins for "traditional" applications. Especially the *German Research Center for Artificial Intelligence* (DFKI) and their partners have developed plug-ins for several office applications such as e-mail clients, web browsers and even the Windows File Explorer (Maus et al., 2013b, p. 2). Since a user's PIMO can thus be fed (with semantically annotated data) in various ways, the DFKI's Semantic Desktop prototype (hereafter only referred to as *the* Semantic Desktop) is already a relatively sophisticated system with rich personal information models (assuming an appropriate usage of the system).

Vision In summary, we formulate our vision of creating a **PIMO-based diary that writes itself**. Expressing this in a less catchy phrase: we would like to develop an application that generates diaries on demand based on personal information models, allowing users to retrospect on their (private and/or professional) lives without much effort. By this we also address the results of our previously mentioned survey. The Semantic Desktop is able to deal with all media mentioned in item 1 and since the diaries are generated by the system, the users do not need to invest much time (item 2). Items 3 and 4 are covered just by providing the application.

Such a diary requires several AI[3]-based methods, whose conception (or adaption) and implementation are the core challenges of this thesis.

An outline is given in the next section.

[3] AI: artificial intelligence

1.3. Outline

In **Chapter 2** we substantiate some of the already mentioned concepts and also introduce additional ones.

Next, we evaluate related works and applications in research and industry (**Chapter 3**).

The concept of our application is given in **Chapter 4**. This comprises a more detailed problem description as well as requirements, especially usage scenarios and use cases.

In **Chapter 5** we present our system's architecture and component design.

Chapter 6 is about the implementation and how several sub-problems are solved. It also contains an example, in which we present the author's own diary generated for the time of this thesis.

A user experience evaluation of our app[4] can be found in **Chapter 7**.

Chapter 8 concludes this thesis by giving a short summary and an outlook on possible future work.

[4] app: short for application

2. Background

In this chapter we introduce the conceptual and technical background of this thesis by explaining some previously mentioned concepts in more detail and introducing additional ones.

2.1. Diaries

A diary is "a record of events, transactions, or observations kept daily or at frequent intervals". There is a special form called a *journal*, which is "a daily record of personal activities, reflections or feelings" (Merriam-Webster Dictionary).

According to Mohrmann et al. (2005) the form of a diary as we often understand it today – texts in which people are concerned with their experiences, thoughts and feelings – first emerged in the 19th century. Before, there were several antecedents like *chronicles* in the Middle Ages. In these chronicles, events concerning monasteries, cities or families were recorded in irregular time intervals. Information about important events like births, deaths, natural disasters, fires or wars were written down in order to be preserved for later generations.

The amount of personal information in diaries – in contrast to more group-related information earlier – began to increase since the end of the 15th century. Two of the most famous historic diaries today are the ones by Anne Frank, a Jewish girl who kept a diary during the time of Nazi Germany, and the one by Samuel Pepys, a naval administrator and member of the British parliament that recorded his daily life for almost ten years in the 17th century (Wikipedia Encyclopedia). In 1771 Johan Caspar Lavater introduced the diary as a literary category. For a more detailed overview of diaries and their history please see (Mohrmann et al., 2005).

We already mentioned in Chapter 1 that there are more kinds of diaries than the classical *"teenager diary"*, for example: *baby-*, *family-*, *reading-*, *project-*, *partnership-*, *pregnancy-*, *care-*, *travel-*, or *scientific diary*. More examples can be found on Bücher-Wiki. We also gave reasons why we chose the form of a diary, one of them was that it is a well-known concept to most people. This assertion is supported by several studies from 1925 to 1985 mentioned in (Seiffge-Krenke, 2001, p. 3). They revealed that throughout these decades there was a relatively constant rate of 30% to 60% adolescents writing diaries. In another sample of 1987 this rate was 40% (Seiffge-Krenke, 2001, p. 4). These are only numbers concerning adolescent diary *writers*. Since they cover several decades and the set of people *writing* diaries is a subset of the people *knowing* or *reading* diaries, it is fairly safe to assume that the number of people (all ages) knowing this concept is much higher.

Closely related to diaries is the visualization concept of a timeline, which is introduced more thoroughly in the following.

2.2. Timelines

Loosely based on Sauer (2005, p. 197), a timeline is an spatial illustration of the abstract concept of time: a sequence of dates belonging to events or periods are plotted on a (usually) horizontal line.[5] Kullberg et al. (1995, p. 7) define a timeline as "an atlas of history, a map of events in time". According to them, *"we use timelines for some of the same reasons we use geographical maps:*

- *to locate an event in time, as we would locate a city on a geographical map;*
- *to see the time elapsed between events, as we would see the distance between two cities;*
- *to get an overview while being able to focus on detail in its correct context, as we would view a city in the larger context of its state while being able to discern information particular to the city.*

When examining events in time, we are not only concerned with finding the what, when, where. We also look for causal relationships. We look at other events and the historical context, and try to understand why and how."

Timelines can help us in providing an appropriate visual overview, since diary entries can be represented as single events or periods plotted on them.

An example of a timeline showing some events of November 22nd, 1963 – the day John F. Kennedy was shot – is shown in Figure 3.14 (Chapter 3.3.3).

Also closely related to diaries are blogs, which are the topic of the next section.

2.3. Blogs

Blogs (or *web logs*) "are frequently updated webpages with a series of archived posts, typically in reverse-chronological order. Blog posts are primarily textual, but they may contain photos or other multimedia content" (Nardi et al., 2004, p. 222). Today, many tools are available which support less technically experienced users in creating their own blogs, e.g. *WordPress* or *Tumblr*.

Blogs vary widely in nature and content and – while growing in popularity – increasingly became online diaries or personal journals (Nardi et al., 2004, p. 222). According to Nardi et al., Herring et al. (2004) found three primary types of blogs: individually authored *personal journals*, *"filters"* (because they select and provide commentary on information from other websites) and *"knowledge logs"*. The majority in their sample were of the first type (70%).

We will see in Chapter 4 that one of our main requirements is that the created application should have the look and feel of blogs typically created with *WordPress* or *Tumblr*, as shown in Figure 4.1.

[5] Original quotation: „Die Zeitleiste ist eine räumlich-anschauliche Umsetzung des abstrakten historischen Zeitverlaufs. Auf einer (in der Regel) waagerechten Geraden werden Jahreszahlen abgetragen." (Sauer, 2005, S. 197)

A frequently used term in the context of diaries, timelines or blogs is that of a *memory landmark*, which is defined in the next section.

2.4. Memory Landmarks

Horvitz et al. (2004, p. 1) explained the term of a *memory landmark* (referring to several studies of the human memory) as follows:

> *"Studies of memory support the assertion that people make use of special landmarks or anchor events for guiding recall and for remembering relationships among events. Such landmarks include both public and autobiographical events. More generally, there has been significant study and modeling of episodic memory, where memories are considered to be organized by episodes of significant events, including such information as the location of an event, attendees, and information about events that occurred before, during, and after each memorable event. Memory has been shown to also depend on the reinstatement of not only item-specific contexts, but also on more general context capturing the situation surrounding events."*

In our diary app we can utilize memory landmarks to build abstractions. As an example, let us consider a person's wedding. All information items associated with this event, e.g. photos, the menu, bills for the restaurant or the wedding trip, etc. can be summarized by a landmark called "wedding".

Another example could be a project manager, who – like many other knowledge workers – might have an important meeting as a memory landmark. Thus, all information items like text documents, emails, or a presentation can be summarized under this landmark.

Since we already mentioned the term of a *knowledge worker*, we shall catch up on providing its definition in the following.

2.5. Knowledge Workers

Our goal is to create an application that can be used to reminisce about private as well as professional life. Considering the latter, our app can probably be most effectively used in the jobs of *knowledge workers*, which, in short, are persons that *think for a living* (Davenport, 2005, p. 3). Their jobs typically include activities like *processing information, communicating, making decisions, creating documents*, etc. All these activities are *usually associated with office work, innovation, leadership roles and are commonly known as "knowledge work"* (Dengel and Bernadi, 2012, p. 5). Examples for knowledge workers are engineers, scientists, doctors, lawyers or managers.

Thus, a knowledge worker's PIMO will be populated with information very fast and thoroughly by simply using his computer "the usual way" – besides adding some semantic annotations. Maus et al. (2013b, p. 1) describe today's work life of knowledge workers as follows:

> *"The modern working environment places high requirements on knowledge workers: they are confronted with various applications, are involved in several projects and processes, work in changing teams, are on the road with a mobile office, and finally, face an ever increasing flow of information. The resulting knowledge spaces are complex, dynamic, distributed over several applications, and use different vocabulary."*

The second last aspect is also known as the *(project) fragmentation problem in personal information management (PIM)*. Figure 2.1 shows a typical scenario in which semantically related data is distributed over several applications, in this case the file system, e-mail folders and bookmarks in a web browser. For details please see Dengel (2007) or Bergman et al. (2006).

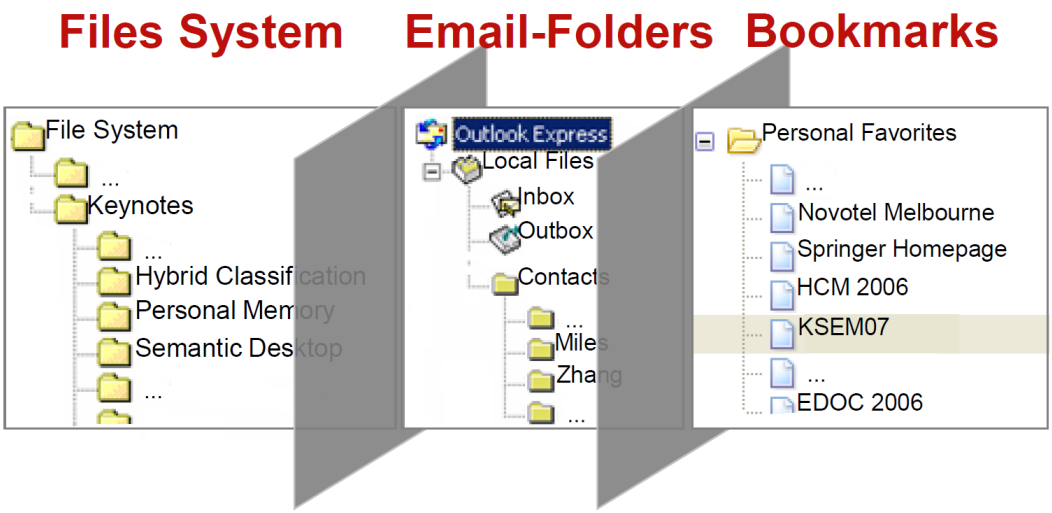

Figure 2.1: Fragmentation problem in PIM (Dengel, 2007, p. 4)

Thus, "it is hard to keep the overview in the resulting personal knowledge space. This challenge is addressed with the concept of the Semantic Desktop" (Maus et al., 2013b, p. 1).

2.6. Semantic Desktop

In Chapter 1 we already gave a short description of what the Semantic Desktop is. A more detailed definition is as follows:

> *"A Semantic Desktop is a device in which an individual stores all her digital information such as documents, multimedia and messages. These are interpreted as Semantic Web*

resources, each is identified by an URI[6] and all data is accessible and queryable as RDF[7] graph. Ontologies allow the user to express personal mental models and form the semantic glue interconnecting information and systems, and Semantic Web protocols are used for inter-application communication. The use of Semantic Web standards allows existing web resources to be incorporated into the personal knowledge space, and does also facilitate the sharing of knowledge with others, for example within a work-group. The Semantic Desktop is an enlarged supplement to the user's memory."

(Schwarz et al. (2012, p. 333) referring to
Sauermann et al. (2005) and Sauermann (2009))

The Semantic Desktop helps in overcoming the problems of parallel organizational structures, fragmentation of interconnections (like mentioned in the last section and depicted in Figure 2.1) and many incompatible APIs[8] (Schwarz et al., 2012, pp. 331).

According to Schwarz et al. (2012, p. 333) the realization of this vision can be summarized in the following steps:

1. Represent all data as RDF.
2. Make all data accessible via RDF.
3. Organize and connect all data in a PIMO.
4. Adopt existing desktop applications to these new possibilities or create new interfaces.

The third step – organizing and connecting data in a PIMO – is what the next section is about.

2.7. Personal Information Model (PIMO)

In order to provide a thorough definition of a *personal information model* (or *PIMO* for short) some auxiliary terms are needed first (Sauermann et al., 2007, p. 2):

- *personal knowledge workspace*: embraces all data needed by an individual to perform knowledge work
- *native resources*: part of the personal knowledge workspace, e.g. personal files of the user, e-mails, and other PIM related resources, such as appointments or contacts

[6] URI: uniform resource identifier – a string of characters used to identify a name of a resource (Wikipedia Encyclopedia)
[7] RDF: resource description framework
[8] API: application programming interface

- *native structures*: categorization schemes for native resources such as file-system folders, bookmark folders, e-mail folders, or tags
- *mental model*: part of the cognitive system of a person; Subjective to a person, it is individual and cannot be externalized thoroughly. The PIMO aims to represent parts of the mental model necessary for knowledge work.

Using these terms a PIMO can be defined as follows:

> *"A PIMO is a Personal Information Model of one person. It is a formal representation of parts of the users Mental Model. Each concept in the Mental Model can be represented using a Thing or a subclass of this class in RDF. Native Resources found in the Personal Knowledge Workspace can be categorized, then they are occurrences of a Thing.*
>
> *The vision is that a Personal Information Model reflects and captures a user's personal knowledge, e.g., about people and their roles, about organizations, processes, things, and so forth, by providing the vocabulary (concepts and their relationships) for required expressing it as well as concrete instances. In other words, the domain of a PIMO is meant to be 'all things and native resources that are in the attention of the user when doing knowledge work'."*
>
> (Sauermann et al., 2007, p. 2)

In the definitions of the Semantic Desktop and the PIMO, representing data in RDF has been mentioned. A resource in RDF is identified with its URI – a typically unique string of characters. Using these URIs statements about resources can be made. An RDF statement consists of three parts: subject, predicate and object, which all are resources themselves, except for the object which may also be a literal (comparable to basic data types in programming languages).

PIMO Example Let us consider a small example: Dr. Heiko Maus, the advisor of this thesis, has given a talk on Cebit 2012. The semantic network (RDF graph) representing this scenario is depicted in Figure 2.2. All entities in this graph are either *classes*, indicated by blue bubbles, or *things* (red bubbles), which are instances of classes. The resource representing Heiko's task of giving a talk at the Cebit 2012 (indicated with the yellow notes sheet) has a URI which reads as *pimo:event:Cebit2012*. This resource *is* of type task and *has the topics* of Semopad and Cebit. These topics are resources themselves: Semopad *is a* project and has a URI called *http://www.dfki.de/semopad*, Cebit *is an* event having the URI of *http://www.cebit.de*. Heiko himself is represented by a resource which *is* of type person and associated with a URI called *mailto:heiko.maus@dfki.de*. We also learn from this graph that Heiko *attends* the Cebit, is

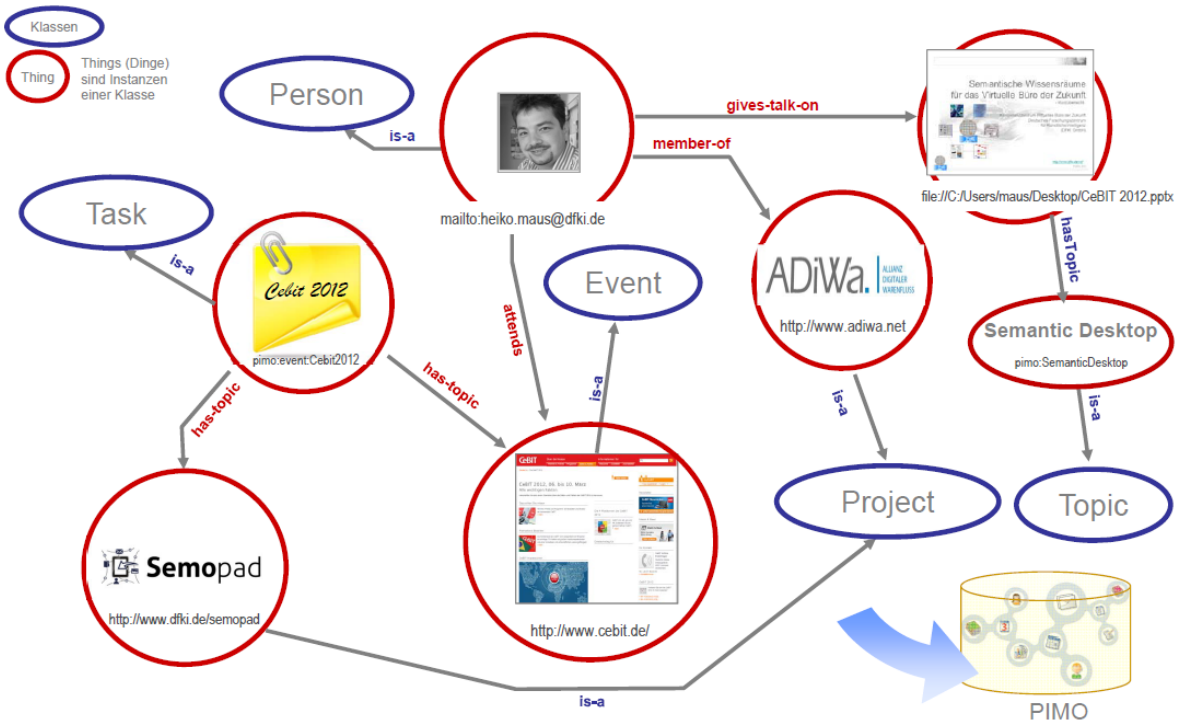

Figure 2.2: A schematic excerpt of a PIMO (Maus et al., 2013b, p. 1)

member of another project called ADiWa and his talk *given* by him *has the topic* of the Semantic Desktop.

GIMO This graph is only an excerpt of Heiko's PIMO. There is also the possibility to share (parts of) his PIMO with others, e.g. his colleagues at DFKI. Since more and more social features were added to the Semantic Desktop during the last years, for example a group calendar or a cloud service that can also be used for sharing files with others (Schettler-Köhler, 2014), one could also speak of a *group information model (GIMO)* instead of a PIMO (Maus et al., 2013b, p. 77).

More information about the Semantic Desktop and PIMO can be found in (Schwarz et al., 2012). They can also be classified as DFKI projects related to our work. We decided to already present them in this chapter due to their fundamental character. Other related projects, works and applications (DFKI and third-party) follow in the next chapter.

We will later have to determine similarities between different information items mainly based on their associated texts and concept annotations. Thus, this chapter's last section is about similarity calculation.

2.8. Similarity Calculation

In our diary app, we will have to identify information items belonging to the same project, life situation or topic, for example. Thus, their labels and text bodies as well as their concept annotations need to be analyzed in order to find similarities.

Text Similarity In order to analyze the similarity of different texts (documents), they are usually first processed by a *text analyzer* which, for example, applies measures of *stemming*[9] and *stop word*[10] *elimination*. We only apply the latter, which sorts out all stop words given by a pre-defined list. From the remaining list of terms a *term vector* is created. The approach of creating term vectors, also called *index vectors*, in order to compare (and retrieve) documents was – to our best knowledge – first proposed by Salton et al. (1975). The basic idea is as follows:

> *"Consider a document space consisting of documents D_i, each identified by one ore more index terms T_j; the terms may be weighted according to their importance, or unweighted with weights restricted to 0 and 1." In a t-dimensional index space each item (document) is identified by up to t distinct terms and thus "each document D_i is represented by a t-dimensional vector $D_i = (d_{i1}, d_{i2}, ..., d_{it})$, d_{ij} representing the weight of the j-th term."*
>
> (Salton et al., 1975, p. 613)

This vector space model is depicted in Figure 2.3.

> *"Given the index vectors of two documents, it is possible to compute a similarity coefficient between them, which reflects the degree of similarity in the corresponding terms and term weights. Such a similarity measure might be the inner product of the two vectors, or alternatively an inverse function of the angle between the corresponding vector pairs; when the term assignment for two vectors is identical, the angle will be zero, producing a maximum similarity measure."*
>
> (Salton et al., 1975, p. 613)

[9] stemming: *the process for reducing inflected (or sometimes derived) words to their stem, base or root form – generally a written word form; e.g. a stemming algorithm reduces the words "fishing", "fished", and "fisher" to the root word "fish"* (Wikipedia Encyclopedia)

[10] stop words: *words which are filtered out before or after processing of natural language data (text); any group of words can be chosen as the stop words for a given purpose; for some search engines, these are some of the most common, short function words, such as "the", "is", "at", "which", and "on"* (Wikipedia Encyclopedia)

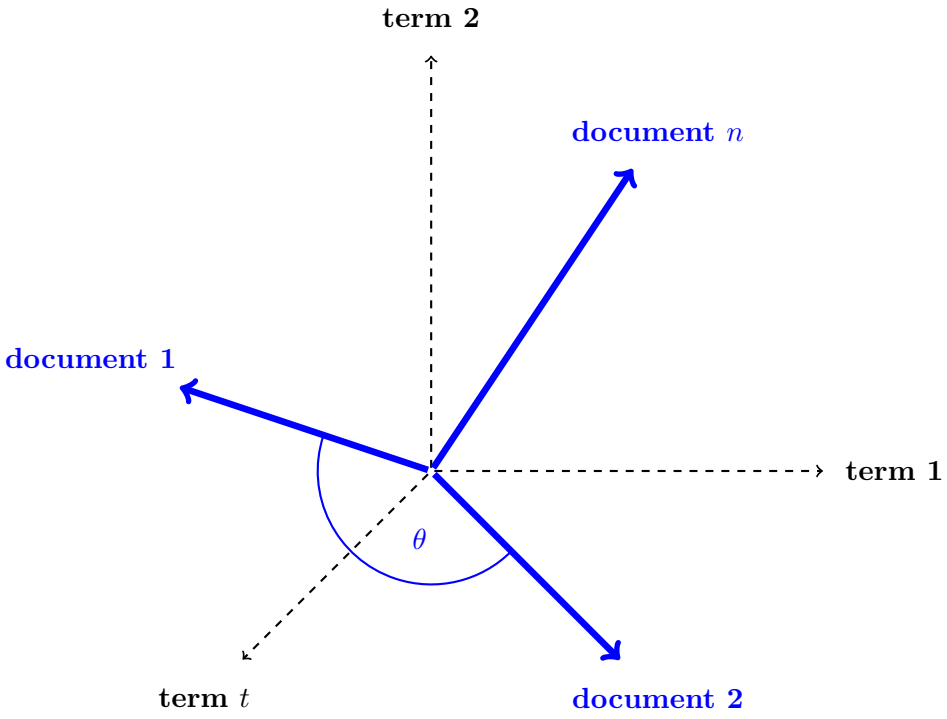

Figure 2.3: Vector space model

In our case, we use the *cosine similarity measure* $cos(\theta)$. Let V and W be two term vectors and $||V||$ the magnitude of vector V, then $cos(\theta)$ is defined as follows:

$$cos(\theta) = \frac{V \cdot W}{||V|| \cdot ||W||} = \frac{\sum_{i=1}^{n} v_i \cdot w_i}{\sqrt{\sum_{i=1}^{n}(v_i)^2} \cdot \sqrt{\sum_{i=1}^{n}(w_i)^2}} \tag{1}$$

Like stated before, the results range from 0 to 1. (The cosine usually ranges from -1 to 1, but since the term vectors do not contain any negative values, all results are positive.) A value of 1 indicates that the angle between both vectors is 0, which means that they are identical and thus have all terms in common in our scenario (in general this indicates that all terms have the same weights in both vectors, see second quotation above). The opposite (no terms in common) is the case if the cosine is 0, which means that both vectors are orthogonal to each other.

Let us consider an example to illustrate the concepts introduced in this section. Suppose the following two sentences (documents) are given:

1. This is a test sentence we would like to process with the analyzer.
2. And this is a second sentence that helps in explaining even more.

Processing them with our text analyzer returns the following term lists:

1. (test, sentence, process, analyzer)
2. (second, sentence, helps, explaining, even)

Next, indices are assigned to the different terms:
1: analyzer, 2: even, 3: explaining, 4: helps, 5: process, 6: second, 7: sentence, 8: test

Thus, both documents can be represented by the following term vectors d_1 and d_2, respecitively (please note that we use the *unweighted version* mentioned in the first quotation above, i.e. a weight of 1 is assigned if the term is present and 0 otherwise):

$$d_1 = (1, 0, 0, 0, 1, 0, 1, 1)$$
$$d_2 = (0, 1, 1, 1, 0, 1, 1, 0)$$

Finally, the cosine similarity measure for both term vectors results in $\frac{1}{2\sqrt{5}} \approx 0.2236$, which indicates are rather low similarity. This is easily comprehensible, since both term lists (or vectors, respectively) only have the term "sentence" in common.

In addition to associated texts we will later also analyze concept annotations which is discussed in the next section.

Concept Similarity We would like use an example in order to introduce our approach of analyzing concept annotations to determine the similarity between different resources. In the PIMO, everything is a resource and every resource can be annotated with "concepts", which is only a synonym, since they are resources themselves. In the following we only use these different terms in order to clearly differentiate between the informations items and their annotations, although their are all resources.

Suppose that there are three resources R_1, R_2 and R_3 which are annotated with different concepts:

- Annotations of R_1: Diary, Heiko
- Annotations of R_2: PIMO
- Annotations of R_3: PIMO, Heiko

First, we index all concepts:
1: diary, 2: Heiko, 3: PIMO

For all resources R_1, R_2 and R_3 we next define *concept vectors* in analogy to the aforementioned term vectors, i.e. we assign a weight of 1 if the resource is annotated with a particular concept and 0 otherwise. Strictly speaking, the three resources R_1 to R_3 would also be related to themselves (each one to itself), thus the term vectors would have three more elements. In our example, we omitted this self-relatedness for the sake of simplicity.

Concept vectors are generally used to determine similarities in knowledge bases. A similar approach can be found in (Liu et al., 2010), although we neglect their aspect of concept hierarchies and instead incorporate a form of spreading activation (as explained later).

The concept vectors in our example are as follows:

$$d_1 = (1, 1, 0)$$
$$d_2 = (0, 0, 1)$$
$$d_3 = (0, 1, 1)$$

Using the cosine similarity measure we could now evaluate the resources' similarities based on their concept annotations. This would result in d_1 and d_3 as well as d_2 and d_3 having a similarity of 0.5. The one of d_1 and d_2 is 0, since they do not have any concepts in common. Nevertheless, we would like to take one more intermediate step. In addition to explicit annotations we would also like to include those concepts that they imply, e.g. since the PIMO is part of the Semantic Desktop, explicitly annotating a resource with it implies additionally annotating the resource with the "Semantic Desktop" to some extend. The difference between explicit and implicit annotations can be reflected by using lower weights for implicit ones, for example. A semantic network showing further relations between the concepts of our example is depicted in Figure 2.4.

Fully drawn arrows represent the explicit annotations. They go from the resource nodes R_1, R_2 and R_3 to the three aforementioned concept nodes (Diary, Heiko and PIMO). The dashed and dotted arrows represent implicit annotations. In order to obtain them, we adapted the approach of *spreading activation* (Crestani, 1997) to the PIMO, e.g. by considering the different types of relations accordingly (see below). First, all explicitly annotated resources are *activated*. In a next step, the activation is *spread* to their neighboring nodes. But with each transition, the activation that is passed on is reduced by a *decay factor*, until it is finally below a certain threshold, the so-called *firing threshold*. If this is the case, the spreading stops at those nodes for which the firing threshold was not reached anymore. Additionally, the spreading is terminated if a node is reached for the second time within the same iteration, which is detected by keeping track of previously used paths. This is often the case since

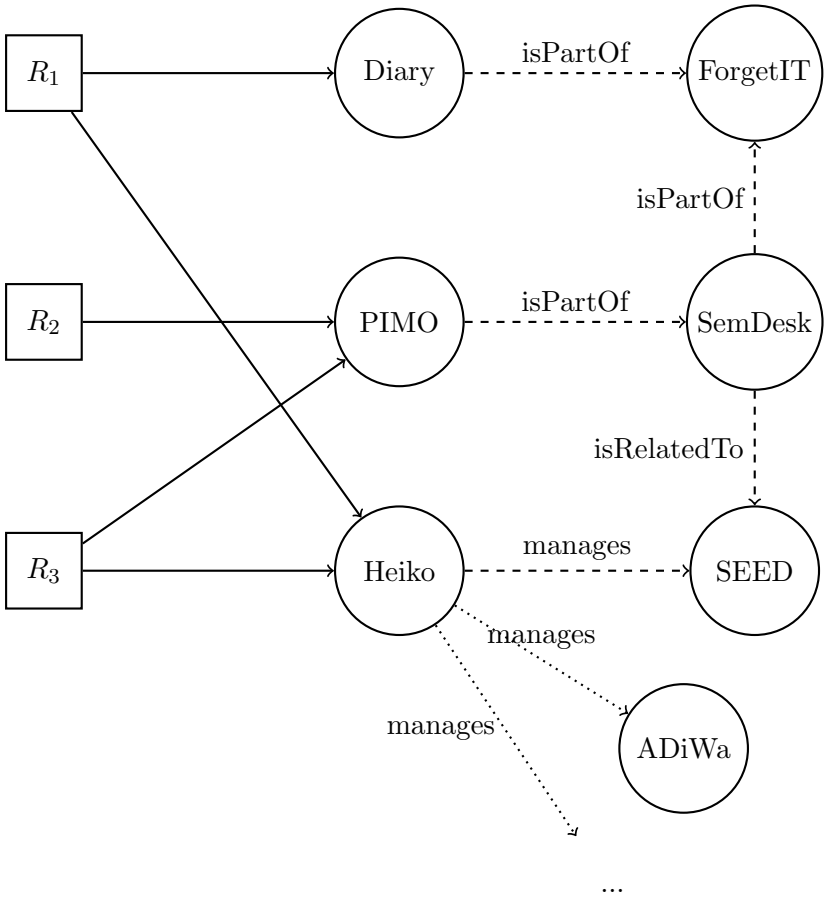

Figure 2.4: Explicit and implicit concept annotations

relations in the PIMO are usually bidirectional and concepts are often highly connected. We omitted this fact in our example for the sake of simplicity.

In addition, different semantic relations can be reflected by assigning corresponding weights. For example, *isPartOf* is a stronger relation than *isRelatedTo*, thus it could be associated with a value of 0.75, whereas the other value is 0.1. Another aspect in this regard is assigning weights according to the number of incoming arcs having the same type. For example, a project has 20 members (relation *isMemberOf*) but is only managed by a single person (relation *manages*). Thus, assigning *manages* with a much higher value than *isMemberOf* will likely lead to better results due to its higher expressive character.

Furthermore, we suggest the exclusion of implicit annotations that only have a single connection to the network which leads over a node representing a person. In our example this is the case for "ADiWa" and other projects managed by Heiko (indicated with "..."). We therefore drew the corresponding relations using dotted arcs. Since Heiko's other project called

"SEED" is also reached from another path (coming from "SemDesk" – short for "Semantic Desktop"), it stays part of the network. By doing so, only meaningful implicit annotations are included. A user would possibly wonder himself why there are resources appearing that do not have anything in common with "his" topics or the currently viewed ones, respectively. The purpose of our app is not the proactive delivery of new or related topics, but to ease retrospection.

Coming back to our example, let us assume that we got the following information from our spreading algorithm:

1. Diary $\Rightarrow$ ForgetIT (0.9)
2. Heiko $\Rightarrow$ SEED (0.2)
3. PIMO $\Rightarrow$ SemDesk (0.9)
4. SemDesk $\Rightarrow$ ForgetIT (0.9) $\wedge$ SEED (0.1)

The third "implication" reads as follows: if a concept is annotated with "PIMO" (the assigned weight is 1, since it is an explicit annotation), it is also (implicitly) annotated with "SemDesk" having a weight of 0.9. Additionally (fourth "implication"), being annotated with "SemDesk" implies being also annotated with ForgetIT ($0.9 \cdot 0.9 = 0.81$) and SEED ($0.9 \cdot 0.2 = 0.18$).

Using this information we can extend our concept vectors. First, we also index the new concepts found by the spreading algorithm:
4: ForgetIT, 5: SEED, 6: SemDesk

Next, we perform the actual *concept vector extension*:

$$d'_1 = (1.00, 1.00, 0.00, 0.90, 0.00, 0.00)$$
$$d'_2 = (0.00, 0.00, 1.00, 0.81, 0.18, 0.90)$$
$$d'_3 = (0.00, 1.00, 1.00, 0.81, 0.38, 0.90)$$

Later, similar resources are clustered to diary entries. Let us assume this was the case for the resources of our example. When merging them to form a single entry, their concept vectors are added. Using the original vectors d_1, d_2 and d_3, the arising composite concept vector d_{123} would be $d_{123} = (1, 2, 2)$. So, the most prominent annotations would be "Heiko" and "PIMO". Doing the same using the extended vectors d'_1, d'_2 and d'_3 results in $d'_{123} = (1.00, 2.00, 2.00, 2.52, 0.56, 1.80)$. We see that "ForgetIT", a concept that was not mentioned explicitly, has become the most prominent one with a value of 2.52. We can comprehend this by looking at the annotations: "diary" implies "ForgetIT" directly and "PIMO" implies "SemDesk"

which itself also implies "ForgetIT". Thus, this concept receives "scoring points" from various sources. Imagine having even more information items: a major project like ForgetIT would receive such "scoring points" from annotated topics like "forgetting", "preservation", "diary", etc. Although all these values might be rather small, they possibly add up to make this concept the most prominent one. By this, something like the "lowest common denominator" of several information items might become a suitable summarization (or abstraction) for them.

In the next chapter we will present and evaluate related works and applications.

3. Related Work

This chapter contains related works and applications in research and industry.

Please note that one of this thesis' guidelines was to focus on timelines and diaries, mainly disregarding topics like life logging, trend and topic detection or (mostly sensor-based) augmented (personal) memories.

We already mentioned in the previous chapter that timelines are mainly a visualization concept, whereas diaries are a specific form of text. Since the transition between a timeline decorated with rich background information for all of its items and a diary presenting its (usually rather detailed) entries in chronological order is fluid, we decided to differentiate projects by their main focus. Is it the visualization that is primarily addressed or the "editorial preparation" of events as a text. We will therefore have two sections in this chapter, one about rather timeline-focused works and the other on diary-focused ones.

In a pre-step we first introduce further DFKI projects related to our thesis. All works and applications mentioned afterwards are third-party.

3.1. DFKI Projects

Besides the Semantic Desktop and Personal Information Models that we already introduced in the previous chapter, there are other related projects by the DFKI, which are introduced in the following.

3.1.1. Semantic Editor (SEED)

Although we create an application that generates diaries, we also want to enable users to actively (and directly) contribute and shape their diaries if desired (beyond implicit manipulation coming from their usage of the PIMO). This can be accomplished by introducing *notes* as a new media type to the Semantic Desktop. In addition, we also want to associate these notes with concepts that already exist in the user's PIMO and are also mentioned in the text. This evokes several challenges to be tackled, e.g. solving ambiguities. The word "Paris" appearing in a text might, for example, refer to a city or the name of a person – or even several persons in a user's PIMO share the same first name. One of DFKI's tools called *Semantic Editor (SEED)* tries to solve this problem and (correctly) map concepts to words mentioned in the text by applying natural language precessing (Papadopoulou et al., 2014, p. 17).

Figure 3.1 shows the user interface of SEED. The first line contains the label (or headline) of a note, in this case "Besprechung mit Heiko" (Meeting with Heiko). This line is followed by the note's date. We will later see that mapping an information item to a specific date or time

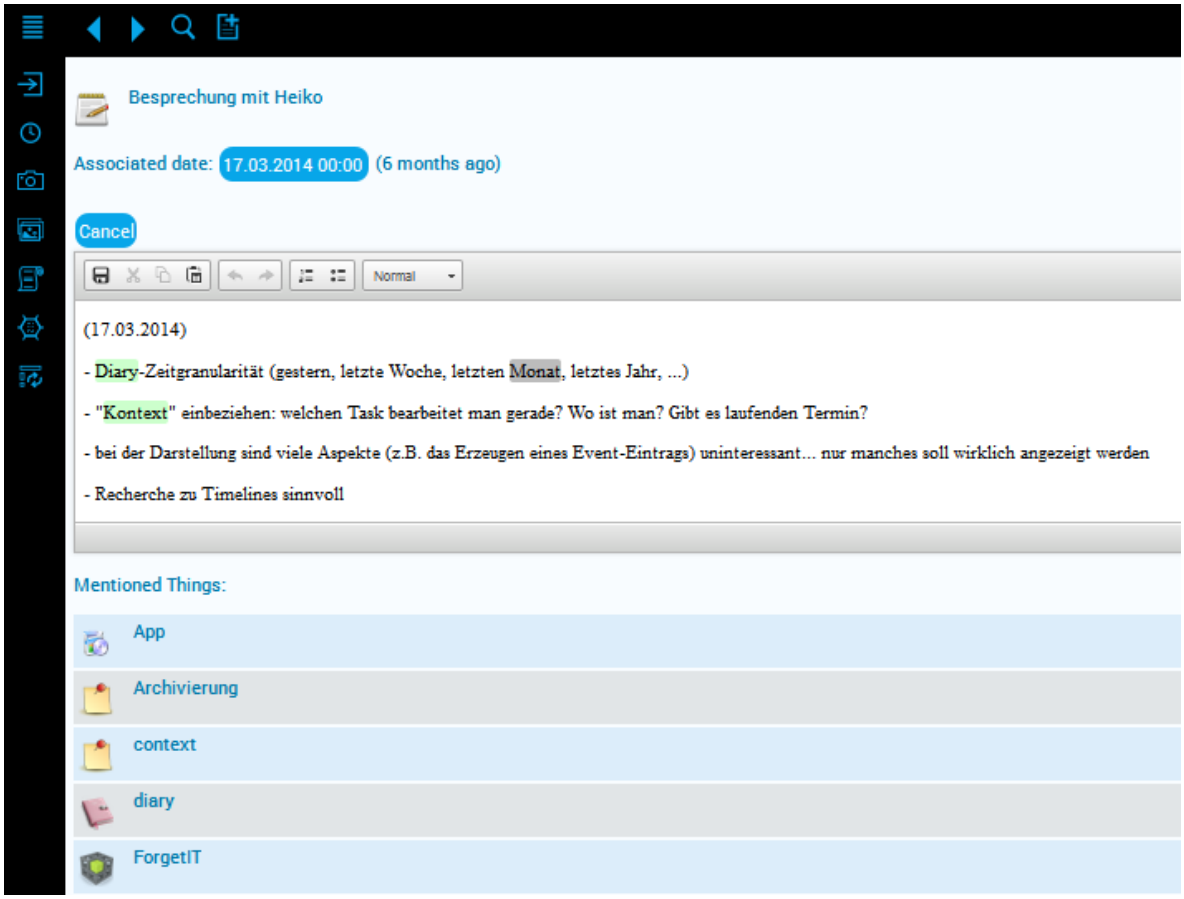

Figure 3.1: SEED: Semantic Editor (by DFKI)

period is a highly non-trivial task, which (in parts) even exceeds this thesis' scope. In the case of notes, we assume that they always refer to a single point in time, which can either be their *creation date* or an *associated date* given by the user. In this example, the note was written in April but refers to a meeting on March 17th. In consequence, the date of March is displayed. The main part of the user interface is the actual text editor. Words possibly matching to concepts found in the user's PIMO (or in external sources like *Freebase*[11] or *DBPedia*[12]) are highlighted in gray. If the user confirms a mapping, the word is afterwards highlighted in green. In our example, this is the case for "diary" and "context". All concepts mapping to words in the text are additionally listed in the lower part of the application (section entitled with "Mentioned Things"). We see that concepts like "app", "archiving" and "ForgetIT" were also found in the text (actual text passage not visible on the screenshot).

Like described before, we use SEED as a service in order to get notes as an additional – but very direct – input for our diary.

[11] Freebase: "a community-curated database of well-known people, places, and things" (*www.freebase.com*)
[12] DBPedia: "a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web" (*www.dbpedia.org*)

3.1.2. PIMO Reminiscence (PIMORE)

PIMO Reminiscence (or *PIMORE* for short) is a tool developed in the context of a Bachelor's thesis written at DFKI, which was about the "process of accessing information as an interplay between forgetting and remembering" (Jerke, 2013). This tool helps users in composing a family photo collection.

According to studies mentioned in that thesis, people especially want to remember (or preserve) certain moments or situations in life. Thus, three sample life situations were defined: *birth*, *vacation* and *wedding*. The right-hand side of Figure 3.2 shows that for every life situation (examples on the left-hand side) there is a *pool of photos*, which may be tagged by the user of being *preferred*, *disliked* or *scrapped*. As a consequence, the life situation is later probably best expressed by only including the preferred photos, since they were explicitly chosen by the user reflecting a high emotional relevance (Jerke, 2013, p. 32).

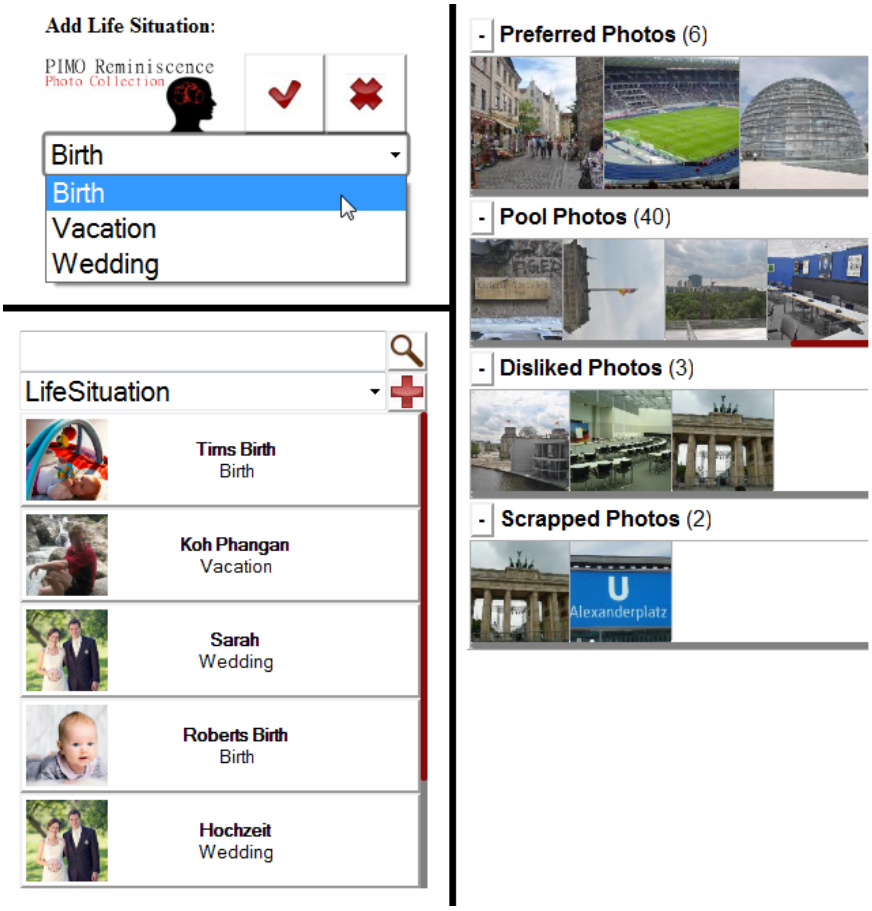

Figure 3.2: PIMORE: PIMO Reminiscence (by Jerke (2013), DFKI)

In our diary app we can utilize these life situations in order to create abstractions for several single information items, e.g. all photos belonging to a certain situation.

3.1.3. PIMO Timeline

More closely related to our diary app is a DFKI application called *PIMO Timeline*. It is a tool that shows which things (own or shared) in a user's PIMO were created in which periods of time. Its user interface is depicted in Figure 3.3.

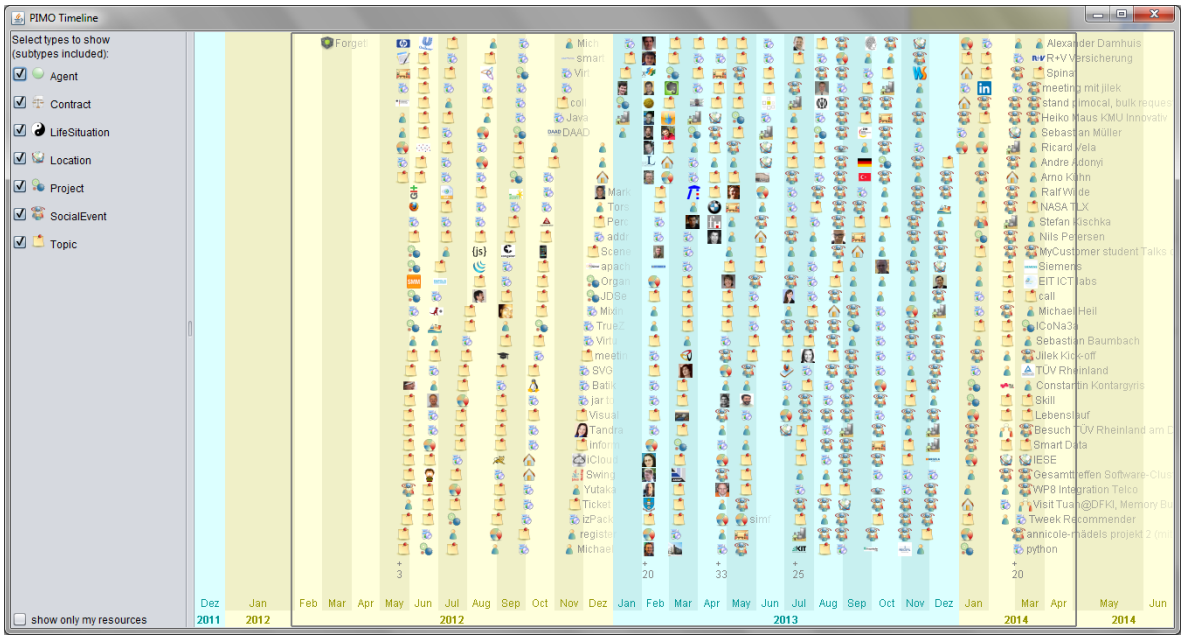

Figure 3.3: PIMO Timeline (by DFKI)

On the left-hand side different types of things can be filtered: *agents*, *contracts*, *life situations*, *locations*, *projects*, *social events* and *topics*. The main part of the user interface contains columns representing time intervals (days, months or years) and icons representing concepts created in these periods.

We do not intend to be overly critical, since we know it is a makeshift solution, but the screenshot already indicates that there are problems concerning clarity. Viewing larger periods of time (or even only short ones, depending on individual usage of the PIMO) often leads to an overwhelming mass of icons that is presented to the user. There is a feature to widen selected time periods (similar to a fish eye lens), which leads to less icons being crowded together in these particular periods, but leaves the very confusing situation in the surroundings before and after these periods.

In our diary app we would like to include more types of things and provide a better overview than *PIMO Timeline*. Besides creating diary entries – text bodies with headlines and possibly attached media files like photos – we also want to utilize semantic relations between the different concepts in order to better structure the data or build abstractions.

3.1.4. ForgetIT

The DFKI is involved in the *ForgetIT* project funded by the European Community, for example by contributing the PIMO or SEED.

> *"ForgetIT combines three new concepts for easing the adoption of preservation in personal and organizational contexts:*
>
> - *Managed forgetting should complement, not copy human memory by supporting resource selection for preservation and creating immediate benefit from preservation adoption.*
> - *Synergetic preservation enables a smooth transition between active use and preservation.*
> - *Contextualized remembering keeps the archive understandable and useful.*
>
> *ForgetIT brings together an interdisciplinary team of experts including cognitive psychology."*
>
> (Niederée, 2013, p. 2)

Being the application scenario of ForgetIT's deliverable about personal preservation, our diary is also related to this project (Maus et al., 2013a, pp. 22). We therefore adopted and extended their personas in our usage scenarios found in Section 4.1.

After having introduced all related DFKI projects we will continue with third-party works, starting with the rather diary-related ones.

3.2. Diary-related Works

This section is about applications whose focus is generating diaries. Like we stated in Chapter 1.2, two very important problems in the context of diary systems are acquiring and handling possibly huge amounts of data and presenting them to the user in a comprehensible, not overwhelming way. Cho et al. (2007, p. 66) also point out these problems explicitly. Concerning the first aspect, the following applications share the idea of acquiring the data by reading out the logs of mobile devices. Since many people carry a smart phone with them all day, it appears obvious to ask this "digital companion" about a user's experiences and events (Liao et al., 2012, p. 1). Regarding the second aspect the authors follow different approaches as discussed below.

3.2.1. ComicDiary (2002)

The first diary tool we would like to present is called *ComicDiary*. Like the name suggests, in addition to short texts reflecting the events and experiences of a person, the tool adds small

comic-style images as depicted in Figure 3.4.

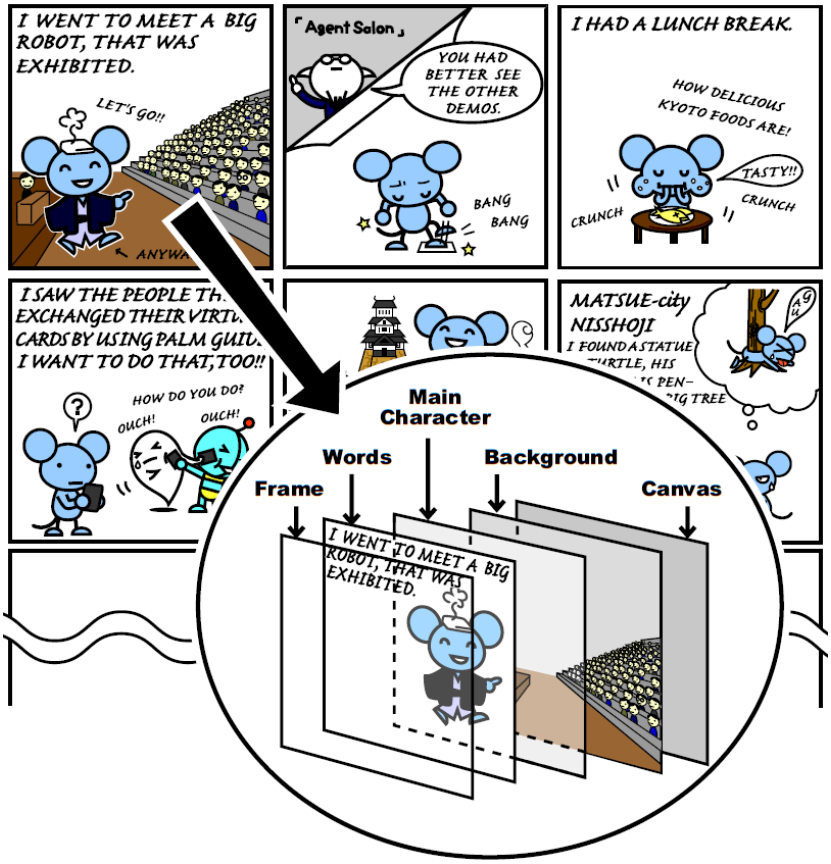

Figure 3.4: ComicDiary: Cartoons (Sumi et al., 2002, p. 24)

ComicDiary was built as a sub-system of a project called *C-MAP* which aimed at developing a personal guidance system for exhibition touring at museums, trade shows, academic conferences, cities and so on (Sumi et al., 2002, p. 16). The authors created two prototypes, one is a web service and the other runs on hand-held/palm devices. Both were designed and tested in the context of an academic conference. They generate comic diaries from a user's touring history and records of interactions with others, e.g. virtual business card exchanges and accesses to the "AgentSalon" – a meeting facilitator. Data used to generate diaries are divided into two parts: personal and community data. The personal data includes age, gender, participant type (i.e. whether a person has an own presentation on a conference or not), touring history (attended presentations and their ratings), and the previously mentioned interaction records. The community data comprises plenary events (e.g. reception and invited talks), tourist information, or socially shared impressions (e.g. popularity of a presentation). To have a greater variety of comic frames from limited resources, each image is created by combining different layers, e.g. words, main character or background, like it is shown in

Figure 3.4 (Sumi et al., 2002, p. 23). The system's architecture is depicted in Figure 3.5.

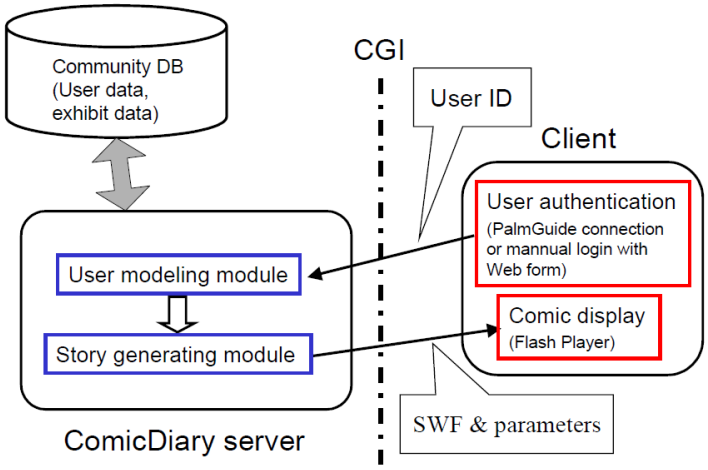

Figure 3.5: ComicDiary: System Architecture (Sumi et al., 2002, p. 21)

A similar but more recent approach is called *AniDiary* and is discussed in the next section.

3.2.2. AniDiary (2007)

AniDiary (which stands for *Anywhere Diary*) can detect and visualize memory landmarks and transform numerous (mobile device) logs into user-friendly cartoon images like depicted in Figure 3.7 (Cho et al., 2007, p. 66). Figure 3.6 shows its system architecture.

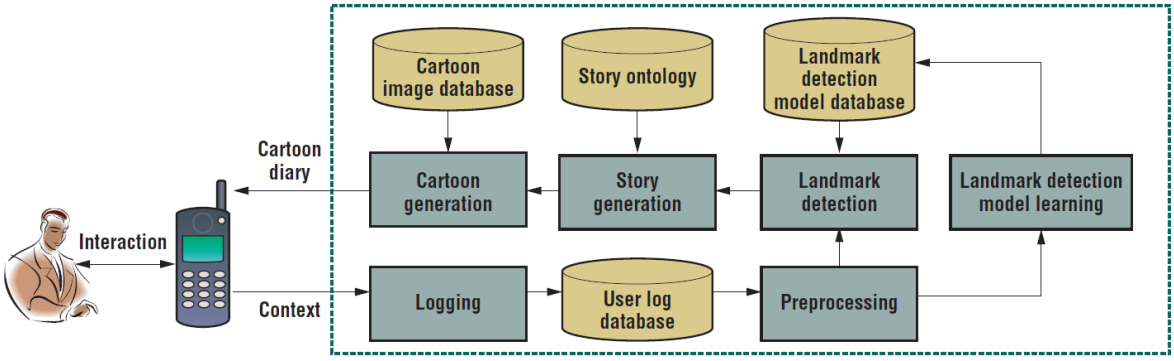

Figure 3.6: AniDiary: System Architecture (Cho et al., 2007, p. 67)

The system's *logging* component continuously records GPS data, accesses call logs and the address book, stores SMS texts and logs usage information of the photo viewer and MP3 player. Using a web service the GPS data is mapped to the nearest building and a label like the street or building name is assigned. The user may also customize these labels by reassigning values like "my home", "my office" or "my friend's home" (Cho et al., 2007, p. 67).

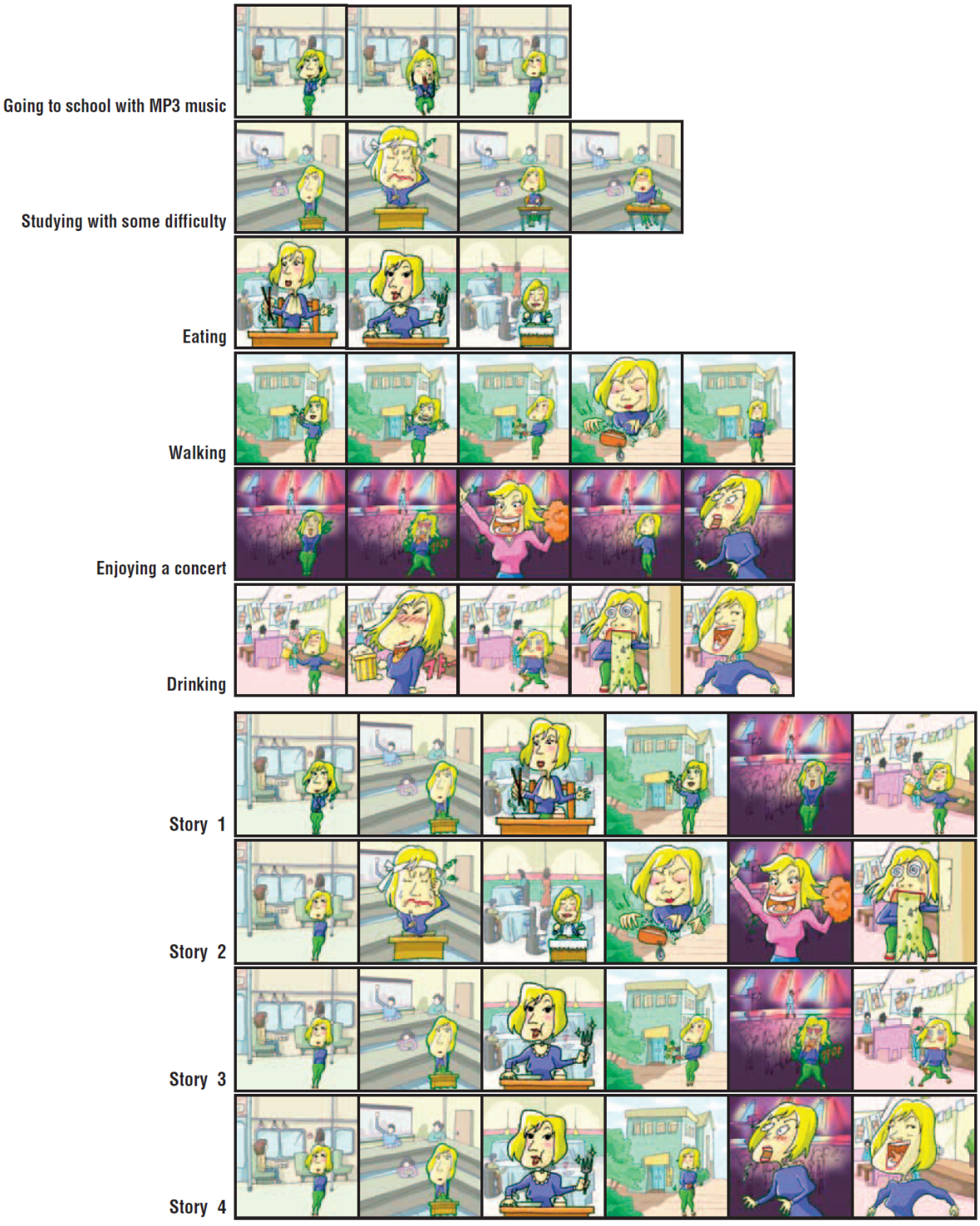

Figure 3.7: AniDiary: Cartoons (Cho et al., 2007, p. 72)

In a *preprocessing* step the systems tries to detect statistical variations in the raw data in order to find informative situations. This analysis comprises simple techniques like "determining the average, maximum, and minimum values or the frequency over the time domain" (Cho et al., 2007, p. 67). In our diary app we will follow a similar approach as described in later chapters.

To infer landmarks (*landmark detection* component) the system applies machine learning techniques by means of (modular) Bayesian networks. We will address these models in more detail in Section 3.3.6. The authors also published two more papers about this topic in addition to the AniDiary paper (Hwang and Cho (2006) and Hwang and Cho (2009)).

Concerning the actual *cartoon generation* they suggest reorganizing the detected landmarks to avoid a boring or redundant story which might be the case if the landmarks are ordered chronologically (Cho et al., 2007, p. 69). Fostering a high diversity within the diary is also a key requirement in our application (please see Section 4.6.1). As with *ComicDiary*, the cartoon images are created by combining different layers: text, main character, sub-character, main background and sub-background (Cho et al., 2007, p. 69). We see in Figure 3.7 that there are several images for the same situation, e.g. three for eating and five for walking (upper half of the figure). In order to create a concrete story, these images are combined (lower half of the figure).

AniDiary has been implemented for Nokia and Windows Mobile smart phones.

The application we would like to discuss next uses a different presentation style: instead of comics actual texts are generated to reflect the user's experiences (see Figure 3.8). It is called *Smart Diary*.

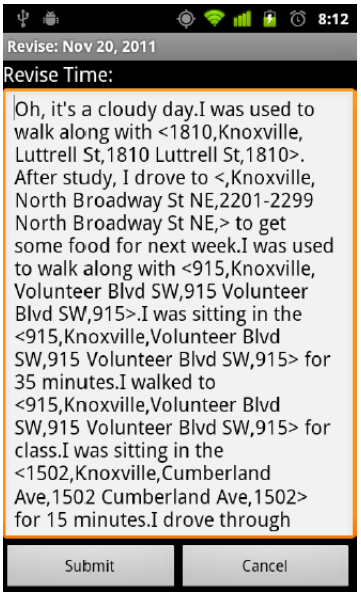

Figure 3.8: Smart Diary (Liao et al., 2014, p. 9)

3.2.3. Smart Diary (2012/2014)

Like the other diary systems mentioned before, *SmartDiary* acquires its data by reading out sensors and app usage statistics of smart phones. Its system architecture, which consists of four layers, is shown in Figure 3.9.

The lowest layer is the *raw data collection*, in which data from six different sources is collected: motion activity, location data, app usage, calendar events, phone calls or SMS messages, and the web history (Liao et al., 2014, p. 3).

In the layer above, which is called *context analysis*, multiple events from the users' life, classified either as entertainment activities, social activities or health conditions, are extracted from the raw data by means of several mining components. The authors propose a so-called *sustainable mining model*, "which decomposes a mining component's algorithm procedures into separate processing units. These units will continuously shuffle raw data, and provide the relevant ones to all the mining components where events are assembled". Processing of events "either adopts existing algorithms or relies on user-specific logic rules" (Liao et al., 2014, p. 3).

The events extracted by the context layer analysis will in a next step be evaluated whether they are more or less important to the user. In this process, which is called *(event) personalization* by the authors, a combination of event ranking and filtering is applied in order to find the most relevant ones according to the users' preferences. Importance evaluation is also a major topic in our diary application.

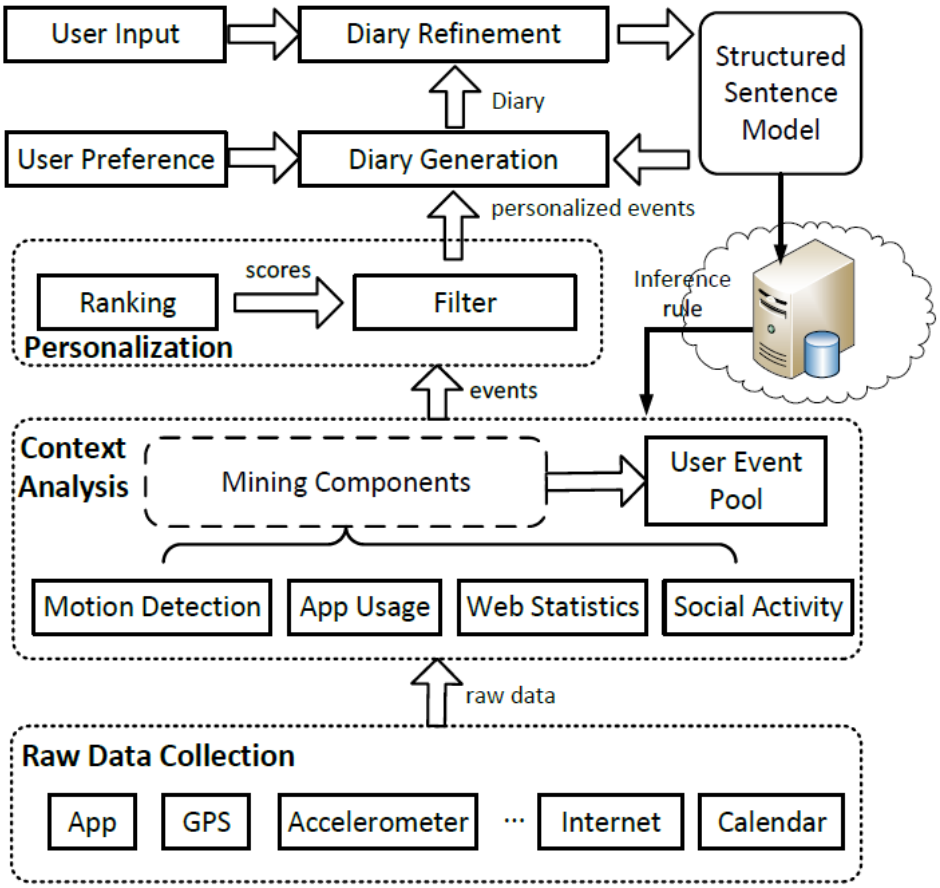

Figure 3.9: Smart Diary: System Architecture (Liao et al., 2014, p. 3)

In the highest layer, the *diary generation*, personalized events are translated into human-

readable sentences. For this purpose the authors propose a *narrative structured sentence model*, of which one key feature is to use regular expression formats in order to construct natural language sentence templates (Liao et al., 2014, p. 3). "During runtime, real event properties, such as the time of the day and the user's activities replace these wild cards in the regular expression templates to generate diary outputs. Furthermore, to make language output natural, each type of events has multiple corresponding structured sentence templates for use" (Liao et al., 2014, p. 4).

Users may provide additional feedback regarding the generated diaries in an optional step (*diary refinement*). For example, they may want to share sentences with others or revise an existing sentence. "In practice, this stage is not only useful for improving the quality of the diaries, but also for enhancing the narrative structured sentence model by adopting better structured sentences for each event" (Liao et al., 2014, p. 4).

Smart Diary is implemented on Android smart phones and a sample of the generated diary entries is given in Figure 3.8.

3.2.4. Other Diary Applications

Apart from the previously mentioned – rather academic – examples, we could not find any application that actually generates diaries, i.e. texts based on users' experiences and events reflected by their information items (data tracks). All applications found are either diary apps in the sense that they help in organizing or writing daily notes, i.e. a kind of specialized text editor, or more or less PIM tools providing timelines reflecting the history of users' interactions with files, emails, etc.

Smart Diary Suite Both aspects are covered by the *Smart Diary Suite*[13], an application we exemplarily picked out of the set of commercial tools. It is developed and distributed by Programming Sunrise[14]. Figure 3.10 shows three screenshots of the application, that, in particular, depict the system's overview panel and the diary and notes sections. The screenshots indicate that there might be a problem concerning (global) clarity – a problem most of these tools have. Since semantic interconnections or abstractions are not inferred, a large sequential list of individual information items like notes, appointments, diary entries, etc. is presented to the user in all parts of the application. Finding out what actually happened in a given time interval, for example last year, is only possible if potentially lots of individual items are scanned and mentally connected or summarized by the user. We will address this problem again in this chapter's conclusion (Section 3.4).

Next, we will examine several timeline-related works.

[13] although both (partly) share the same name, this tool has nothing to do with the formerly discussed application by Liao et al.

[14] for details please see *http://www.sdiary.com*

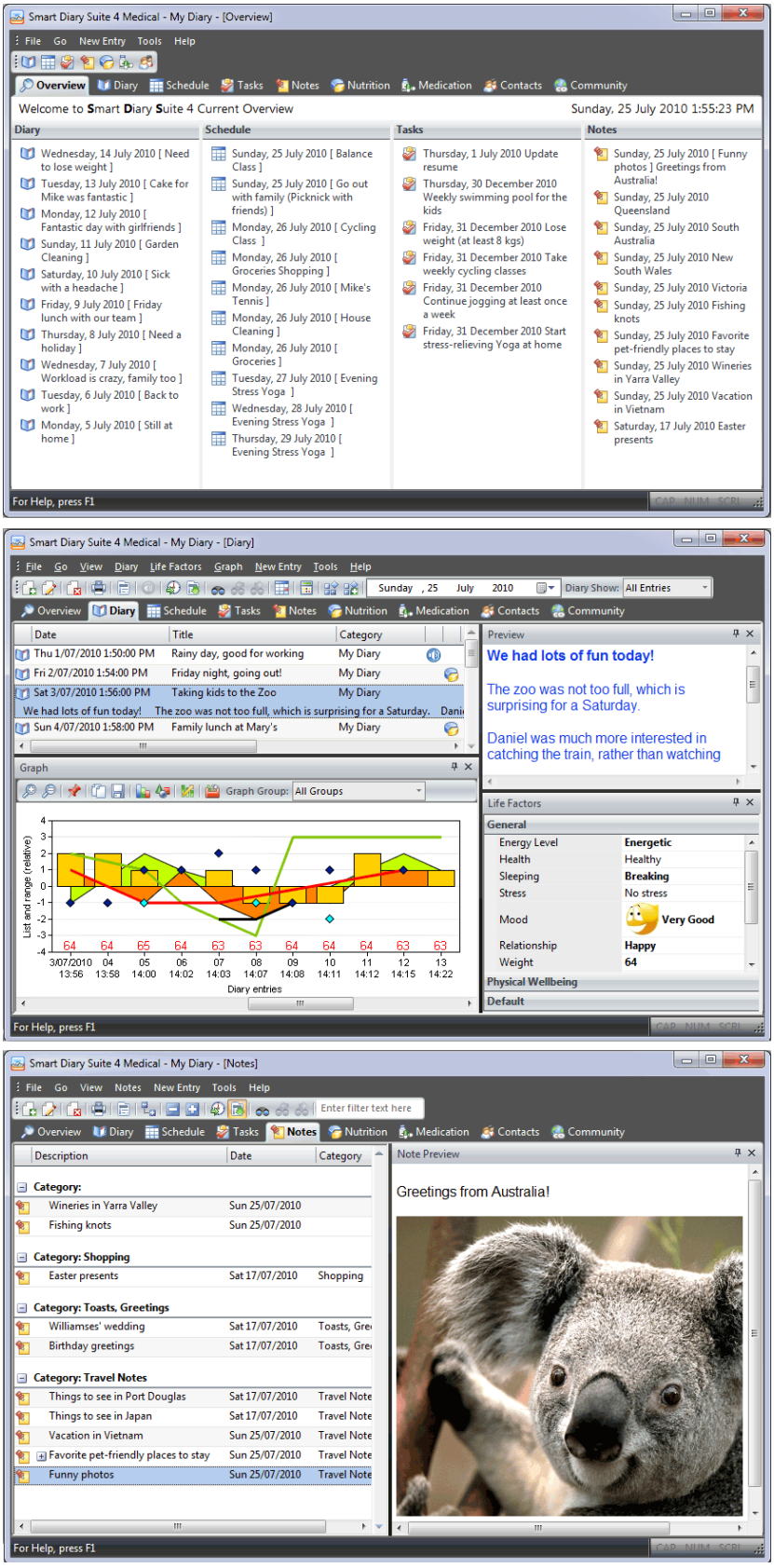

Figure 3.10: Smart Diary Suite (by Programming Sunrise)

3.3. Timeline-related Works

After having presented different diary applications in the previous section we will focus on timeline-related works in the following.

3.3.1. LifeLines (1996/1998)

The first and oldest system we would like to discuss is called *LifeLines*. It provides "a general visualization environment for personal histories that can be applied to medical and court records, professional histories and other types of biographical data" (Plaisant et al., 1996, p. 2). According to André et al. (2007, p. 103) *LifeLines* is the first system "to bring together the full gamut of problems facing timelines: the overview, hierarchy, rescaling, interrelationships, and layout issues". This is not surprising since Ben Shneiderman, author of the frequently cited *visual information seeking mantra* is among the authors. His mantra, which reads as *"Overview first, zoom and filter, then details-on-demand."* (Shneiderman, 1996, p. 337), was published in the same year and obviously also coined this application.

Like depicted in Figure 3.11, multiple facets of the records are displayed as regions on the screen, e.g. problems, allergies etc. Stories or aspects with varying status (e.g. medical conditions or legal cases) are displayed as horizontal lines, while icons indicate discrete events. "Line color and thickness illustrate relationships or the significance of events. *LifeLines* always begin with a one screen overview of the record, and rescaling tools or filters allow users to focus on part of the record and see more details" (Plaisant et al., 1996, pp. 2).

Since we will adopt some of *LifeLines'* visualization features in our application, we will discuss them more thoroughly in the following (Plaisant et al., 1998, pp. 78):

- **Details on demand:** By clicking on events, detailed information appears in a separate page covering part of the display or optimally in tiled windows on the side. Hovering over an event displays its (extended) label.
- **Zooming:** Zooming in and out can be done either by a slider or by clicking on the regions of the screen to be zoomed (left mouse button zooms in, right mouse buttons zooms out).
- **Highlighting Relationships:** In addition to the implicit horizontal and vertical relationships, searching for a specific term highlights all its occurrences in the record.
- **Coding attributes:** There are settings in *LifeLines* that allow mapping the main display attributes (label, color and line thickness) to the data attributes. Depending on the preferences of the user, severity can thus be signaled by a red color or a thick line, for example.

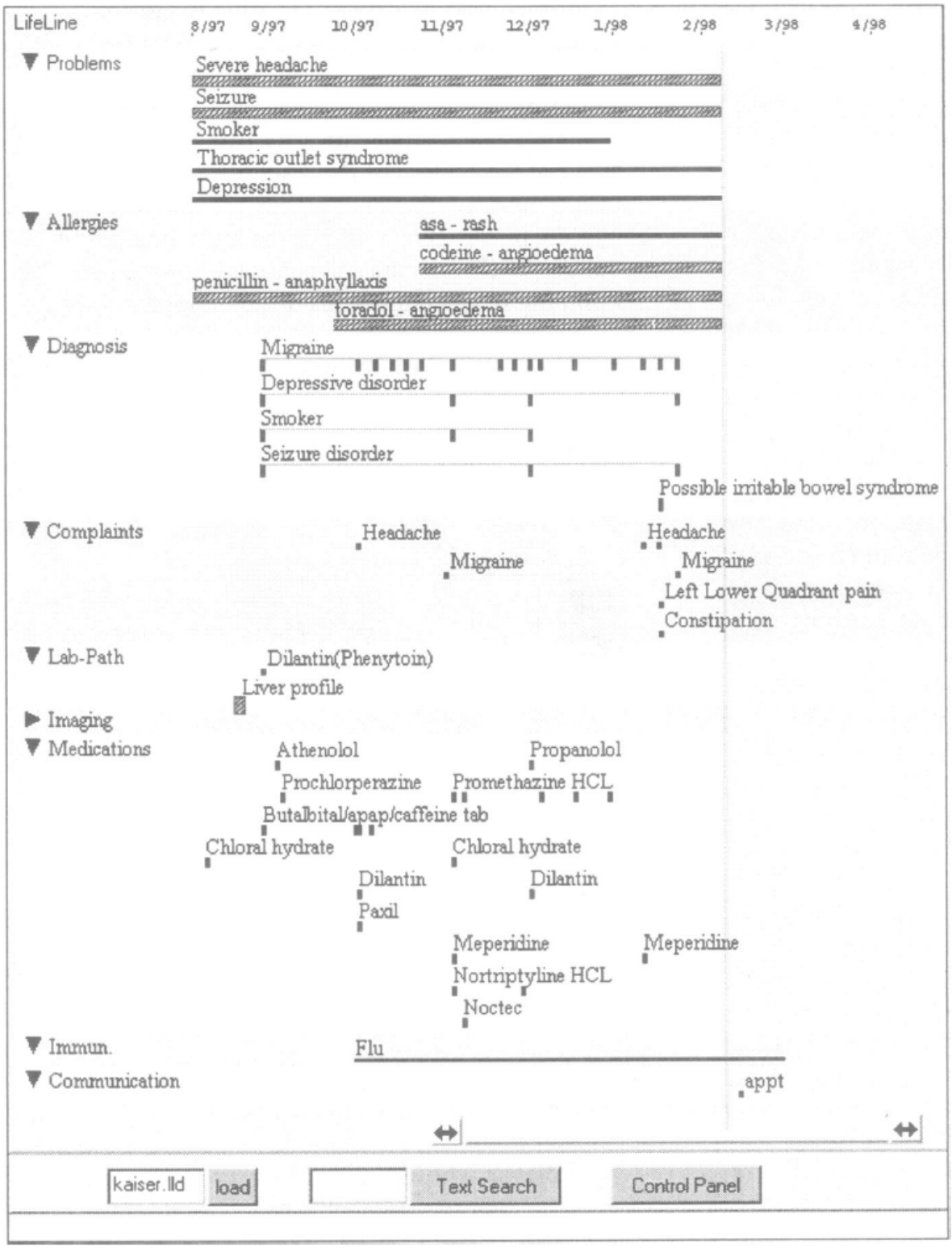

Figure 3.11: LifeLines (Plaisant et al., 1998, p. 77)

- **Outlining:** "Facets can be opened and closed in an outliner fashion. A closed facet only reveals the 'silhouette' of a record, i.e. compact lines with no labels, but color is preserved. Those silhouettes are useful to estimate the volume and type of information available and to guide users to the most important parts of the record." (Plaisant et al., 1998, p. 79)

- **Summarizing:** The app allows a set of events within a facet to be recursively aggregated and replaced with summary events, e.g. a series of *athenolol* and *propanolol* prescriptions can be aggregated as *beta-blockers*.

Plaisant et al. (1996, p. 4) name four possible benefits of *LifeLines*:

1. Reduce the chances of missing information.
2. Facilitate the spotting of anomalies and trends.
3. Streamline the access to details.
4. Remain simple and tailorable to various applications.

Concerning their use cases of medical and court records, Plaisant et al. (1996, p. 2) wrote the following:

> *"Once gathered in a single record, the information is often in the form of a puzzle and the reader has to browse the date in order to form the big picture of the record."*

This is a statement we would like to adopt as a metaphor for our diary application. The life of a user – or actually his PIMO – can be seen as a large puzzle. The individual information items are the pieces that need to be sorted, turned around, and rearranged in similar looking groups to step by step put together larger parts until finally the big picture manifests.

The second timeline application we would like to present is a tool built on top of a system called *Stuff I've Seen*.

3.3.2. Stuff I've Seen (SIS) (2003)

SIS *Stuff I've Seen (SIS)* is a personal search engine developed by Dumais et al. (2003) that provides a unified index of personal content (Ringel et al., 2003, p. 184). We are especially interested in the timeline visualization feature that Ringel et al. (2003) built on top of it.

The creation of *SIS* was motivated by studies revealing "that 58-81% of web pages accessed were re-visits to pages previously seen. Similar re-access patterns have been observed in usage of Unix commands, library book borrowing, and human memory" (Dumais et al., 2003, p. 72). As a consequence, *SIS* was developed with the intention of making it easy for people to find things they have seen before. This is supported by two key features. "First, the system

provides a unified index of information that a person has seen on their computer", e.g. emails, web pages, documents, media files, calender appointments, etc. "Second, because a person has seen the information before, rich contextual cues such as time, author, thumbnails and previews can be used to search for and present information" (Dumais et al., 2003, p. 72).

Figure 3.12 shows the search results of a *SIS* query.

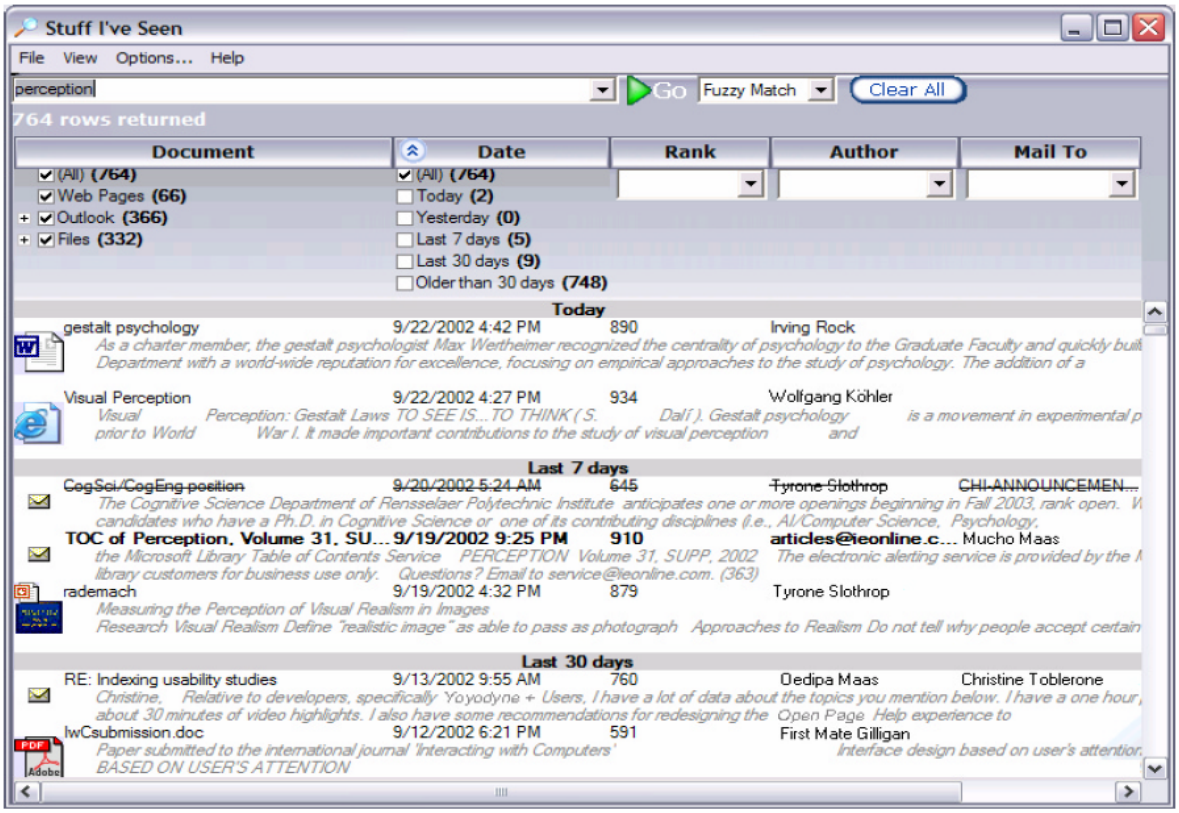

Figure 3.12: Stuff I've Seen (SIS) (Dumais et al., 2003, p. 74)

We see that documents can be filtered by several types. Date categories are *today*, *yesterday*, *last 7 days*, *last 30 days* and *older than 30 days*. The preview of each individual search result contains the first 300 characters of a message or text file as well as thumbnails for images or presentations. Search results can be sorted by different criteria like date or rank.

SIS Timeline Visualization The additional timeline visualization feature was developed to probe "the value of timelines and temporal landmarks for guiding search over subsets of personal content". In contrast to earlier approaches in the second half of the 1990s and around the turn of the millennium, the system "uses a mix of personal and public landmarks as memory cues" (Ringel et al., 2003, p. 184). Its user interface is depicted in Figure 3.13.

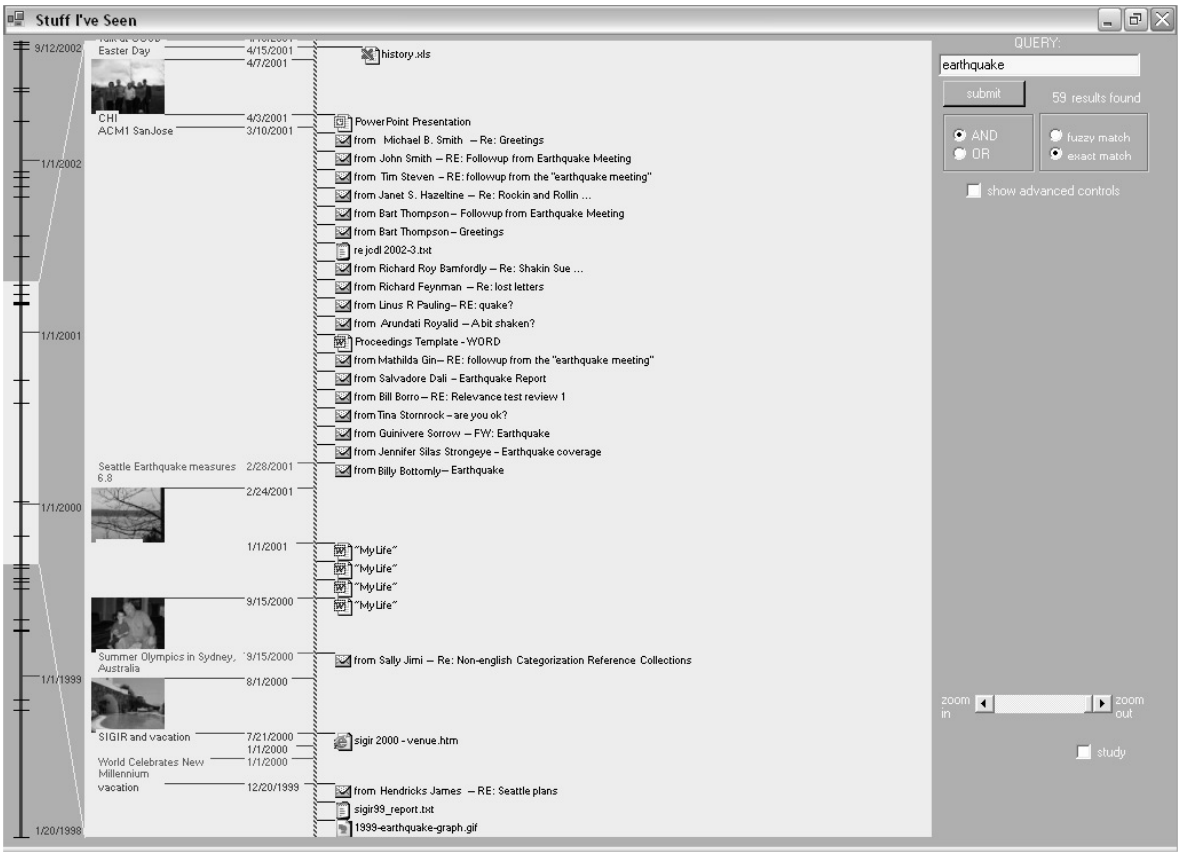

Figure 3.13: SIS Timeline Visualization (Ringel et al., 2003, p. 186)

It basically consists of four parts. On the left edge of the display there is the overview timeline, having the date of the earliest search result at the bottom and the one of the newest at the top. If the search results span over a year, the borders between years are also labeled with a date. We see that a part of this timeline is highlighted (lighter gray in the screenshot). This part is expanded in the detailed area, which covers the rest of the display. "Users can interact with the overview timeline as if it were a scroll bar, by grabbing the highlighted region with their mouse cursor and dragging it to a different section of the timeline, thus changing the segment of time that is displayed in the detailed view" (Ringel et al., 2003, p. 185). Landmarks as well as their dates are located in the left part of the detailed view. We will later see that there are four types of landmarks, all of them appear in a different color. The right part of the detailed view contains the titles and icons of all documents most recently modified (for most files) or the time an email message was received. Hovering over a search result pops up a summary containing more detailed information about the object, i.e. the full path, a preview of the first 512 characters of the document as well as *from* and *cc* information in case of an email. Clicking on a result opens the target item with the appropriate application (Ringel et al., 2003, p. 186).

Like mentioned before, there are public and personal landmarks. The public ones "are drawn from events that a broad base of users would typically be aware of" (Ringel et al., 2003, p. 186): *holidays* and *news headlines*. Personal landmarks are unique for each user: *calendar appointments* and *digital photographs*.

One use case of our diary app is to embed the user's own diary into a historic context, which is comparable to inserting public landmarks into a timeline of personal data.

3.3.3. SIMILE Timeline (2006-2009)

The *SIMILE Timeline* is an application developed at the Massachusetts Institute of Technology (2006 to 2009) and later maintained by the SIMILE Widgets Community. It is a popular open-source Web 2.0 widget, that is widely available and highly interactive (André et al., 2007, pp. 102).

Like the screenshot in Figure 3.14 shows, "a uniform overview timeline presents context while a more detailed view focuses on a specified area within the time space. Hierarchy and relationships are not dealt with explicitly, but permitted to certain extends through controls such as color" (André et al., 2007, p. 103).

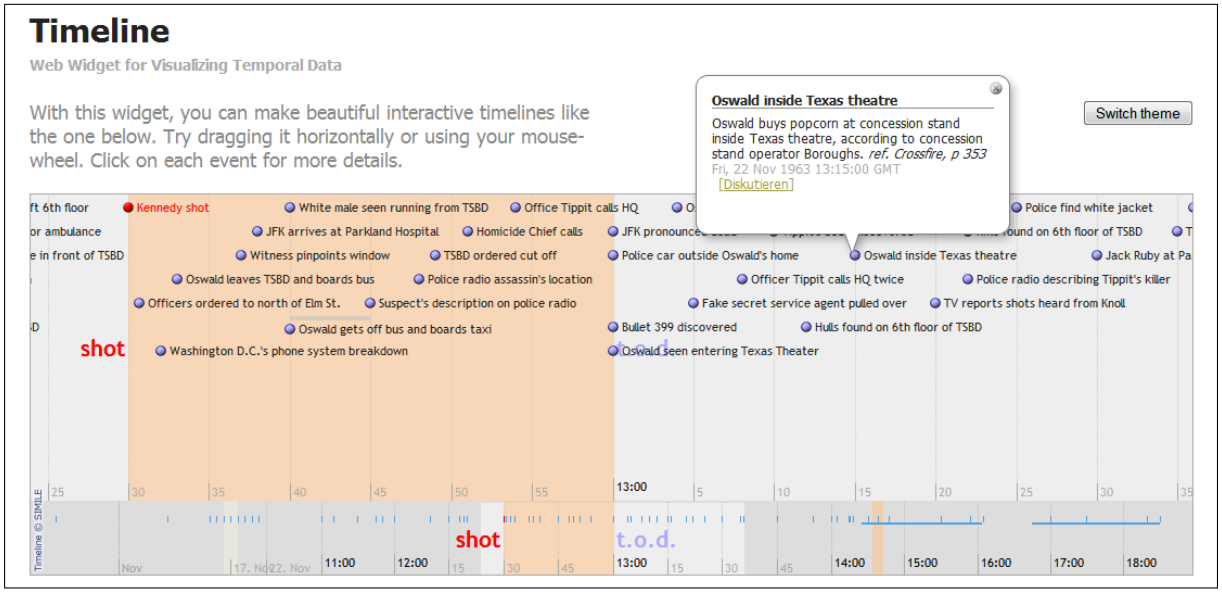

Figure 3.14: SIMILE Timeline (Massachusetts Institute of Technology, 2009)

Clicking on individual events leads to detailed information popping up, like it is also shown on the screenshot. Zooming effects can be realized using so-called *hot zones* which distort parts of the timeline creating more space for time periods containing lots of individual events (SIMILE Widgets Community, 2010).

3.3.4. Continuum (2007)

André et al. (2007) developed a tool for faceted temporal browsing called *Continuum*. It enables

- "hierarchical relationships in temporal data to be represented and explored;
- relationships between events across periods to be expressed;
- user-determined control over the level of details of any facet of interest so that the person using the system can determine a focus point, no matter the level of zoom over the temporal space" (André et al., 2007, p. 101).

The motivation behind the system is that "temporal events, while often discrete, also have interesting relationships within and across times; larger events are often collections of smaller more discrete events (battles within wars; artists' works within a form); events at one point also have correlations with events at other points (a play written in one period is related to its performance, or lack of performance, over a period of time). Most temporal visualizations, however, only represent discrete data points or single data types along a single timeline" (André et al., 2007, p. 101). The authors therefore propose *Continuum*, whose user interface is shown in Figure 3.15.

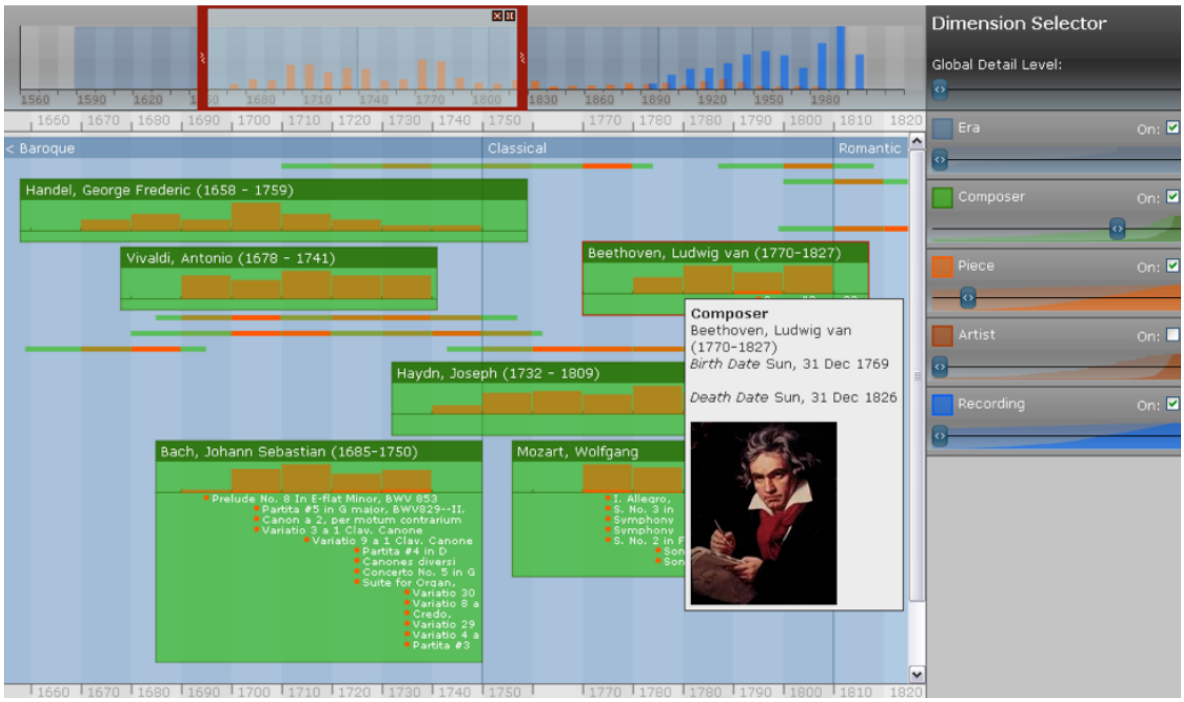

Figure 3.15: Continuum (André et al., 2007, p. 101)

They chose the context of classical music for their presentation of the tool. Its interface basically consists of three panels. On the right edge of the display there is the *dimension filter panel* having several sliders to regulate the amount of *global details* and in the particular

example of classical music the details of five different facets: *era*, *composer*, *piece*, *artist* and *recording*. In the main part of the view there is an *overview panel* at the top, which always provides a complete representation of the whole dataset by means of a scalable histogram. "Typically, timeline visualizations that include an overview, such as the SIMILE timeline, simply show the same information as the detail view, but on a much smaller scale. However, for such tools, as the detail view overflows, so does the overview" (André et al., 2007, p. 104). The third panel is the already mentioned *detail view* which occupies most of the display's space. We see that "child nodes", in this case composers, are drawn as green boxes within their "parent nodes", which in this case are eras represented by blue boxes in the background (André et al., 2007, p. 105).

For our diary app the scalable histogram overview as well as the ideas concerning the representation of hierarchical data could be of interest. Since *Continuum* has been developed as a standalone JavaScript widget, we could possibly use it (or parts of it) in our app. Unfortunately, a public beta version announced for the end of 2007 (mSpace Project, 2007) has – to our best knowledge – not yet been released.

While the main focus of *Continuum* was searching by browsing faceted temporal data, the app we will introduce next focuses on contextual search. It is called *YouPivot*.

3.3.5. YouPivot (2011)

YouPivot *YouPivot* is "a contextual history based search tool" (Hailpern et al., 2011, p. 1522). It was motivated by studies of cognitive science. When people try to retrieve a document, a website, or a file, semantic information (e.g. name, URL, system, path, etc.) cannot be recalled, but environmental factors can. Theses factors may be music they have listened to while working at the document, a special place they have been, or a phone call that interrupted them, etc. Those temporally related activities are referred to as *contextual cues*. They do not have to match semantically to the search target. While using these cues is a natural method of recall, modern computers do not yet support this form of contextual search (Hailpern et al., 2011, p. 1521). *YouPivot* bridges this gap: it allows users to search through their digital history (e.g. files, URLs, physical location, meetings, and events) for the context they *do* remember. Users "can then *Pivot*, or see everything that was going on while that context was active. Further, *YouPivot* displays a visualization of the user's activity, providing another method for finding context" (Hailpern et al., 2011, p. 1521).

TimeMarks Another interesting feature introduced in *YouPivot* is a new annotation method called *TimeMarks*. Like depicted in Figure 3.16, a user has the possibility to mark a moment in time as being important. "This leaves a temporal landmark for later contextual recall.

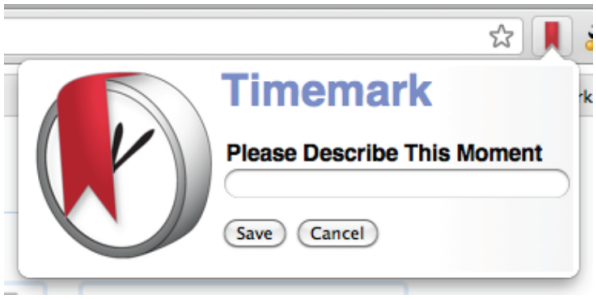

Figure 3.16: YouPivot: TimeMarks (Hailpern et al., 2011, p. 1525)

Because *YouPivot* logs a user's personal activity, *TimeMarks* effectively bookmarks all the user's activity at that moment (open webpages, files, songs, physical location etc.) for easier recall. Manually placing memory landmarks might also be an interesting feature for our diary app.

YouPivot's user interface is depicted in Figure 3.17.

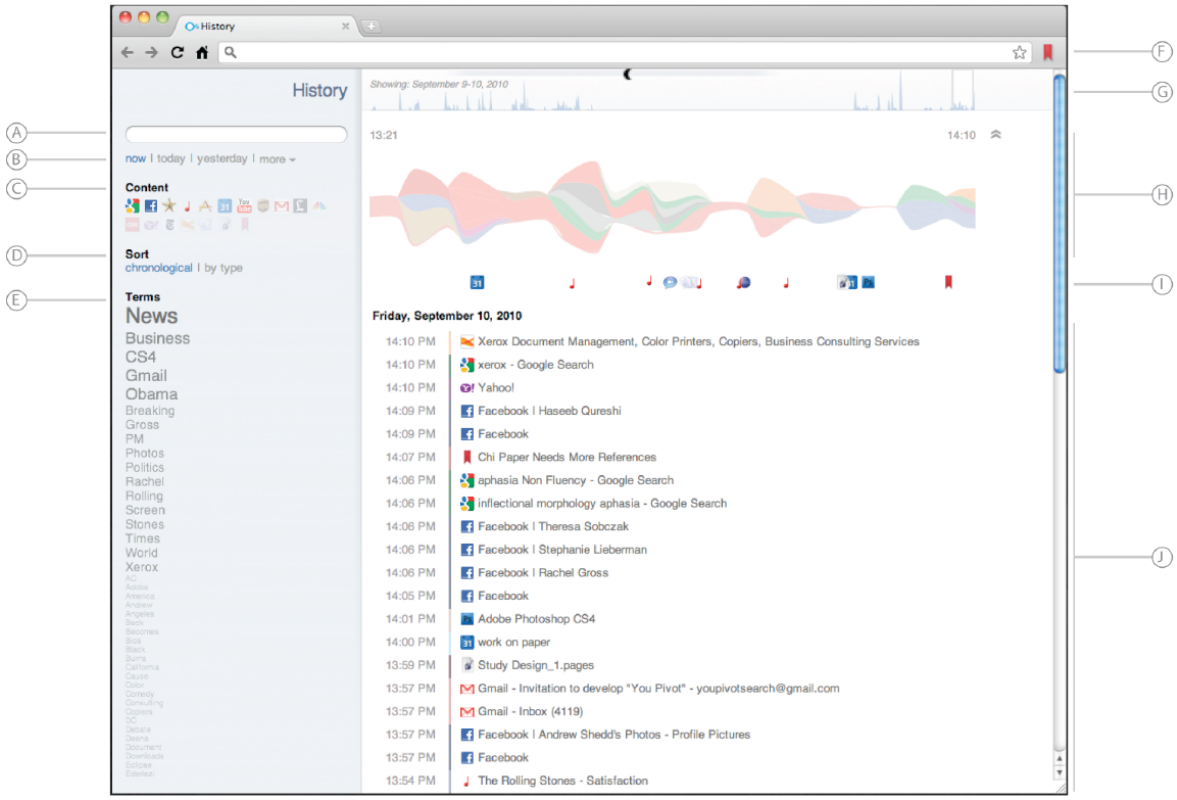

Figure 3.17: YouPivot (Hailpern et al., 2011, p. 1523)

Besides the usual search and sorting options (letters A to E) there is a button to place

a *TimeMark* (F), a *24-hour histogram of activities* (G), a *modeled activity visualization* by means of a stream graph[15] (H), a *non-modeled activity visualization* (I) and a *history list* (J).

The application is freely available as an HTML5 app (Hailpern, 2012).

3.3.6. Life Browser and Memory Lens (2004/2012)

Life Browser In 2012 an application called *Life Browser* was presented by Microsoft. It is a system that learns to predict memory landmarks and uses those landmarks to help users navigate through large stores of their own personal information (Microsoft Corporation, 2012). The system includes photos, search and browsing activity, documents, appointments etc. Users have the possibility to browse their desktop by navigating a (big) timeline containing memory landmarks as well as information items. Figure 3.18 shows its user interface captured from a presentation video by its creator Eric Horvitz (Microsoft Corporation, 2012).

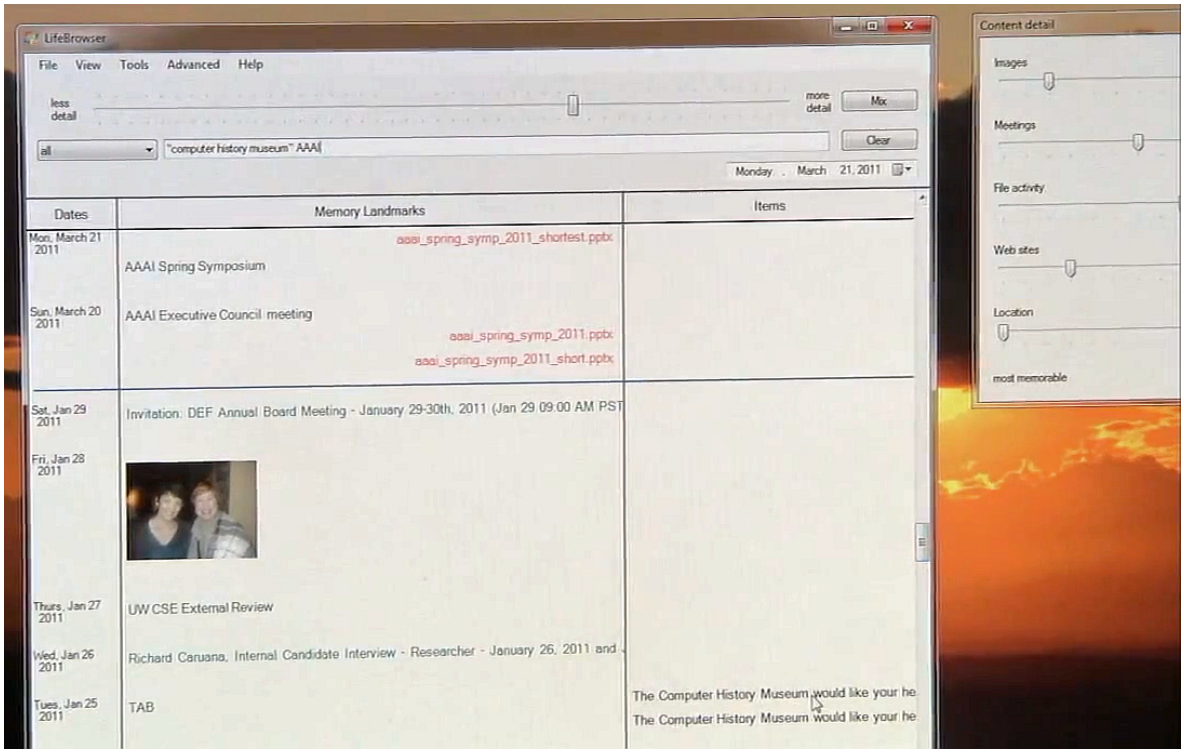

Figure 3.18: Life Browser (Microsoft Corporation, 2012, video position 01:20)

On the screenshot we see two windows: the larger main frame on the right and a smaller options panel, labeled *Content detail*, on the left. The main part contains a timeline consisting of three columns: *dates*, *memory landmarks* containing photos or images as well as text, and information *items* in the rightmost column. Clicking an information item directly opens it

[15] stream graph: a type of stacked area graph which is displaced around a central axis, resulting in a flowing, organic shape (Wikipedia Encyclopedia)

in the appropriate application. In the upper part of the main frame there is a slider that enables the user to view *less* or *more detail*. Although not explicitly presented in the video, the options panel on the left also contains various sliders labeled with *most memorable* and *all* at their respective ends. There are sliders for *images*, *meetings*, *file activity*, *web sites* and *location*. We assume that by using these setting filters like "the *most memorable meetings* in *all* possible *locations*" can be set.

Memory Lens In an interview (Webster, 2012), Horvitz confirmed that *Life Browser* originates from another project called *Memory Lens* published in 2004 (Horvitz et al., 2004). Looking at its user interface (see Figure 3.19) reveals the obvious similarity.

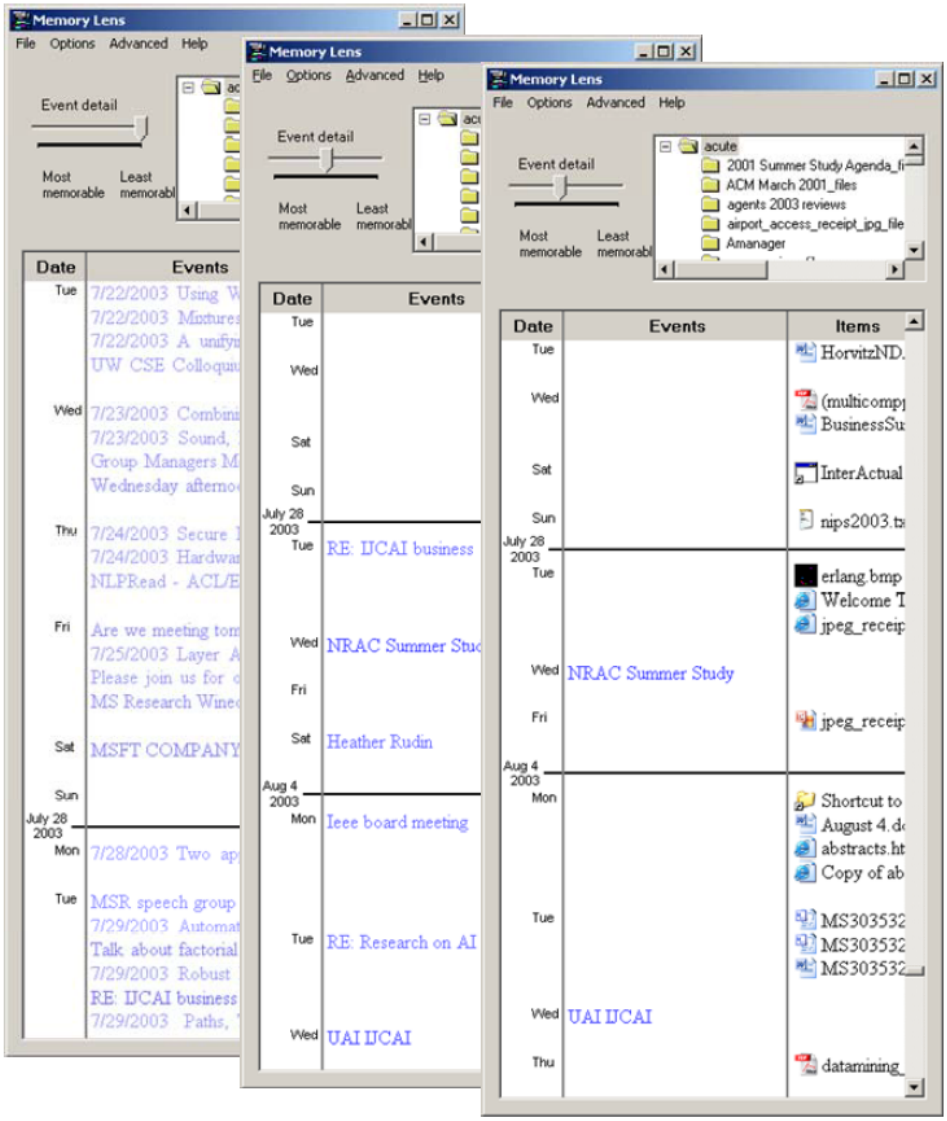

Figure 3.19: Memory Lens (Horvitz et al., 2004, p. 5)

The screenshots show three situations in which the *event details* slider (most likely comparable to the general details slider in *Life Browser*) is set to different positions between *most* and *least memorable*, which changes "the threshold on the likelihood required for an event to be considered a memory landmark" (Horvitz et al., 2004, p. 5).

Memory Landmark Inference Horvitz et al. (2004, p. 1) "developed a calendar event crawler that works with the Microsoft Outlook messaging and appointment management system. The crawler analyzes a user's online calendar to create a case library of events and properties associated with each event. The calendar crawler extracts approximately 30 properties for each event", for example *subject*, *event duration*, *location*, *organizer*, etc. One of its subsystems accesses the Microsoft Active Directory Service to identify organizational relationships among the user, the organizer, and the invitees, noting for example, whether the organizer and attendees are *organizational peers*, *direct reports*, *managers*, or *managers of the user's manager*. Beyond this data, Horvitz et al. (2004, p. 2) "created several derived properties representing statistics about *atypical situations*, based on the intuition that *rare contexts might be more memorable than common ones*". They compute the "measure of rarity for *atypical organizers, attendees and locations* of events by considering the portion of all meetings over all events under consideration or for a fixed period of time (e.g. events over a year) in which the property under consideration has the same value it has in the event at hand". To compute the value of *location atypia* for events, they "first compute the number of times each location has appeared in a user's calendar over a fixed period. The system then discretizes the *location atypia* variable into a set of states, capturing a range of percentiles, and the *location atypia* variable for each event acquires a particular value based on the rarity of the location associated with that event. An analogous derivation is used for computing *organizer atypia* and *attendee atypia*. A meeting acquires the *organizer atypia* or *meeting atypia* value associated with the least frequent attendee or organizer of the meeting" (Horvitz et al., 2004, p. 2).

In a next step, Horvitz et al. created Bayesian network structures based on supervised training data to provide a probability that an event is a memory landmark. For details we kindly refer the reader to (Horvitz et al., 2004).

Figure 3.20 displays a Bayesian network "showing all of the variables and the dependencies among them. A sensitivity analysis demonstrated that key influencing variables in the model for discriminating whether an event is a memory landmark are the *subject*, *location string*, *meeting sender*, *meeting organizer*, *attendees*, and whether the meeting is *recurrent*." Besides, "*atypically long durations*, *non-recurrence of events*, a user *flagging a meeting as busy* or *out of office*, and *atypical locations* or *special locations* had significant influence on the inferred probability." They also found that "meetings marked as *recurrent* rarely served as memory landmarks" (Horvitz et al., 2004, pp. 2). Other influences on this probability mentioned by the authors are the likelihood of meeting attendance, acoustical energy during meetings, and preparatory or follow-up activity associated with appointments (Horvitz et al., 2004, pp. 6).

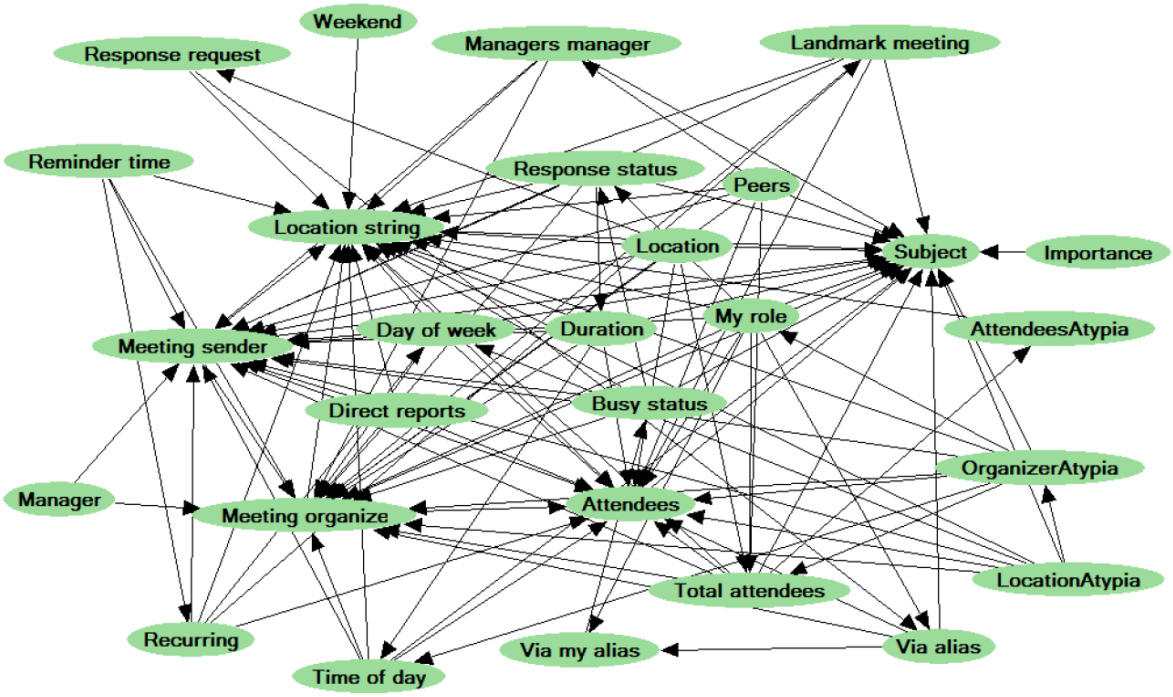

Figure 3.20: Bayesian network to infer memory landmarks (Horvitz et al., 2004, p. 3)

In our diary app we mainly adopt the idea of evaluating the rarity of certain properties.

3.3.7. Timeline Generation: Tracking Individuals on Twitter (2014)

Li and Cardie (2014, p. 1) discuss "the problem of reconstructing users' life history based on their Twitter stream and propose an unsupervised framework that creates a chronological list of *personal important events (PIEs)* of individuals".

We would like to explain these *PIEs* more thoroughly in the following. Like the term suggests, events considered a PIE, should be

- **important:** the event is referred to many times by an individual or his followers;
- **time-specific:** a unique event delineated by specific start and end points – rather than a general, recurring and regularly tweeted event over a long period of time;
- **personal:** an event of interest to himself or to his followers – rather than events of interest to the general public (Li and Cardie, 2014, p. 1).

In a next step, the authors characterize tweets into one of four categories: *public time-specific*, *public time-general*, *personal time-specific* and *personal time-general* (Li and Cardie, 2014, p. 2). Last, based on these categories the authors name criteria whether events can be considered PIEs depending on the person of interest being a *ordinary Twitter user* or a *celebrity*. Please see the paper for more information about the different models proposed by the authors. Concerning our diary app we just wanted to present an alternative idea of finding memory landmarks, in this case in the context of Twitter streams.

3.3.8. Other Timeline Applications

So far we only presented those timeline apps (highly) related to our work. Nevertheless, there are several other timeline applications available. For example, some of the "global players" were also concerned with timelines during recent years.

Google Timeline Google added a timeline tool for historical searches to their search engine in 2007. By clicking on the *Timeline* option, the search engine would break down the number of results for a search by year in bar graph format (Goodwin, 2011). An example is depicted in Figure 3.21).

Figure 3.21: Google Timeline (Goodwin, 2011)

However, the feature has been discontinued in 2011 (Goodwin, 2011). Since it "quietly vanished" (discontinuance only confirmed by a Google employee in a web search help post), only speculations about the reasons can be made. Goodwin (2011), for example, assumes that "not enough Google users made use of it".

Google+ Stories and Movies In May 2014, Google introduced two new features to its social media platform *Google+* called *Stories* and *Movies* (Sabharwal, 2014).

Google+ Stories can automatically generate interactive photo stories from photographs uploaded by the user. The system sorts photos according to time and location, for example by analyzing the meta data (e.g. date or geodata) or the contents itself (image recognition to find popular places). The resulting stories may be shared with others but cannot be changed

by the user (Donath, 2014).

Google+ Movies "can produce a highlight reel of your photos and videos automatically – including effects, transitions and a soundtrack (Sabharwal, 2014).

Facebook Timeline Facebook introduced a timeline feature in 2011 enabling user to "create profiles with photos and images, lists of personal interests, contact information, memorable life events, and other personal information, such as employment status." (Wikipedia Encyclopedia and Facebook, Inc. (2011)).

Bing Timeline Richard Qian, a member of Microsoft's Bing Index and Knowledge Team, announced a timeline feature for their *Bing* search engine in his blog post on February 21st, 2014 (Qian, 2014).

There is another timeline project by Microsoft called *Project Greenwich*, which is the last timeline application we would like to present.

Project Greenwich *Project Greenwich* is "a website that allows people to create timelines of any subject", for example "by uploading photos to the site as well as drawing on other sources from across the web". Timelines can then be shared with others. There is also a possibility "to compare two different timelines in order to add new context to each" (Microsoft Corporation, 2014). Once the timeline has been created it can, for example, be embedded in a blog.

Figure 3.22 shows an exemplary screenshot of the tool.

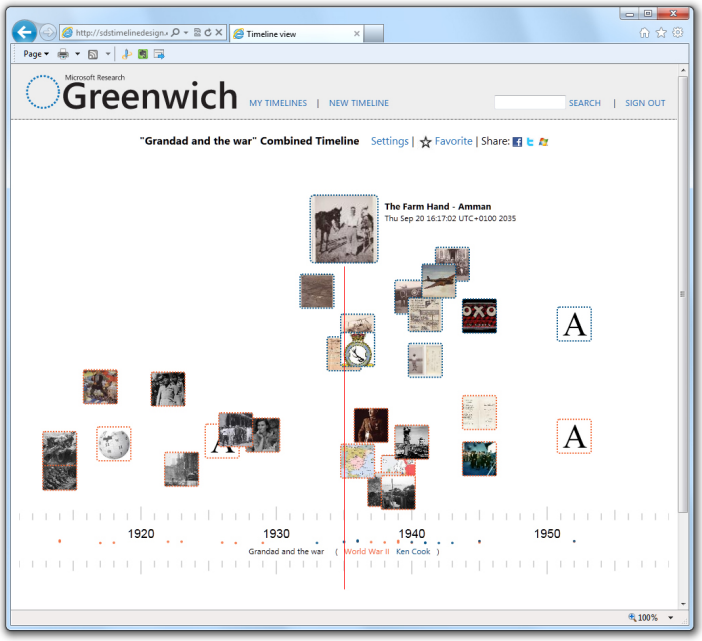

Figure 3.22: Project Greenwich (Microsoft Corporation, 2014)

Project Greenwich is "a research project by the Socio-Digital Systems team in Microsoft Research, and not an official Microsoft product." They want to explore "how people think about time, how they go about the process of telling a story through time, and what it means to reflect on chronological content to think about the past." Additionally, they "are interested in how the act of sitting down and manually crafting a timeline encourages reflection, learning and provides insights into relationships between the different elements within it" (Microsoft Corporation, 2014).

Completing this chapter, we will sum up all related works in the next subsection.

3.4. Conclusion

DFKI Projects: Concerning the different DFKI projects presented in this and the previous chapter, we can sum up as follows:

- Our app will be based on the *Semantic Desktop*, which provides the information items (source material) necessary for diary generation as well as the semantic interrelations between them.
- We utilize *SEED* in order to get *notes* for our diary app.
- The *life situations* defined in *PIMORE* can help us in creating abstractions (in addition to other means).
- *PIMO Timeline* – as a first approach to incorporate a timeline into the DFKI's Semantic Desktop prototype – succeeded in giving us an impression of how many items a PIMO might include for given periods of time.
- Conceptually, the *ForgetIT* application scenario of a diary provides us with a greater context, in which we can formulate our usage scenarios, for example.

Diary-related Works: Summarizing our previous findings we can conclude that only a few approaches of creating applications that generate diaries have been made. In two cases mentioned before (*ComicDiary* and *AniDiary*), the focus of presentation was on showing cartoon images having only small amounts of text. In our opinion, this is far too unspecific to capture the large bandwidth of experiences and events of a person's life. Large amounts of available data like documents, photos, etc. remain unused. In addition, it is doubtable whether cartoons actually help in remembering moments of one's life when years or decades have passed – besides, real photos might be preferred.

Smart Diary was the only application we could find that actually generates diaries having text entries only. On the one hand their *narrative structured sentence model* is an interesting approach, since it produces human-readable sentences. But on the other hand, this model

still seems to be too immature to actually deliver a diary which is exciting (or fun) to read. Nevertheless, we encourage the reader to decide on this for himself. Please have a look at the example depicted in Figure 3.8, again. The diary entry reads as (we replaced the location labels with "..."): *"Oh, it's a cloudy day. I was used to walk along with ... After study, I drove to ... to get some food. I used to walk along with... I was sitting in... for 35 minutes. I walked to ... for class. I was sitting in ... for 15 minutes. I drove through ..."*. Besides the rather boring repetition of similar phrases, finding out what actually happened in a large period of time, for example last year, will surely be very hard on the basis of these rather "low-level" activities. Like mentioned earlier, a mass of individual information items has to be scanned and mentally connected or summarized by the user. Higher level abstractions (beyond connecting sensor values with other logs) or summarizations of a set of events are not created, although this would increase the overview of the particular time period as well as clarity. These abstractions could, for example, be made by inferring semantic interrelations between the information items. These findings are by the way also true for *ComicDiary* and *AniDiary* as well as all commercial tools we investigated during our studies (e.g. the *Smart Diary Suite*).

In contrast to some of the timeline applications like *SIS* or *Life Browser*, the presented diary apps do not attach the individual information items (documents, photos, etc.) to the memory landmarks represented by diary entries. When reading about an event (e.g. a wedding), users will probably want to see that there are photos associated with it – not to speak of photos automatically displayed next to the generated entry.

Timeline-related Works: Open-source (or freely available) tools like *SIMILE* or *Continuum* could (partly) be incorporated into our app to realize some of our use cases (see Section 4.4) or further extend its functionality. Besides, we can adopt some of the visualization principles presented in this chapter, for example:

- details-on-demand, outlining, zooming and summarization (*LifeLines*)
- a "zoomable" and/or scalable histogram, always showing the whole dataset (*SIS* and *Continuum*, respectively)
- representation of hierarchical data (*Continuum*)
- clicking an information item opens it in the appropriate application (*SIS*, *Life Browser*)

Memory Landmarks: *Life Browser* and *AniDiary* present sophisticated methods and models to infer memory landmarks, which can (partly) be adopted by us, or at least they provide hints on how this problem can be tackled and solved. These aspects are supplemented by additional ideas like:

- *TimeMarks* to explicitly set memory landmarks (*YouPivot*)
- division into public and personal landmarks (*SIS*)
- distinction between "ordinary" and celebrity users (*"Twitter Timeline"* approach)

In general, by basing our diary application on the Semantic Desktop and personal information models we have the advantage of being provided with (partly very detailed) semantic information already available on the system. This is an advantage all other mentioned applications did not have. They had to evaluate several data sources like sensor data or usage logs in order to derive certain knowledge. It is up to us to optimally exploit this advantage.

Having introduced all fundamentals, we will next present our diary application's concept (Chapter 4), followed by the documentation of its system architecture (Chapter 5) and implementation details (Chapter 6).

4. Concept

To elicitate the requirements we basically followed the approach of *Task and Object-oriented Requirements Engineering (TORE)*. Since we lack the time to provide a fully detailed requirements documentation, we confine ourselves to only document some of the framework's core aspects in order to present our concept (more details about *TORE* are given in Appendix D).

First, we will introduce the stakeholders of our application and especially their goals.

4.1. Stakeholders and their Goals

Stakeholders Like mentioned in Section 3.1.4, this thesis is related to the *ForgetIT* project. We therefore adopt and extend their **personas**, which are described as follows:

> *"**Peter Stainer** likes traveling, taking photos, and his hobby is whisky. He works at a consulting company. The family has a computer and a tablet. Peter manages the family's files on the computer and has a separate hard disk for backups. Peter is married to **Jane**. Jane likes music, theatre, and comedy. Peter and Jane have smartphones. Jane has a grandmother in her eighties. Jane is in her late thirties, Peter is in his mid thirties. They have two children, **Sandra** and **Tim**, both teenagers."*
>
> (Maus et al., 2013a, Ch. 2.4, pp. 22)

Both, Peter and Jane, use the Semantic Desktop (or the PIMO, respectively) in various ways in their daily lives.

Goals They are looking for an easy way to retrospect on their lives. On the one hand, this retrospection should be fed with various types of media like photos, notes, calendar events, etc. But on the other hand, collecting and maintaining this data should not be too time consuming. (Please note that these goals are derived from our survey about social media usage and personal reminiscence – please see Section 1.1 and Appendix A).

The individual tasks Peter and Jane perform today (or would like to perform in the future) are discussed in the following.

4.2. Tasks and Usage Scenarios

In this section, we will describe several tasks which can also be seen as usage scenarios of our diary application. How these tasks are currently performed (if at all) and how they will look like in the future using our new app, is discussed afterwards in Section 4.3.

Scenario 1a: Keeping a "digital diary" (i.e. highly active in collecting and contributing data for diary generation) Jane is a very communicative person. She kept a diary when she was a teenager. Nowadays, she uses the PIMO as a kind of "digital diary" to document her personal life and that of her family. In particular, this means she tries to capture special moments in her life by taking (and uploading) photos and adding comments to them. She also writes down lots of thoughts and ideas (although she will probably not be able to realize all of them) and archives all kinds of texts, like articles from a magazine, web pages she visited or meeting reports from her job. In some cases, she also attaches files to them. Instead of sticking notes to the casing of her monitor she writes them down in her PIMO.

Scenario 1b: Not explicitly keeping a "digital diary" but interested in having a generated one (i.e. less active in collecting and contributing data for diary generation) In contrast to Jane, Peter is a less communicative person. Being a consultant, which is one of the classic knowledge working jobs (see Section 2.5), he primarily uses the PIMO for his job. His calendar events, emails, documents and some visited web pages are linked with (or stored in) his PIMO. He also saves texts like meeting reports, todos, ideas or some thoughts about a project. From time to time, he also uploads a (commented) photo of a memorable event. In summary, he basically uses the PIMO the same way as Jane does, but not nearly as extensive and less focused on private life.

Nevertheless, both, Peter and Jane, like the idea of having their data rehashed in the form of a diary on demand, but differ in the extent of (explicitly) contributing data for it.

Scenario 2: Reminiscing or review From time to time, they would like to browse their diary in order to remember various periods and events of their lives. In Peter's case this also has a professional background: it is sometimes very useful to have an overview which projects took place in which periods of time. This might answer questions like *"were some projects temporally interrelated?* or *"which projects are still running?"*.

Scenario 3: Embed own diary in historical context or in other peoples' contexts Placing your own history depicted by your PIMO-based diary in another context might reveal very interesting and surprising facts. For example, Jane might ask herself: *"what have I been doing during the Ukrainian crisis?"*, *"what was my friend Alice doing during that time?"*, *"was my mother's career proceeding similar to mine?"* or *"what was my father doing while he was at my age?"*.

Peter – in a more business-centric scenario – has the impression that more and more of his work is related to social media and cloud computing since their greatest competitor introduced a new cloud service last year and thus, his customers increasingly ask for similar services now,

too. Placing the mentioned event as a cesura in his diary and analyzing the topics before and after it might confirm (or disconfirm) his impression.

Scenario 4: Utilizing a person's episodic memory to start a search process Peter is often very busy. Every day lots of emails, documents, etc. arrive at his desk. He has to write a report here, sign a document there, archive an invoice in this place, store an email in that place. Being flooded with so much data, he is often not able to find a certain document right away. The situation gets worse, if the "missing" document has not been term indexed for search, yet. Let us think of an invoice Peter once captured using the camera of his cell phone on a business trip to France. The document is called *invoice_127582.jpg* and is stored somewhere in Peter's PIMO, but he cannot remember the filename (or path), nor the exact time period. He only remembers that he created the document on the formerly mentioned business trip and that he argued about it with his colleague John on a meeting a few months later. Peter can especially remember this meeting, since it was the last one before his colleague and friend John left for a new job in Australia. In this case, Peter could narrow down the relevant time period for a search process using the entries in his diary. He could browse his diary for trips to France and search in these time periods. If Peter has not been to France very often, this might already be enough to retrieve the document. If Peter often visits France, but remembers that it was on his last trip (or on one of the last few trips, or a trip in the 1990s, etc.) he can set a lower time limit for the search. Another possibility is to set the event of John leaving for his new job as an upper bound for the time interval of the search process. In all these cases, the search space will be reduced, possibly leading to a much faster retrieval of the document.
Please note that especially using additional information originating from the user's PIMO significantly helps in the search process described above. As stated before, Peter does not remember any concrete dates, so feeding a classic (i.e. non-semantic) search engine with temporal meta data about the missing invoice file is not possible. What Peter does remember are events (being to France or John leaving to Australia), which can be searched for and found in the diary. These events' (temporal) meta data can then be used, also by a classic search engine, to narrow down the time interval of a search. Still, the mentioned missing invoice file and John's leaving to Australia are two unrelated things except for that single connection in Peter's mental model, depicted by his PIMO. It is this semantic interrelation which cannot be utilized by classic search engines.

Scenario 5: Sharing diary with others Jane's grandpa died before she was born. Imagine, her grandfather would have had the possibility to keep a PIMO-based diary during his lifetime and also would have shared this digital memory with his family and friends before his death. For Jane it would be very exciting to browse her grandpa's diary generated from his PIMO, since she only knows him from stories told by her grandmother or old photos shown to her.

This usage scenario can also be transferred to a company. A new co-worker can much faster get in contact with his future colleagues if they share some parts of their (mostly job-related) diary with him. After some time of browsing through these diaries, especially on a higher abstraction level, revealing projects and topics his new colleagues were in touch with during the last years, he is probably able to get a quite good impression of who already gained much (or at least some) experience in certain topics. Future problems can then be tackled and possibly solved much easier, since reference persons (and/or experts) are already identified.

Having introduced all of Peter's and Jane's relevant tasks, we will next focus on how they are performed now and in the future using our diary app, respectively.

4.3. Activities and System Responsibilities

Before having access to our diary application, Peter's and Jane's tasks consisted of different *as-is activities*, which are subject of being later replaced by the corresponding *to-be activities* evoked by the availability of our app.

Scenarios 1a and 1b Usage scenarios 1a and 1b have in common that they are about **collecting and contributing data for a diary**. For these scenarios the *as-is-* and *to-be activities* are mostly identical. What possibly changes is peoples' eagerness in contributing data, since the (automatically) generated diaries are more rich and accurate having a broader data basis to work with. Since both, Peter and Jane, use the Semantic Desktop, their PIMOs might already contain lots of data that is thus also available for diary generation.

Scenario 2 *As-is activity:* In order to **retrospect on their past**, all media like photos, notes, documents, etc. have to be accessed manually. Although the Semantic Desktop helps in storing and finding these information items, especially by exploiting semantic interconnections between them, it is still up to the user to sort and summarize, or abstract from the data in order to get an actual overview of a certain period of time.

To-be activity: In the future, the task of summarizing and abstracting from plenty of individual information items in favor of short and concise descriptions like projects names, life situations etc. should be performed by the system. (If desired, these abstractions can easily be resolved by the user.) Except for providing the data once (i.e. by implicitly or explicitly using the PIMO), the additional effort needed by the user should be close to zero.

Scenarios 3 to 5 All other usage scenarios are similar to the second one. The *to-be activities* differ from the *as-is activities* by automating or supporting a functionality that was previously performed manually.

System Responsibilities Concerning system responsibilities (i.e. the decisions whether *to-be activities* are performed only by the system (*automated*), only by a human (*manual*) or by a human interacting with the system (*system-supported*)), we note that most of our *to-be activities* are *system-supported*. As a completely *automated* activity we could name a kind of "diary crawler", which pre-generates diaries in order to reduce the response time when a user actually invokes a diary generation – we will address this matter as well as the crawler more thoroughly in Section 4.6.2.

From the *system-supported to-be activities*, that mostly correspond to the previously mentioned usage scenarios, we can derive use cases, which is done in the next section.

4.4. Interactions and Use Cases

Like stated in the last section, the *system-supported to-be activities* focus on the interactions between the system and the user. In a next step towards the actual implementation, these interactions are refined to use cases.

Use case 0: Enter notes Users should be able to enter notes in order to actively and directly contribute and shape their diary – apart from implicit manipulations induced by using their PIMO. (We denoted this use case with zero, since this functionality is not provided by us. Instead, we use *SEED* as a service, please see Section 3.1.1.)

Use case 1: Generate diaries for given periods of time Initially starting the app results in a generated diary we refer to as the *standard diary* or *standard settings diary* (since no specific settings except for the period of time have been set). By default the system may view the standard diary of the current day or week, for example. The user then has the possibility to browse different periods of time, e.g. by clicking *earlier* or *later*. Each click initiates a re-generation of the diary covering the newly chosen period, e.g. the next week or month (depending on the chosen time granularity, see use cases 6 and 7).

There is also the possibility to manually influence the diary generation process. This is motivated by two reasons. First, the user may want to shift the diary's emphases, which can be initiated by explicitly including or excluding certain concepts (please see use cases 2 and 3). The other reason is that, although the condensation and abstraction process is performed on best effort, the result might still be unsatisfying to the user. Due to sparsely or wrongly annotated resources, for example, the importance of events as well as their potential for condensation might be misestimated. Thus, the user may readjust the diary by giving higher or lower priorities to selected concepts. Another possibility to influence the generation process is by adjusting several detail- or expert settings (see use case 10).

Use case 2: Update diary while explicitly excluding selected concepts Like stated before, there are several reasons to exclude certain concepts from the currently viewed diary. Excluded concepts are associated with a lower priority which leads to reduced mentions or even total absence in the dynamically re-generated diary.

Use case 3: Update diary while explicitly including selected concepts In analogy to use case 2, the user may also update the current diary by explicitly including selected concepts. This also triggers a dynamic re-generation process. Due to our special requirement of diversity (see Section 4.6.1) it may happen, that some concepts are not explicitly mentioned in the diary's entries, because they would appear too often. Since they should then appear in the *concept context* – an overview of the things that were rather important or prominent in the currently viewed period of time – the user may force their integration into the diary (ignoring the previously mentioned and later deepened requirement of diversity). Explicitly included concepts are associated with a higher priority and thus, their mentions in the diary are at a maximum.

Use case 4: Zoom out of a diary interval We already mentioned in Chapter 3 that many diary and timeline applications have problems concerning overview and clarity. Our diary app should enable the user to actually get an *overview* of his past. When retrospecting on several months or years, users should not need to browse a mass of individual information items like documents, notes, calendar events, etc., in order to comprehend what actually happened during the selected period. As a consequence, our application should condense and summarize the individual items and build abstractions for them. For example, instead of showing every individual meeting, presentation or note within a project, the system simply displays the project's name as well as the respective time and a short summary of the condensed items. In addition, characterizing information like photos and thumbnails or links to important documents could be attached to the diary entries. This condensation can be triggered by *zooming out* of the currently selected diary interval, e.g. switching from weeks to months.

Use case 5: Zoom into a diary interval The previously mentioned condensations or abstractions can easily be resolved by users *zooming (back) in* a diary interval. Clicking on a specific diary entry zooms into the time period covered by this entry. A second, less specific possibility is to press a general zoom-in button which simply increases the time granularity, e.g. switching from months to weeks. Instead of using the time period of an individual entry for zooming in, the overall time period of the currently shown diary is used. *Zooming in* is possible as long as the actual information items, i.e. the basic material like documents, photos, etc., are not reached. In our app, we decided that the highest time granularity possible is the level of *days*. Clicking on the diary entry of a day is the topic of the next use case.

Use case 6: Jump from diary entries to actual contents Like stated before, on the time granularity level of *days* the diary entries are not rehashed and thus directly show the individual information items. Clicking on any of these entries directly opens the respective item. In this way the diary also enables direct browsing of information items.

Use case 7: Use diary entries to set time interval of a search Like described in usage scenario 4, the time periods of the currently viewed diary or any selected individual entry can be used to narrow down the time span of a search process.

Use case 8: Embed current diary in other context Usage scenario 3 was about the possibility of embedding one's own diary into another context, e.g. the one of another person or a historic context. The system should offer some pre-defined themes or a function to load (historic) user data, e.g. the biography of a celebrity or chronicle events. These *historic entries* are then incorporated into the user's own diary enabling him to compare different timelines or seeing things in another (or greater) context.

Use case 9: Share diary with other users Sharing your diary with others or reading the diaries that others shared with you opens a wide range of new possibilities and features. Since this also implicates a lot of open issues which are out of this paper's scope (e.g. access rights, definition of user groups, etc.), we just want to mention this use case for the sake of completeness here.

Use case 10: Advanced or expert mode in diary generation The diary generation process can be influenced not only by providing the desired time period and a few other parameters like desired number of entries, etc. Additionally, advanced users or experts should have in-depth access to the main algorithms, for example to experiment with different weights or thresholds, etc.

Having discussed various interaction scenarios between the system and the user, we will next focus on showing how these use cases are actually presented to the user.

4.5. User Interface Structure and Data

Our diary application should have the look and feel of a modern web log. Similar to an example created with *Tumblr*, which is given in Figure 4.1, the diary entries should primarily consist of text summarizing the information items they represent. Every entry should also have a label (headline) and an associated time referring either to an event (i.e. a single point in time) or a time period. In addition, characterizing icons or thumbnails as well as links to further information can also be part of the entry. The entries may, for example, be ordered

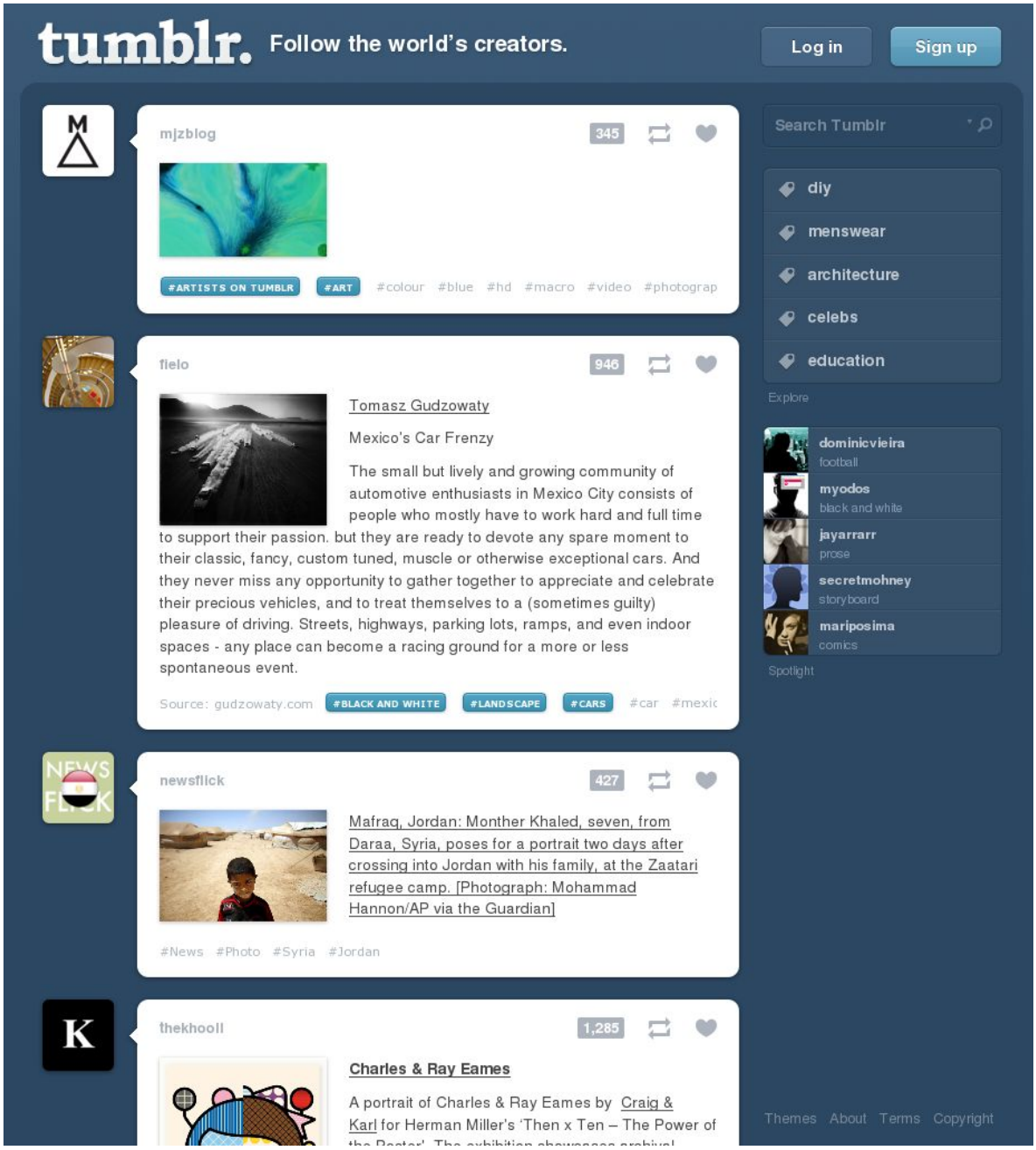

Figure 4.1: Tumblr (screenshot by Web Rater (2014))

chronologically, like in traditional diaries, or reverse-chronologically, like it is typically the case in blogs.

Representing all information items (such as documents, emails, photos, calendar events, etc.) in the form of a web log is a rather innovative idea, although it has the disadvantage of loosing an overview of temporal coherences *within* the individual entries. Let us consider an example. If a project running for a longer period of time is represented by a single diary

entry, the user is not able to see from this entry whether there were "hot spots" having lots of individual events or whether there were periods without any progress and thus no events, for example. Even though the user may zoom into the particular time period represented by the entry (and thus find these "hot spots" himself), we would like to provide an overview right away. This is accomplished by an auxiliary view we call *Topic Lanes*. Like depicted in Figure 4.2, for every diary entry (which typically represents a certain topic, project, life situation, etc.) there is a corresponding lane in this view.

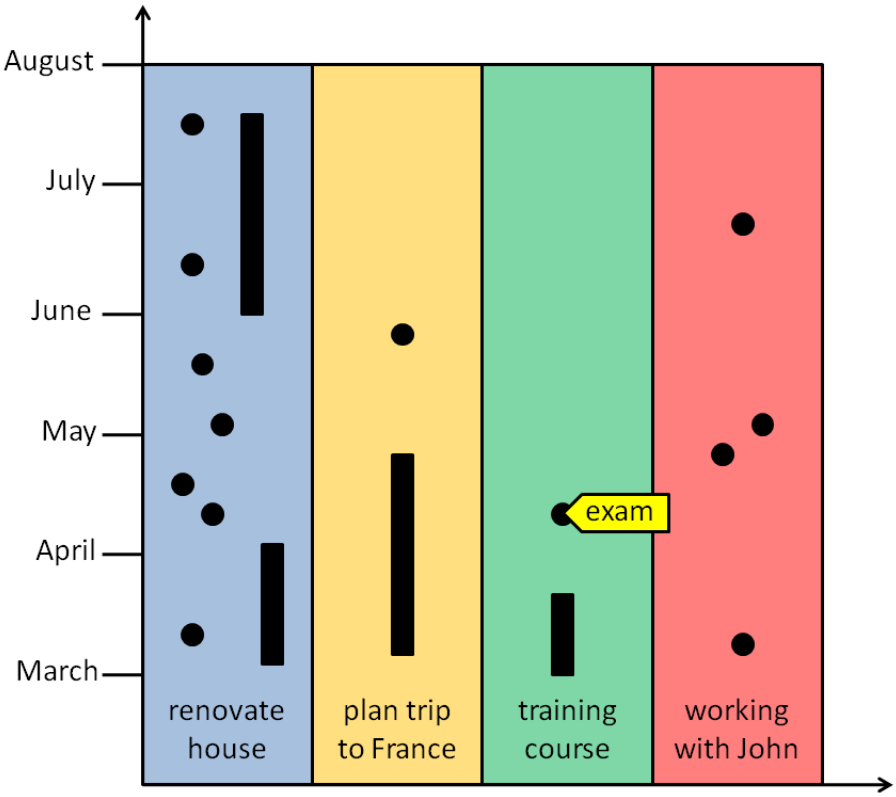

Figure 4.2: Topic lanes

Within each lane the individual information items belonging to the entry are shown in the usual style of a timeline. Single events may be represented by dots and time periods by bars, for example. Clicking on these items should pop up detailed information, e.g. label/headline, associated time, thumbnails, links, etc. In our example, the pop-up message next to the event in April on the *training course* lane reveals that it is the course's final exam. Please note, that this is only a UI mock-up and thus more advanced features like they were mentioned in Chapter 3, e.g. histograms or zoomed-up excerpts, are not incorporated here. By using this view, a more targeted zoom-in is possible, since potential periods of interest are easier found.

Representing all information items in the form of a blog-style dairy does not completely

reflect our idea. An additional important aspect is ensuring a high diversity within the diary. This special requirement as well as others are the topic of the next section.

4.6. Special Requirements

There are some functional and non-functional requirements we consider to be very critical to our application's success. We therefore address them more thoroughly in this section.

4.6.1. Diversity

In order to avoid repetitive and thus rather boring contents, like we found it in the applications discussed in Chapter 3.2, we want to ensure a high diversity within our generated diaries.

Imagine Peter writing a diary for the last three months of a year – in our example, these are the calendar weeks CW40 to CW52 of 2013. Peter's diary entry for CW43 could, for example, be composed of three individual notes, which together read as follows:

(CW43) Oct. 21st – Oct. 27th, 2013:
I attended a training course. [Note #1]
I worked with John Smith. [Note #2]
I renovated our garden shed. [Note #3]

Other diary entries (composed of several information items) can be found in Table 4.1. Please note that we printed the items/entries horizontally and also used rather simple and non-varying sentences (in structure) in order to better illustrate our idea.

time	information items / diary entry		
CW 40	I attended a training course.		
CW 41	I attended a training course.	I worked with John Smith.	
CW 42	I attended a training course.		
CW 43	I attended a training course.	I worked with John Smith.	I renovated our garden shed.
CW 44	I worked with John Smith.	I renovated our garden shed.	
CW 45	I attended a training course.	I worked with John Smith.	I renovated our garden shed.
CW 46	I worked with John Smith.	I renovated our garden shed.	I planned my trip to France.
CW 47	I worked with John Smith.	I renovated our garden shed	I planned my trip to France.
CW 48	I worked with John Smith.	I renovated our garden shed.	
CW 49	I worked with John Smith.	I renovated our garden shed.	
CW 50	I worked with John Smith.	I renovated our garden shed.	I planned my trip to France.
CW 51	I worked with John Smith.		
CW 52	I renovated our garden shed.	I planned my trip to France.	

Table 4.1: Diary of three months with low diversity

Analyzing Peter's diary, we see that in total four different topics are mentioned: a training course, working with John Smith, planning a trip to France and the renovation of his garden shed. We highlighted them using different colors. Instead of always using the same sentences for each topic, we also could have chosen different ones. The main point is that there are several information items – in this case notes – sharing the same topic in different parts (periods) of the current diary view. Using always the same sentences only makes this aspect more clear. The resulting diary, consisting of 28 notes and covering three months of Peter's life, would be rather boring to read, because similar things are mentioned over and over leading to *low diversity* within the diary.

To ensure higher diversity, our application should create clusters from items that share the same topic and represent them in a condensed way. The result for doing this for Peter's diary is shown in Table 4.2.

time	**diary entry**
CW 40-45	I attended a training course.
CW 41-51	I worked with John Smith.
CW 43-52	I renovated our garden shed.
CW 46-52	I planned my trip to France.

Table 4.2: Diary of three months with high diversity

The condensed diary now contains only four entries – one for each of the aforementioned topics. All of them differ from each other.

We also see the earlier discussed problem of loosing the temporal overview *within* an entry. The training course is listed to be from CW40 to CW45, although it actually did not take place in CW44, for example. Like stated before, the user therefore has the possibility to zoom into the individual entries if more specific information is desired.

Please note, that our real condensation and abstraction algorithm would have created summaries for the different items belonging to a diary entry. Since these items are all notes using the same sentences in this simple example, their summary is identical to their actual text, which is not the case in general.

Another important requirement is a reasonable response time, which is discussed in the following.

4.6.2. Reasonable Response Time

Processing possibly thousands of individual information items will be an ordinary task for our application, especially if a year's (or even a decade's) diary is generated. Our goal is to keep the time a user waits for his diary to be generated as small as possible. Since much data needs to be processed and each item may possibly be interconnected to others, distributing the data to several machines in order to parallelize the calculation process is a non-trivial task. But even if this succeeds, the calculation results of several hubs still need to be reassembled in the end. Thus, the resulting response time will eventually still be undesired.

Diary Crawler Another possibility to tackle this problem is introducing a *diary crawler* which we already mentioned in Section 4.3. Constantly running in the background, this crawler generates diaries for every user of the system and for each period of time in advance. In practice, this can only be realized if the number of possible time intervals is limited. For this reason we decided to only include pre-defined intervals in our app, e.g. days, weeks, months, quarters, half-years, years, etc. On a conceptual level this can also be justified. Users will probably rather ask what they were doing in typical time intervals like weeks or months (e.g. *"What have I been doing last week?"* or *"What have I been doing in 2011?"*), instead of asking for "irregular" intervals like *"What have I been from May 5th until September 2nd, 2009?"*. This is probably even more the case when comparing a user's own diary to those of others (*"My friend talked about May 2005 – what have I been doing during that month?"*). A drawback of this design decision is that zooming works less smoothly (due to possible realignments) and the diary of "irregular" intervals has to be embedded into a larger regular one. For example, if only the time granularities of days, weeks and months are available, generating a diary covering ten days of a month is only possible by either generating the diary for the whole month or creating several diaries for each week which can then be browsed in order to retrospect on the ten days in multiple stages.

Please note that this crawler would only produce the aforementioned *standard diaries* (due to the great number of possible settings). If a user, however, applies different settings, e.g. by explicitly including or excluding certain concepts, a new diary has to be generated. In order to avoid doing this completely from scratch, at least some intermediate results could be pre-calculated, e.g. text analyses.

After having introduced the concept of our app, we proceed in presenting the system's design in the next chapter.

5. System Design

We designed our diary tool to be a distributed client/server application. Its basic architecture as well as the design of its different components are presented in next sections.

5.1. System Architecture

Like stated before, our application implements a client-server model. In particular, our prototype's server part is a *JAVA servlet* and its client is embedded into the DFKI's so-called *PIMO5* app, which basically is a *HTML5* web client offering various tools like our diary or a notes app, for example.

Further details about the server as well as the client are given in the next section.

5.2. Component Design

Server Our server component is integrated into the DFKI's Semantic Desktop prototype. This prototype is currently implemented as "a cloud-based service and provides a service API based on JSON RPC". Its service API "defines a set of methods to access and manipulate the PIMO" (Maus et al., 2013b, p. 72), for example a *Query API*, a *User API* or a *Manipulation API*. Additional details can be found in (Schettler-Köhler, 2014, pp. 6), for example. "In contrast to typical semantic web approaches, the service API does not allow direct access of the core data. Instead, a designated set of methods guarantees a consistent and privacy-safe access to the PIMO." (Maus et al., 2013b, p. 72).

Figure 5.1 shows our server component's basic structure. We see that by integrating our diary app, the Semantic Desktop now additionally has a *Diary API*, which is used by the client to get the *diary data* (i.e. diary entries as well as meta data, URIs of linked resources, etc.).
The actual diary component (gray box within the server component on the right) is shown more close-up on the left-hand side of the figure. It uses the *Query API* as well as the *User API* (both currently – but not necessarily – running on the same server as the diary).

The main part of the diary component is the *PIMO Diary Impl* class. Besides providing the actual *Diary API* it generates the diary entries in collaboration with the *Entry Condensation Manager*. Depending on the *diary query options* (i.e. the parameters given to generate a diary) the generation process runs in slightly different ways. In order not to loose results or calculate several aspects twice, relevant intermediate results are stored in a *Basic Diary Data* object, which is passed between both classes. Later, the final diary entries are assembled using this temporary data. The *Entry Condensation Manager* utilizes the *Text Analyzer*, which itself uses the *Apache Lucene* software library (Apache Software Foundation, 2014) in order to analyze the labels and text bodies of the information items.

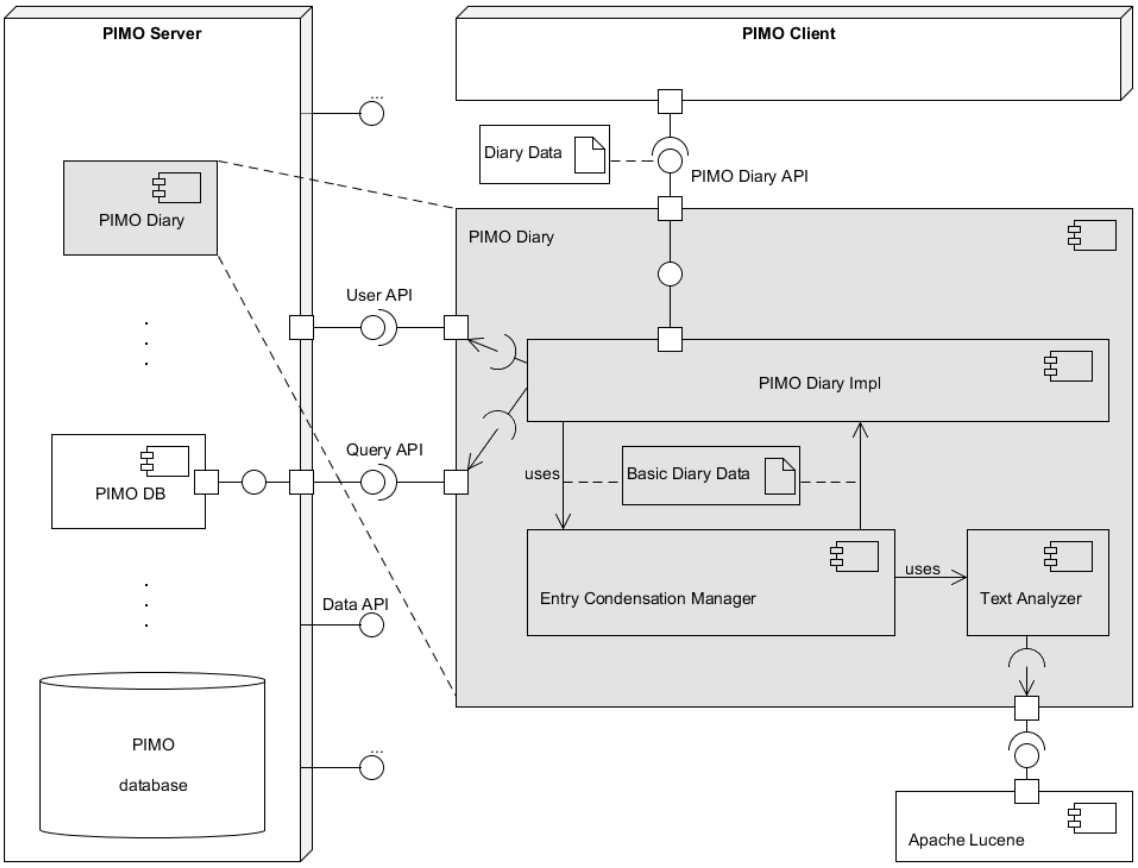

Figure 5.1: Server components (diary app highlighted in gray)

Client The basic structure of the DFKI's *PIMO5* client is depicted in Figure 5.2. Like mentioned before, this *HTML5* web client contains several individual apps (red box in the figure) of which one is our diary application (drawn in gray). Using an *App Selector* or the *Menu* (lower part of the figure), the user may choose which app should be loaded. After selecting an app, a *Router* component is responsible for actually loading and viewing the app. Since all apps use their own URL[16] hash ("#"), this component is also a listener for hash changes. (Manually entering a different hash address is like selecting an app.)

The apps themselves, especially our diary app – are designed according to the *Model-View-ViewModel (MVVM)* pattern, which is a special case of the well-known *Model-View-Controller (MVC)*[17] architectural pattern. In short, the *ViewModel* is similar to the original *Controller*, but it is less general by only serving an individual *View* (East, 2008). As a consequence, the same *ViewModel* instance cannot be used to serve different (concurring) *Views*. In the *PIMO5* app, the *ViewModels* and *Views* are conceptually forming the *presentation layer*. We

[16] URL: uniform resource locator – a web address
[17] for Details please see *http://en.wikipedia.org/wiki/Model-view-controller*

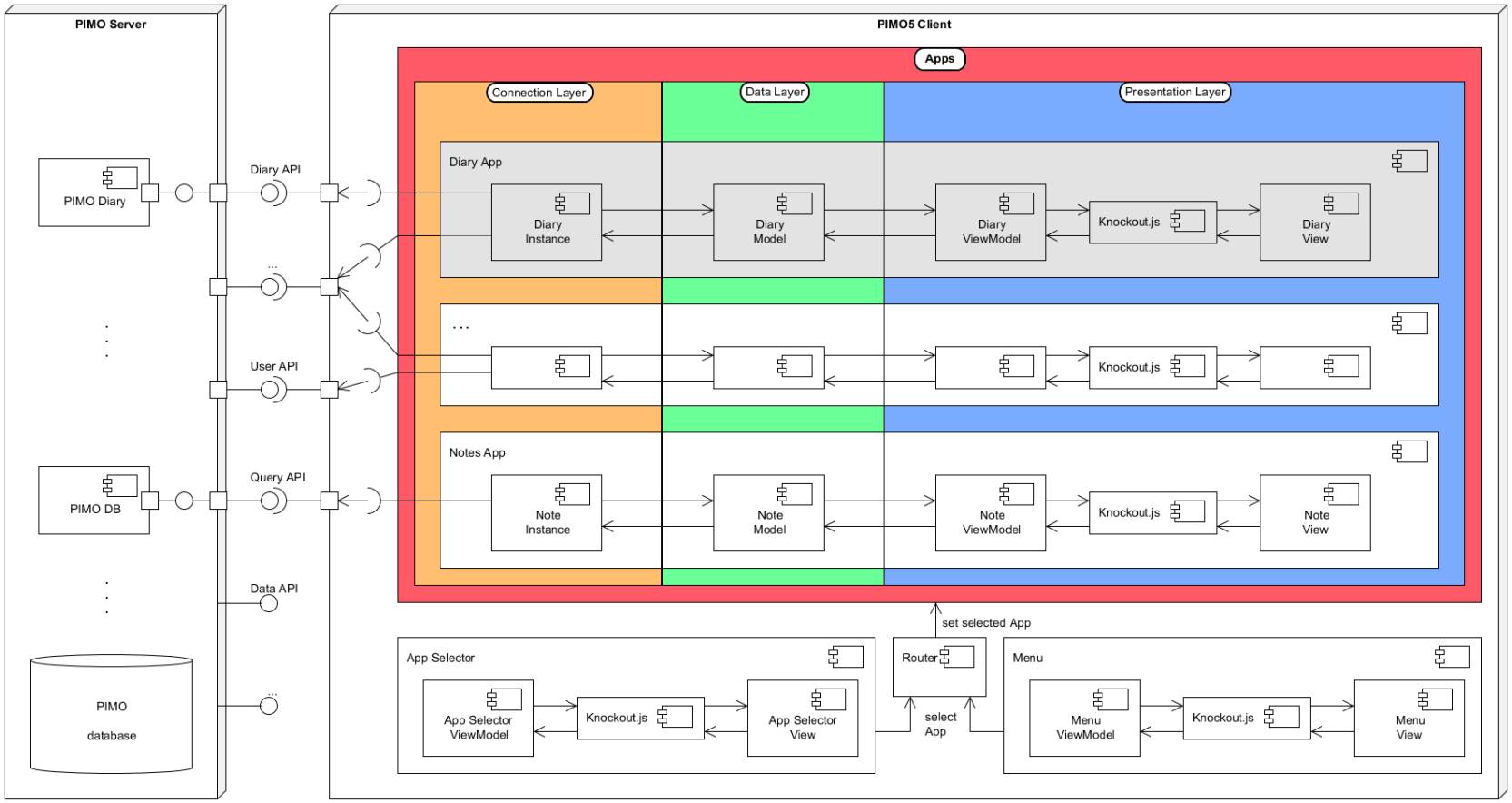

Figure 5.2: Client components (diary app highlighted in gray)

use the *Knockout.js* framework (Knockoutjs.com, 2014) in order to provide the data bindings between them.

The actual data is stored in the *Model* which belongs to the *data layer*. In order to reduce loading times and efficiently use bandwidth, the *Model* basically caches PIMO resources. If a resource cannot be found in the cache, a request is sent to the lower *connection layer*, which contains an *Instance* object for every app, e.g. a *Diary Instance* object for the diary app. These *Instances* encapsulate the connections to the respective apps' server counterparts. Thus, if the client is disconnected from the server, the apps should – at least to some extend – remain functional as long as resources can be found in the cache or no updates need to be sent to the server, etc.

From our concept discussed in Chapter 4 and the system design just presented we created a prototype of our diary app which is described in the next chapter.

6. Implementation

In this chapter we present a proof of concept implementation of our diary application. For several sub-problems arising from our basic idea *one* possible solution is presented. Thus, we decided not to include these detail solutions in Chapter 4, although they are partly discussed on a rather conceptual level.

This chapter consists of three parts. The two main parts are about the user interface (client side) and the actual diary generation, which mainly runs on the server side. In a third section we present an example, in which the author generated a diary from his PIMO for the time of this thesis.

6.1. User Interface

The user interface (or UI for short) of our diary application can be divided into six sections as depicted in Figure 6.1 and described more thoroughly in the following.

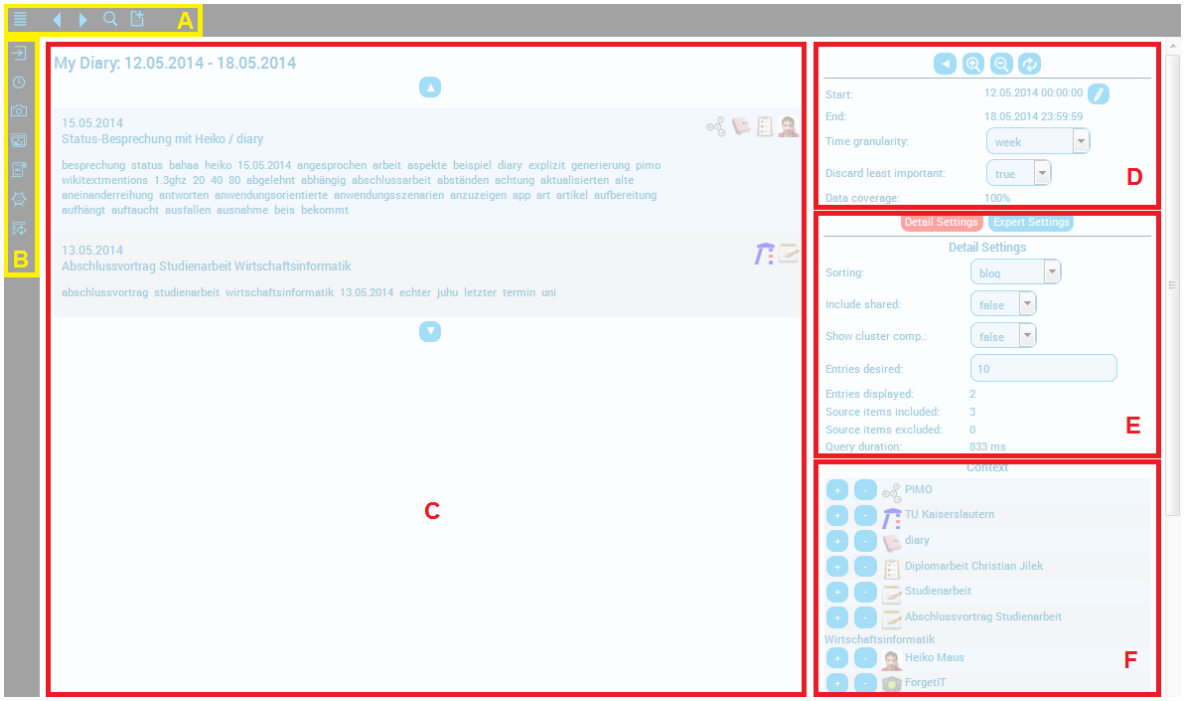

Figure 6.1: User interface sections: (A) menu, (B) app selection, (C) diary entries, (D) basic settings, (E) detail and expert settings, (F) (concept) context

6.1.1. Screen Structure

Like mentioned in Section 5.2, our tool is an HTML5 app integrated into the DFKI's *PIMO5* client. Thus, this client's general menu (A) and its app selection bar (B) frame our application. Since they are actually out of our responsibility, we highlighted them in a different color (yellow).

Our app is basically divided into two parts: a large main part on the left containing the diary entries (C) and a settings bar consisting of several sub-sections on the right (D to F). In this bar the user finds the basic settings (D), a detail and expert settings area that can be expanded on demand (E), and the *context* (or *concept context*), an overview of things that were important or prominent in the currently viewed time period (F). A larger and non-highlighted screenshot of the UI can be found later in this section (Figure 6.4).

Figure 6.2 shows the fully expanded settings bar except for the concept context (usually below the expert settings), which we left out for the sake of readability. We consecutively numbered all 39 settings (or information labels, respectively) in order to precisely address them later on. Since many of these settings, especially those belonging to the expert mode, concern the actual diary generation (Section 6.2), we will later refer to this figure several times.

In the following sections we will discuss the basic and detail settings, while the expert settings are deepened in Section 6.2. Furthermore, we will show how a typical diary entry and the concept context look like.

6.1.2. Basic Settings

Basic Buttons (1-4) The first row of the basic settings panel contains a button to go one step *back* (1) or forth (*next*) in the diary browsing history (*next* button not visible on the screenshot). In addition, there is a *refresh* button (4) as well as two buttons to *zoom into* (2) or *zoom out of* (3) a period of time (see UC[18]4 and UC5).

Time Interval (5-7) The *start* (5) and *end* (6) of the currently viewed time period are displayed right below these buttons. The start time of the diary to be generated can be set manually by clicking on the *set start time* button (7) and enter the desired date in the appearing window.

Time Granularity (8) Zooming in (UC5) and out (UC4) of the current time period is either possible by using the aforementioned buttons (2 and 3) or by setting the *time granularity* (8) manually. This allows skipping intermediate granularities, for example months, quarters and half-years when zooming out from weeks to years.

[18] UC: short for *use case*

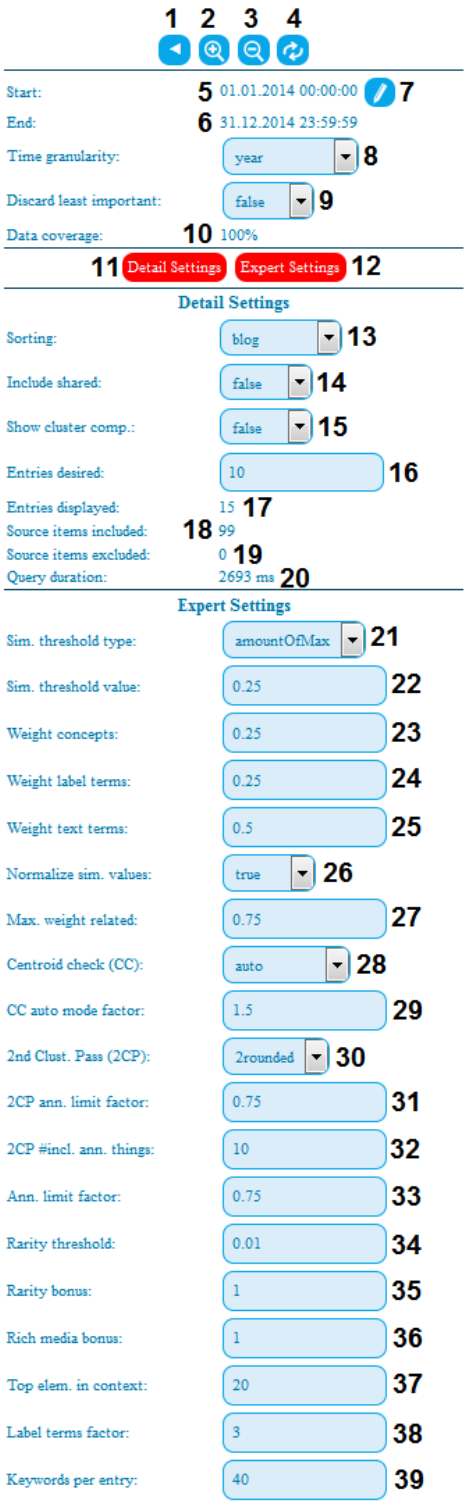

Figure 6.2: Settings bar

Discard least important Entries (9) Like previously stated and explained in Section 6.2, generating a diary (UC1) is an interplay of merging (clustering) and filtering (importance evaluation) information items. The user may indicate whether he wishes to see all entries generated in this process or whether the *least important ones should be discarded* (10). If this option is set to *true*, possibly not all information items belonging to the given period of time may be incorporated into the diary.

Data Coverage (10) To give the user an impression of how much of his data has been rehashed to generate the diary he is currently viewing, we created a status information called *data coverage* (10). A value of 10%, for example, means that 90% of a user's information items belonging to the selected time period are not incorporated into the diary currently displayed. A low data coverage may occur if the *number of desired diary entries* (16, see detail settings) is too low or if a period's information items are very heterogeneous and thus cannot be condensed very well. Low data coverage should not be equated with low quality of entries or the like. A period of time could, for example, also include many information items that are not very memorable, e.g. a note that a user has written years ago in order to remind himself to buy butter on his next trip to the mall. After being at the mall and buying the butter, this note is probably never mentioned (referenced) again and is thus evaluated to be not very important for that period's diary.

Show/Hide Advanced Settings (11-12) The advanced settings (UC10) are divided into *detail settings* and *expert settings*, which can be shown or hidden using the buttons 11 and 12. Both categories differ in their targeted user group. Whereas detail settings are still meant for standard users, the expert settings are – like the name suggests - for experts only. Detail settings extend the basic ones by some additional information labels and features which are yet less technical than those of the expert mode. Details are given in the next section.

6.1.3. Detail Settings

In this section we will describe the *detail settings*, which are part of UC10, more thoroughly.

Sort Diary Entries (13) Using the *sorting* option (13), the user may toggle whether the diary entries are ordered chronologically (*diary style*), reverse-chronologically (*blog style*) or according to their *importance*. The last case can especially be useful if a user is interested in a specific period of time without caring about the temporal order of entries (events) *within* this period. Consider the example of retrospecting on a specific month which is many years ago, e.g. May 2005. It is maybe neglectable for the user whether an event happened in the first or the third week of this month. Sorting the entries according to the importance evaluated by the system, the user is able to see the most important things upon first sight. The more he scrolls down, the less important are the entries he gets to see.

Include shared Data (14) Although we intend to generate *personal* diaries, we implemented the experimental option of also *including shared things* (14), i.e. the things of a GIMO instead of a PIMO (see Chapter 2.2). This feature has to be refined in later versions to fully realize UC9, especially after the currently rather simple data sharing model of the DFKI's Semantic Desktop prototype gets extended in the future. (Currently there is only the option to either have private or public things. The definition of user groups with specific access rights to certain things is not possible, yet.)

Show Entries' Composition (15) Enabling the option of *showing the composition of entries (clusters)* (15) reveals which information items were merged to form the different entries. Additionally, details about the importance evaluation are provided (please see Section 6.2.4 and especially Figure 6.11, since it depicts a situation in which this option is enabled). Originally, this was a debugging feature, but we decided to transform it into a standard one, due to the interesting insights it may provide.

Number of desired Entries (16) Setting the *number of desired entries* (16) gives a recommended value to system of how many diary entries the user approximately expects. It was intended to be a kind of "soft limit" that can be undershot or slightly exceeded by the system.

The current version of our diary app, however, treats this value as an upper bound that can by undershot by the system if plausible.

Number of displayed Entries (17) Label 17 shows how many *diary entries are currently displayed*.

Detailed Data Coverage (18-19) Label 18 shows the number of information items (source items) that were incorporated into the currently viewed diary, whereas label 19 shows those, that were not (due to being too unimportant). Dividing the number of included items by the sum of both values results in the data coverage displayed in label 10.

Query Duration (20) How long the generation of a dairy took on the server can be read off label 20. Intended as a debugging feature, we later decided to keep it as a standard one, since it also provides some interesting insights.

After having presented the basic structure of our app's user interface as well as several user settings, there is only one important UI aspect missing: the diary entries themselves.

6.1.4. Diary Entries

A typical diary entry is depicted in Figure 6.3. In its middle part there is a date or time period given (A), followed by a label/headline right below (B), and a text body (C) containing the summary of the information items this entry represents. If the time granularity of *days* is set (which means that condensation is turned off and thus there is a diary entry for every information item), the text bodies (if available) are directly adopted from the information items without summarization. In this case, clicking an entry would directly open the underlying item (UC6) instead of zooming in further. If an entry is associated with an image, it is displayed on the left side next to the middle part (E). Currently this is the case if the entry is either composed of an item that *is* an image or an item that *is associated with* an image. An entry's most important concept annotations are drawn as icons on the right-hand side (D).

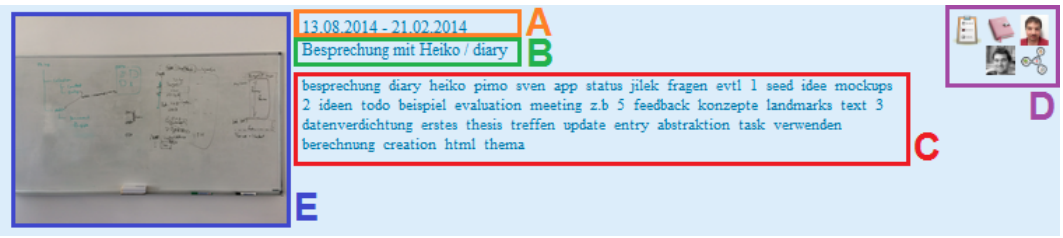

Figure 6.3: Diary entry layout: (A) date/time period, (B) label/headline, (C) text body, (D) concept annotations, (E) image

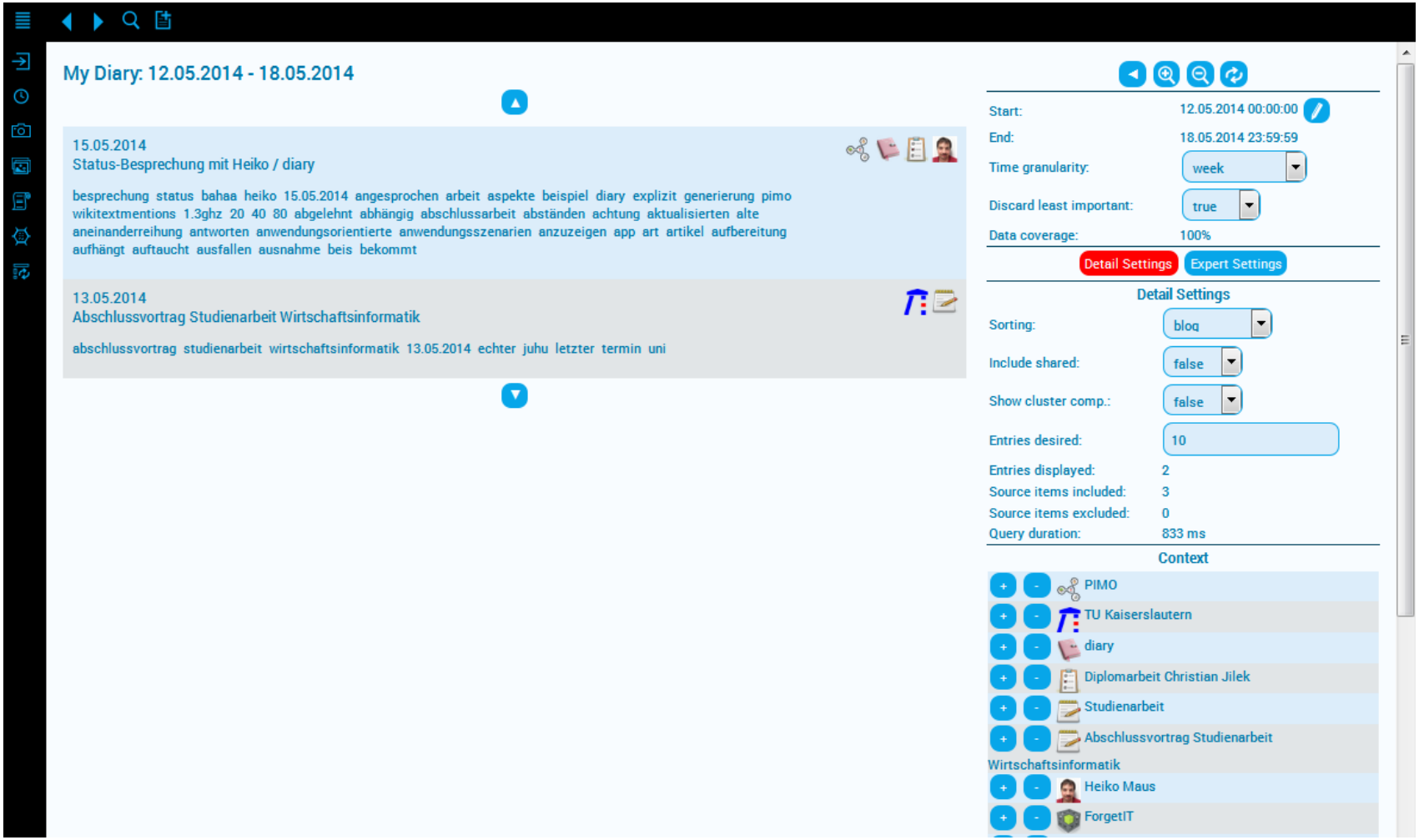

Figure 6.4: User interface (detail settings expanded, expert settings hidden)

6.1.5. Concept Context

Like mentioned before, the *concept context* is an overview of things that were important or prominent in the currently viewed time period. This overview is generated from *all* information items of the given period, whether they were sorted out in a possible importance evaluation or not. Since the concept context can be thought of as a top ten (or top twenty) ranking list of concepts, it is unlikely that rather unimportant topics make it into this list, though it may happen if a period's data is very heterogeneous. In the UI, each concept's name (label) as well as its image icon are displayed. More details about the concept context are provided in Section 6.2.5.

Its basic (left-hand side) and extended version (right-hand side) are depicted in Figure 6.5.

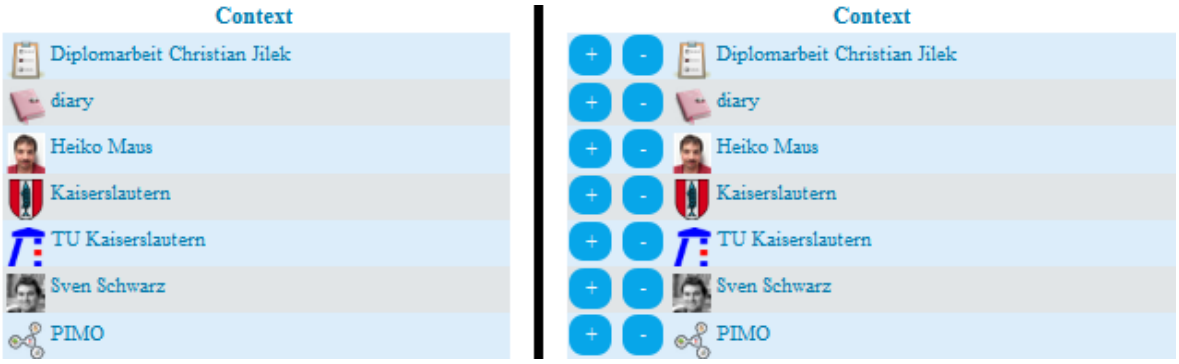

Figure 6.5: Concept context in basic (left) and extended version (right)

Enabling the detail settings (button 11 in Figure 6.2) also extends the context by adding two more buttons to each of its elements. By using these buttons the manual inclusion (+) or exclusion (–) of a concept can be triggered (see UC3 and UC2, respectively).

In the next section we will explain the core functionality of our app, which is the actual diary generation from a user's PIMO.

6.2. Diary Generation

The main part of our application is the actual generation of diaries from the users' personal information models. In principle, the client requests a diary by sending a set of parameters (*diary query options*), e.g. period of time, number of desired entries, etc., to the server, who generates it accordingly. The core of this process is the *getEntries()* method, which is also the main method of the Semantic Desktop's *Diary API* (see Section 5.2 and especially Figure 5.1). Its basic outline is depicted in Figure 6.6.

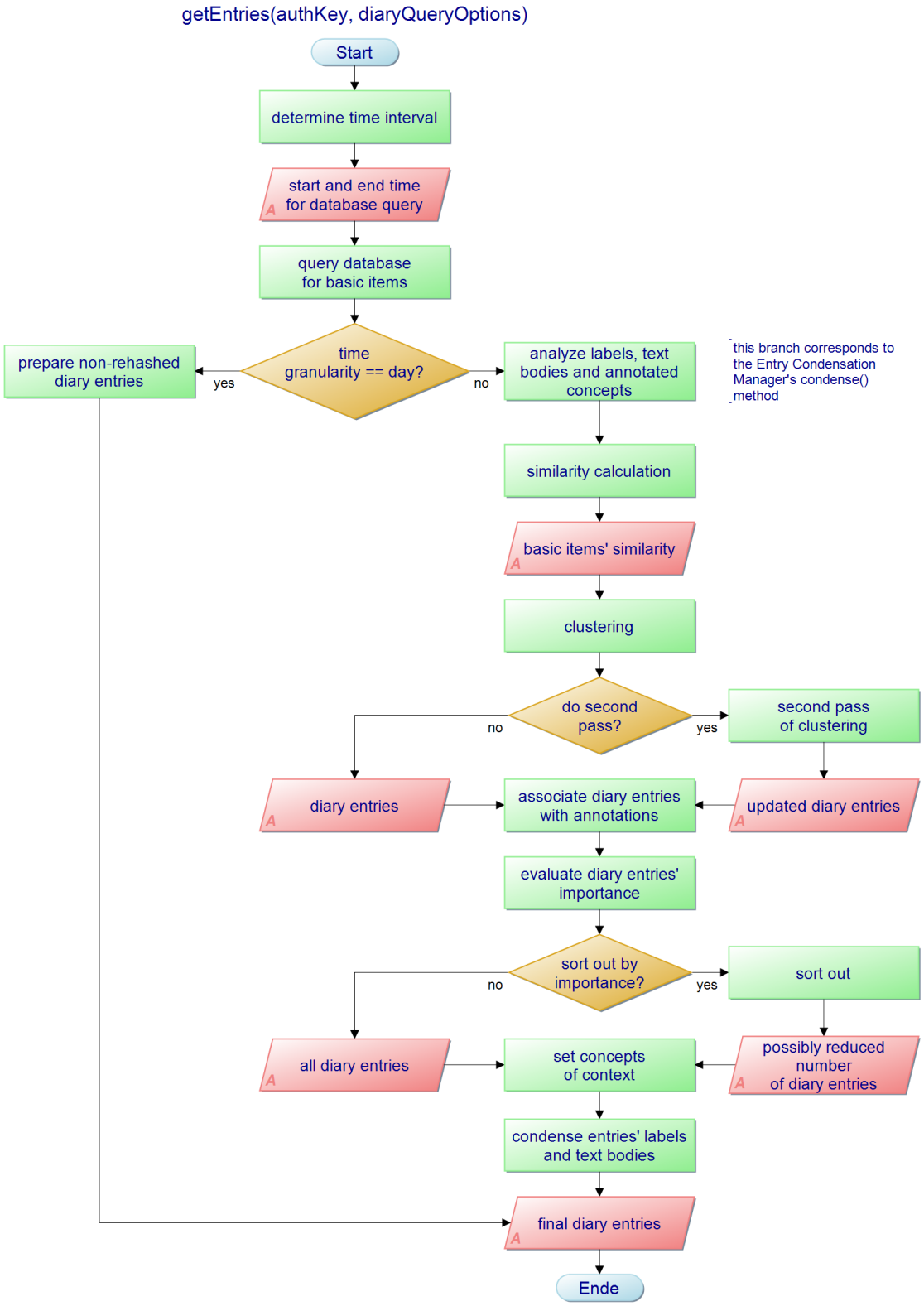

Figure 6.6: Diary generation

Since we decided to only offer pre-defined time intervals (see our special requirement of a reasonable response time, Section 4.6.2), the first step of the *getEntries()* method is to determine which pre-defined time interval matches the period requested by the user. For example, setting the time granularity to *month* and providing the start date of *March 21st, 2014* would lead to the same diary as providing a start date of *March 5th, 2014*, namely a diary from *March 1st, 2014* until *March 31st, 2014*.

After setting the appropriate time period, the database can be queried in order to retrieve all source items, like documents, notes, photos, etc., for the diary generation. (To be more precise, we mostly query for the resources in a user's PIMO that *represent* each of the information items. Getting an actual information item that is stored as a file would be done in another request.) If a time granularity of *days* is chosen, these items are directly transformed to diary entries without any condensation (left branch of Figure 6.6). If a different time granularity is given, the *condense()* method of the *Entry Condensation Manager* is called in order to start the actual rehashing of the basic material (right branch of the figure).

This process consists of several steps and the rest of this section is structured accordingly.

First, the information items provided for diary generation need to be analyzed. This is primarily done in order to find similarities and lay the basis for the subsequent clustering of items that share the same topic or belong to the same project or life situation, etc. More details are given in the next section.

6.2.1. Data Analysis and Similarity Calculation

To assess the information items' similarity we take three different measures into account:

- the label (headline) similarity s_L,
- the text body similarity s_T, and
- the similarity in annotated concepts s_C.

The first two measures are subject of the *text analysis* and the third one is addressed in the *concept annotation analysis*.

Text Analysis We use the *Text Analyzer* of the *Apache Lucene Software Library* (Apache Software Foundation, 2014) in order to process the information items' labels (headlines) and text bodies. In particular, this analyzer eliminates all stop words from the texts according to a given stop word list, that has already been used in other DFKI projects. Thus, it delivers us with a list of remaining terms from which we create the term vectors. Please note that we analyze labels and text bodies separately, so there are two different term vector spaces in our model. Finally, determining the (pairwise) label- (s_L) and text body similarities (s_T) of all of information items is done using the cosine measure as described in Section 2.8.

Concept Annotation Analysis In order to calculate the (pairwise) concept similarity s_C of the information items, we create concept vectors from their explicit annotations and extend them by implicit ones determined using our variant of spreading activation. For details please see Section 2.8. In addition to the basic remarks of that section, we will provide further implementation details (or design decisions, respectively) in the following:

- We used a slightly different spreading algorithm in the current version of our diary app. Since there was already a spreading algorithm available for the Semantic Desktop, which mainly differs in the way semantic relations are weighted, we deferred implementing our own and primarily focused on other problems. Still, implementing all ideas as described above remains an objective for possible future work (please see Section 8.2).
- We principally exclude the diary owner's *own person thing*, since it does not help in discriminating information items. (In your own personal diary everything is somehow linked to you.)
- There is an option (setting 27 in Figure 6.2) which enables a user to limit the weight of implicit annotations. In our implementation, all implicit annotations for a thing are normalized after spreading and multiplied with the weight given in setting 27. As a consequence, a user may completely deactivate including implicit annotations by setting a weight of zero.

After completing the text- and concept annotation analyses, the (overall) similarity of the information items can be calculated.

Similarity Calculation After the analysis phase, the label similarity $s_L(x, y)$, the text body similarity $s_T(x, y)$ and the similarity in annotated concepts $s_C(x, y)$ have been calculated for all pairs information items x and y. In order to compute the overall (combined) similarity $sim(x, y)$ of two items (x and y) we use a weighted sum that reads as follows:

$$sim(x, y) := w_L \cdot s_L(x, y) + w_T \cdot s_T(x, y) + w_C \cdot s_C(x, y) \tag{2}$$

The three weights w_L, w_T and w_C correspond to the settings 23, 24 and 25 (see Figure 6.2). Setting 26 is a boolean value that determines whether the result of the sum is normalized (by default this value is *true*).

Like mentioned before, calculating the similarity of information items provides the basis for the subsequent clustering process, which is described more thoroughly in the following.

6.2.2. Clustering

In order to the cluster information items to diary entries, we use a slightly modified version of the so-called *single-link* clustering algorithm which originates from Sneath and Sokal (1973) (Jain et al., 1999, p. 275). It belongs to the group of *hierarchical* clustering approaches, that "produce a nested series of partitions (while *partitional* methods produce only one)". Like depicted in Figure 6.7 (left-hand side) the *distance* d_{SL} between two clusters, X and Y, is the *minimum* of the distance between all pairs of patterns drawn from the two clusters (one pattern from X, the other from Y) (Jain et al., 1999, pp. 275):

$$d_{SL}(X, Y) := \min_{x \in X,\ y \in Y} d(x, y) \tag{3}$$

The algorithm "suffers from a *chaining effect* (Nagy, 1968). It has a tendency to produce clusters that are straggly or elongated" (Jain et al., 1999, p. 276), see right-hand side of Figure 6.7.

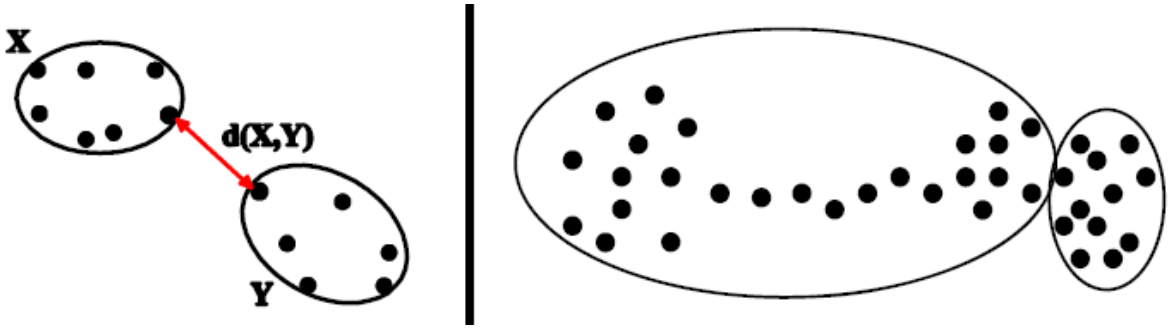

Figure 6.7: Single-link clustering algorithm: distance (left-hand side) and chaining effect (right-hand side) (Schubert and Zimek, 2011, p. 5)

Jain et al. state the algorithm in the following *agglomerative* version, which means that it "begins with each pattern in a distinct (singleton) cluster, and successively merges clusters together until a stopping criterion is satisfied" (Jain et al., 1999, pp. 274):

1. *Place each pattern in its own cluster. Construct a list of interpattern distances for all distinct unordered pairs of patterns, and sort this list in ascending order.*

2. *Step through the sorted list of distances, forming for each distinct dissimilarity value d_k a graph on the patterns where pairs of patterns closer than d_k are connected by a graph edge. If all the patterns are members of a connected graph, stop. Otherwise, repeat this step.*

3. *The output of the algorithm is a nested hierarchy of graphs which can be cut at*

a desired dissimilarity level forming a partition (clustering) identified by simply connected components in the corresponding graph.

Strictly speaking, we use our similarity measure $sim(x, y)$ introduced in the previous section instead of the distance/dissimilarity measure $d(x, y)$ described above. So, instead of using the *minimum dissimilarity*, our algorithm uses the *maximum similarity* to find the next pair of clusters to be merged:

$$sim_{SL}(X, Y) := \max_{x \in X,\ y \in Y} sim(x, y) \tag{4}$$

Alternatively, we could define $d(x, y)$ as follows and keep the original algorithm (using the *minimum dissimilarity*). Let our similarity measure $sim(x, y)$ be normalized to 1 (see setting 26 in Figure 6.2). Then we define $d(x, y)$ to be:

$$d(x, y) := 1 - sim(x, y) \quad \forall x, y \tag{5}$$

Our initial motivation to use this algorithm was as follows:

- Understanding and implementing it is fairly easy.
- An efficient implementation having a runtime complexity of $\mathcal{O}(n^2)$ is possible (Schubert and Zimek, 2011, p. 5).
- Since we want to cluster possibly thousands of individual items to only a few diary entries, the aforementioned chaining effect might help.

Centroid Check However, we later implemented a small modification, which we call the *centroid check*, in order to moderately alleviate the chaining effect and obtain slightly better results in our tests. When two clusters are subject to be merged next, we first calculate the similarity of their *centroids* (using the same measures as before). If this value is below a given *similarity threshold* t_S, the clusters are not merged. Instead, the algorithm proceeds with the next pair of clusters having the second highest similarity. By applying this method we slightly alleviate the aforementioned chaining effect. For the sake of completeness we also need to mention that the stopping criterion (step 3) given by Jain et al. has to be extended as follows: If there is no pair of clusters left to be merged due to failing the centroid check, the algorithm also terminates.

We found the centroid check to be especially useful if there are only very few information items available in a given period of time, e.g. less than ten. Applying a more moderate clustering in this case may result in six or seven clusters instead of two or three, for example. The criteria when to apply the centroid check can be set by the user in setting 28 (see Figure 6.2). Leaving this at its default value of *auto* turns on the centroid check if the number of

information items available for a given time period is below a certain threshold t_{CC}. Let n be the number of desired entries (setting 16), then t_{CC} is calculated as follows:

$$t_{CC} := a_{CC} \cdot n \tag{6}$$

a_{CC} is called the *centroid check auto mode factor* and may be altered in setting 29. If the number of desired entries n is set to 10 and a_{CC} is 1.5, the centroid check is only used if the number of information items available for a given period of time is below 15, for example. Apart from setting the centroid check to *auto*, there are two other modes. The first one is to *always* apply it. This also includes the cases in which a cluster consisting of more than one element is subject to be merged with a single-elemented one. The last option is called *clusters only* and applies the centroid check only if two clusters, both containing more than one element, are subject to be merged.

Similarity Threshold We already mentioned the *similarity threshold* t_S, which is a very important parameter, since it sets the clustering algorithm's termination criterion. It corresponds to the "cut at a desired (dis)similarity level" mentioned in step 3 of the algorithm outline given by Jain et al.. In addition to its ordinary usage, we also used this parameter as a threshold in the previously introduced centroid check. (Please do not confuse t_S with t_{CC} in this context: the first sets a threshold for the clustering process, whereas the second determines whether the centroid check is applied or not.)

The value of t_S can be set in three different ways (setting 21 in Figure 6.2). Let I_1, ..., I_m be the information items of a given period of time. The first possibility is using the *average* similarity of all information items:

$$t_S^{avg} := \frac{2}{m^2 - m} \cdot \sum_{i=1}^{m} \sum_{j=i+1}^{m} sim(I_i, I_j) \tag{7}$$

Suppose all pairwise similarities of the information items are given in a (square) matrix. Since the similarity is a symmetric relation ($sim(I_i, I_j) = sim(I_j, I_i)\ \forall i, j$), we thus only need the average of the $\frac{m^2-m}{2}$ values above the diagonal, which is reflected by the equation given above.

Alternatively, the user may also set this threshold to be a fraction of the maximum similarity found (*amount of max*):

$$t_S^{frac} := \max_{1 \le i \le m, 1 \le j \le m, i \ne j} sim(I_i, I_j) \cdot f_{TS} \tag{8}$$

f_{TS} is a factor that can be altered by the user in setting 22. The third option is to provide a *constant* similarity threshold. In this case, f_{TS} is used as an absolute value:

$$t_S^{const} := f_{TS} \tag{9}$$

Like mentioned in the previous section, if two clusters are merged, their term- and concept vectors are added. Additionally, the new cluster's centroid is updated.

Core Algorithm (First Pass) All ideas described so far determine the core of our clustering algorithm. Since we also implemented an additional *post-processing* method (i.e. a *second pass* of clustering), we thus refer to this core algorithm as the *first pass*. An example depicted in Figure 6.8 shows clustering results after the first pass.

We see five different diary entries that were not merged any further, although they have some annotated concepts in common, e.g. diary, PIMO and pizza. This is mainly due to a very low similarity in the labels and text bodies. Let us illustrate this using the third and fifth entry that are about pizza. We compare their term- and concept vectors by using the intersection of sets (for the sake of readability):

- labels: { pizza } ∩ { pizzabacken } = ∅
- text bodies: { pizzaessen, pizza, 29.03.2014, lecker } ∩
  { pizzabacken, pizzabacken.jpg, 05.04.2014, family, friends } = ∅
- concepts: { NoteX, Pizza, NoteY } ∩ { NoteY, ImageY, Pizza } = { NoteY, Pizza }

The term sets are both disjoint, but the intersection of the concept sets contains two elements. Nevertheless, the overall similarity is too low in order to induce a merging of these entries.

Like stated before, we therefore implemented a *post-processing* method, which is discussed in the following.

Post-Processing (Second Pass) In order to solve possible clustering problems like those mentioned before, we implemented a *post-processing* method (also referred to as a *second pass* of clustering), that tries to focus more on the semantics given by the entries' annotated concepts than the plain numeric values representing the similarity.

When merging clusters, their term- and concept vectors are added. In each iteration, this influences (and possibly changes) the ranking of an entry's most important concepts. The core idea of the second clustering pass is to establish (and compare) *prominent concept sets* for every diary entry created in the first pass. Due to the possibly very different composition of entries (i.e. some are composed of maybe a hundred items, others only have a few or a

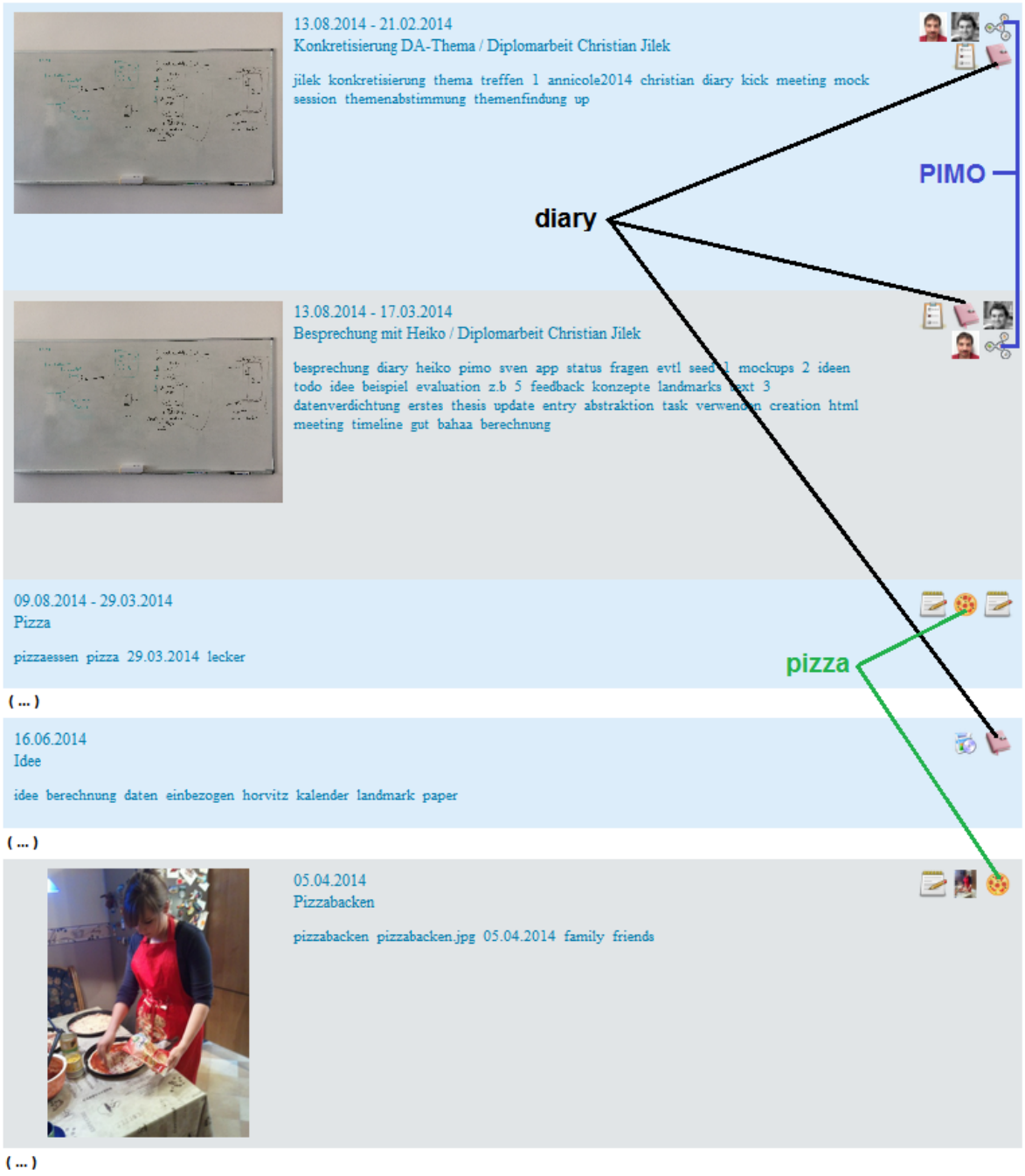

Figure 6.8: Example of clustering results after first pass

single one), we decided to use a relative threshold in order to determine which concepts may enter the *prominent concept sets*. Let E be an entry that is annotated with the concepts c_1, ..., c_k, then its *prominent concept set* $prom_S(E)$ is defined as follows:

$$prom_S(E) := \{\ c_i \mid c_i \geq f_L \cdot \max_{1 \leq i \leq k} c_i\ \} \tag{10}$$

f_L is a factor that is multiplied with the highest concept weight of E to determine the entering threshold. It can be modified using setting 31 (in Figure 6.2). Additionally, the maximum size (cardinality) of $prom_S(E)$ can be limited to a value given in setting 32, e.g. at most ten concepts may enter the set.

Please note that these are not the final concept annotations, which are determined later (see next section) and that $prom_S(E)$ is actually a set. On the one hand, this means that the former order of its concepts is ignored. On the other hand, this eases post-processing since the only criterion to check is whether a concept is present in several sets, specific or varying ranking positions may be ignored. Refining this procedure may be a subject for future work.

We also implemented a possibility to regulate this feature's impact. The post-processing algorithm basically runs for two iterations (rounds), the first one only includes *higher priority concepts* and the second *lower priority ones*, respectively. Things having a higher priority are for example *life situations*, *events* or *projects*, whereas *topics* belong to the second group. In general, these weights should correspond to the potential of a resource type of being a landmark. Please note that we currently do not use *persons*, *organizations* or *tasks* in this process (due to their low "discriminative power" in many cases). This could also be a subject for future work. Whether one or two rounds should be performed can be controlled using setting 30. A third option is to completely turn off this second pass of clustering.

Coming back to our example depicted in Figure 6.8, applying a second pass of clustering merged the two entries about pizza as well as the other three which share the topics of diary or PIMO, respectively. The result is shown in Figure 6.9.

Concept annotations were already in the focus of this section. In the following we will give some additional details about them and how they are finally set.

6.2.3. Concept Annotations of Diary Entries

Like stated before, the clusters' term- and concept vectors are added when they are merged, which influences (and possibly changes) the ranking of an entry's most important concepts.

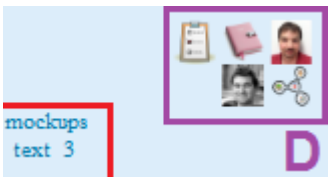

Figure 6.10: Concept annotations

To finally annotate diary entries with certain concepts, which are later also visible on their right edge (see D in Figure 6.3 or its close-up version in Figure 6.10), a *prominent concept list* $prom_L(E)$ is created, which is a variant of the aforementioned *prominent concept sets* $prom_S(E)$. They mainly differ in two aspects. First, the former is a *list* instead of a *set*, i.e. the concepts' order is relevant. Second, since we now determine the

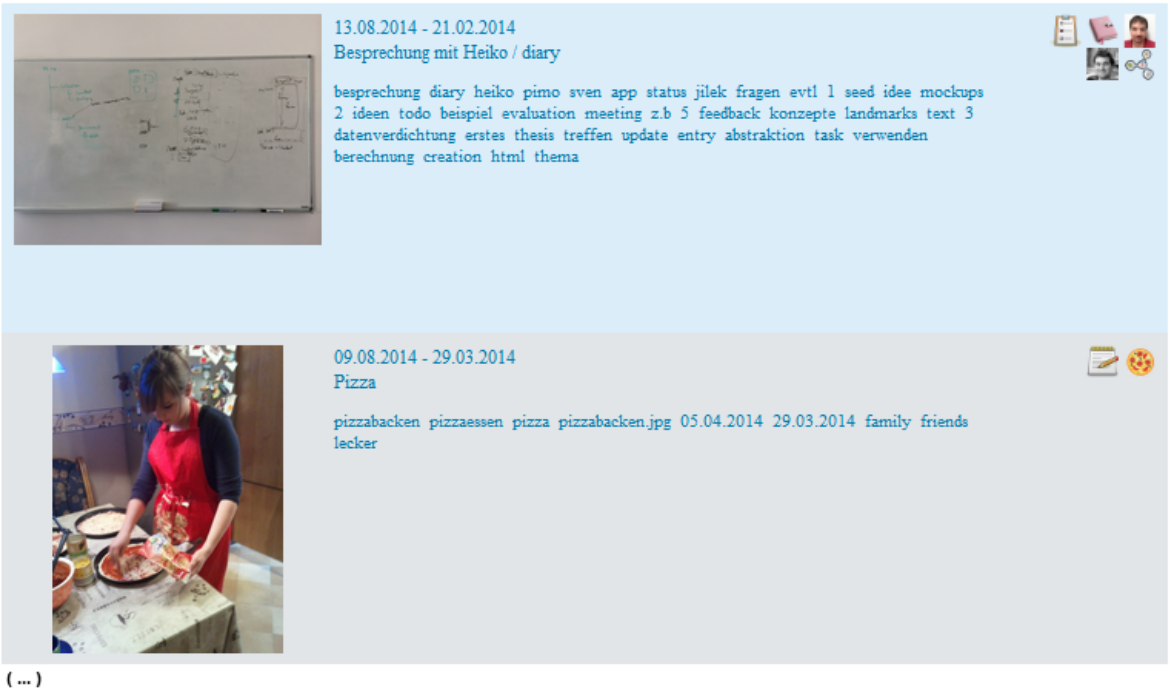

Figure 6.9: Example of clustering results after second pass

final concept annotations we do not limit these lists in size, like it was the case for the sets (which only contained intermediate concepts). Nevertheless, there is also a factor f'_L that corresponds to f_L, which sets the entering threshold. It can be modified using setting 33 (see Figure 6.2).

The clustering process might yield more entries than desired. If this is case, the most important ones need to be found, which is addressed in the next section.

6.2.4. Importance Evaluation

Our process of evaluating an entry's importance considers several factors that are described more thoroughly in the following.

Annotation Intensity (I) The first and most influential factor can be thought of as the *"annotation intensity"*. For an entry E_i having a concept vector d_i the annotation intensity $I(E_i)$ is defined as follows:

$$I(E_i) := \sum_j d_{ij} \tag{11}$$

In other words: all elements of the entry's concept vector are summed up. Thus, this

measure, for example, comprises the following aspects:

- **cluster size:** Since an entry's concept vector is the sum of all individual concept vectors, $I(E)$ is likely to be higher in larger clusters, since more (possibly non-zero) values are summed up.
- **number of an entry's annotations:** The more annotations an entry has, the more non-zero values are present in its concept vector. Thus, $I(E)$ will be higher in entries having more annotations.
- **connectivity of an entry's annotated concepts:** An entry having (explicitly) annotated concepts that are highly annotated will also have more implicit ones. Thus, there are more non-zero values in its concept vector.

Let us illustrate some possible effects:

- If an entry is not annotated at all, $I(E)$ is zero.
- An entry having several averagely connected concepts may have the same value as another one having fewer concepts that therefore have a higher connectivity.
- A single-elemented cluster associated with a highly connected concept may have a higher value than a cluster of maybe three items that is associated with only sparsely connected concepts.

In order to damp the high values of large clusters compared to smaller or single-elemented ones, we decided to measure this variable on a logarithmic scale in our final importance calculation (see the end of this section).

High Priority Things (H) The second factor is more focused on the *quality* of an entry's annotations. Similar to the *higher priority concepts* of the second clustering pass (see Section 6.2.2) we associate each of an entry's annotated concepts with a certain weight, for example: *life situation (9)*, *collection (8)*, *project (7)*, *event (6)*, ..., *note (2)*. For all things that are not in this list, a value of 1 is assigned. The *high priority things* factor H is then determined by the highest weight assigned to any of the entry's annotations. If an entry is not annotated at all, H is 0.

Like in the case before, we also apply the logarithm to H in the final importance calculation. This is motivated by the idea that we do not want to differentiate high priority concepts as much as the low priority ones. For example, whether an entry is associated with a life situation (weight of 9) or the collection of photos belonging to it (8) should be less of a difference than an entry being "somehow" annotated (1) or being a note (2). Although these differences are one in both cases, using the logarithm changes them to 0.05 and 0.18, respectively[19]. Thus,

[19] The exact formula we used is $log_{10}(w+1)$ in order to have the logarithm's zero for a weight of $w = 0$.

the difference for higher priority concepts is only about a fourth of the one for low priority ones afterwards.

Rarity (R) Our next factor is inspired by Horvitz et al. (2004). It determines whether an entry contains *rare* concepts. Let us assume we had a function $count_C(c_i)$ which counts the total number of occurrences of a concept c_i in the annotations of the diary entries. Furthermore, every concept c_i is associated with a certain type t_j ($j \in \mathbb{N}$) with a function $type(c_i) = t_j$ returning it. Additionally, we assume that a function $count_T(t_j)$ counts all occurrences of a certain type t_j in the annotations of the diary entries.

Thus, the *rarity* factor $rarity(c_i)$ for a concept c_i can be calculated as follows:

$$rarity(c_i) := \frac{count_C(c_i)}{count_T(type(c_i))} \tag{12}$$

Let us illustrate this using an example. Assume we have a diary consisting of several entries, all annotated with multiple concepts. The concept of "Heiko", which is a *person*, is mentioned four times, whereas *persons* in general are mentioned ten times. (This may mean that besides the four times "Heiko" was mentioned, another person has been mentioned six times. It could also mean that two other persons are mentioned three times each, and so forth.) Therefore, the *rarity* of "Heiko" is $\frac{4}{10} = 0.4$.

Assume an entry E is annotated with k concepts. We then evaluate the rarity for all of these concepts c_i ($1 \leq i \leq k$) and define this entry's overall rarity factor $rarity_E(E)$ to be the minimum of those values:

$$rarity_E(E) := \min_{1 \leq i \leq k} rarity(c_i) \tag{13}$$

Finally, if the resulting value of $rarity_E(E)$ is below a given threshold t_R (e.g. 1%), a *rarity bonus* b_R is associated with the entry. Thus, $R(E)$ is defined as follows:

$$R(E) := \begin{cases} b_R, & \text{if } rarity_E(E) < t_R \\ 0, & \text{otherwise} \end{cases} \tag{14}$$

The *rarity threshold* t_R can be set using setting 34 and the *rarity bonus* b_R can be modified in 35 (see Figure 6.2).

Rich Media (M) Since we want the generated diaries to be high in diversity (see special requirements, Section 4.6) and interesting to view and read, we additionally assign a bonus b_M to entries associated with *rich media*, e.g. photos or images. We therefore define the *rich*

media factor $M(E)$ for an entry E to be:

$$M(E) := \begin{cases} b_M, & \textit{if } E \textit{ is associated with rich media} \\ 0, & \textit{otherwise} \end{cases} \tag{15}$$

Again, b_M can be set by the user (setting 36).

Importance Value Calculation Summarizing this section, an entry's (overall) importance value $imp(E)$ can be calculated as follows:

$$imp(E) := log[I(E)] + log[H(E)] + R(E) + M(E) \tag{16}$$

Figure 6.11 shows the importance values as well as their components for four different diary entries. We especially would like to highlight two aspects. The first entry is the only one having a *rich media bonus* (in this example a value of 1), which in this case also makes it the most important entry. Without this bonus, the entry would be the least important one. This is mainly the effect we wanted to accomplish using the *rich media bonus*, although the impact is usually not that high (here: from last to best, usually: just a bit higher in the ranking). Secondly, the forth entry is the only one having a *rarity bonus*.

6.2.5. Concept Context Generation

We already introduced the *concept context* in Section 6.1.5. In summary, these are its key aspects:

- It as an overview of the things that were rather important or prominent in the currently viewed period of time.
- It is generated from *all* information items of the given period, whether they were sorted out in a possible importance evaluation or not.
- It can be thought of as a top ten ranking list of concepts (or top twenty, etc.).

To create this context, the concept vectors of all clusters (whether sorted out by importance or not) are added up and its f_C most prominent concepts are chosen. f_C can be set by the user (setting 37 in Figure 6.2).

Since the concept context also serves the purpose of enabling a user to manually include or exclude concepts, all concepts belonging to entries currently visible need to be incorporated into the context at their appropriate position. Nevertheless, this leaves the f_C top most concepts untouched. (To be more precise: let n be the number of all concepts annotated to

Figure 6.11: Entry composition and importance evaluation

diary entries in a given period of time. Then the context contains the $min(f_C, n)$ top most annotated concepts of this period as well as those incorporated for the reason mentioned before.)

The merging (clustering) and filtering (importance evaluation) of information items has produced all diary entries that should be presented to the user. In a last step, the labels and text bodies of these entries need to be processed to make them an expressive and meaningful summary of the items they are composed of. More details are provided in the following.

6.2.6. Label/Headline Generation

The principle of associating different concepts with varying weights according to their potential of being a landmark, like we introduced it in the context of our clustering post-processing method, can also be applied to the label generation. We basically have two groups to evaluate. On the one hand, there are the information items a diary entry is composed of. On the other hand, there are the concept annotations associated with the entry (to be more precise: the concepts belonging to this entry's final *prominent concept list*). Both sets are evaluated separately. First, all concepts are associated with a weight corresponding to their type, e.g. *life situation (9)*, *collection (8)*, *project (7)*, *event (6)*, etc. Then, three maximum values are determined: m_C as the maximum weight of all annotated concepts and m_I as the one of all information items, respectively. Additionally, $m_T = max(m_C, m_I)$ is the maximum of both maxima.

A diary entry's label is then generated as follows:

- $m_T > t_L$ and $m_C \geq m_I$: the name of the annotated concept having the highest weight is used as a label for the entry
- $m_T > t_L$ and $m_C < m_I$: the name of the information item having the highest weight is used as a label for the entry
- $m_T < t_L$: a *split label* is generated (see below)

t_L is a kind of *quality threshold*, which ensures that if both, concepts and information items, do not contain any *high priority item*, a *split label* is generated. This label combines the names of the annotated concept as well as the information item having the highest weight in their particular group. In our diary app's prototype we set t_L to have a value of 6, which means that if no *life situation*, *collection*, *project* or *event* is found in both groups, a *split label* is generated.

If there is more than one concept (or information item) associated with the highest weight within its group, an arbitrary one is chosen.

The last step in our condensation algorithm is the text summarization, which is the topic of the next section.

6.2.7. Text Summarization

In order to summarize the different information items a diary entry is composed of, we decided to present a list of the most prominent keywords to the user. This is motivated by two aspects. First, we did not have any text summarization tool at hand. Second, creating one by ourself would go beyond this thesis' scope. This is one of the major subjects for possible future work (please see Section 8.2).

The final diary entries are already associated with the summed up term vectors for labels and text bodies, which were created in the data analysis process. To create our keyword list for an entry we can therefore combine both term vectors and simply extract their most prominent terms.

Since a resource's label is usually something a human being has created as a kind of summary for this resource, we consider it more important than the text body in the context of summarization. So, before combining an entry's term vectors we first multiply the label's term vector with a value of f_L, that can be set by the user in setting 38. Thus, terms occurring in an entry's label undergo a slight boost. Nevertheless, the most prominent keywords may also come from the text bodies, if they are used frequently enough to compensate the label terms' boost.

Furthermore, the number of keywords per diary entry can be adjusted using setting 39.

In the last section of this chapter we present an example in which the author generated a diary from his PIMO for the time of this thesis.

6.3. Example: The Author's Diary for the Time of this Thesis

Since we assume that there are readers who do not have access to the Semantic Desktop and especially our diary application, we would like to provide a larger example having more realistic data. In particular, we generated a diary from the author's PIMO mainly covering the time of this thesis, although the major parts of the data belong to the months of March to June – the identification stage and design phase of this project.

The author's diary can be primarily seen as a scientific diary enriched with some personal remarks or notes. It is based on 99 information items, mainly notes, that were condensed to 15 diary entries, which we consecutively numbered starting with the newest entry (*blog style*).

Since this diary's screenshot covers three pages, Figure 6.12 provides an overview of how the partial screenshots given in Figures 6.13, 6.14 and 6.15 belong together.

Figure 6.12: Diary overview

Next, we will comment on some of this diary's entries.

Diploma thesis (entry 1) Entry 1 covers most parts of this project. The concept annotations show a task labeled with *Diploma Thesis of Christian Jilek*, the topics of *PIMO* and *diary* as well as two photographs of the author's advisor *Dr. Heiko Maus* and his co-advisor *Dr. Sven Schwarz*. The entry is entitled with *Meeting with Heiko* and *diary*, whereas the most prominent keywords are basically a composition of the terms already mentioned. A photograph showing the white board of an early brainstorming session is associated with the entry.

Pizza (entry 2) Since the author met family and friends to make or eat pizza several times throughout the time of this thesis, these rather personal notes were clustered together in a diary entry called *pizza*. Additionally, a photograph showing the author's cousin making pizza is associated with the entry.

Other personal events (entries 6, 8, 10 and 13) Other personal events like having lunch on Mother's Day or being invited to a friend's confirmation were taken over into the diary as single-elemented clusters, since they lack any annotations. Two of them (entries 10 and 13) are associated with a photo.

Events at university (entries 7 and 12) During the time covered by this diary there were two important events at the university, which is also indicated by the university's logo being the most prominent annotated concept in these entries. They are about the final presentation of the author's student research paper (entry 7) and the inspection of this last exam (entry 12). Please note that the labels are also very expressive: the first one reads as *student research paper* and the other one is a *split label* (see Section 6.2.6) called *"last exam / University of Technology Kaiserslautern"*.

Visiting a soccer match (entry 9) Entry 9 is about visiting a soccer match of the local club "1. FC Kaiserslautern". The entry is annotated with this club's logo as well as the city coat of arms of Kaiserslautern. Additionally, a photo of the ticket is associated with the entry.

Concept context The concept context lists *Diploma Thesis of Christian Jilek*, *diary*, *Dr. Heiko Maus*, *Kaiserslautern*, *University of Technology Kaiserslautern*, *Dr. Sven Schwarz*, *PIMO*, *student research paper*, *DFKI* and *diploma thesis* among the top twenty concepts for this period of time, which we consider is a good overview of the very important or prominent things.

In order to also get an unbiased (or at least less biased) assessment of our application, we additionally did a user experience evaluation with several DFKI-external testers whose results are described in the next chapter.

My Diary: 2014

13.08.2014 - 21.02.2014
Besprechung mit Heiko / diary

besprechung diary heiko pimo sven app status jilek fragen evtl 1 seed idee mockups 2 ideen todo beispiel evaluation meeting z.b 5 feedback konzepte landmarks text 3 datenverdichtung erstes thesis treffen update entry abstraktion task verwenden berechnung creation html thema

1

09.08.2014 - 29.03.2014
Pizza

pizzabacken pizzaessen pizza pizzabacken.jpg 05.04.2014 29.03.2014 family friends lecker

2

04.08.2014 - 02.07.2014
Umlaute-Bug in Label-Generierung von Notes repariert / Notes

generierung label notes repariert umlaute bug bugfix überarbeitete überprüfe bahaa absprache abstimmung behoben bekommen evtl hoffentlich möglichkeit newlines nochmal seed text verbessert

3

30.06.2014 - 23.06.2014
Anmeldung Diplomarbeit

diplomarbeit anmeldung bestätigung 30.06.2014 montag offiziell vorsitzenden 30.09.2014 abgabedatum abgegeben andreas angemeldet angemeldete arbeit ausgabedatum auskunft beginn dengel dr genehmigt kommende kommenden offizieller portal prof prüfer prüfungsamt prüfungsausschuss prüfungsausschusses qis tage wi zumindest

4

27.06.2014
Spreading Activation

activation spreading beachten bestimmte einsetzen gewichtungen haspart hassubtopic hassupertopic idee ismanagerof ismemberof ispartof manages person projekt relationen topic unterschiedliche z.b zumindest

5

01.06.2014
Kerwe-Essen mit Family

essen family kerwe 01.06.2014 hähnchen lecker schwarzwaldständchen zeug

6

13.05.2014 - 03.04.2014
Studienarbeit

studienarbeit wirtschaftsinformatik abschlussvortrag 03.04.2014 13.05.2014 abgeholt conclusion echter feedbackgespräch formulierenung hinsichtlich insb juhu kapitel kapiteleinteilungen kleinere lang letzter merke termin tipps uni zahlen

7

11.05.2014
Muttertagsessen

muttertagsessen 11.05.2014 anlässlich essen familie großteil muttertags pasta pizza salat

8

Figure 6.13: The author's diary for the time of this thesis (part 1/3)

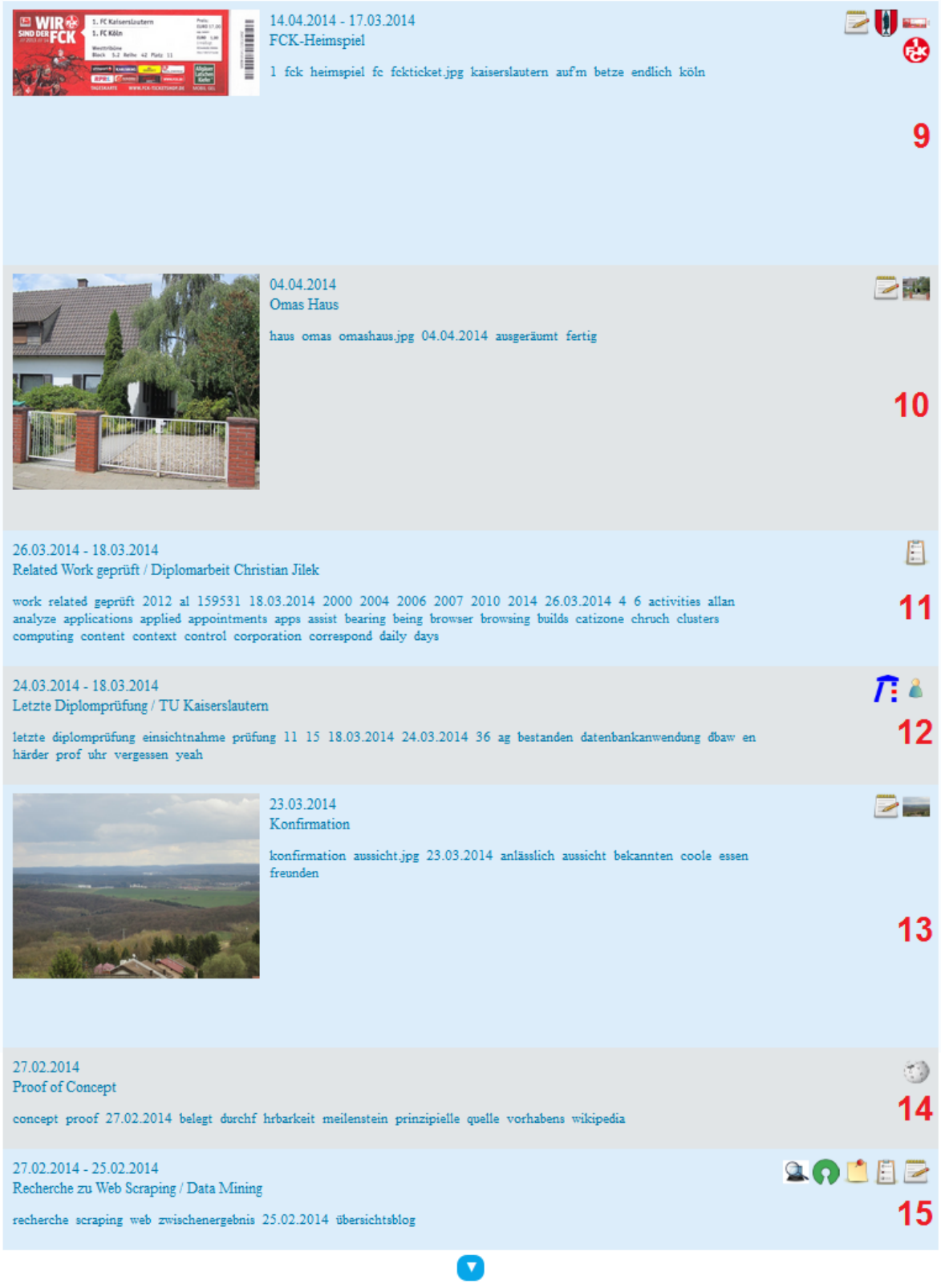

Figure 6.14: The author's diary for the time of this thesis (part 2/3)

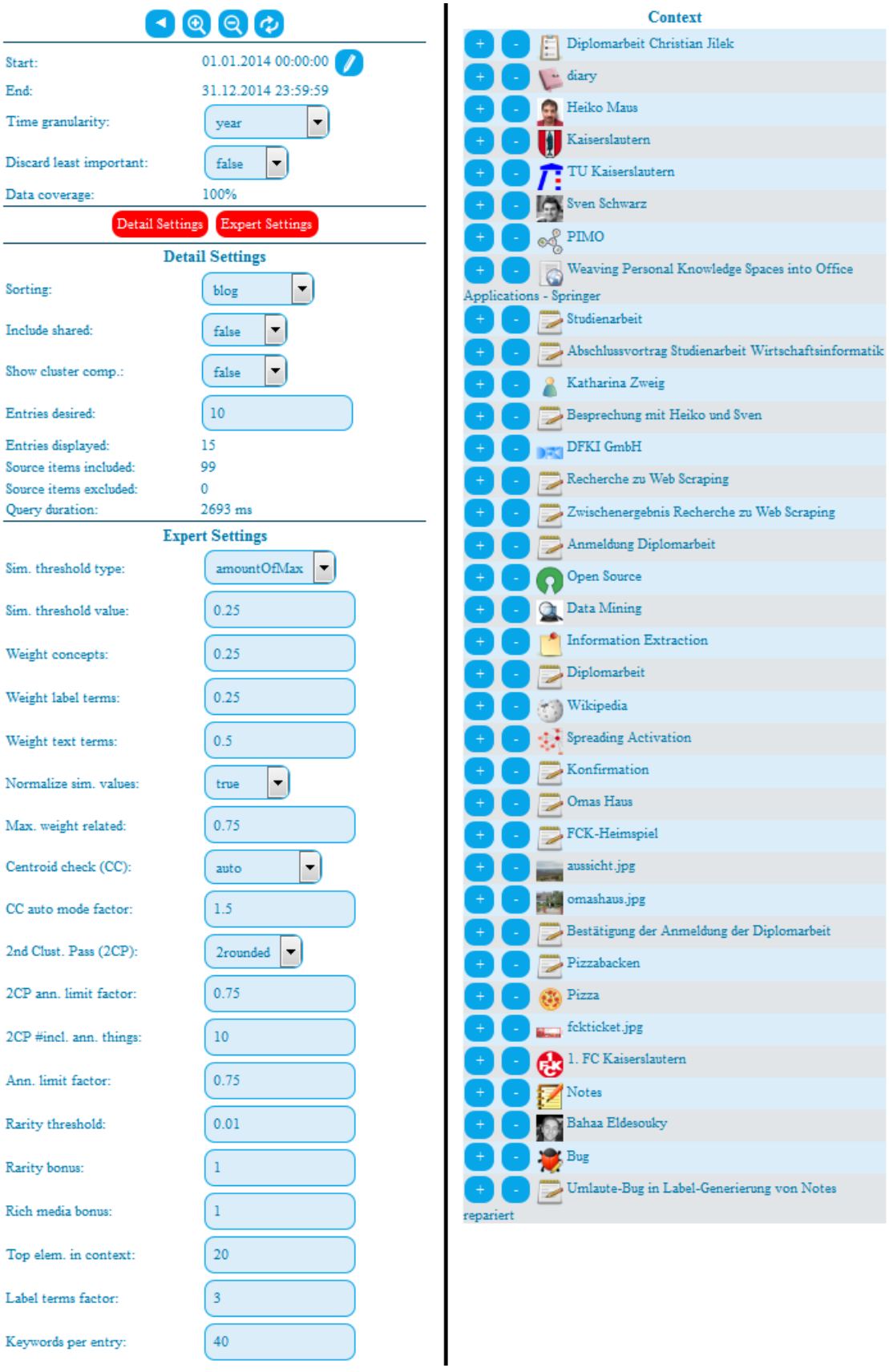

Figure 6.15: The author's diary for the time of this thesis (part 3/3)

7. User Experience Evaluation

Having started this project with a survey revealing people's interest in an application that would ease retrospection (see Section 1), we developed an app that generates diaries from users' personal information models on demand. We incorporated features like zooming in and out of time periods evoking concretization- and condensation processes, or a concept context providing the user with an overview of those things he seemed to be concerned with the most during a given period of time. Furthermore, all data is presented using the look and feel of a blog. Since we think these ideas are rather innovative compared to other approaches seen in the past (see Section 3), we thus would like to finally evaluate, whether our diary app satisfies people's needs.

7.1. Setting

In order to assess our app, it was delivered to a four-headed group of DFKI-external testers we hereafter call *PANiC*, which is an acronym of their first names. PANiC are four students (two female, two male) of computer science and economics in their final year before earning their master's degree. Before our application was delivered, they already had access to the Semantic Desktop for about four months. Their PIMOs therefore were fed with hundreds of information items, in some cases even more than a thousand. They were able to test our diary app for three weeks in total. During the first week they provided early feedback that was in parts directly incorporated into our app. For another two weeks they were able to test the final version and afterwards participated in our user experience evaluation.

We used the *USE questionnaire* proposed by Lund (2001). *USE* stands for *usefulness*, *satisfaction* and *ease of use*. It is applicable in several domains like software, hardware, services and user support materials. The questionnaire is constructed using a *seven-point Likert scale* ranging from *strongly disagree* to *strongly agree*, while the items were constructed with intention to be *as simply worded as possible* and *as general as possible* (Lund, 2004)[20]. The author presents 30 items (questions) distributed over four different categories (factors), which are *usefulness*, *ease of use*, *ease of learning* and *satisfaction*. Some items do not load as strongly as others on these factors (Lund, 2004). We used a *short form of the questionnaire* that can be "constructed by using the three or four most heavily weighted items for each factor" (Lund, 2004). Beside these twelve "standard items", we added eight more questions directly concerning our diary app's core features. All questions were formulated in a way that higher ratings are better in each case. The 20 items of our questionnaire are as follows (the actual questionnaire is depicted in Figures B.1 and B.2 in Appendix B):

[20] To our best knowledge (Lund, 2004) is just a reprint of (Lund, 2001).

Usefulness:

1. It helps me be more effective.
2. It helps me be more productive.
3. It is useful.

Ease of use:

4. It is easy to use.
5. It is user friendly.
6. It requires the fewest steps possible to accomplish what I want to do with it.

Ease of learning:

7. I learned to use it quickly.
8. I easily remember how to use it.
9. It is easy to learn to use it.

Satisfaction:

10. I am satisfied with it.
11. I would recommend it to a friend.
12. It is fun to use.

Core features:

13. The way information items are clustered to diary entries makes sense to me.
14. The labels (i.e. headlines of diary entries are chosen meaningfully.
15. The text bodies of diary entries provide good summaries of the information items they refer to.
16. If the number of desired diary entries is limited the most important ones are actually chosen.
17. By zooming in and out of time periods I am able to find my desired level of details.
18. Manually including or excluding concepts shifts the diary's emphases as expected.
19. The app allows an appropriate and satisfactory retrospection on those parts of my life that are reflected by my PIMO.
20. The concept context provides a good impression, i.e. a quick overview, of those things (reflected by my PIMO) that concerned me the most in the chosen period.

Apart from these closed questions we added a commentary field (open question), in which the participants could express further feedback – positive or negative – concerning our app.

The results of our evaluation are presented in the next section.

7.2. Results

In summary, our diary app achieved very good results in the evaluation, which are discussed more thoroughly in the following.

Closed Questions The average results of 18 closed questions of our evaluation (items 3 to 20) range from 6.00 to 6.75. The remaining two items (items 1 and 2) were rated 5.50 and 5.25 on average. They are about being "more effective" and "more productive" using the diary app. In our opinion, these items were rated worse than the others due to the questions being inadequate in this context. It is doubtable whether reminiscing about your past can help you be more effective or productive. There may be cases where this is true, but the diary setting is rather associated with leisure time, relaxation, etc. Nevertheless, we still included these questions since they are part of the *USE* standard set.

Figure 7.1 provides an overview of the results, whereas Table 7.1 contains more details.

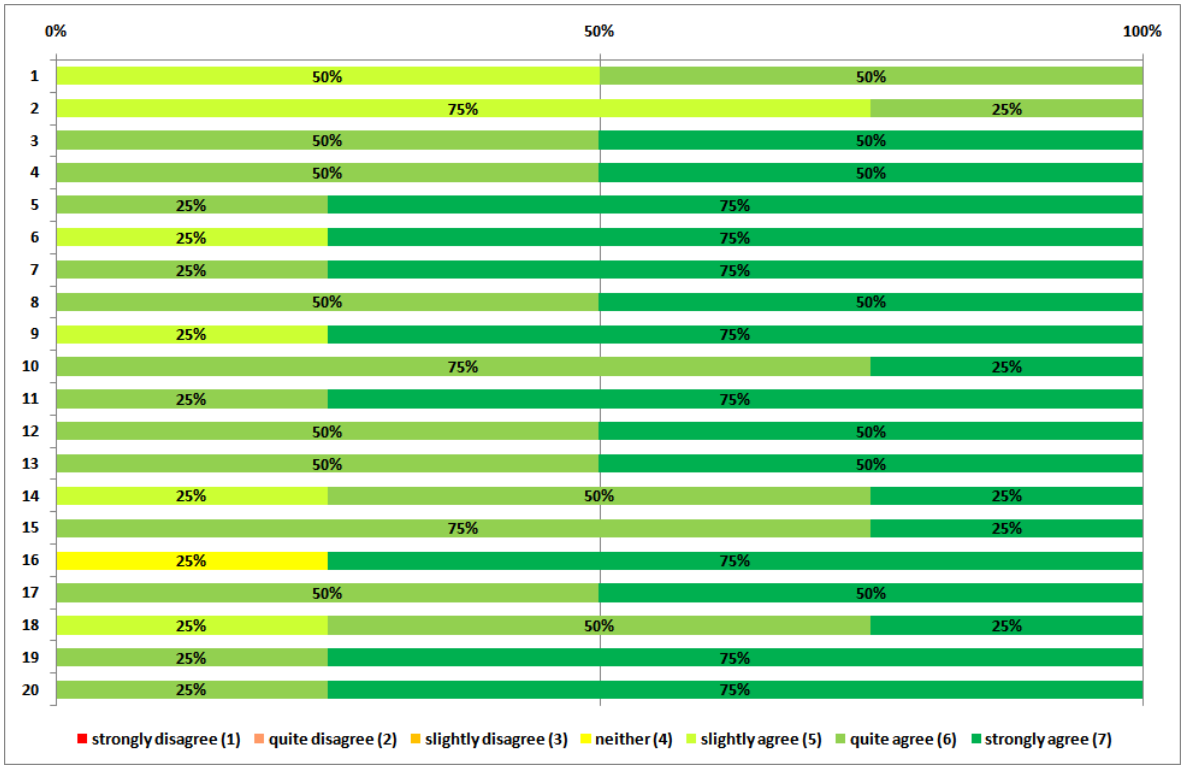

Figure 7.1: Condensed answers to closed questions of user experience evaluation (a seven-point Likert scale was used, questions were formulated in a way that higher ratings are better in each case)

We see that 40 of 80 items were rated with the highest value (7), 30 with a rating of 6, 9 with 5 and a single one with 4. Ratings less or equal to 3 have not been assigned.

Participant	1	2	3	4	avg.
Gender	♂	♂	♀	♀	
1. be more effective	6	5	5	6	5.50
2. be more productive	5	6	5	5	5.25
3. is useful	7	7	6	6	6,50
4. is easy to use	7	7	6	6	6.50
5. is user friendly	7	7	7	6	6.75
6. requires fewest steps possible	7	7	7	5	6.50
7. can be learned quickly	7	7	7	6	6.75
8. usage can easily be remembered	7	7	6	6	6.50
9. usage easily learned	7	7	7	5	6.50
10. satisfied with it	6	7	6	6	6.25
11. recommend it to a friend	7	7	7	6	6.75
12. is fun to use	7	7	6	6	6.50
13. sensibly clustering items to entries	6	7	6	7	6.50
14. meaningful labels/headlines	6	7	5	6	6.00
15. expressive text bodies	6	7	6	6	6.25
16. importance evaluated correctly	7	7	7	4	6.25
17. zooming in/out is useful to adjust details	7	6	6	7	6.50
18. manual in-/exclusion works as expected	7	6	5	6	6.00
19. enables satisfactory retrospection	7	7	7	6	6.75
20. context provides quick overview	7	7	6	7	6.75

Table 7.1: Detailed answers to closed questions of user experience evaluation (a seven-point Likert scale was used, questions were formulated in a way that higher ratings are better in each case)

Concerning our main questions whether our app enables an easy (items 4 to 6) and satisfactory retrospection on one's life (item 19) as well as a good overview of important things that concerned a person during a given period of time (item 20), we can observe that the corresponding items were all rated with 6.50 or 6.75 on average, which we think is clearly a success.

Open Question In addition to these results, the participants also provided verbal feedback using the commentary field (open question) of the questionnaire. Their comments in full detail can be found in Chapter B.1 of the appendix. Since they are in German we provide a translated summary in the following:

- **(+) innovative:** PANiC found our application to be very innovative (participant 2, or "P2" for short). One tester stated, that he has not seen a tool providing this or a similar functionality, yet (P1).
- **(+) fun to use:** They also stated that they had fun using our app to retrospect on their past (P1 and P3).
- **(+) ease of use:** PANiC was surprised by the ease of use of our application (P1).
- **(+) high quality of results:** They were also surprised by the high quality of the results (P1).
- **(−) response time:** The response time of the system is still unreasonable in some cases, especially if the diary for a longer period of time is generated. This is due to our diary crawler not being implemented, yet (please see Section 8.2). A participant suggested implementing a progress bar indicating how long the generation process will still take (P1).
- **(−) low online support/guidance:** PANiC also suggested providing a help section (or help file) in the tool as well as additional information when hovering with the mouse over certain settings (P1). These support- or guidance features are mainly missing due to our app being a proof of concept implementation.

Like in the case of the closed questions, these comments are predominantly very positive with some advice for further improvement.

In the next chapter, we will conclude this thesis by giving a short summary and an outlook on possible future work.

8. Conlusion and Outlook

This chapter contains this thesis' final conclusion as well as an outlook on possible future work.

8.1. Conclusion

In the beginning of this project, we carried out a survey which revealed people's interest in an application that would ease retrospection. Furthermore, this app should be capable of handling various media and reduce the time users have to spent in order to preserve their memories and experiences (Section 1). Addressing this, we came up with the idea of creating an application that generates diaries on demand based on users' personal information models. After providing the conceptual and technical background (Section 2), we evaluated several works and applications in research and industry (Section 3). In these studies, we found out that there were only very few approaches of generating diaries from users' data tracks in the past. All diary-related projects we presented were based on reading out users' mobile devices (primarily smart phones), e.g. sensor data or activity logs. By applying means of artificial intelligence, these applications had to associate the acquired data with inferred semantics. Using the Semantic Desktop we have the advantage of being already provided with semantically annotated data. Thus, it was up to us to optimally exploit this advantage and provide an app that surpasses several shortcomings of earlier approaches.

One of the main problems existent in the aforementioned applications was the lack of providing an actual overview of a user's past. In most cases, users are exposed to an overwhelming mass of individual information items, like documents, notes, photos, calendar events, etc., when retrospecting on a selected period of their life. As a consequence, they are not able to easily comprehend what actually happened in this period. Although some timeline-related approaches already tried to tackle this problem, condensations or abstractions concise and meaningful enough to enable a satisfactory retrospection have – in our opinion – not been presented, yet. We tried to solve this problem by introducing the feature of *zooming in and out of time periods*, which evokes concretization- and condensation processes (Section 4). Instead of showing hundreds and thousands of individual information items when, for example, retrospecting on a year, our app provides abstractions like project names, life situations, events, etc. This data is presented as diary entries having the look and feel of a modern web log. Since we want to present an adequate number of entries, merging (*clustering*) and filtering (*importance evaluation*) of the information items is necessary. Especially the former also facilitates a *high diversity within the diary*, making it interesting to view and read. Furthermore, the entries contain a *textual summary* of all information items they consist of, and they are supplemented by icons representing *annotated concepts* as well as *photos* they are associated with. In addition, we implemented another feature called the *concept context*

which provides an overview of the most prominent or important things of a selected period of time. By looking at this context, users may quickly get an impression of what they were concerned with the most in the given period.

We designed our diary tool to be a distributed client/server application (Section 5) and created a proof of concept implementation whose client component is an *HTML5 app* belonging the DFKI's so-called *PIMO5 client* and whose server component is a *JAVA servlet* (Section 6). In total, the software developed in this project consists of about 7200 lines of code (including approximately 1500 experimental lines which were only used temporarily).

The app achieved very good results in an evaluation by a four-headed group of DFKI-external testers (Section 7). They found it to be very innovative and fun to use. Additionally, they were surprised by its ease of use and the high quality of the delivered results. In summary, they confirmed that our application enables an easy and satisfactory retrospection on one's life and additionally also provides a good overview of the things a person was concerned with during a selected period of time. Besides, some hints for improvement were given, which we address (among others) in the next section.

8.2. Outlook

Since our diary application is only a proof of concept implementation, there are plenty of aspects that can be improved. Many of them can probably be solved by adding just a few lines of code. In this section we therefore confine ourselves to only mention the major and/or rather conceptual subjects for possible future work.

Response time / diary crawler Probably *the* major problem of our diary app is the unreasonable response time in some cases, especially if longer periods of time are processed. We already designed a *diary crawler* as one counter measure for this problem (see Section 4.6.2), but it has not been implemented, yet. Nevertheless, all necessary preparations are complete. A *Diary Data* object (associated with the Semantic Desktop's *Diary API*, see Figure 5.1) fully contains a user's diary and can easily be stored persistently, since it is available in JSON format (i.e. as a text string). If this object is, for example, saved to a database together with the diary's start date and time granularity as well as a timestamp of the diary creation time, the system could first query this database instead of generating a user's diary. If this query succeeds, no generation process is necessary and the result may immediately be presented to the user significantly improving the response time. After checking this diary's creation timestamp the user may decide whether he wishes a re-generation (maybe some things have changed and the diary would look a bit different now). In this case as well as in the cases of users manually including or excluding concepts, a new diary has to be generated (the diary crawler gets left out).

Since our diary app's server component is directly integrated into the Semantic Desktop

infrastructure, the same would be true for its diary crawler. Thus, an additional database (or schema) has to be created and the crawler has to be set up as a permanently running service, which is a task for the DFKI's Semantic Desktop administrators. In addition, a crawler constantly generating diaries in the background would also drain a significant amount of the system's CPU time, which is then missing for other components of the Semantic Desktop.

Topic lanes Another feature already described in our design but not implemented yet are the *topic lanes* (see Section 4.5). Especially zooming *into* time periods is therefore not as comfortable as we planned it to be. Since the user does not have an overview of the temporal coherences *within* a diary entry, he is not able to easily find "hot spots" or more interesting sub-periods allowing him a more targeted zoom-in.

Mobile UI Although the usage of our app is possible on mobile devices, typically having a smaller screen than desktop computers, it is not very comfortable for the user. A solution we had planned is implementing the setting bar (see Figure 6.2) as a panel that may slide in and out from the right, thus freeing more space for the actual diary entries.

Text summarization / natural language One of our initial intentions to choose the form of a diary was having some kind of *editorial preparation of text*. Thus, it would be preferable if dairy entries contained text written in natural language (i.e. complete sentences) instead of a keyword list. Since we did not have any text summarization library available and its creation obviously goes beyond this thesis' scope, incorporating a text summarization component producing natural language remains a major subject for possible future work. A second topic closely related to this aspect are varying languages. For example, mixing the German and English language in the diary most likely hurts the performance of the condensation and abstraction algorithms. To our best knowledge, this is also the case for SEED. Again, this goes deep into the topic of natural language processing, which is mostly out of our thesis' scope.

Things having time spans Except for events (single point in time or *one* time span), we have not included PIMO resources having time spans, yet. For example, these are documents, non-continuously written by a person for several weeks or months, or tasks performed during various periods of time.

Imagine our advisor writing an assessment of this thesis, which may take several days or weeks. If the document is complete, he just leaves a little space at the end for inserting the final grade after discussing with his colleagues and the professor. Some time later, the grade is inserted. The problem now is to determine the "right spots" in the diary this document actually belongs to. Should it be shown on all days the user has changed something in the document, or only on its completion day, or the period in which most changes occurred?

Please think of another scenario in this matter. A person corrects a typing error several months or years after creating a document: is this modification worth showing the document again in the new period?

Solving this problem and showing these resources in the "right" periods of a diary requires more time for brainstorming and experimenting that we could afford during this thesis.

Remaining use cases We implemented the following seven of ten uses cases presented in Section 4.4:

- generate diaries for given periods of time (UC1)
- update a diary by explicitly excluding or including selected concepts (UC2 and UC3)
- zoom in and out of a diary interval (UC4 and UC5)
- jump from diary entries to actual contents (UC6)
- advanced- or expert mode in diary generation (UC10)

Nevertheless, there are three use cases which are not implemented, yet. We will discuss them as well as the particular reasons for deferring their implementation in the following:

- **Use case 7: Use diary entries to set time interval of a search:** This use case basically only requires a button which transfers the time of a selected diary entry or time period to the Semantic Desktop's search interface. Since its search capabilities are currently rather limited, we deferred implementing this use case until further improvements are available in this area.
- **Use case 8: Embed current diary into other context:** It is easy to imagine how our diaries would look like if there were some "third-party" or historic entries between the own ones of a user. Since this raises more problems than the conceptual benefit obtained, we deferred the implementation of this feature. Possible problems are mainly how to store and manage historic datasets like the biography of a celebrity or chronicles of historic events like the Ukrainian crisis, e.g. create database structures, set access rights, provide a user interface to enter or update the data, create possible components automatically searching the web for historic data, etc.
- **Use case 9: Share diary with others:** A very interesting feature not yet implemented is the sharing of (parts of) a user's diary with others. Since all resources of a user's PIMO can currently only be private or public, the same would be true for diary entries. We therefore deferred implementing this feature until a more advanced sharing model is available in the Semantic Desktop, e.g. the definition of user groups having access to certain resources, etc.

Social media interface Closely related to the sharing aspect mentioned in the last use case is the idea of connecting our diary app to social media platforms like *Facebook*, *Google+*, *Twitter*, etc. This includes both directions, importing data for the diary generation from a user's social media profile as well as sharing the created diary entries with others using these platforms. Implementing this use case can primarily be reduced to connecting a user's PIMO to these social media platforms. If this is accomplished, probably not too much work is left to be done in the diary app. In addition, we then could also incorporate some of the ideas (e.g. the distinction between "ordinary" users and celebrities) found in the paper about tracking individuals on Twitter (see Section 3.3.7).

Algorithm- or parameter tuning In general, much work can still be put into optimizing the different heuristics and especially their parameter sets. We could, for example, think of profiles for *very few*, *an "usual" amount* or *lots of* information items. Applying the centroid check during clustering (Section 6.2.2) only if rather few items are available for condensation is one step towards these profiles.

Other starting points are the selection of the best fitting photo in a collection associated with an entry, fully implementing our variant of spreading activation, the similarity calculation (what if resources do not have any text associated with them – should the different weights be altered accordingly?) or the clustering algorithm.

Let us illustrate the last mentioned example. Currently, the second clustering pass (post-processing) compares the *prominent concept sets* $prom_S(E)$ of different entries. Suppose we have four different concepts, A to D, and three different information items having the concept vectors d_1 to d_3, which are given below. D_1 to D_3 are the single-elemented clusters containing these items. If the threshold factor f_L for entering $prom_S(E)$ is for example 0.75, the resulting entering threshold is also 0.75 (=$1.00 \cdot 0.75$). Thus, concept D (the fourth element of the concept vectors) would not be in any of the prominent concept sets, since its weight of 0.7 is too low in each vector. The post-processing method would merge all three clusters, since D_1 and D_2 share the concept of C, and the sets of D_1 and D_3 both contain A. As a consequence, the individual concept vectors would be added, making D the highest weighted concept in the resulting entry's *prominent concept list* $prom_L(E)$. If we furthermore assume that the threshold factor f'_L for entering this list is also 0.75, C – which was one of the reasons for clustering these items – would not be part of the final list, since its weight of 1.5 is less than the resulting threshold of 1.575 (= $2.10 \cdot 0.75$).

$$
\begin{aligned}
d_1 &= (1.00,\ 0.00,\ 0.75,\ 0.70) \quad \Rightarrow \quad prom_S(D_1) &&= \{\ A,\ C\ \} \\
d_2 &= (0.00,\ 1.00,\ 0.75,\ 0.70) \quad \Rightarrow \quad prom_S(D_2) &&= \{\ B,\ C\ \} \\
d_3 &= (1.00,\ 0.00,\ 0.00,\ 0.70) \quad \Rightarrow \quad prom_S(D_3) &&= \{\ A\ \} \\
\\
d_{123} &= (2.00,\ 1.00,\ 1.50,\ 2.10) \quad \Rightarrow \quad prom_L(E_{123}) &&= (\ D,\ A\ )
\end{aligned}
$$

Even though this is a constructed example, it may be an undesired effect in our application, making it harder to comprehend the reasons why certain items were merged to be become a diary entry. One possibility to solve the problem above would, for example, be setting f'_L dynamically according to the intermediate results, thus making C appear in the entry's final concept list. In summary, we recommend further tuning and optimization of the used algorithms or their parameter sets, respectively.

The aforementioned aspects provide a basis for further improvement. Regardless of their actual realization, we are confident that our diary app would already satisfy the needs of a larger audience considering the very good results it achieved in a first user experience evaluation.

Bibliography

S. Adam, J. Doerr, M. Eisenbarth, and A Gross. Using task-oriented requirements engineering in different domains – experiences with application in research and industry. In *Requirements Engineering Conference, 2009. RE '09. 17th IEEE International*, pages 267–272, 2009.

P. André, M. L. Wilson, A. Russell, D. A. Smith, A. Owens, and M. C. Schraefel. Continuum: Designing timelines for hierarchies, relationships and scale. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, pages 101–110, New York, NY, USA, 2007. ACM.

Apache Software Foundation. Apache Lucene: Ultra-fast search library and server. Software Framework, 2014. URL `http://lucene.apache.org/`. Last modification: September 3rd, 2014, Accessed: September 25th, 2014.

Bücher-Wiki. Das Tagebuch. Website. URL `http://www.buecher-wiki.de/index.php/BuecherWiki/Tagebuch`. Accessed: September 25th, 2014.

O. Bergman, R. Beyth-Marom, and R. Nachmias. The project fragmentation problem in personal information management. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI 2006)*, pages 271–274. ACM Press, 2006.

S.-B. Cho, K.-J. Kim, K.S. Hwang, and I.-J. Song. AniDiary: Daily cartoon-style diary exploits bayesian networks. *Pervasive Computing, IEEE*, 6(3):66–75, July 2007.

F. Crestani. Application of spreading activation techniques in information retrieval. *Artificial Intelligence Review*, 11(6):453–482, 1997.

T. H. Davenport. *Thinking for a living: how to get better performances and results from knowledge workers*. Harvard Business Press, 2005.

A. Dengel. Knowledge technologies for the social semantic desktop. In *Knowledge Science, Engineering and Management*, pages 2–9. Springer, 2007.

A. Dengel and A. Bernadi. Einleitung. In A. Dengel, editor, *Semantische Technologien: Grundlagen – Konzepte – Anwendungen*, pages 3–19. Spektrum Akademischer Verlag, 2012.

A. Donath. Stories: Google Plus erstellt automatisch Fotogeschichten. Website, 2014. URL `http://www.golem.de/news/stories-google-plus-erstellt-automatisch-fotogeschichten-1405-106619.html`. Last modification: May 21st, 2014, accessed: September 25th, 2014.

S. Dumais, E. Cutrell, J.J. Cadiz, G. Jancke, R. Sarin, and D.C. Robbins. Stuff I've Seen: A system for personal information retrieval and re-use. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, pages 72–79, New York, NY, USA, 2003. ACM.

R. East. Overview of the Model View ViewModel (MVVM) pattern and data-binding. Website, 2008. URL `http://russelleast.wordpress.com/2008/08/09/overview-of-the-modelview-viewmodel-mvvm-pattern-and-data-binding/`. Last modification: August 9th, 2008, accessed: September 25th, 2014.

Facebook, Inc. Facebook Timline. Website, 2011. URL `https://www.facebook.com/about/timeline`. Accessed: September 25th, 2014.

D. Goodwin. Google's timeline search option is history. Website, 2011. URL `http://searchenginewatch.com/article/2124563/Googles-Timeline-Search-Option-is-History`. Last modification: November 11th, 2011, accessed: September 25th, 2014.

J. Hailpern. YouPivot & TimeMarks. Application software, 2012. URL `http://youpivot.com/`. Last modification: January 26th, 2012, accessed: September 25th, 2014.

J. Hailpern, N. Jitkoff, A. Warr, K. Karahalios, R. Sesek, and N. Shkrob. YouPivot: Improving recall with contextual search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 1521–1530, New York, NY, USA, 2011. ACM.

S.C. Herring, L.A. Scheidt, S. Bonus, and E. Wright. Bridging the gap: A genre analysis of weblogs. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences, Big Island, Hawaii*. IEEE, 2004.

E. Horvitz, S. Dumais, and P. Koch. Learning predictive models of memory landmarks. In *Proceedings of the CogSci 2004: 26th Annual Meeting of the Cognitive Science Society*, Chicago, USA, August 2004.

K.-S. Hwang and S.-B. Cho. Modular bayesian networks for inferring landmarks on mobile daily life. In A. Sattar and B.-H. Kang, editors, *AI 2006: Advances in Artificial Intelligence*, volume 4304 of *Lecture Notes in Computer Science*, pages 929–933. Springer Berlin Heidelberg, 2006.

K.-S. Hwang and S.-B. Cho. Landmark detection from mobile life log using a modular bayesian network model. *Expert Systems with Applications*, 36(10):12065–12076, 2009.

A.K. Jain, M.N. Murty, and P.J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31 (3):264–323, 1999.

S.R. Jerke. PIMO-basierte Reminiszenz: Mentales Model, Vergessen und Erinnern für eine Familien-Fotokollektion. Bachelor's thesis, 2013. Kaiserslautern, University of Technology, Department of Computer Science.

Knockoutjs.com. Knockout: Simplify dynamic JavaScript UIs with the Model-View-View Model (MVVM) pattern. Software Framework, 2014. URL `http://www.knockoutjs.com/`. Last modification: 2014, accessed: September 25th, 2014.

R. L. Kullberg, W. J. Mitchell, and S. A. Benton. Dynamic timelines - visualizing historical information in three dimensions. Technical report, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1995. URL `http://dspace.mit.edu/bitstream/handle/1721.1/29098/34236359.pdf?sequence=1`.

J. Li and C. Cardie. Timeline generation: Tracking individuals on Twitter. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14, pages 643–652. International World Wide Web Conferences Steering Committee, 2014.

J. Liao, Z. Wang, Q. Cao, and H. Qi. Smart Diary: the narrative of your daily life. Technical report, University of Tennesse, Knoxville, TN, 2012. URL `http://web.eecs.utk.edu/~zwang32/publications/cscn-smartDiary.pdf`.

J. Liao, Z. Wang, L. Wan, Q. Cao, and H. Qi. Smart Diary: A smartphone-based framework for sensing, inferring and logging users' daily life. *Sensors Journal, IEEE*, PP(99), 2014.

H. Liu, J. Wang, and D. Xu. A simple and effective Concept Vector for WordNet semantic measurement. In *Advanced Computer Theory and Engineering (ICACTE), 2010 3rd International Conference on*, volume 2, pages 342–345, 2010.

A.M. Lund. Measuring usability with the USE questionnaire. *Usability interface*, 8(2):3–6, 2001.

A.M. Lund. Measuring usability with the USE questionnaire. Website, 2004. URL `http://www.stcsig.org/usability/newsletter/0110_measuring_with_use.html`. Last modification: 2004, accessed: September 25th, 2014.

Massachusetts Institute of Technology. SIMILE Widgets: Timeline. Application software, 2009. URL `http://www.simile-widgets.org/timeline/`. Last modification: 2009, accessed: September 25th, 2014.

H. Maus, O. Dobberkau, M. Wolters, and C. Niederée. ForgetIT Deliverable 9.1: Application Use Cases & Requirements Document. Technical report, ForgetIT Project, 2013a. URL `http://www.forgetit-project.eu/fileadmin/fm-dam/deliverables/ForgetIT_WP9_WP10_D9.1.pdf`.

H. Maus, S. Schwarz, and A. Dengel. Weaving personal knowledge spaces into office applications. In M. Fathi, editor, *Integration of Practice-Oriented Knowledge Technology: Trends and Prospectives*, pages 71–82. Springer Berlin Heidelberg, 2013b.

Merriam-Webster Dictionary. Diary. Website. URL `http://www.merriam-webster.com/dictionary/diary`. Accessed: September 25th, 2014.

Microsoft Corporation. Microsoft Research: Lifebrowser. Video, 2012. URL `http://research.microsoft.com/apps/video/default.aspx?id=159531`. Last modification: February 27th, 2012, accessed: September 25th, 2014.

Microsoft Corporation. Microsoft Research: Project Greenwich. Web application, 2014. URL `http://research.microsoft.com/en-us/projects/greenwich/`. Last modification: 2014, accessed: September 25th, 2014.

R.-E. Mohrmann, C. Cantauw, L. Volmer, B. Spies, S. Altemühle, and U. Rogier. Mein 18. November. Technical report, Volkskundliche Kommission für Westfalen, 2005. URL `http://www.lwl.org/LWL/Kultur/mein_18_November/`. Last modification: 2005, accessed: September 25th, 2014.

mSpace Project. Continuum: A timeline visualisation using space and time to scale and represent meaningful views at all levels of zoom. Technical report, School of Electronics and Computer Science, University of Southampton, 2007. URL `http://research.mspace.fm/projects/continuum`. Accessed: September 25th, 2014.

G. Nagy. State of the art in pattern recognition. *Proceedings of the IEEE*, 56(5):836–863, 1968.

B.A. Nardi, D.J. Schiano, and M. Gumbrecht. Blogging as social activity, or, would you let 900 million people read your diary? In *Proceedings of the 2004 ACM conference on Computer supported cooperative work*, pages 222–231. ACM, 2004.

C. Niederée. ForgetIT Brochure. Technical report, ForgetIT Project, 2013. URL `http://www.forgetit-project.eu/fileadmin/fm-dam/downloads/2013-05-24_forgetit_brochure.pdf`.

B. Paech and K. Kohler. Task-driven requirements in object-oriented development. In *Perspectives on Software Requirements*, pages 45–67. Springer, 2004.

O. Papadopoulou, V. Mezaris, V. Solachidis, A. Ioannidou, B.B. Eldesouky, H. Maus, and M.A. Greenwood. ForgetIT Deliverable 4.2: Information Analysis, Consolidation and Concentration Techniques, and Evaluation. Technical report, ForgetIT Project, 2014. URL `http://www.forgetit-project.eu/fileadmin/fm-dam/deliverables/ForgetIT_WP4_D4.2.pdf`.

C. Plaisant, B. Milash, A. Rose, S. Widoff, and B. Shneiderman. LifeLines: Visualizing personal histories. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '96, pages 221–227, New York, NY, USA, 1996. ACM.

C. Plaisant, R. Mushlin, A. Snyder, J. Li, D. Heller, and B. Shneiderman. LifeLines: using visualization to enhance navigation and analysis of patient records. In *Proceedings of the AMIA Symposium*, pages 76–80. American Medical Informatics Association, 1998.

Programming Sunrise. Smart Diary Suite 4.8.0. Application software. URL `http://www.sdiary.com/`. Last modification: July 31st, 2013, accessed: September 25th, 2014.

R. Qian. Timeline: Understanding important events in people's lives. Website, 2014. URL `http://blogs.bing.com/search/2014/02/21/timeline-understanding-important-events-in-peoples-lives/`. Last modification: February, 21st, 2014, accessed: September 25th, 2014.

M. Ringel, E. Cutrell, S. Dumais, and E. Horvitz. Milestones in time: The value of landmarks in retrieving information from personal stores. In *Proc. Interact*, volume 2003, pages 184–191, 2003.

A. Sabharwal. Google official blog: Google+ stories and movies: memories made easier. Website, 2014. URL `http://googleblog.blogspot.de/2014/05/google-stories-and-movies-memories-made.html`. Last modification: May 20th, 2014, accessed: September 25th, 2014.

G. Salton, A. Wong, and C.-S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

M. Sauer. Die Zeitleiste. In H.-J. Pandel and G. Schneider, editors, *Handbuch Medien im Geschichtsunterricht*. Wochenschau Verlag, 3rd edition, 2005.

L. Sauermann. *The Gnowsis Semantic Desktop approach to Personal Information Management*. PhD thesis, Kaiserslautern, University of Technology, Department of Computer Science, 2009. URL `http://www.dfki.uni-kl.de/~sauermann/papers/Sauermann2009phd.pdf`.

L. Sauermann, A. Bernardi, and A. Dengel. Overview and outlook on the semantic desktop. In *Proceedings of the 1st Workshop on The Semantic Desktop at ISWC*, 2005.

L. Sauermann, L. Van Elst, and A. Dengel. PIMO – a framework for representing personal information models. *Proceedings of I-Semantics*, 7:270–277, 2007.

J.-G. Schettler-Köhler. PimoCloud: a cloud-based, versioning document storage as a service for the PIMO. Master's thesis, Kaiserslautern, University of Technology, Department of Computer Science, 2014.

M. Schubert and A. Zimek. Knowledge Discovery in Databases I – Chapter 5: Clustering (lecture slides). Technical report, Ludwig-Maximilians-Universität München, Fakultät für Mathematik, Informatik und Statistik, Institut für Informatik, Lehr und Forschungseinheit für Datenbanksysteme, 2011. URL `http://www.dbs.ifi.lmu.de/Lehre/KDD/WS1011/skript/kdd-5-clustering4.pdf`. Last modification: June 28th, 2011, accessed: July 30th, 2014.

S. Schwarz, H. Maus, M. Kiesel, and L. Sauermann. Wissensarbeit am Desktop. In A. Dengel, editor, *Semantische Technologien: Grundlagen – Konzepte – Anwendungen*, pages 315–368. Spektrum Akademischer Verlag, 2012.

I. Seiffge-Krenke. "Dear Kitty, you asked me...": imaginary companions and real friends in adolescence. *Praxis der Kinderpsychologie und Kinderpsychiatrie*, 50(1):1–15, 2001.

B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE, 1996.

SIMILE Widgets Community. SIMILE Widgets: Timeline Documentation Wiki. Website, 2010. URL `http://simile-widgets.org/wiki/Timeline`. Last modification: 2010, accessed: September 25th, 2014.

P.H.A. Sneath and R.R. Sokal. *Numerical taxonomy. The principles and practice of numerical classification.* Freeman, London, UK, 1973.

Y. Sumi, R. Sakamoto, K. Nakao, and K. Mase. ComicDiary: Representing individual experiences in a comics style. In G. Borriello and L.E. Holmquist, editors, *UbiComp 2002: Ubiquitous Computing*, volume 2498 of *Lecture Notes in Computer Science*, pages 16–32. Springer Berlin Heidelberg, 2002.

Web Rater. How good is your site? Website, 2014. URL `http://www.web-rater.com/`. Last modification: 2014, accessed: June 16th, 2014.

A. Webster. Microsoft's Lifebrowser learns what's important so you can browse through personal milestones. Website, 2012. URL `http://www.theverge.com/2012/3/16/2878127/microsoft-research-lifebrowser-eric-horvitz-private-data-mining`. Last modification: March 16th, 2012, accessed: September 25th, 2014.

Wikipedia Encyclopedia. Facebook. Website, 2014a. URL `http://en.wikipedia.org/wiki/Facebook`. Last modification: September 25th, 2014, accessed: September 25th, 2014.

Wikipedia Encyclopedia. Samuel Pepys. Website, 2014b. URL `http://en.wikipedia.org/wiki/Samuel_Pepys`. Last modification: September 20th, 2014, accessed: September 25th, 2014.

Wikipedia Encyclopedia. Stemming. Website, 2014c. URL http://en.wikipedia.org/wiki/Stemming. Last modification: September 2nd, 2014, accessed: September 25th, 2014.

Wikipedia Encyclopedia. Stop words. Website, 2014d. URL http://en.wikipedia.org/wiki/Stop_words. Last modification: September 4th, 2014, accessed: September 25th, 2014.

Wikipedia Encyclopedia. Stream graph. Website, 2014e. URL http://en.wikipedia.org/wiki/Stream_graph. Last modification: May 7th, 2014, accessed: September 25th, 2014.

Wikipedia Encyclopedia. URI. Website, 2014f. URL http://en.wikipedia.org/wiki/Uniform_resource_identifier. Last modification: September 11th, 2014, accessed: September 25th, 2014.

List of Figures

List of Tables

List of Abbreviations

♀	female
♂	male
AI	artificial intelligence
API	application programming interface
app	application
avg.	average
blog	web log
CPU	central processing unit
CW	calendar week
DFKI	Deutsches Forschungszentrum für Künstliche Intelligenz / German Research Center for Artificial Intelligence
etc.	et cetera
GIMO	group information model
gr.	group
HTML	hypertext markup language
min	minute(s)
MVC	Model-View-Controller (architectural pattern)
MVVM	Model-View-ViewModel, special version of the MVC pattern
PIM	personal information management
PIMO	personal information model
PIMORE	PIMO Reminiscence (project by DFKI)
RDF	resource description framework
SEED	Semantic Editor (project by DFKI)
SIS	Stuff I've Seen (project by Dumais et al. (2003))
TORE	Task and Object-oriented Requirements Engineering (framework)
UC	use case
UI	user interface
URI	uniform resource identifier
URL	uniform resource locator

Abkürzungsverzeichnis / List of German Abbreviations

App	kurz für Applikation
bspw.	beispielsweise
bzw.	beziehungsweise
DFKI	Deutsches Forschungszentrum für Künstliche Intelligenz
etc.	et cetera
ggf.	gegebenenfalls
inkl.	inklusive
KI	Künstliche Intelligenz
PIMO	Persönliches Informationsmodell
sog.	sogenannte
usw.	und so weiter

List of Notations

a_{CC}	centroid check auto mode factor
b_R	rarity bonus that may be assigned to an entry E
c_j	concept j or j-th element of a concept vector (see also: d_{ij})
d_i	term- or concept vector i
d_{ij}	j-th element of a term- or concept vector i (see also: c_j)
D_i	document i (definitions by Salton et al. only)
d_k	dissimilarity (definitions by Jain et al. only)
$d(x, y)$	distance between two information items x and y
$d_{SL}(X, Y)$	distance between two clusters of information items X and Y
E	diary entry
f_C	number of the concept context's top most concepts
f_L	factor to determine the threshold for a concept to enter $prom_S(E)$
f'_L	factor to determine the threshold for a concept to enter $prom_L(E)$
f_{TS}	factor to determine t_S^{avg} and t_S^{frac}
$H(E)$	high priority things factor of an entry E
I_i	information item i
$I(E)$	annotation intensity of an entry E
$imp(E)$	(overall) importance value of an entry E
m_C	maximum weight associated with an annotated concept
m_I	maximum weight associated with information item
m_T	maximum of m_C and m_T
$M(E)$	rich media factor of an entry E
n	number of (desired) diary entries
$prom_S(E)$	prominent concept set of an entry E
$prom_L(E)$	prominent concept list of an entry E
R_i	PIMO resource
$R(E)$	rarity factor of an entry E
s_L	label similarity
s_T	text body similarity
s_C	similarity in annotated concepts
$sim_{SL}(X, Y)$	(overall) similarity of two clusters of information items X and Y
$sim(x, y)$	(overall) similarity of two information items x and y
t_{CC}	threshold to determine whether the centroid check should be applied
T_j	an index term (definitions by Salton et al. only)
t_L	threshold for associating an entry E with a split label
t_R	threshold for associating an entry E with a rarity bonus
t_S	similarity threshold (in general)

List of Notations (cont'd)

t_S^{avg}	similarity threshold given by the average similarity of all items
t_S^{frac}	similarity threshold given by a fraction of the maximum similarity of all items
t_S^{const}	similarity threshold given by a constant
V, W	term- or concept vector
w_L	weight of label similarity
w_T	weight of text body similarity
w_C	weight of similarity in annotated concepts
x, y	single information items
X, Y	clusters of information items

Appendices

A. Survey about Social Media Usage and Personal Reminiscence

In order to find out more about the way people *retrospect* on their lives and the way they *preserve* (or *"document"*) memories and events, we carried out a survey in the early phase of this diploma thesis project. To better evaluate the results afterwards, we also asked several questions about the participants' *social media usage*. In Chapter 1 we already gave the four most remarkable insights we concluded from the survey. This section contains additional details about the concrete setting, the questionnaire and the given answers.

A.1. Setting

We conducted this survey with two different groups. The first group consisted of 17 persons aged between 14 and 66 years. For details please see Table A.1. We also tried to cover a broad bandwidth of jobs, habits, educational achievements, etc.

Age	0-19	20-24	25-29	30-34	35-39	40-44	45-49	50-54	55-59	60-64	65-69
♀	0	2	0	2	3	0	1	0	1	0	1
♂	1	1	1	1	2	0	0	1	0	0	0

Table A.1: Demographic data of the first group of participants. It consisted of 17 persons aged between 14 and 66 years, 59% were female and 41% male.

The second group is the four-headed DFKI-external testing group called *PANiC* that we already mentioned in Chapter 7.

Comparing both groups, we can see that the results are quite similar, although there might be a slight bias towards technophilia in the second group, most likely related to their educational and professional background.

A.2. Questionary and Answers

This section contains the actual questionary (Figures A.1 and A.2) as well as the different answers in condensed and detailed version (Tables A.2 to A.7).

Please note that we omitted printing the detailed answers to open questions, since they only differ from the condensed ones in item 8 that was misunderstood and thus wrongly answered by some participants (they wrote *what* they actually did during that time, instead of *how* they would retrospect on it). So, in order to protect their privacy we omitted printing these answers here.

	statements about usage					reasons (if no or low usage)			
	no	a few times per year	a few times per month	a few times per week	daily	no interest	too time-consuming	concerns about privacy/security	other (please explain)
1) Do you post entries in social networks like *Facebook* or *Google+*?	☐	☐	☐	☐	☐	☐	☐	☐	☐
2) Do you post entries using microblogging services like *Twitter*?	☐	☐	☐	☐	☐	☐	☐	☐	☐
3) Do you *blog*?	☐	☐	☐	☐	☐	☐	☐	☐	☐
4) Do you use a *diary app*?	☐	☐	☐	☐	☐	☐	☐	☐	☐
5) Do you keep a *(classical) diary*?	☐	☐	☐	☐	☐	☐	☐	☐	☐

6) Have you kept a diary in the past (e.g. while being a teenager)?				
☐ no	☐ a few times per year	☐ a few times per month	☐ a few times per week	☐ daily

7) How do you "document" things in your life? (in terms of keeping memories, sharing memories with others, etc.)				
☐ notes (paper)	☐ notes (digital)	☐ photos (paper)	☐ photos (digital)	☐ videos
☐ other (please explain)				

Figure A.1: Questionary of our survey (page 1/2)

8) How would you retrospect on last week / last month / last year?

9) How time-consuming is the retrospection mentioned in the previous question for you?				
☐ not at all	☐ little	☐ medium	☐ considerable	☐ extraordinary

10) If somebody gives you an arbitrary chosen time period, e.g. September 2008 or spring 2003, could you name five things you were concerned with the most in that period after 1 min of thinking time?			
☐ no	☐ rather no	☐ rather yes	☐ yes

11) Are you interested in an app that eases retrospection?				
☐ no	☐ rather no	☐ maybe	☐ rather yes	☐ yes

Figure A.2: Questionary of our survey (page 2/2)

Question	Answer	Gr. 1	Gr. 2	all
1. Do you post entries in social networks like *Facebook* or *Google+*?	1. no	59%	25%	52%
	2. a few times per year	18%	50%	24%
	3. a few times per month	18%	25%	19%
	4. a few times per week	6%	0%	5%
	5. daily	0%	0%	0%
	6. no interest	38%	25%	35%
	7. too time-consuming	6%	25%	10%
	8. privacy/security concerns	50%	50%	50%
	9. other	0%	50%	10%
2. Do you post entries using microblogging services like *Twitter*?	1. no	100%	100%	100%
	2. a few times per year	0%	0%	0%
	3. a few times per month	0%	0%	0%
	4. a few times per week	0%	0%	0%
	5. daily	0%	0%	0%
	6. no interest	53%	75%	57%
	7. too time-consuming	0%	0%	0%
	8. privacy/security concerns	41%	0%	33%
	9. other	0%	50%	10%
3. Do you *blog*?	1. no	100%	100%	100%
	2. a few times per year	0%	0%	0%
	3. a few times per month	0%	0%	0%
	4. a few times per week	0%	0%	0%
	5. daily	0%	0%	0%
	6. no interest	71%	50%	67%
	7. too time-consuming	6%	75%	19%
	8. privacy/security concerns	0%	0%	0%
	9. other	0%	25%	5%
4. Do you use a *diary app*?	1. no	100%	100%	100%
	2. a few times per year	0%	0%	0%
	3. a few times per month	0%	0%	0%
	4. a few times per week	0%	0%	0%
	5. daily	0%	0%	0%
	6. no interest	71%	25%	62%
	7. too time-consuming	0%	25%	5%
	8. privacy/security concerns	6%	0%	5%
	9. other	0%	50%	10%

Table A.2: Condensed answers to closed questions of our survey (part 1/2)

Question	Answer	Gr. 1	Gr. 2	all
5. Do you keep a *(classical) diary*?	1. no	94%	100%	95%
	2. a few times per year	0%	0%	0%
	3. a few times per month	0%	0%	0%
	4. a few times per week	6%	0%	5%
	5. daily	0%	0%	0%
	6. no interest	69%	50%	65%
	7. too time-consuming	6%	50%	15%
	8. privacy/security concerns	0%	0%	0%
	9. other	0%	25%	0%
6. Have you kept a diary in the past?	1. no	65%	75%	67%
	2. a few times per year	18%	0%	14%
	3. a few times per month	6%	0%	5%
	4. a few times per week	12%	25%	14%
	5. daily	0%	0%	0%
7. How do you "document" things in your life?	1. notes (paper)	53%	50%	52%
	2. notes (digital)	29%	100%	43%
	3. photos (paper)	41%	25%	38%
	4. photos (digital)	82%	100%	86%
	5. videos	53%	50%	52%
	6. other	12%	0%	10%
9. How time-consuming is the retrospection mentioned in the previous question for you?	1. not at all	17%	0%	14%
	2. little	44%	50%	45%
	3. medium	39%	25%	36%
	4. considerable	6%	0%	5%
	5. extraordinary	0%	0%	0%
10. ... could you name five things you were concerned with the most ... ?	1. no	12%	50%	19%
	2. rather no	65%	50%	62%
	3. rather yes	24%	0%	19%
	4. yes	0%	0%	0%
11. Are you interested in an app that eases retrospection?	1. no	11%	0%	9%
	2. rather no	17%	0%	14%
	3. maybe	39%	25%	36%
	4. rather yes	17%	50%	36%
	5. yes	17%	25%	18%

Table A.3: Condensed answers to closed questions of our survey (part 2/2)

Participant	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Gender	♀	♀	♂	♀	♀	♂	♀	♀	♂	♀	♂	♀	♂	♀	♀	♂	♂
Age	66	59	54	48	38	37	35	35	35	33	33	32	27	24	23	20	14
Answer 1.1	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗							
Answer 1.2											✗	✗					✗
Answer 1.3													✗	✗	✗		
Answer 1.4																✗	
Answer 1.5																	
Answer 1.6	✗	✗	✗	✗				✗				✗					
Answer 1.7										✗							
Answer 1.8	✗		✗	✗	✗	✗	✗		✗		✗						
Answer 1.9																	
Answer 2.1	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Answer 2.2																	
Answer 2.3																	
Answer 2.4																	
Answer 2.5																	
Answer 2.6	✗	✗	✗	✗				✗		✗	✗	✗				✗	
Answer 2.7																	
Answer 2.8	✗		✗	✗	✗	✗	✗		✗								
Answer 2.9																	
Answer 3.1	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Answer 3.2																	
Answer 3.3																	
Answer 3.4																	
Answer 3.5																	
Answer 3.6	✗	✗	✗	✗	✗		✗	✗	✗	✗	✗	✗				✗	
Answer 3.7						✗											
Answer 3.8																	
Answer 3.9																	
Answer 4.1	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Answer 4.2																	
Answer 4.3																	
Answer 4.4																	
Answer 4.5																	
Answer 4.6	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗					
Answer 4.7																	
Answer 4.8			✗														
Answer 4.9																	
Answer 5.1	✗	✗		✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Answer 5.2																	
Answer 5.3																	
Answer 5.4			✗														
Answer 5.5																	
Answer 5.6	✗	✗		✗	✗	✗	✗	✗	✗	✗	✗	✗					
Answer 5.7																✗	
Answer 5.8																	
Answer 5.9																	
Answer 6.1	✗	✗			✗	✗		✗	✗	✗	✗				✗	✗	✗
Answer 6.2				✗			✗					✗					
Answer 6.3													✗				
Answer 6.4			✗											✗			
Answer 6.5																	
Answer 7.1	✗	✗	✗	✗	✗						✗	✗			✗	✗	
Answer 7.2			✗		✗				✗						✗	✗	
Answer 7.3		✗					✗	✗		✗	✗	✗		✗			
Answer 7.4	✗			✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗		✗
Answer 7.5		✗				✗	✗	✗		✗	✗		✗	✗			✗
Answer 7.6	✗		✗														
Answer 8	✗		✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗			✗
Answer 9.1		✗													✗	✗	
Answer 9.2	✗		✗			✗	✗			✗		✗	✗				✗
Answer 9.3	✗				✗			✗	✗		✗	✗		✗			
Answer 9.4																	
Answer 9.5																	
Answer 10.1	✗				✗												
Answer 10.2		✗		✗				✗	✗	✗		✗	✗	✗	✗	✗	✗
Answer 10.3			✗			✗	✗				✗						
Answer 10.4																	
Answer 11.1					✗					✗							
Answer 11.2		✗					✗		✗								
Answer 11.3	✗			✗		✗		✗			✗	✗			✗		
Answer 11.4	✗		✗											✗			
Answer 11.5													✗			✗	✗

Table A.4: Detailed answers of group 1 to closed questions of our survey

- Answer *i.j* denotes the *j*-th predefined answer to the *i*-th question, e.g. Answer 3.1 corresponds to the answer *"no"* to the question *"Do you blog?"*.
- Abbreviations: ✗: participant ticked this answer, ✓: statements were usable (some participants misunderstood question 8 and thus made unusable statements)

Participant	1	2	3	4
Gender	♂	♂	♀	♀
Answer 1.1				✗
Answer 1.2	✗		✗	
Answer 1.3		✗		
Answer 1.4				
Answer 1.5				
Answer 1.6				✗
Answer 1.7		✗		
Answer 1.8	✗		✗	
Answer 1.9		✗		✗
Answer 2.1	✗	✗	✗	✗
Answer 2.2				
Answer 2.3				
Answer 2.4				
Answer 2.5				
Answer 2.6	✗	✗		✗
Answer 2.7				
Answer 2.8				
Answer 2.9			✗	✗
Answer 3.1	✗	✗	✗	✗
Answer 3.2				
Answer 3.3				
Answer 3.4				
Answer 3.5				
Answer 3.6		✗		✗
Answer 3.7	✗		✗	✗
Answer 3.8				
Answer 3.9				✗
Answer 4.1	✗	✗	✗	✗
Answer 4.2				
Answer 4.3				
Answer 4.4				
Answer 4.5				
Answer 4.6		✗		
Answer 4.7	✗			
Answer 4.8				
Answer 4.9			✗	✗

Participant	1	2	3	4
Gender	♂	♂	♀	♀
Answer 5.1	✗	✗	✗	✗
Answer 5.2				
Answer 5.3				
Answer 5.4				
Answer 5.5				
Answer 5.6		✗		✗
Answer 5.7	✗			✗
Answer 5.8				
Answer 5.9			✗	
Answer 6.1	✗	✗		✗
Answer 6.2				
Answer 6.3				
Answer 6.4			✗	
Answer 6.5				
Answer 7.1	✗			✗
Answer 7.2	✗	✗	✗	✗
Answer 7.3				✗
Answer 7.4	✗	✗	✗	✗
Answer 7.5			✗	✗
Answer 7.6				
Answer 8	✓	✓		
Answer 9.1				
Answer 9.2		✗		✗
Answer 9.3			✗	
Answer 9.4	✗			
Answer 9.5				
Answer 10.1	✗	✗		
Answer 10.2			✗	✗
Answer 10.3				
Answer 10.4				
Answer 11.1				
Answer 11.2				
Answer 11.3			✗	
Answer 11.4	✗	✗		
Answer 11.5				✗

Table A.5: Detailed answers of group 2 to closed questions of our survey

- Answer $i.j$ denotes the j-th predefined answer to the i-th question, e.g. Answer 3.1 corresponds to the answer *"no"* to the question *"Do you blog?"*.
- Abbreviations: ✗: participant ticked this answer, ✓: statements were usable (some participants misunderstood question 8 and thus made unusable statements)

Answer	Statements	Gr. 1	Gr. 2	all
1.9	concerns about future employers	0%	50%	50%
	no benefit	0%	50%	50%
2.9	concerns about future employers	0%	50%	50%
	no time	0%	50%	50%
3.9	concerns about future employers	0%	100%	100%
4.9	no diary app known	0%	50%	50%
	not needed yet	0%	50%	50%
5.9	not necessary	0%	100%	100%
7.6	calendar	100%	0%	100%
	folder system on the computer	50%	0%	50%
	mails	50%	0%	50%
8	activity reports	7%	0%	6%
	archives (e.g. newspapers etc.)	7%	0%	6%
	calendar	29%	50%	31%
	certificates (of employment)	7%	50%	13%
	conversations with others	7%	0%	6%
	diary	7%	0%	6%
	images / photos	57%	50%	56%
	posts in social networks	0%	50%	6%
	memory	57%	100%	63%
	videos	43%	0%	38%

Table A.6: Condensed answers to open questions of our survey.

- The percentages reflect the amount of participants that actually answered the question. Thus, a value of 100% does not mean *all participants* but *all participants that answered this particular question* (which could possibly be just a single one; please additionally see Tables A.2 and A.3 for this matter).
- Answer *i.j* denotes the the *j*-th predefined answer to the *i*-th question, which in this case always reads as *"other (please explain)"*.

Question / Medium	Answer 6 (no interest)	Answer 7 (too time-consuming)	Answer 8 (concerns about privacy/security)	Answer 9 (other reasons)	Sum
1. social networks	7	2	10	2	21
2. microblogging services	12	1	7	2	22
3. blog	14	4	0	1	19
4. diary app	13	1	1	2	17
5. (classical) diary	13	3	0	1	17
Sum	59	11	18	8	96
	61%	11%	19%	8%	100%

Table A.7: Reasons for no or low usage of social media (both groups included)

B. User Experience Evaluation

B.1. Detailed Answers to Open Questions

In addition to the 20 closed questions of the evaluation, the participants were asked in a last (open) question whether there is any kind of feedback about this diary app – positive or negative – that they would like to express. This section contains the detailed and unaltered answers. Since they are in German we provided a translated summary in Chapter 7.

Participant 1 (♂):

- Ich kenne kein anderes Programm, das Vergleichbares leistet, von daher sehr interessant und ein weiteres „Alleinstellungsmerkmal“ der Pimo
- Ich war positiv überrascht von der intuitiven Bedienung und der Qualität der Ergebnisse. Die Nutzung hat mir viel Spaß gemacht
- Als Verbesserung würde ich mir wünschen, längere Ladezeiten (bspw. bei Jahresauswertung, ca. 44 Sekunden) mit einer Art „Fortschrittsanzeige“ zu überbrücken
- Die Funktionsweise der Detailsettings dürfte nicht jedem User direkt klar sein, hier wäre eine Hilfeanzeige/Beschreibung beim „Hovern“ mit der Maus z.B. über „Show clustercomp“, etc. sinnvoll, oder generell ein Button, der zu einer Hilfedatei führt.

Participant 2 (♂):

- Es ist sehr innovativ, sowas habe ich noch nie gesehen.
- Es ist nicht mit einem „echten“ Diary vergleichbar. Bei deinem Diary werden nämlich (wichtige) Dinge aufgelistet, die ich manchmal nicht in mein echtes Diary schreiben würde (Nur so reines Bauchgefühl, da ich kein echtes Diary schreibe).

Participant 3 (♀):

- Interesssantes feature, macht Spass sich anzugucken was man so gemacht hat (auch wenn es nur einige Monate her ist).
- Ergänzt die Pimo sehr schön, und hilft einen Überblick zu behalten (was habe ich wann gemacht)
- Dauert lange bis neue Icons übernommen werden.
- Automatisches aktualisieren wäre schön.

Participant 4 (♀): -unanswered-

B.2. Questionary

This section contains the actual questionary (Figures B.1 and B.2).

Usefulness								
1.) It helps me be more effective.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree
2.) It helps me be more productive.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree
3.) It is useful.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree

Ease of Use								
4.) It is easy to use.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree
5.) It is user friendly.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree
6.) It requires the fewest steps possible to accomplish what I want to do with it.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree

Ease of Learning								
7.) I learned to use it quickly.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree
8.) I easily remember how to use it.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree
9.) It is easy to learn to use it.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree

Satisfaction								
10.) I am satisfied with it.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree
11.) I would recommend it to a friend.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree
12.) It is fun to use.								
disagree	strongly ☐	quite ☐	slightly ☐	neither ☐	slightly ☐	quite ☐	strongly ☐	agree

Figure B.1: Questionary of user experience evaluation (page 1/2)

Core Features								
13.) The way information items are clustered to diary entries makes sense to me.								
disagree	strongly	quite	slightly	neither	slightly	quite	strongly	agree
	☐	☐	☐	☐	☐	☐	☐	
14.) The labels (i.e. headlines) of diary entries are chosen meaningfully.								
disagree	strongly	quite	slightly	neither	slightly	quite	strongly	agree
	☐	☐	☐	☐	☐	☐	☐	
15.) The text bodies of diary entries provide good summaries of the information items they refer to.								
disagree	strongly	quite	slightly	neither	slightly	quite	strongly	agree
	☐	☐	☐	☐	☐	☐	☐	
16.) If the number of desired diary entries is limited the most important ones are actually chosen.								
disagree	strongly	quite	slightly	neither	slightly	quite	strongly	agree
	☐	☐	☐	☐	☐	☐	☐	
17.) By zooming in and out of time periods I am able to find my desired level of details.								
disagree	strongly	quite	slightly	neither	slightly	quite	strongly	agree
	☐	☐	☐	☐	☐	☐	☐	
18.) Manually including or excluding concepts shifts the diary's emphases as expected.								
disagree	strongly	quite	slightly	neither	slightly	quite	strongly	agree
	☐	☐	☐	☐	☐	☐	☐	
19.) The app allows an appropriate and satisfactory retrospection on those parts of my life that are reflected by my PIMO.								
disagree	strongly	quite	slightly	neither	slightly	quite	strongly	agree
	☐	☐	☐	☐	☐	☐	☐	
20.) The concept context provides a good impression, i.e. a quick overview, of those things (reflected by my PIMO) that concerned me the most in the chosen period.								
disagree	strongly	quite	slightly	neither	slightly	quite	strongly	agree
	☐	☐	☐	☐	☐	☐	☐	

Comments
Please feel free to express all kinds of feedback here – positive or negative – e.g. **- Is there anything in or about this app you found very satisfying or unsatisfying?** **- What should be added or improved in future versions – except for a better response time, which is already scheduled for improvement ;-)?** **Thank you for your efforts and participation!**

Figure B.2: Questionary of user experience evaluation (page 2/2)

C. User Interface Mock-ups

For the sake of completeness we present our user interface mock-ups in this section.

Design Iteration 1 Figure C.1 shows our first UI mock-up, which covers twelve weeks of the author's studies in 2011.

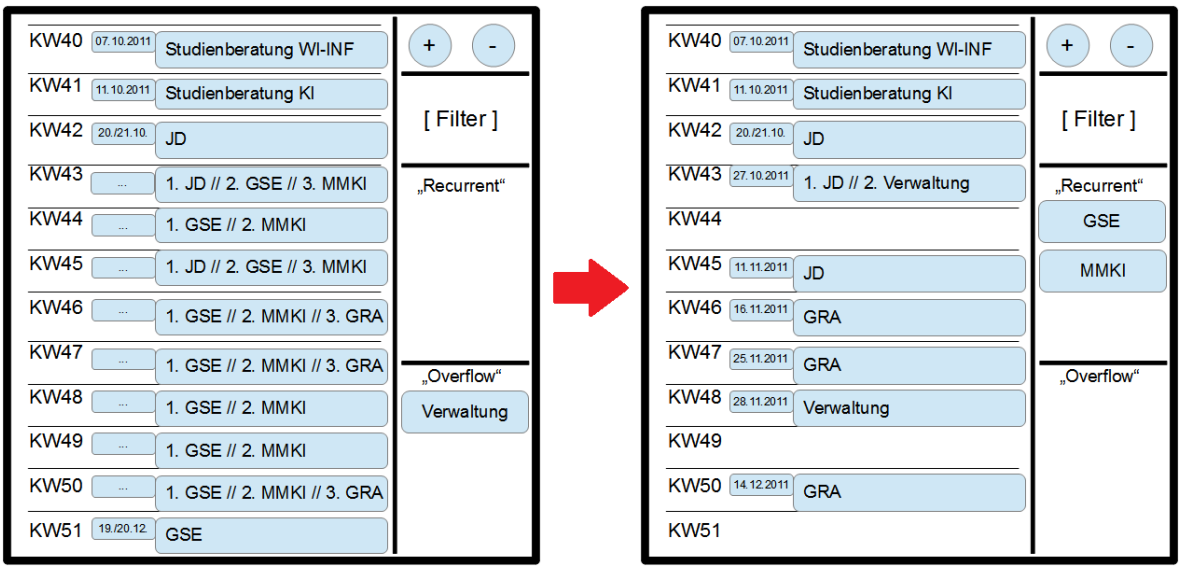

Figure C.1: First design iteration UI mock-up

It was created in a time when we thought about the aspect of diversity (see Chapter 4.6.1). On the right-hand side we see a diary having low diversity, since all the lectures (GSE, MMKI and GRA) are mentioned over an over. Since the diary is already too crowded, a concept called "Verwaltung" cannot be placed into the entries and is left in a "overflow" section. To solve this, we moved the very frequently mentioned topics to a kind of "recurrent" section, thus increasing the diary's diversity and resolving the crowded areas (see left-hand side). As a consequence, the formerly missing concept can then be incorporated into the dairy and therefore leaves the "overflow" section.

The "recurrent" and "overflow" sections were later merged in favor of a general "concept context" (see Chapter 6.2.5). In addition, the functionality of the final diary app differs from these early ideas.

Design Iteration 2 The UI mock-ups shown in Figures C.2 and C.3 are from the second design iteration. Since they depict use cases quite similar to those described in Chapter 4.4, we omit discussing them in more detail here. Again, please note that the functionality of the final diary app differs from these early ideas – although less than in design iteration 1.

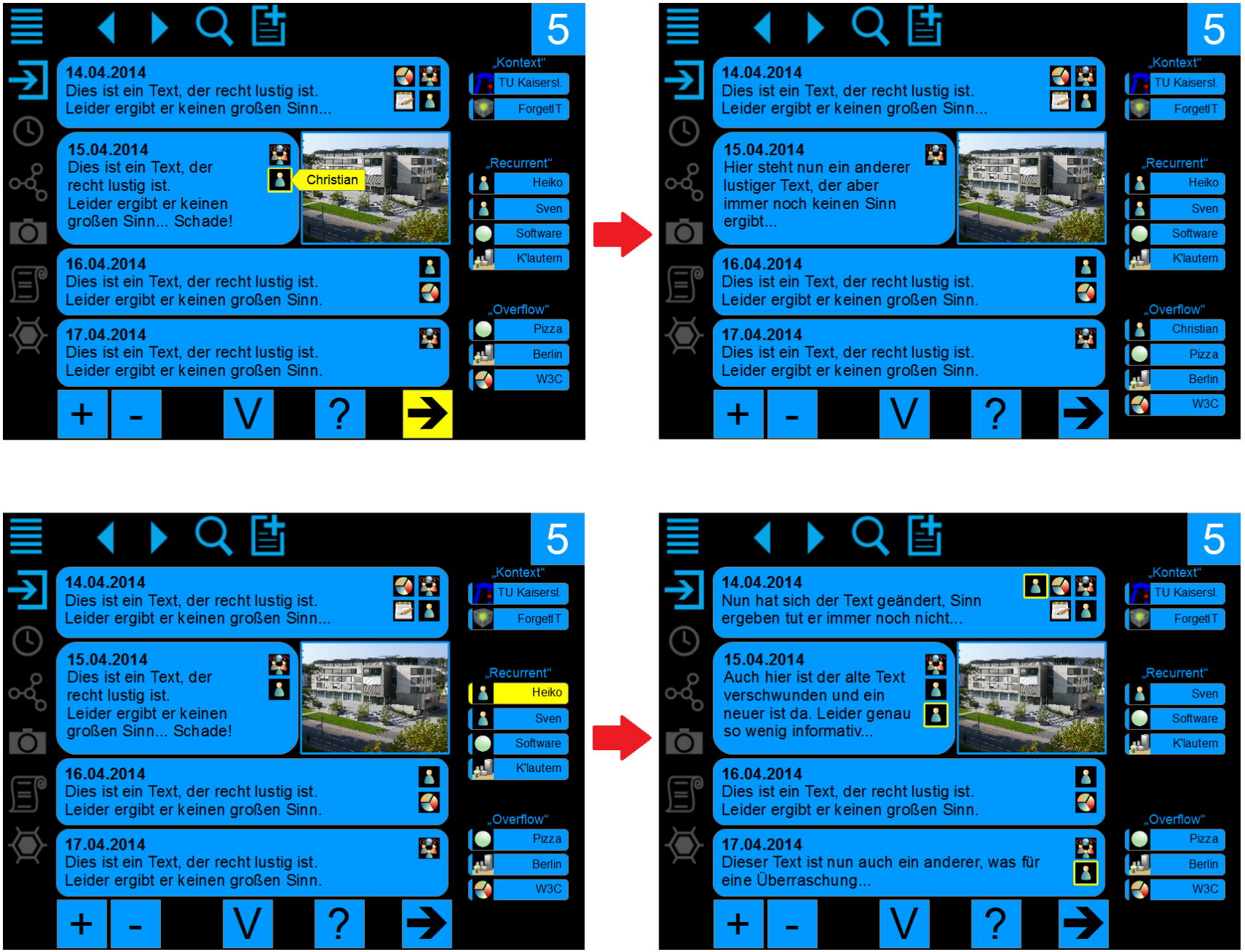

Figure C.2: UI mock-ups of manual concept exclusion (top) and inclusion (bottom)

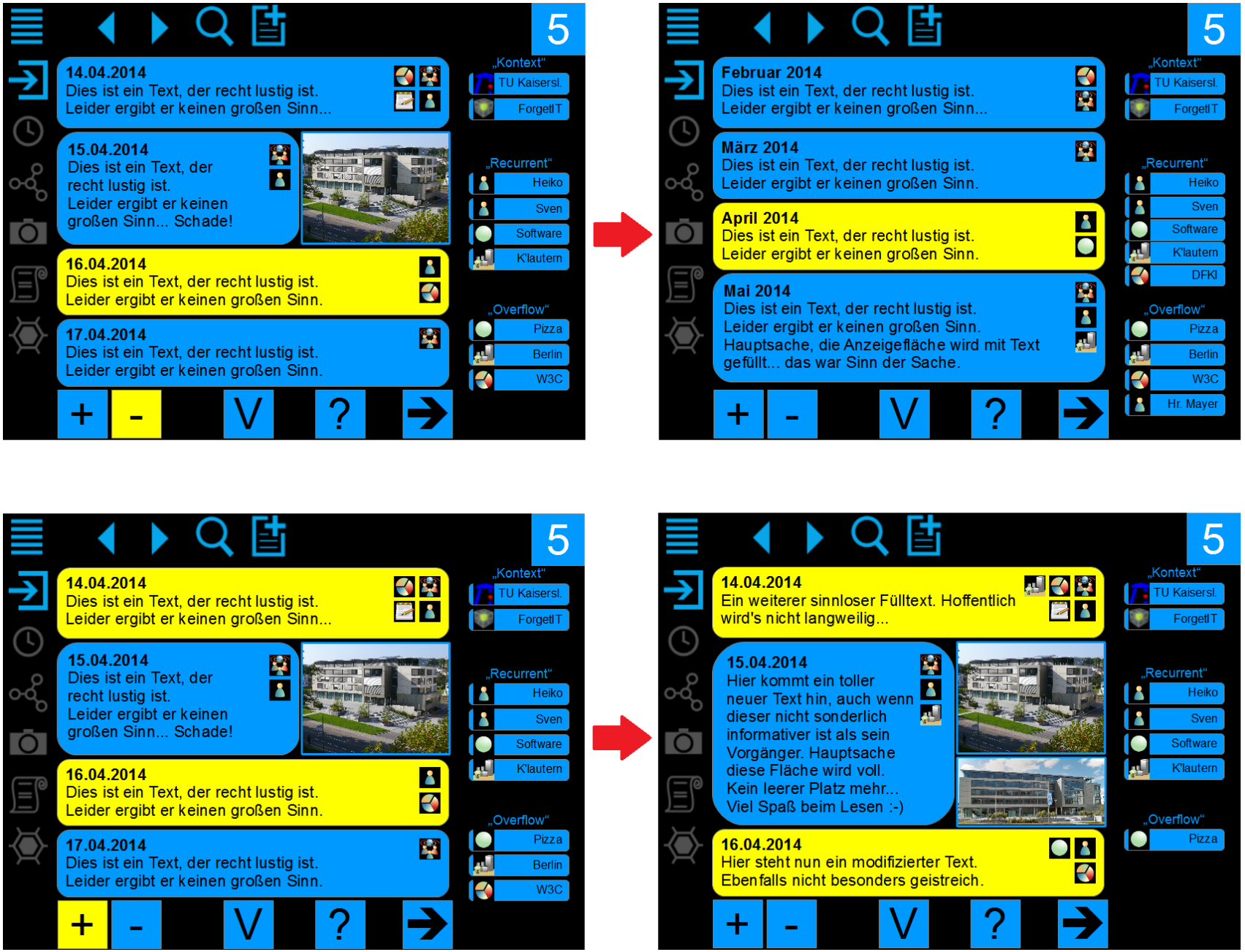

Figure C.3: UI mock-ups of zooming in (bottom) and out (top)

D. Task and Object-oriented Requirements Engineering (TORE)

In (Adam et al., 2009) a summary of *TORE* is provided, whereas details can be found in (Paech and Kohler, 2004).

> *"TORE is a decision framework that encapsulates 18 decisions on four different levels of abstraction that have typically to be made during requirements engineering for interactive (information) systems (see Figure D.1). The benefit of thinking in these decisions is that it can serve as a conceptual model independent of concretely used processes or notations allowing a high applicability in many different contexts."*
>
> (Adam et al., 2009, p. 268)

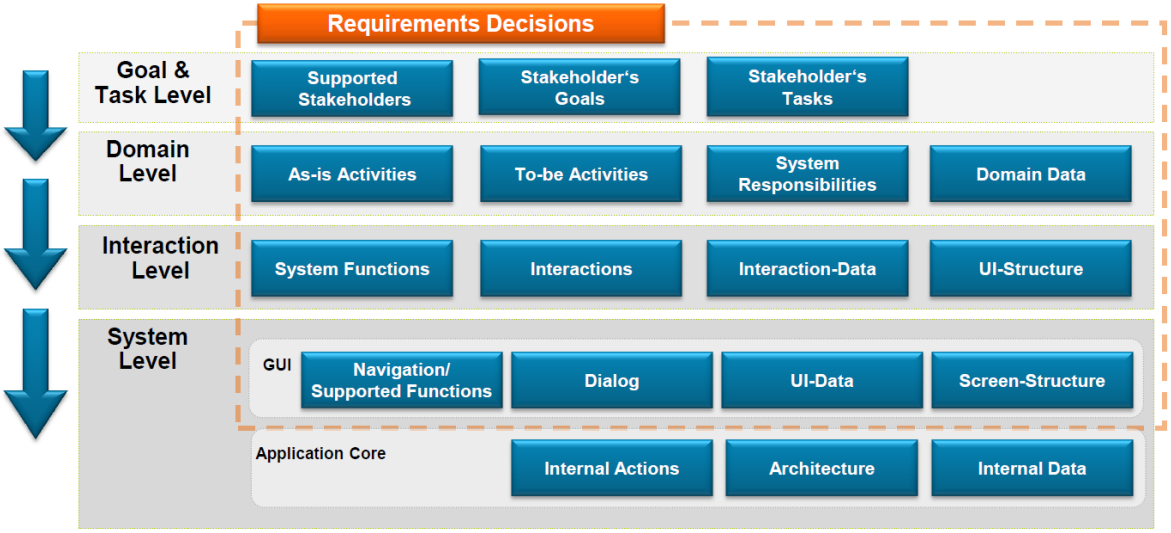

Figure D.1: Decision points in the TORE framework (Adam et al., 2009, p. 268)

Adam et al. applied the *TORE* framework as the conceptual basis for their requirements engineering activities in more than ten projects with partners from research, industry, and the public domain and found it to be "highly beneficial even in systems that do not seem to be 'traditional interactive systems' at a first glance" (Adam et al., 2009, p. 267).

In particular, they name the following benefits (Adam et al., 2009, pp. 271):

- support for a systematic way of thinking,
- increased understanding,
- systematic functional decomposition,
- separation of concerns,
- integration of usability / UI aspects,
- supports for requirements engineering education and technology transfer.

E. Digital Files

The disc on this page contains:

- this document in digital form,
- used literature and archived web pages,
- additional files of the appendix (concerning survey and user experience evaluation),
- meeting slides created by the author,
- the created software (final version and experimental code).