Projektauftrag
Gruppe 3
188.405 ASE
WS 2009/10

Andrea Füresz, Christian Hattinger,
Dominik Hofer, Alex Leutgöb,
Manuel Maly, Michael Petritsch

Inhaltsverzeichnis

Projektbezeichnung & Projektteam

Das Projekt, das von Gruppe 3 durchgeführt wird, trägt den Arbeitstitel "TM", oder "Task-Manager for Mac OSX, with Main Focus on Cloud-based Services Integration".
Der Projekttitel für den Enduser wird "WellDone" lauten.
Da sich TM als Abkürzung praktisch erweist, wird während der Entwicklungszeit dieser Arbeitstitel für das Projekt verwendet.
Das Projektteam setzt sich zusammen aus folgenden Mitgliedern:

Manuel Maly, 0526121, Teamkoordinator, User Interface Stellvertreter
Michael Petritsch, 0126861, Dokumentbeauftragter, Teamkoordinator Stellvertreter
Christian Hattinger, 0427438, Software Tester
Dominik Hofer, 0626629, Software Architekt Stellvertreter, Software Tester Stellvertreter
Andrea Füresz, 0525548, User Iterface Beauftragte, Dokumentbeauftragte Stellvertreterin
Alex Leutgöb, 0625881, Software Architekt

Ausgangssituation

Immer mehr administrative Hilfsprogramme werden in cloud-based Services ausgelagert. So existiert für die meisten Hilfsprogramme, die früher nur für Desktop-Computer und teils Palmtops und Handys verfügbar waren, ein Cloud-Pendant. Der Service "toodledo" lagert die Speicherung und Bearbeitung von Todo-Listen in eine Web-Application aus, die durch eine umfangreiche Web-Service-API ergänzt wird, welche zumindest für nicht-kommerzielle Projekte in der Benutzung frei ist.

"toodledo" verfügt zusätzlich zum Online-Modus über keinen Offline-Modus, der dem Benutzer auch Zugriff auf die Todo-Listen geben würde, wenn keine Verbindung zum Server besteht.

Es ist leicht zu sehen, dass die Webservice-API von "toodledo" nicht ohne Grund existiert. Zusätzlich zur besseren Offline-Nutzung empfielt sich eine Synchronisation der Daten mit dem eigenen Computer aus Sicherheitsgründen - die exklusive Verwaltung von Daten bei einem fremden Anbieter erweist sich als riskant, da der Benutzer keinerlei Einsicht in Backup-Mechanismen des Systems hat.

Es existiert ein Dashboard-Widget für OSX, sowie eine Adobe Air Application, die jeweils oberflächliche Übersichten und rudimentäre Bearbeitungsfunktionen der zu erledigenden Tasks bietet.
OSX-Desktop-Tools mit umfangreicher Verwaltung, Schnellzugriff, Notifikation bei Ablaufen eines Todo-Tasks, etc. in Verbindung mit "toodledo"-Integration sind der Projektgruppe zur Zeit der Erstellung dieses Dokuments nicht bekannt.

Projektbeschreibung

Es wird ein Werkzeug benötigt welches Benutzern ermöglicht, den umfangreichen Service von "toodledo" in einer Desktop-Umgebung komfortabel und ohne Verlust von Funktionalität zu nutzen. Das Werkzeug soll keineswegs von "toodledo" abhängig sein, weder in funktioneller Weise noch in Hinsicht auf die Usability. Zudem ist eine Synchronisation mit anderen Cloud-Services denkbar (siehe Nice-to-have-Features). Es soll sich so gut wie möglich in die Funktionen des Betriebssystems (OSX) integrieren und den Benutzer bei der Verwaltung von Tasks, und die Erinnerung an ebendiese unterstützen. Der Benutzer soll zudem auf verschiedenen Wegen (Email, iCal-Termin, iCal-Todo,...) vom Ablaufen eines Todo-Tasks notifiziert werden können - dies soll immer funktionieren, wenn das Betriebssystem im betriebsfähigen Zustand ist. Der Projektgruppe ist bewusst, dass sowohl iCal als auch "toodledo" über Notifikationen für abgelaufene Tasks verfügen, es ist jedoch trotzdem eine umfassende TM-interne Notifikationsfunktion vonnöten, da TM auch als eigenständige Applikation bestehen können soll. Zudem dient eine zentrale Steuerung für Notifikationen der besseren Usability.
Das Projektteam von "TM" soll diese Anwendung von Grund auf und vollständig umsetzen.

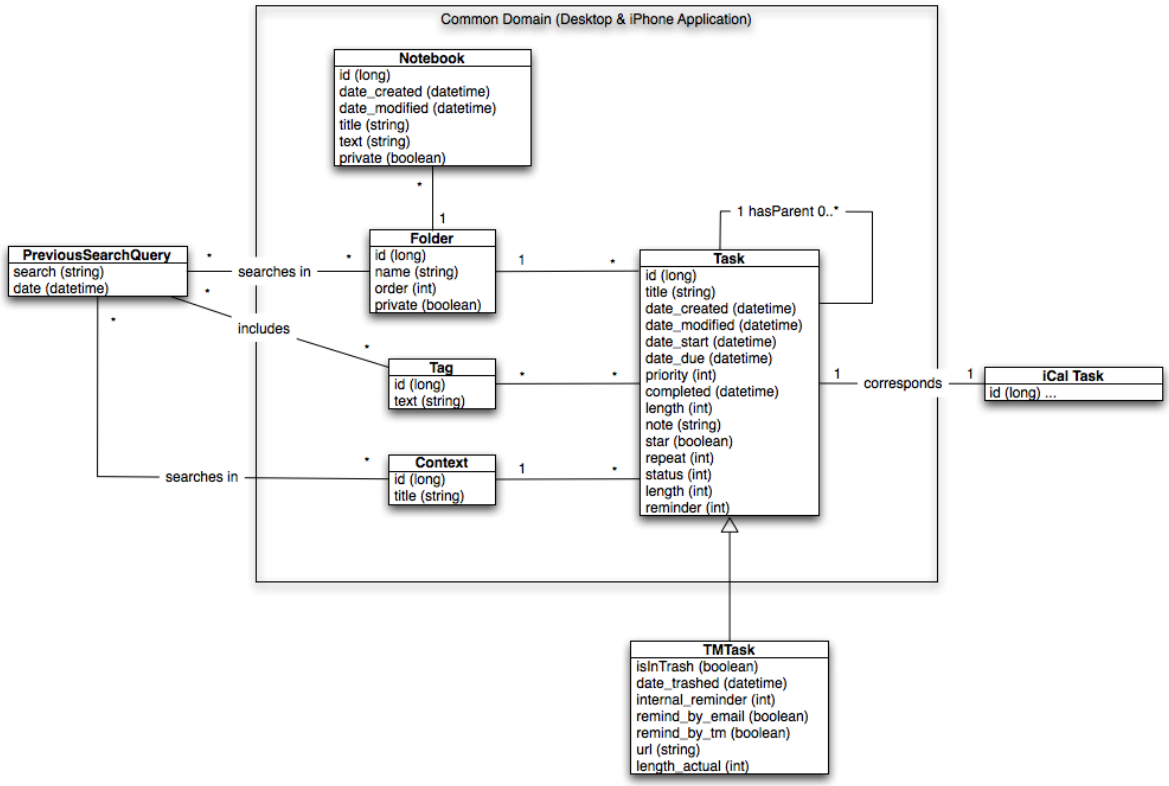
Zielgruppen

Benutzer des Cloud-Services "toodledo", die OSX auf ihrem Computer installiert haben, sollen von "TM" durch die umfangreiche Synchronisation mit verschiedenen Services profitieren. Weiters fehlt bisher eine Offline-Backup-Lösung für Benutzer unter OSX, was zu Unsicherheit bezüglich der Datensicherheit führt.

Dazu zählen insbesondere **Mitglieder von Projektgruppen**, die eine große Anzahl an atomaren Aktionen durchführen, die teils mit anderen Mitgliedern koordiniert werden müssen. Große Zahlen an Tasks können schnell zu Verwirrung führen, wenn diese nicht übersichtlich präsentiert werden. Zudem erfordern große Projekte durch Ausfall von Mitarbeitern etc. eine Einsicht in freigegebene Tasks, um diese übernehmen oder umschichten zu können. Die Applikation "TM" verwendet daher die Plattform "toodledo", damit auf diese Weise Tasks von anderen "toodledo"-Mitgliedern eingesehen und bearbeitet werden können (in der "toodledo" Web-Application).

Benutzer der Kalendersoftware iCal können "TM" verwenden, um existierende Task-Einträge in iCal mit zusätzlichen Informationen wie Task-Beginn-Datum zu versehen, und diese in "toodledo" zu übertragen. Der geringe Umfang und die fehlende Synchronisierung mit anderen Services oder Tools verhindert zur Zeit die umfangreiche Benutzung von Task-Daten in iCal.

Domain-Modell



Anmerkung: Diagramm ist als Omnigraffle-Projekt auch in der Projekt-Drop-Box (/ASE_03/DomainModel_03_v2.graffle).

Informationswesen

Kommunikation & Besprechungen

Die Kommunikation des Projektteams zur Realisierung von "TM" ist in folgende Teilbereiche gegliedert (sortiert nach absteigender Entscheidungskraft):

- Langfristig geplante Gruppentreffen
    - Rhythmus: Bestimmte Anzahl über gesamte Projektdauer
    - Sind entweder Management Reviews oder Internal Reviews
    - Anwesenheit der meisten Mitglieder ist obligatorisch
    - Koordinieren strategische, langfristig wirkende Entscheidungsprozesse
    - Stellen eine wichtige Schnittstelle zu den Auftraggebern dar

- Gruppentreffen in Anwesenheit des Tutors (IR)
    - Rhythmus: Jede 2. Woche
    - Werden regelmäßig durchgeführt
    - Anwesenheit von einigen Mitgliedern ist obligatorisch
    - Koordinieren taktische, kürzer wirkende Entscheidungsprozesse
    - Können auch zur Entscheidung von Feinheiten verwendet werden
- Gruppentreffen ohne Tutor
    - Rhythmus: Jede Woche, auch kurzfristig angesetzt
    - Werden nach Bedarf durchgeführt
    - Anwesenheit der Mitglieder je nach Bedarf
    - Koordinieren operative Entscheidungsprozesse, sollten also nur zur Entscheidung von Feinheiten verwendet werden
- Strukturierte digitale Kommunikation
    - Rhythmus: Wird täglich durchgeführt
    - Findet auf den zentralen, digitalen Anlaufstellen des Projektteams statt (Google Docs, Google Sites, Issue Tracker,...), oder per Email an alle Projektmitglieder
    - Sollte von allen Mitgliedern zur Kenntnis genommen werden
    - Festgehaltene Informationen werden als Dokumentation und als Grundlage für Gruppentreffen herangezogen
    - Koordiniert operative Entscheidungsprozesse, für die kein Gruppentreffen notwendig oder terminlich unerreichbar ist
- Informelle Kommunikation
    - Rhythmus: Wird fast ständig durchgeführt
    - Findet per Email, Skype, SMS und Telefon statt
    - Koordiniert keine Entscheidungsprozesse, sondern wird nur zur Klärung von Unklarheiten/Missverständnissen benutzt

Vorgehensmodell

Das Vorgehensmodell für die Umsetzung des Projekts "TM" ist eine angepasste Version des "Scrum"-Modells.
Folgende Anpassungen werden vorgenommen:

- Sprintlänge von 1 Woche
- Kein "Daily Scrum", sondern selbstständige Meldung von Projektmitgliedern per E-Mail, wenn eine Aufgabe voraussichtlich bei der Erfüllung des Sprints Probleme oder Verspätungen verursachen wird
- Keine Rolle "Scrum Master"
- "Iceberg-List" mit Waterline, alles oberhalb der Linie entspricht Sprint-Backlog, die ganze Liste entspricht Product-Backlog
- Keine "User Stories", sondern Usecases in Iceberg-List
- Kein "Impediment-Backlog", sondern sofortige Bearbeitung von organisatorischen Problemen durch den Teamkoordinator
- Burndown-Chart wird einmal pro Woche erstellt

Das wichtigste organisatorische Artefakt des Projekts ist das Product-Backlog, das im Falle der Projektgruppe "TM" der kompletten Iceberg-List entspricht. In dieser Liste sind alle Arbeitspakete enthalten, die zur Erfüllung der Projekt-Ziele benötigt werden. Die Iceberg-List ordnet den Arbeitspaketen die Release-Zahl zu, die aussagt, in welchem Sprint die jeweiligen Arbeitspakete erledigt werden sollen. Die geschätzte Zeitdauer zur Erledigung der Pakete darf nicht die voraussichtlich verfügbare Arbeitszeit übersteigen.
Vor jedem Sprint hat ein Treffen der Gruppenmitglieder stattzufinden, sodass die Arbeitspakete verteilt werden können. Während des Sprints (1 Woche lang) erarbeiten die Team-Mitglieder selbstständig die zugewiesenen Arbeitspakete, und melden sich per Email, wenn ein Paket Probleme verursacht oder höhere Komplexität aufweist als angenommen. Während des Sprints werden auch Burndown-Charts erstellt (einmal pro Woche), die Abweichungen vom Zeitplan gut aufzeigen. Nach jedem Sprint wird die Retrospektive durchgeführt, wo Ereignisse von allen Mitgliedern aufgezeigt werden und Verbesserungsmöglichkeiten in die Iceberg-List aufgenommen werden (und somit bei späteren Sprints bearbeitet werden); organisatorische Verbesserungsmöglichkeiten werden sofort vom Teamkoordinator wahrgenommen.

Kollaborations-Tools

- Google Documents: Wird für die zentrale Verwaltung von Projekt-Dokumenten wie Protokollen, Iceberg-List (Projekt-Backlog), etc. verwendet
- Dropbox: Für Dateien welche nicht in google docs gehostet werden können
- Google Sites: Wird für Inhalte verwendet, die nicht in Google Docs passen, wie Projekt-Blog
- Github: Wird für source code Hosting (git als SCM) und Issue-Tracking (Sprint-Backlog) verwendet

Besonderheiten

Kosten für Hard/ Software sowie für Räumlichkeiten

Sämtliche Teammitglieder entwickeln und testen auf ihren privaten Geräten sowie in privaten Räumlichkeiten. Gruppentreffen finden entweder in Räumen der TU Wien oder in privaten Wohnungen statt.
Im Zuge von Management Reviews (MRs) werden die Fortschritte mittels gedruckten Artefakten und/oder Bildschirm- bzw. Beamerpräsentationen vorgeführt. Ein dazu benötigtes Notebook ist vorhanden, private Drucker können verwendet werden und auch ein Beamer befindet sich in dem Raum, in welchem die MRs stattfinden. Es fallen also für Hard- oder Software sowie für Räumlichkeiten keine Kosten für das Projektteam an.

Folgende Hardware und Software wird einheitlich von allen Teammitglieder verwendet:

- *Macintosh*-Computer (Notebook oder Desktop Rechner)
- Betriebssystem *Mac OS X Snow Leopard (Version 10.6)*

- Entwicklungsumgebung *Xcode 3.2*
- Programmiersprache *Objective C 2.0*
- git scm Version 1.6.4

Finanzielle Aspekte

Die Erstellung eines Business Planes ist derzeit nicht geplant. Es ist des des weiteren noch nicht definiert, ob das Projekt als Open-Source oder Closed-Source der Comunity zu Verfügung gestellt wird.

Abgrenzungen des Projekts

Inhaltliche Abgrenzung

Der Funktionsumfang der Anwendung beschränkt sich implizit auf die definierte feature-Liste. Nicht teil dieses Projekts sind daher unter anderem:

- Die Synchronisationsmöglichkeit mit Notizen von Apple Mail, iPods, iPhones oder anderen mobilen Endgeräten
- Die Synchronisationsmöglichkeit mit anderen Web-Anbietern außer toodledo.com
- Eine vollständiger Ersatz für iCal, die Anwendung soll zum Verwalten von todo's, und nicht für Termine dienen
- Eine 1:1-Replikation von toodledo.com
- Eine vollständige Internationalisierung der Anwendung (es ist nur "de" und "en" vorgesehen)
- Die Implementierung einer Kontaktverwaltung

Zeitliche Abgrenzung

Für die zeitliche Abgrenzung kann der Projektstrukturplan herangezogen werden. Zeitliche Aufwendungen für das Marketing oder den Support sind für dieses Projekt nicht vorgesehen.

Soziale Abgrenzung

In das Projekt involvierte Personen:

Auftraggeber	- (indirekt siehe Zielgruppen)
Informationsgeber	- (toodledo-Informationen: API-Doc)
Ansprechpartner der LV	Peter Steinberger, Gerald Futschek
Projektteam	siehe Team

Funktionale Anforderungen

Must-haves

1 Offline Task-Verwaltung

1.1 Tasks

Es sollen Todo's in Form von Tasks einfach hinzugefügt werden können. Falls Änderungen notwendig sind, können die Tasks bearbeitet sowie des weiteren gelöscht werden. Gelöschte Tasks werden zuerst in den Papierkorb verschoben, von wo aus sie wieder hergestellt werden können. Um einen Task endgültig zu löschen, muss der Papierkorb geleert werden.

1.2 Ordner

Um die Organisation der Tasks zu vereinfachen, können Ordner erstellt bzw. diese danach auch wieder editiert werden. Diese hierachischen Ordner können ergänzend oder alternativ zu Tags verwendet werden. Ordner können mit Notebooks, also kleinen Notizen, versehen werden. Wie bei den Tasks, werden auch Ordner nach dem Löschen zuerst in den Papierkorb verschoben, von wo aus sie ebenfalls wieder hergestellt werden können. Das endgültige Entfernen erfordert ein Leeren des Papierkorbes. Beim Löschen eines Ordners muss darauf geachtet werden, dass enthaltene Tasks die nicht gelöscht werden sollen, zuvor manuell verschoben werden müssen. Ansonsten werden diese auch entfernt.

1.3 Tasks in Ordnern gruppieren/verschieben

Tasks können zu Ordnern hinzugefügt werden um zusammengehörige Tasks zusammen zu fassen. Dabei ist auch ein verschieben der Tasks in andere Ordner problemlos möglich.

1.4 Tags

Tasks können mit Tags versehen werden eine besser Übersichtlichkeit zu erlangen (als Ergänzung zu einer hierarchischen Ordnerstruktur). Ein Task kann mehrere Tags haben, die auch geändert oder entfernt werden können.

1.5 Undo-Funktionalität

Aktionen im Zusammenhang mit Tasks sollen wieder Rückgängig gemacht werden können. Arbeitsschritte wie das Hinzufügen, Ändern und Bearbeiten von Tasks, Ordnern und Tags soll bis zu der im Arbeitsspeicher vorhandenen Historie rücksetzbar sein.

1. 6 Papierkorb-Verwaltung

Tasks, Tags oder Ordner, die in den Papierkorb gelegt wurden, können wiederhergestellt oder alternativ ganz gelöscht werden. Da Tasks in

Ordnern in Toodledo nicht gelöscht, sondern ordner-los werden, werden diese nach der Wiederherstellung wieder dem Ordner zugeordnet.

2 Anzeigen von Tasks

2.1 Listenansichten ("heute", "diese Woche", "dieses Monat")

Der Benutzer soll zwischen verschiedenen Ansichten umschalten können. Dabei werden Ansichten für den aktuellen Tag, die aktuelle Woche und den aktuellen Monat unterstützt.

3 Optische / akustische Benachrichtigungen bei abgelaufenen oder kurz bevorstehenden Tasks

Der Benutzer soll durch einen Dialog, ein blinkendes Symbol im Menubar und evtl. zusätzlich über ein akkustisches Signal darüber informiert werden, wenn Tasks ablaufen oder kurz bevorstehen. Dabei ist die Art der Benachrichtigung sowie der Zeitpunkt und die Wiederholungen dieser Benachrichtigungen individuell im Einstellungsmenü definierbar.

4 Synchronisation

Es besteht die Möglichkeit den lokalen Datenbestand mittles PlugIns mit anderen ToDo Services zu synchronisieren. Vorerst werden zwei PlugIns erstellt: Eines welches die Synchronisation mit der online ToDo Verwaltung "toodledo" ermöglicht. Ein weiters Plugin ermöglicht die Synchornisation mit iCal Daten (z.b. iCal Tasks/Notizen). Der Benutzer hat bei der Synchronisation die Möglichkeit Konflikte zu beheben, dies soll ähnlich wie bei der iPhone/iCal Synchronisation funktionieren.

5 Suche in Tasks

Sämtliche Tasks können mit einer Suchfunktion durchforstet werden. Diese Suche soll an die an die Suche von "Finder" angelehnt sein. Dies ermöglicht das schnelle Auffinden eines gewünschten Tasks. Es stehen dabei diverse Filterkriterien (Zeitraum der Suche, Task Tags, etc.) zu Verfügung. Zusätzlich soll die Möglichkeit bestehen, vergange Sucheinträge zu wiederholen, ähnlich wie es eine solche Funktion im Programm "PathFinder" gibt.

6 Automatisierte Backups & Logs

In regelmäßigen Abständen - welche in den Programmeinstellungen individuell konfigurierbar sind - werden automatische Backups der Datenbasis erstellt. Des weiteren wird jede Änderung des Datenbestanden, wie durch Benutzereingabe oder Synchronisation, protokolliert.

7 Zugriff auf wichtigste Features über Menubar

Über ein Programmsymbol in der Systemleiste am oberen Bildschirm (Mac OS X Menübar, z.b. neben der Uhrzeit) können die wichtisten Programmfeatures schnell ausgeführt werden, ohne das Programm selbst gestartet zu haben. Damit können z.b. Features wie Task anlegen, manuelle

Synchronisation durchführen, etc. schnell durchgeführt werden ohne das aktuelle Programm (Word, Mail, etc.) zu verlassen.

8 Programmeinstellungen vornehmen

In einem Einstellungsbereich können sämtliche Programmeinstellungen vorgenommen werden. Dies sind u.a. Spracheinstellungen, Acounteinstellungen, Synchronisationseinstellungen und dergleichen.

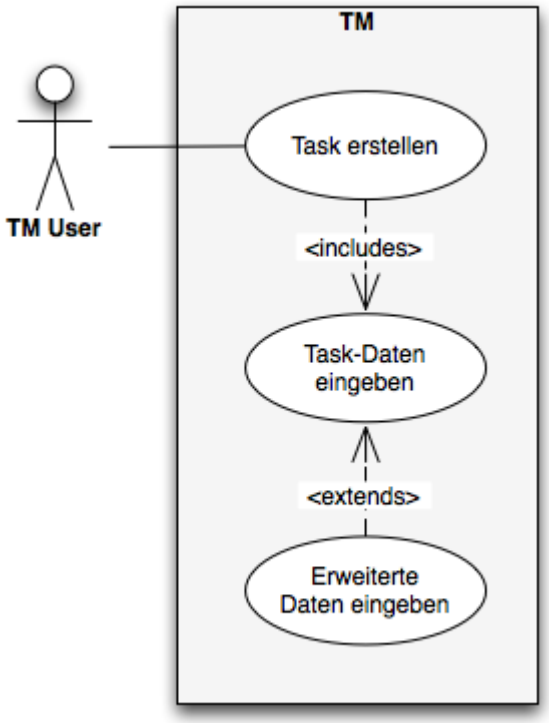
Nice-to-haves

- Erweiterter Zugriff, Anzeige von Tasks im Menubar
- Erweitere Task-Ansicht "Timeline", auch Anzeige im Menubar
- Suche kann als intelligente Liste gespeichert werden
- Offline Task-Verwaltung
    - Batch-processing mehrerer Tasks
    - Import/Export der Tasks von/in CSV-Format
    - Auswertungen zw. estimated und benötigter Zeit
- Automatische Programm-Update Funktion
- Stoppuhr-Funktion für aktive Tasks
- Anzeige und Bearbeitung von Tasks, die von anderen "toodledo"-Benutzern freigegeben wurden
- Andere "toodledo"-Benutzer mit Adressbuch verknüpfen
- Zusätzliche Menü-Einträge
    - Punkt "Hilfe"
    - Punkt "Über..."
- "Live"-Suchfunktion mit Autovervollständigung
- Eigenes Dashboard-widget
- Geo-Locations (Kartendarstellung, Foto-Verknüpfung)
- Schnellzugriff via Tastenkürzel
- Firefox Plug in welches aktuelle Tabs als Task im Programm ablegt ohne (ToRead until XY)
- Schnell-Hinzufügen von Task über Eingabefeld

Use Cases & Diagramme

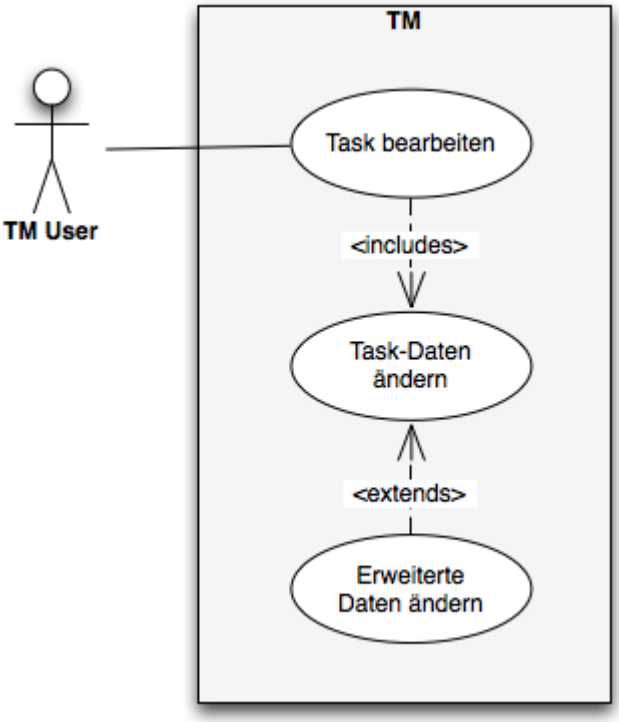
Die identifizierten Use Cases erweitern die idenfizierten "Funktionalen Anforderungen" mit einer höheren Granularität und Diagrammen.
Im Folgenden sind drei der Use Cases angeführt, der Rest befindet sich im Google Docs Dokument "Use Case Beschreibungen und Diagramme".

Task erstellen



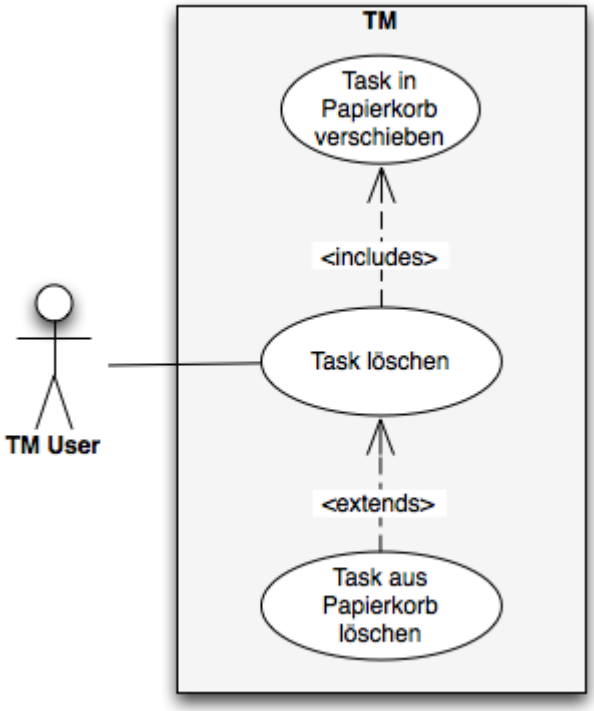
Beschreibung: Der User erstellt einen neuen Task. Für die Dateneingabe öffnet sich daher ein Formular, auf dem die wichtigsten Felder sofort verfügbar sind. Will der Benutzer erweiterte Einstellungen erfassen, klickt er einen entsprechenden Erweiterungs-Button.
Akteure: TM User
Vorbedingungen:
Anmerkung:

Task bearbeiten



Beschreibung: Der User bearbeitet einen existierenden Task. Für die Dateneingabe öffnet sich daher ein Formular, auf dem die wichtigsten Felder sofort verfügbar sind. Will der Benutzer erweiterte Einstellungen erfassen, klickt er einen entsprechenden Erweiterungs-Button.
Akteure: TM User
Vorbedingungen: Der zu bearbeitende Task muss im Datenmodell existieren.
Anmerkung: Bei dieser Art der Formular-basierten Bearbeitung soll immer nur ein Task zu einem Zeitpunkt bearbeitet werden können. Der zu bearbeitende Task muss für die Dauer der Bearbeitung für andere Services/Funktionen gesperrt werden (Synchronisation, Löschung, ...).

Task löschen (Papierkorb)



Beschreibung: Bereits erstellte Tasks sollen gelöscht werden können. Diese werden zuerst in den Papierkorb verschoben, von wo aus sie wieder hergestellt werden können. Um einen Task endgültig zu löschen, muss der Papierkorb geleert werden.
Akteure: TM User
Vorbedingungen: Der zu löschende Task muss im Datenmodell existieren.
Anmerkung: Sobald sich ein Task im Papierkorb befindet, wird dieser beim Synchronisieren vom Server auch entfernt.

Nichtfunktionale Anforderungen

Globale NF Anforderungen

Zuverlässigkeit

a1) Systemreife

Ziel ist eine lauffähige Anwendung bei welcher unvorhergesehene Ereignisse nicht zum Datenverlust führen ("beta").

a2) Wiederherstellbarkeit

Dem Benutzer soll bei vorhandenen Funktionen eine "Rückgängig"-Funktion zur Verfügung stehen. Gelöschte Objekte sollen daher in einem Papierkorb zwischengespeichert werden. Einmal im Papierkorb gelöschte Objekte

können im laufenden Betrieb nicht mehr wiederhergestellt werden.

Automatische backups im Hintergrund sollen weiters einen vollkommenen Datenverlust vermeiden.

a3) Fehlertoleranz

Die Anwendung muss mit unvorhergesehenen Eingaben des Benutzers umgehen können.

Aussehen und Handhabung

a4) Aussehen / Bedienbarkeit

Die Anwendung soll den von Apple ausgegeben "human interface guidelines" folgen. Dadurch kann sichergestellt werden, dass die Bedienung durch mac-typische Ansätze (zB. drag'n'drop) möglich ist.

a5) Erlernbarkeit

Die Notwendigkeit eines Benutzerhandbuches soll möglichst vermieden werden. Die Funktionen der Anwendung sollen dem Benutzer durch kontext-sensitive Tipps (u.a. bei der Eingabe) näher gebracht werden.

a6) Verständlichkeit

Die Anwendung soll den Benutzer bei Eingaben (optional) kontextsensitiv unterstützen. Leere Screens sollen vermieden werden, statt dessen kann der Benutzer auf mögliche "next-steps" hingewiesen werden.

Leistung und Effizienz

a7) Antwortzeiten

Die Anwendung soll in jeder Situation ein flüssiges Arbeiten ermöglichen. Funktionen wie Online-Synchronisation sollen dem User zwar angezeigt werden, diese dürfen die Anwendung jedoch nicht blockieren. Weiters sind online-requests mit einem time-out zu versehen, um andere Aktionen nicht zu blockieren.

a8) Ressourcenbedarf

Nachdem die Anwendung auch im Hintergrund laufen soll, darf diese keine höheren Anforderungen an Speicher und Prozessorlast als vergleichbare

Anwendungen stellen. Weiters soll der Speicherbedarf bei längerem Ausführen konstant gehalten werden.
Betrieb und Umgebungsbedingungen

a9) Kompatibilität

Die Kompatibilität soll für Mac OS X in Versionen 10.5 und 10.6 gegeben sein. Frühere Versionen oder andere Plattformen werden nicht in Betracht gezogen.

Wartbarkeit, Änderbarkeit

a10) Testbarkeit

Die einzelnen Komponenten (zB. web-sync) sollen unabhängig voneinander testbar sein.

a11) Änderbarkeit / Erweiterbarkeit

Die Anwendung soll so entworfen werden, dass auch andere sync-Möglichkeiten in das System integriert werden können (nicht Teil dieses Projekts, siehe Abgrenzung). Weiters sollen die Konzepte I18N und L10N eingesetzt werden (siehe auch a13).

a12) Update-Funktionalität

Die Anwendung soll automatisch prüfen können, ob ein update zur Verfügung steht. Weiters soll dieses automatisch heruntergeladen werden können.
Portierbarkeit und Übertragbarkeit

a13) Anpassbarkeit

Für das Projekt relevante Sprachen sind Deutsch und Englisch, wobei die Sprache automatisch, aufgrund der Einstellung am Mac, eingestellt werden soll. Laufzeitrelevante Einstellungen sollen zentral getätigt werden können. Die Sync-Funktionen können wahlweise aktiviert/deaktiviert werden. An eine sonstige Anpassbarkeit werden keine Anforderungen gestellt.

a14) Installierbarkeit

Die fertige Anwendung soll in einer dmg-Datei gekapselt sein. Durch Öffnen dieser Datei kann die Anwendung mac-typisch per drag'n'drop in einen beliebigen Ordner kopiert werden. Die Anwendung soll also ohne einen Installer lauffähig sein.

Sicherheitsanforderungen

a15) Vertraulichkeit

An die lokalen Daten selbst wird kein hoher Grad an Vertraulichkeit gestellt, eine verschlüsselte Speicherung ist nicht vorgesehen. Die Benutzerdaten eines optionalen sync-Services ("toodledo") müssen jedoch im Schlüsselbund gespeichert werden. Die Übertragung der Daten per online-sync soll, falls implementierbar, verschlüsselt vonstatten gehen.

a16) Datenintegrität

Die Integrität der Daten soll auch durch sync-Vorgänge gewährleistet sein. Bei Problemen soll auf backups zurückgegriffen werden können.

a17) Verfügbarkeit

Da die Anwendung auch ohne sync-Dienste voll einsetzbar sein soll, müssen Daten auch ohne bestehende Internetverbindung uneingeschränkt verwendbar und änderbar sein.

Lieferumfang und Abnahme

Abnahmen, welche im Zuge des MR2 bzw. M3 stattfinden sind sind wie folgt geplant:

MR2: Dabei werden alle unter Funktionsbeschreibung beschriebenen Funktionalitäten in einem Prototyp simuliert und teils auch schon funktionsfähig. Dies soll einen ersten Einblick in die Applikation geben sowie die Disskussionsbasis für das MR3 bilden (wie z.b. die Aufnahme von Nice-To-Haves" in den Funktionsumfang oder der Optimierung von Grundfunktionalitäten)

MR3: Dies ist die endgültige Abnahme im Zuge von dem ASE Projekt. Ziel sind die beim MR2 vereinbarten Features, welche neben den Grundfunktionlalitäten auch die Umsetzung einiger "Nice-To-Have" Features beinhalten sollen.

Anmerkungen: Das Entwicklungsteam hat das unter dem Punkt "Informationswesen - Vorgehensmodell" beschriebene Vorgehensmodell SCRUM aus folgenden Gründen gewählt: Sämtliche Teammitglieder sind sehr motiviert und möchten auch eine Applikation entwickeln, welche von vielen Mac Usern tatsächlich benutzt wird, sich also von vielen ASE Projekten abheben. Das Klima zwischen Tutor und Gruppe scheint gut zu sein, was als Basis für ein erfolgreiches SCRUM Projekt sehr wichtig

ist. Alle Gruppenmitglieder haben gute Programmierkenntnisse, jedoch nur ein Gruppenmitglied Erfahrung mit Objective-C. Deshalb ist es zum Zeitpunkt des Projektauftrages schwer abzuschätzen, welche Features wie schnell umgesetzt werden können bzw. welche aufgrund der Cocoa Frameworks "leicht von der Hand" gehen. Die Abschätzungen werden sich sicher von SCRUM Sprint zu SCRUM Sprint verbessern. Auch die Gruppenübergreifende Zusammenarbeit (Domain Model) ist eine Herausforderung, welche eine seriöse Zeitschätzung durch uns Gruppenmitglieder schwer macht. Ein Treffen mit dem Tutor, welches ca. alle zwei Wochen stattfindet, soll diesem gute Einblicke in den aktuellen Entwicklungsstatus geben. In diesen Treffen werden auch klare Sprint Ziele abgesprochen.Die meisten Nice-To-Have Features sind deshalb als Nice-To-Have angeführt, da diese aus jetztiger Sicht technologisches Neuland für die Gruppe sind. Die Umsetzbarkeit und auch der Umfang einer Umsetzung sind für uns derzeit nicht abschätzbar, jedoch nehmen wir an, dass weitere Nice-To-Have Features während der Entwicklung dazukommen.
Das Team hat sich zum Ziel gesetzt in den ersten SCRUM Sprints bis zum MR2 die Features der Funktions Anforderungen umzusetzen und sich dann in Abstimmung mit dem Tutor schrittweise auch den Nice-to-Have Features anzunehmen bzw. die bestehenden Features verbessern.

Projektplan

Da es sich bei dem Vorgehensmodell des Projekts "TM" um einen agilen Prozess handelt, erfordert die Durchführung auch eine agile Herangehensweise an den Projektplan. Die zeitliche Positionierung der Meilensteine hängt von der Länge der Scrum-Sprints ab, die das Projekt in iterative Schritte teilen.
Die Unterteilung des Projekts in Meilensteinen zusätzlich zu der Unterteilung in Scrum-Sprints dient der einfacheren Steuerung durch die Auftraggeber. Meilensteine 1 und 4 sind länger als

Geplante Scrum-Sprint-Länge: 1 Woche

Übersicht der Meilensteine

Meilenstein	Sprints	Termin
M1	4	25. 11. 2009
M2	2	9. 12. 2009
M3	1	16. 12. 2009
M4	4	13. 1. 2010
M5	1	20. 1. 2010

Anmerkungen:
Der Plan geht von einem Entwicklungsstart am 28. Oktober 2009 aus.
Da der Projekt-Abschluss-Termin noch nicht feststeht, hat der Zeitraum M4-M5 eine Länge von 1 Sprint veranschlagt, der Zeitraum kann jedoch auf 2 Sprints ausgedehnt werden.

Details zu den Meilensteinen:

M1 (25.11.09)

Usecase-Analyse	Usecase-Beschreibungen fertig Usecase-Diagramme fertig

Implementierte Usecases	Task erstellen Task bearbeiten Task löschen Listenansichten
User Interface	20% der Interfaces fertig
User Interface Prototyp	lauffähiger Prototyp fertig
Tests	Testrichtlinien fertig, 20% der Tests fertig
Modelle	Komponenten-Modell, Klassen-Modell, Daten/Domain-Modell fertig

M2 (9.12.09)

Implementierte Usecases	Ordner erstellen Ordner bearbeiten Ordner löschen Tasks in Ordnern gruppieren/ verschieben
User Interface	40% der Interfaces fertig
Tests	40% der Tests fertig

M3 (16.12.09)

Implementierte Usecases	Tag erstellen Tag bearbeiten Tag löschen Task einem Tag zuordnen Papierkorb-Verwaltung
User Interface	70% der Interfaces fertig
Tests	60% der Tests fertig

M4 (13.1.10)

Implementierte Usecases	Undo-Funktionalität Synchronisation Suche in Tasks Optische/akustische Benachrichtigungen bei abgelaufenen oder kurz bevorstehenden Tasks Zugriff auf wichtigste Features über Menubar Programmeinstellungen vornehmen
User Interface	100% der Interfaces fertig
Tests	95% der Tests fertig

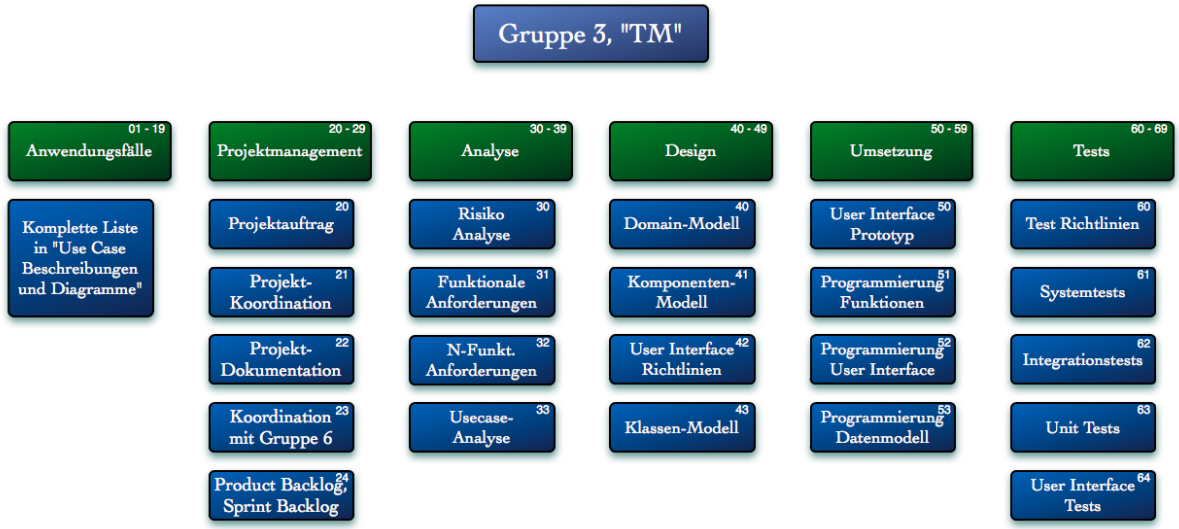
M5 (13.1.10)

Implementierte Usecases	Automatisierte Backups & Logs
User Interface	100% der Interfaces fertig
Tests	100% der Tests fertig

Projektstrukturplan (Work Breakdown Structure)

Im Projektstrukturplan ist eine Übersicht über alle groben Arbeitspakete des Projekts enthalten. Die weitere Verfeinerung und die Zuteilung der Pakete zu Scrum-Sprints (und somit auch Milestones) erfolgt in der Iceberglist (siehe Kapitel "Iceberglist").

Im Folgenden ist eine grobe Übersicht der Work Breakdown Structure angegeben, die im weiteren Projektverlauf auf einzelne Arbeitspakete heruntergebrochen werden wird, um jedes Paket mit einem Fertigstellungs-Termin versehen zu können. Die Nummern der abgebildeten Mainfeatures sind auch in der Iceberg-List wiederzufinden.



Anmerkung: Dokument ist als Omnigraffle-Projekt auch in der Projekt-Drop-Box (/ASE_03/Projekt-Organisation/Projektstrukturplan.graffle).

Iceberglist (erweiterte Feature-Liste)

In der Iceberglist wird die Projektstruktur in Arbeitspakete zerlegt, die jeweils innerhalb eines Arbeitstages erledigt werden können.
Die Gesamtheit der Iceberglist stellt das Scrum-Product-Backlog dar, während das Sprint-Log durch alle Pakete definiert wird, die oberhalb der Waterline liegen.

SIEHE ANHANG A (Auch in Google Docs Repository unter "Projekt-Definition/ Iceberg List")

Komponentendiagramm

Das Komponentendiagramm bietet eine Übersicht über die interne Struktur des zu entwickelnden "TM"-Clients.

SIEHE ANHANG B (Auch in der Projekt-Drop-Box unter /ASE_03/ Komponentendiagramm.graffle)

Rollenverteilung

Teamkoordinator: Manuel Maly, Stellvertretung: Michael Petritsch

Kompetenzen & Pflichten:

- Monitoring Projektforschritt und Reaktion bei Verspätung
- Koordination Gruppentreffen
- Gruppen-Kommunikation im Fluss halten
- Kommunikation mit Tutor
- Kommunikations-Infrastruktur warten (Wikis, Google Docs, etc.)
- Kommunikations-Artefakte geordnet halten

Software Architekt: Alex Leutgöb, Stellvertretung: Dominik Hofer

Kompetenzen & Pflichten:

- Planung der technischen Architektur
- Hilfestellung bei technischen Problemen anderer Team-Mitglieder
- Verwendete Technologien, framworks und tools (+ Versionen) definieren
- Regeln für commits (Kommentare: In welchen files wurde was verändert) festsetzen
- Best-practice-Verzeichnisstrukturen/libraries in Projekt vorschlagen
- Richtlinien für Logging festsetzen
- Definition eines Style-Guides

Software Tester / Qualitätsbeauftragter: Christian Hattinger, Stellvertretung: Dominik Hofer

Kompetenzen & Pflichten:

- Verantwortlich Testplanung, Verteilung der Durchführung und Testbericht
    - In Abstimmung mit TK & Tutor vereinbaren, welche Testartefakte wann benötigt werden
    - Diese erstellen (Struktur, etc.), Inhalt jedoch in Kooperation mit anderen Teammitgliedern
    - Planung und Führung durch statische Tests --> Reviews (e.g. Inspections, Walkthroughs)
    - Verifizierung: Entspricht das Packet den Requirements
    - Validierung: Entsprechen die Requirements den User Anforderungen

        - Feststellung/Dokumentation der Qualität des Sprint-Backlogs (Arbeitspackete)
    - Code-Qualität kontinuierlich prüfen
    - Tätigkeiten die nicht (ausschließlich) vom Tester durchgeführt werden:
        - Alleinige Ausführung und Dokumentierung von Tests
        - Erstellung der Unit Tests (für gesamten Code)
        - Debugging (Qualität wird festgestellt und dokumentiert, später dann durch Team verbessert)

Dokumentbeauftragter: Michael Petritsch, Stellvertretung: Andrea Füresz

Kompetenzen & Pflichten:

- Protokollführung bei Meetings
- Festlegen von Code-Dokumentationsrichtlinien
- Kontrolle ob Code-Dokumentation ordentlich durchgeführt wird
- Bei Dokumentationsmängeln die betreffenden Personen erinnern
- Kontrolle aller erzeugten Dokumente (Diagramme, Skizzen, etc...)

User Interface Beauftrage: Andrea Füresz, Stellvertretung: Manuel Maly

Kompetenzen & Pflichten:

- Festlegen der Masken für den Workflow (Ablauflogik)
- Festlegen welche GUI Elemente eingesetzt werden
- Festlegen von Interaktionspatterns
- Wording von Menüs und Dialogen
- Wahl geeigneter Icon-Metaphern

ANHANG A - Iceberg List

Iceberg-List beginnt auf der nächsten Seite.

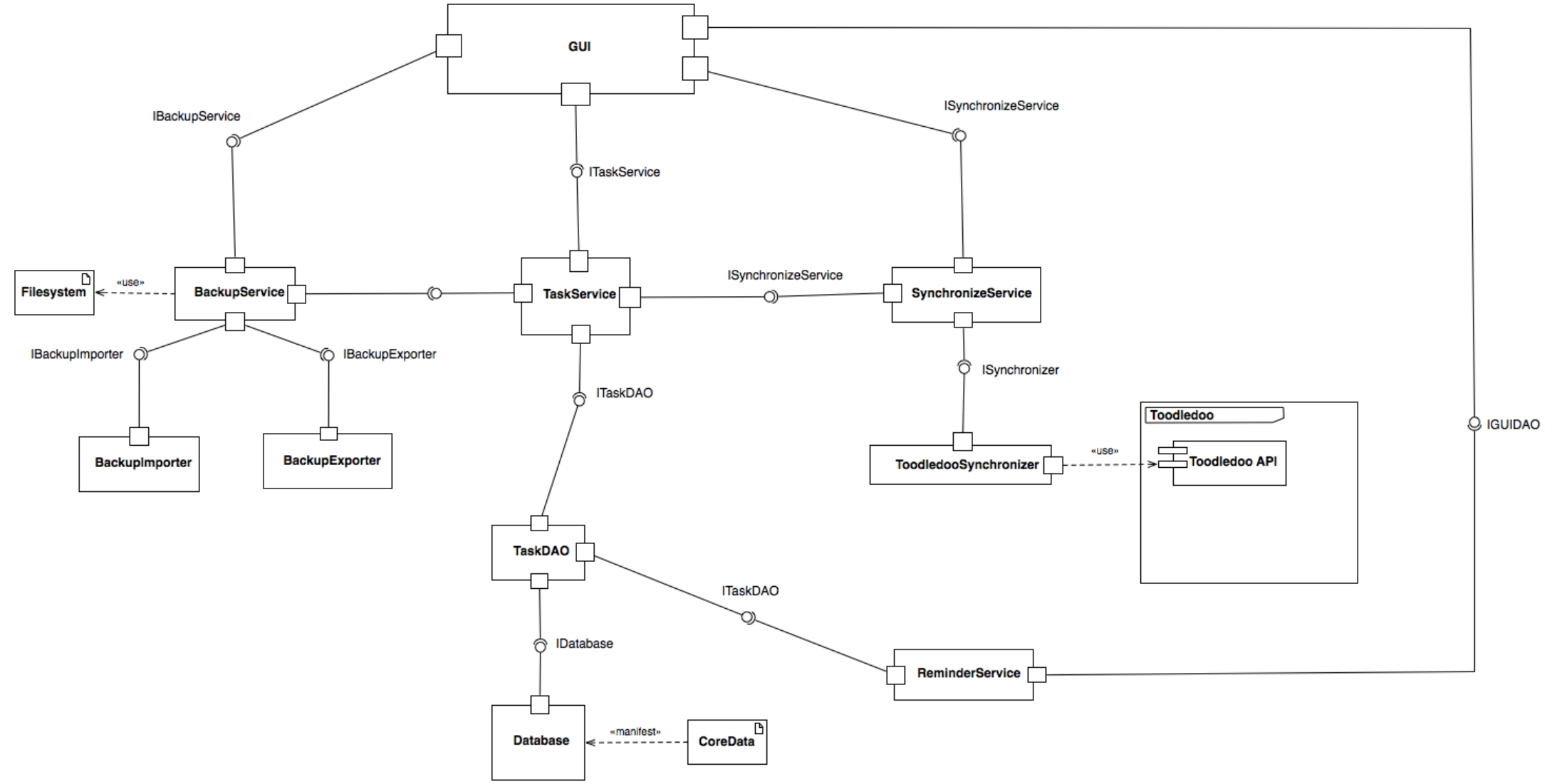
Mainfeature (siehe Projektstrukturplan)	Iceberg Nr.	Anwendungs-Fall	Feature	Feature Beschreibung	Priorität (L,M,H)	Komplexität (1-10)	Verantwortung	Release = Meilenstein
01 - Offline Task-Verwaltung	1	1.1.1	Task hinzufügen	Es sollen Todo's in Form von Tasks einfach hinzugefügt werden können.	H	3	?	1
01 - Offline Task-Verwaltung	2	1.1.2	Task bearbeiten	Der User bearbeitet einen existierenden Task.	H	3	?	1
01 - Offline Task-Verwaltung	3	1.1.3	Task löschen (Papierkorb)	Bereits erstellte Tasks sollen gelöscht werden können.	H	5	?	1
02 - Anzeigen von Tasks	20	2.1	Listenansichten ("Simple Liste", "Intelligente Liste")	Der Benutzer soll zwischen verschiedenen Ansichten umschalten können.	H	4	?	1
20 - Projektauftrag	200	-	Projektauftrag fertigstellen		M	3	Manuel	1
21 - Projektkoordination	210	-	Projektkoordination, Sprint 1		H	6	Manuel	1
22 - Projektdokumentation	220	-	Projektdokumentation Sprint 1		H	2	Michael	1
23 - Koordination mit Gruppe6	230	-	Koordination mit Gruppe 6, Sprint 1		M	4	Alex, Michael	1
24 - Product & Sprint Backlog	240	-	Product & Sprint Backlog, Sprint 1		H	5	Manuel	1
30 - Risiko-Analyse	300	-	Risiko Analyse		L	3	?	1
31 - Funktionale Anford.	310	-	Funktionale Anforderungen		H	2	?	1
32 - N-Funkt. Anford.	320	-	Nicht-Funktionale Anforderungen		H	2	?	1
33 - Use-Case Analyse	330	-	Use-Case Analyse, Sprint 1		H	4	?	1
40 - Domain-Modell	400	-	Domain-Modell		H	7	?	1
41 - Komponenten-Modell	410	-	Komponenten-Modell, Sprint 1		H	8	Dominik	1
42 - User Interface Richtlinien	420	-	GUI Richtlinien		M	7	Andrea, Manuel	1
43 - Klassen-Modell	430	-	Klassen-Modell		H	8	?	1
50 - User Interface Prototyp	500	-	User Interface Prototyp		H	10	Andrea, Manuel	1
53 - Progr. Datenmodell	530	-	Programmierung Datenmodell		H	4	Dominik	1
52 - Programmierung GUI	520	-	Programmierung User Interface, 30% abgeschlossen		H	7	Andrea, Manuel	1
60 - Test-Richtlinien	600	-	Test-Richtlinien		H	7	Christian	1
01 - Offline Task-Verwaltung	4	1.2.1	Ordner erstellen	Der User erstellt einen neuen Ordner.	H	3	?	2

Mainfeature (siehe Projektstrukturplan)	Iceberg Nr.	Anwendungs-Fall	Feature	Feature Beschreibung	Priorität (L,M,H)	Komplexität (1-10)	Verantwortung	Release = Meilenstein
01 - Offline Task-Verwaltung	5	1.2.2	Ordner bearbeiten	Der User bearbeitet einen existierenden Ordner.	H	3	?	2
01 - Offline Task-Verwaltung	6	1.2.3	Ordner löschen (Papierkorb)	Bereits erstellte Ordner sollen gelöscht werden können.	H	5	?	2
01 - Offline Task-Verwaltung	7	1.3	Tasks in Ordnern gruppieren/verschiebe	Tasks können in Ordnern zusammengefasst werden	M	6	?	2
52 - Programmierung GUI	521	-	Programmierung User Interface, 60% abgeschlossen		H	7	Andrea, Manuel	2
01 - Offline Task-Verwaltung	8	1.4.1	Tag erstellen	Der User erstellt einen neuen Tag.	M	3	?	3
01 - Offline Task-Verwaltung	9	1.4.2	Tag bearbeiten	Der User bearbeitet einen existierenden Tag.	M	3	?	3
01 - Offline Task-Verwaltung	10	1.4.3	Tag löschen (Papierkorb)	Bereits erstellte Tags sollen gelöscht werden können.	M	5	?	3
01 - Offline Task-Verwaltung	11	1.4.4	Task einem Tag zuordnen	Der User ordnet einem Task einen oder mehrere Tags zu	M	6	?	3
01 - Offline Task-Verwaltung	13	1.6	Papierkorb Verwaltung	Gelöschte Tasks/Tags/Ordner landen im Papierkorb. Von dort können sie wiederhergestellt oder ganz gelöscht werden	H	5	?	3
52 - Programmierung GUI	521	-	Programmierung User Interface, 100% abgeschlossen		H	7	Andrea, Manuel	3
01 - Offline Task-Verwaltung	12	1.5	Undo Funktionalität	Der User kann Aktionen im Zusammenhang mit Tasks wieder Rückgängig machen	H	8	?	4
03 - Notifikationsdienst	30	3	Optische/akustische Benachrichtigungen bei abgelaufenen oder kurz bevorstehenden Tasks	Der Benutzer wird an seine Tasks "erinnert" in dem er Benachrichtigungen erhält	M	10	?	4

Mainfeature (siehe Projektstrukturplan)	Iceberg Nr.	Anwendungs-Fall	Feature	Feature Beschreibung	Priorität (L,M,H)	Komplexität (1-10)	Verantwortung	Release = Meilenstein
04 - Synchronisationsmodul	40	4	Synchronisation	Dieser Dienst ermöglicht die Synchronisation des lokalen Datenbestandes mit anderen ToDo Services.	H	10	?	4
05 - Suche	50	5	Suche in Tasks	Sämtliche Tasks können mit einer Suchfunktion durchforstet werden.	H	8	?	4
07 - Menubar	70	7	Zugriff auf wichtigste Features über Menubar	Über ein Programmsymbol in der Systemleiste können wichtige Programmfeatures schnell ausgeführt werden	L	5	?	4
08 - Programmeinstellungen	80	8	Programmeinstellung vornehmen	In einem Einstellungsbereich können sämtliche Programmeinstellur vorgenommen werden.	H	4	?	4
06 - Backups & Logs	60	6	Automatisierte Backups & Logs	In regelmäßigen Abständen werden automatische Backups der Datenbasis erstellt.	M	10	?	5
61 - Systemtests	610	-	Systemtests		M	6	?	?
62 - Integrationstests	620	-	Integrationstests		M	6	?	?
63 - Unit-Tests	630	-	Unit-Tests		H	7	?	?
64 - GUI-Tests	640	-	GUI-Tests		H	10	?	?

ANHANG B - Komponentendiagramm

Komponentendiagramm befindet sich auf der nächsten Seite.


GUI
IBackupService
ITaskService
ISynchronizeService
Filesystem
«use»
BackupService
TaskService
ISynchronizeService
SynchronizeService
IBackupImporter
IBackupExporter
ITaskDAO
ISynchronizer
Toodledoo
IGUIDAO
BackupImporter
BackupExporter
ToodledooSynchronizer
«use»
Toodledoo API
TaskDAO
ITaskDAO
IDatabase
ReminderService
Database
«manifest»
CoreData