Konzeptbericht

Bash Vi, Shylux, Kaleb Tschabolt

May 29, 2011

Status	In Arbeit/In Pruefung/**Abgeschlossen**
Projektname	Projektexplorer
Projektleiter	Niklaus Hofer
Auftraggeber	M. Frieden, GIBB
Autoren	Niklaus Hofer
Verteiler	Lukas Knoepfel, Kaleb Tschabolt, Niklaus Hofer

Aenderungskontrolle, Pruefung, Genehmigung

Version	Datum	Beschreibung, Bemerkung	Name oder Rolle
0.1	22.03.2011	Erstellen	Hofer, Knöpfel, Tschabold
1.0	29.05.2011	Verbesserungen, Port nach LaTeX	Hofer

Definitionen und Abkuerzungen

Begriff/ Abkuerzung	Bedeutung
DB	Datenbank

References

[1] Wikimedia Foundation. inotify. `http://en.wikipedia.org/w/index.php?title=Inotify&oldid=404718721`, 2011.

[2] Lukas Knoepfel Kaleb Tschabold Niklaus Hofer. Konzeptbericht. Teil der Abgabedokumente, 2011.

[3] seb m. seb-m/pyinotify. `https://github.com/seb-m/pyinotify`, 2011.

Contents

1 Zweck des Dokuments

Zusammenfassung der Ergebnisse der Phase „Konzept“.

2 Geschäftsprozesse

Da unser Programm einen sehr generellen Zweck hat und nicht für eine bestimmte Arbeitsumgebung optimiert ist, gibt es hier keinen Geschäftsprozess der konret beschrieben werde könnte.
Um die Einfachheit, und trotzdem die Bedeutung die dem Aufgabengebiet anzumessen ist aufzuzeigen haben wir aber ein Diagram, dass den (im Grunde sehr einfachen) Ablauf der zur Öffnung einer Datei nötig ist aufzeigt. Besonders da dieser Ablauf in einem Arbeitsalltag vielemale wiederholt wird kann durch die Optimierung dessen viel Zeit gewonnen und Frustration verhindert werden.

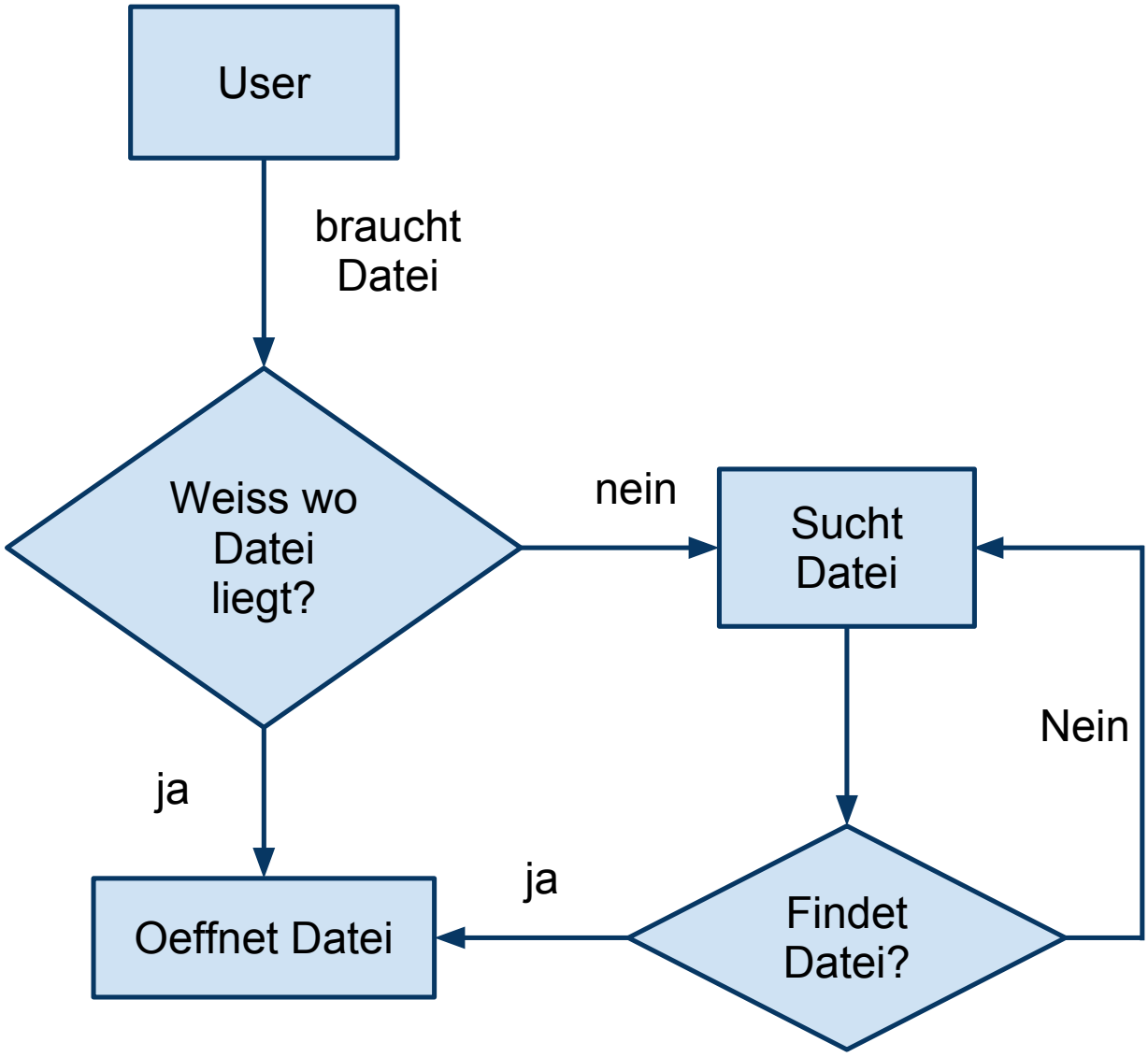

Figure 1: Arbeitsablauf beim Öffnen einer Datei

3 Systemanforderungen

3.1 Anforderungen an die Funktionalität

3.1.1 Funktionen

1. Dateien verwalten
2. Allgemein
   a) Dateien verschieben, kopieren, löschen
   b) Dateien per Doppelklick öffnen
   c) Datei-Eigenschaften anzeigen
      i. Grösse
      ii. Erstelldatum
      iii. Zugriffsrechte
      iv. zuletzt geöffnet
      v. der Datei zugeordnete Tags
   d) Obige Punkte können über ein drop-down erreicht werden, welches wiederum durch Rechtsklick auf eine Datei geöffnet wird.
   e) re-read erzwingen - zwingt das Programm die effektiven Datei-Daten von der Festplatte zu lesen und die Datenbank zu aktualisieren.
   f) Ansicht zwischen hierarchisch und tag-view wechseln.
   g) Link zur Datenbank anzeigen
   h) anzeigen alter Dateiversionen (falls backup eingeschaltet ist)
   i) Drag ‘n’ drop von Dateien/Ordnern in andere Ordner und aus den standard Dateimanagern der Systeme.
3. Hierarchische Ansicht
   a) Dateisystem-Struktur in einer Baumstruktur anzeigen (ähnlich wie in Microsoft Windows Explorer)
   b) Einen bestimmten Ordner anhand seiner URL direkt öffnen
4. Tag-view
   a) Alle Dateien mit einem bestimmten Tag anzeigen
   b) Alle Dateien mit einem bestimmten Tag innerhalb eines Ordners (reqursiv) anzeigen.
   c) Alle Dateien eines bestimmten Tags öffnen
   d) Alle Dateien eines bestimmten Tags kopieren (z.B. zur Sicherung)
   e) Tag umbenennen
5. Archivierung
   a) Kann für das ganze Dateisystem eingeschalten werden
      i. funktioniert aber natuerlich nur bei Dateien für die der User die nötigen Zugriffsrechte hat
      ii. Warnung bei auswahl grosser Gesamtgrösse (500MB?).
      iii. Warnung bei auswahl von Logfiles (viele modifikationen = viele backups).
   b) Kann für einzelne Ordner/ Dateien ein/ausgeschalten werden.
   c) Bei jeder Änderung wird eine Kopie der Datei in einem verschtekten Ordner abgelegt. Der Name der Datei wird dabei um die exakten Datum/Zeit Angaben ergänzt.
   d) Will der User später eine historische Version der Datei ansehen, so kann er dies über die Maske ‘historische Versionen’ die über den Rechtsklick-Dialog der Datei erreicht wird.
   e) Löschen aller historischer Versionen einer Datei
   f) Löschen aller historischer Versionen aller Dateien eines Ordners (reqursiv)
   g) Ob historische Versionen beim Löschen einer Datei auch gelöscht werden sollen kann in den Programmeinstellungen festgelegt werden.

3.1.2 Usecases

3.2 Bemerkungen

- Für das Anzeigen der hierarchischen Struktur der Verzeichnisse und Dateien haben wir hier keinen Usecase beschrieben, da dieser Vorgang wohlbekannt und für viele User Alltag ist.
- Dasselbe gilt für die Dateioperationen (z.B. kopieren, verschieben, . . .)

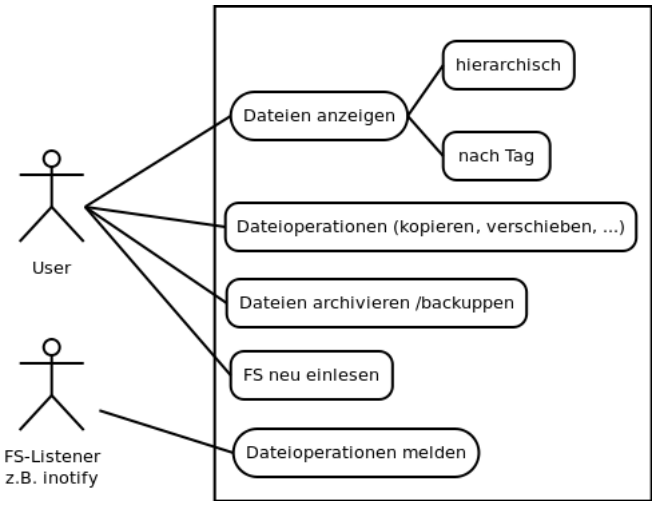

Figure 2: Usecases

Name	**Dateien anzeigen; nach Tag**
Kurzbeschreibung	Alle Dateien eines Tags anzeigen
Akteur(e)	User
Ergebniss(e)	• Alle Dateien die mit dem Tag versehen sind werden angezeigt. • Alle Dateien die mit einem Tag versehen sind und sich innerhalb eines bestimmten Ordners befinden werden angezeigt.
Eingehende Daten	Benutzeraktionen
Vorbedingungen	Das Tag muss vorhanden sein. Dem Tag müssen Dateien zugeordnet sein.
Nachbedingungen	Alle Dateien die mit dem Tag versehen wurden werden angezeigt.
Essenzielle Schritte	1. Bei Agieren aus der Tag-View a) Auswahl des gewünschten Tags aus der Liste der vorhandenen Tags. i. Dateien werden jetzt angezeigt b) Auswahl eines Filters i. Filter nach Ordner ii. Gewünschten Ordner auswählen iii. Dateien des gewünschten Tags die sich im gewählten Ordner (oder einem Unterordner dieses) befinden werden nun angezeigt 2. Bei Agieren aus Hierarchical-View a) Navigieren zum entsprechenden Ordner. b) Rechtsklick; Dateien eines Tags anzeigen; Tag wählen c) Die Ansicht wechselt jetzt automatisch zur Tag-View und Dateien des gewünschten Tags die sich im gewählten Ordner (oder einem Unterordner dieses) befinden werden nun angezeigt.
Offene Punkte	Es ist noch zu klären ob auch weitere Filter als nach Ordner zur Verfügung stehen sollen (z.B. Datumsbereich).
Änderungshistorie	22.03.2011; Lukas Knöpfel; Final

Name	**Archivierung (archivierte Datei ansehen)**
Kurzbeschreibung	Die historische Version eines Dokumentes wird angezeigt
Akteur(e)	User
Auslöser	• In einer alten Version eines Dokumentes wurde etwas gelöscht von dem speter bemerkt wird dass es doch noch benötigt wird. • Die aktuelle Version eines Dokumentes wird mit einer älteren verglichen.
Ergebniss(e)	Eine alte Version einer Datei wird geöffnet.
Eingehende Daten	Benutzeraktionen
Vorbedingungen	• Die entsprechende Datei muss zur Archivierung markiert sein. • Es müssen archivierte Versionen der Datei vorhanden sein.
Nachbedingungen	Eine historische Version der Datei ist geöffnet.
Essenzielle Schritte	1. Navigieren zur Datei. 2. Rechtsklick; Historische Versionen anzeigen. 3. Eine neue Maske öffnet sich in der alle historischen Versionen der Datei zusammen mit dem Datum an dem die Datei archiviert wurde angezeigt werden. 4. Ein Doppelklick auf die entsprechende Datei öffnet diese.
Offene Punkte	Werden in der alten Version der Datei änderungen vorgenommen, wohin werden diese dann gespeichert? Evtl. könnten alte Versionen einer Datei auch als ‘read only’ zur Verfuegung stehen.
Bemerkungen	In der Maske die die historischen Versionen anzeigt, befindet sich neben jeder Version ein Button, mit dessen Hilfe sich die historische Version wiederherstellen lässt. Dadurch wird die aktuelle Version archiviert und durch die archivierte überschrieben.
Änderungshistorie	• 20.03.2011; Niklaus Hofer; WIP • 28.05.2011; Niklaus Hofer; finished

Name	**FS neu einlesen; Systemdaten einlesen**
Kurzbeschreibung	Falls von einem externen Programm Dateisystem Operationen ausgeführt werden (z.B. verschieben einer Datei, Umebenennen einer Datei...) während unser Programm nicht läuft bemerkt das unser Programm nicht. Das kann zu ghost-files (Dateien die im Programm erscheinen, auf FS-Ebene aber nicht erscheinen) führen. Um das möglichst zu verhindern liest das Programm bei jedem Start die Daten vom Dateisystem ein und gleicht diese mit denen in der Datenbank ab. Der User kann aber auch zur Laufzeit manuell ein Neueinlesen der Daten anfordern.
Akteur(e)	User
Auslöser	Eine Datei die innerhalb des Programms noch erscheint kann nicht mehr geöffnet werden da sie auf FS Ebene nicht mehr verfügbar ist.
Ergebniss(e)	Das Programm bemerkt das Nichtvorhandensein der Datei und zeigt diese nicht mehr an.
Eingehende Daten	Benutzeraktionen. Struktur des Dateisystems.
Vorbedingungen	• Eine vom Programm überwachte Datei wurde verschoben. • Die Aktion wurde von Programm nicht bemerkt.
Nachbedingungen	Innerhalb des Programs wird die nicht mehr existente Datei nicht mehr angezeigt.
Essenzielle Schritte	1. In der Menuleiste wird der Eintrag Program; Neueinlesen des Dateisystems gewählt. a) Nun kann ausgewählt werden zwischen dem Aktualisieren aller überwachten Verzeichnisse und dem Aktualisieren bloss eines einzelnen Verzeichnisses (reqursiv).
Bemerkungen	Dieser Vorgang wird auch bei jedem Neustart des Programmes ausgeführt.
Änderungshistorie	• 20.03.2011; Niklaus Hofer; WIP • 28.05.2011; Niklaus Hofer; finished

Name	Dateioperationen melden
Kurzbeschreibung	Eine Datei wird ausserhalb des Programmes verschoben währen dieses läuft.
Akteur(e)	User
Auslöser	Eine Datei wird mit einem anderen Programm (z.B. mv, Windows Explorer, ...) verschoben oder kopiert.
Ergebniss(e)	Das Programm wird über die Aktion informiert und kann die Datenbank anpassen.
Eingehende Daten	Information über die Aktion durch einen FS-Listener.
Vorbedingungen	• Die Datei die verschoben/ kopiert/ gelöscht wird muss in einem überwachten Ordner liegen. • Das Programm muss laufen.
Nachbedingungen	Der Datenbankeintrag der entsprechenden Datei wurde automatisch angepasst. Falls die Datei kopiert wurde, so hat die neue Kopie dieselben Tags wie die alte Version. Sollte die Datei gelöscht werden und für die Archivierung markiert sein so wird sie zuvor noch in den Ordner mit den Archiven kopiert.
Essenzielle Schritte	1. Der Nutzer löst die Aktion aus indem er ausserhalb des Programmes eine Datei verschiebt/ kopirt/ löscht. 2. Ein Filesystem-Listener (unter Linux währe das z.B. iNotify[3], [1] unter OSX FSEvents) informiert das Programm über die Aktion. 3. Das Programm passt den Datenbankeintrag der entsprechenden Datei an.
Änderungshistorie	• 20.03.2011; Niklaus Hofer; WIP • 28.05.2011; Niklaus Hofer; finished

3.3 Anforderungen an die Daten

[2, 4.2]

3.4 Anforderungen an die Informationssicherheit und den Datenschutz

[2, 4.3]

4 Benutzerschnittstelle

4.1 Hierarchische Ansicht

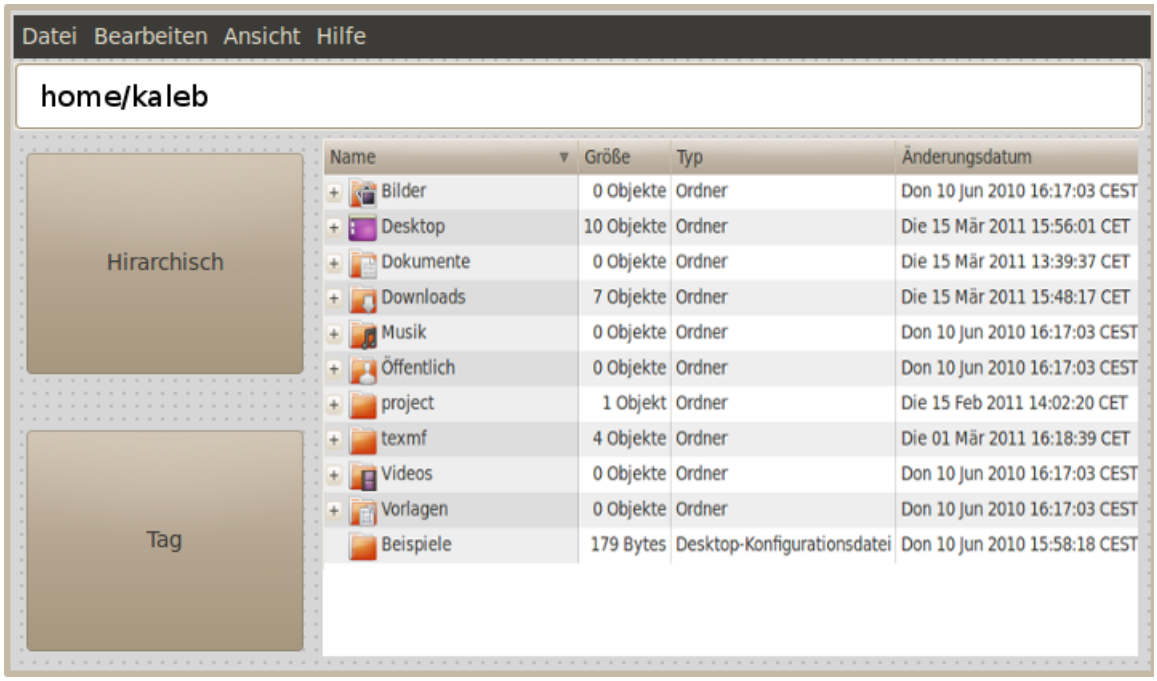

Figure 3: Programm im Modus zur hierarchischen Ansicht

4.2 Tag-Ansicht

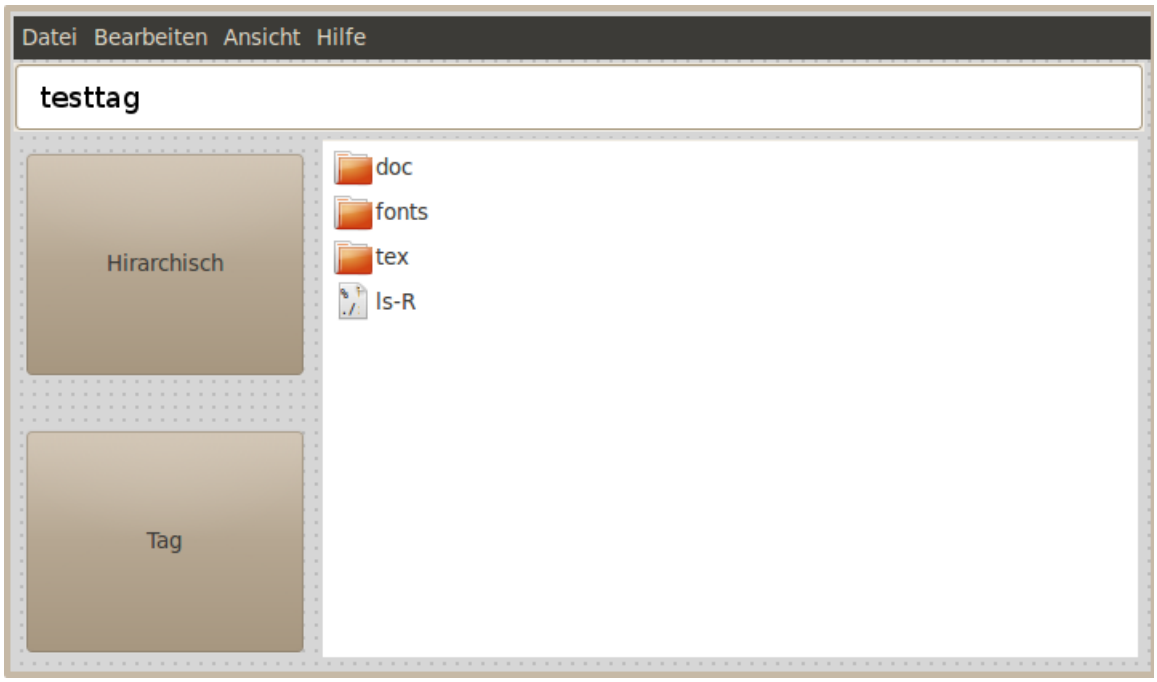

Figure 4: Programm im Modus zur Ansicht der Tags

4.3 Tags mit Datei verknüpfen

Figure 5: Einer Datei Tags zuweisen

4.4 Tags einer Datei ändern

Figure 6: Die einer Datei zugewiesenen Tags bearbeiten

5 Systemarchitektur

5.1 Gliederung der Lösung in Module

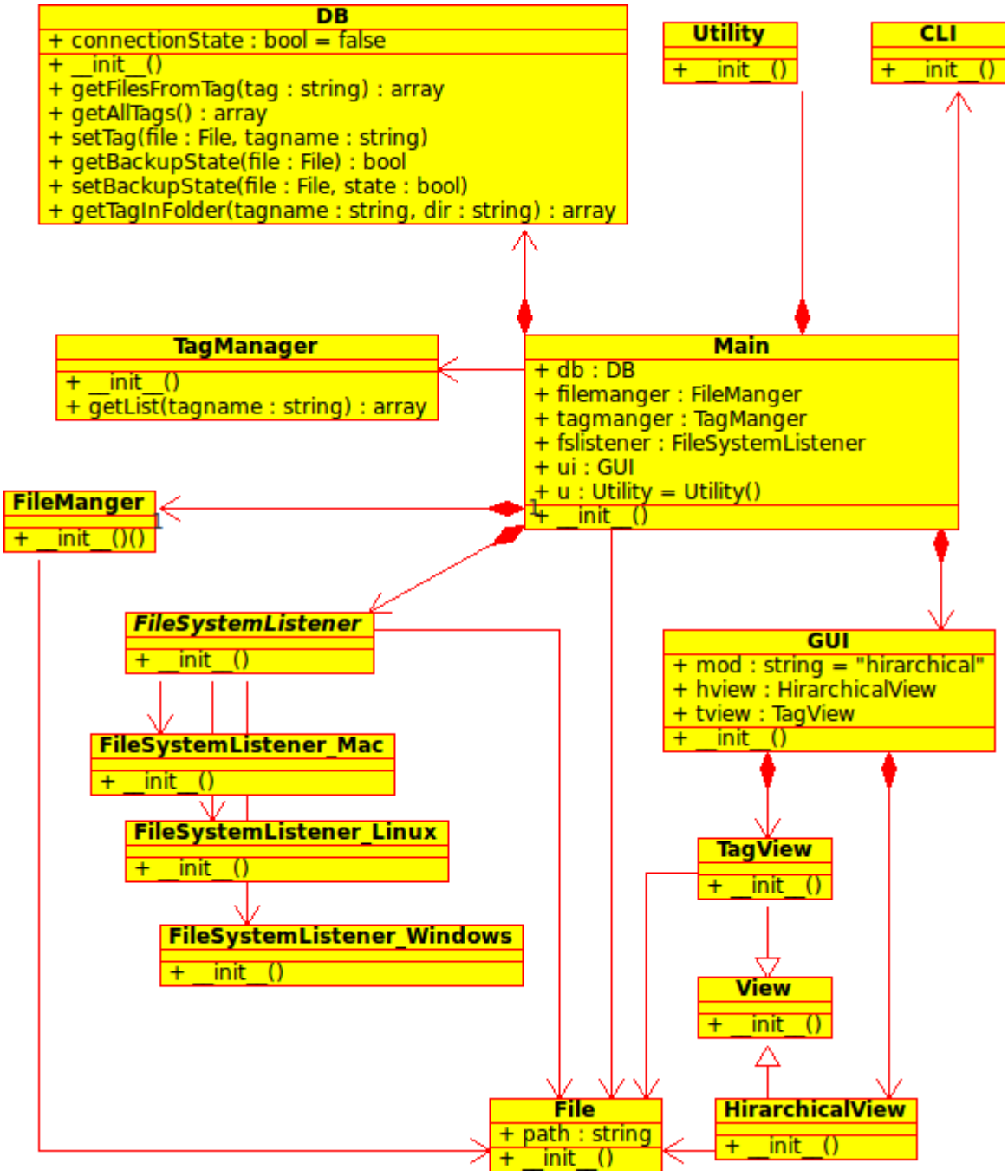

Figure 7: Entwurf des Klassendiagrammes

Die Klasse DB dient als Schnittstelle zur Datenbank und damit zum Datenmodell, wie unten aufgezeigt.
Eine Instanz der Klasse File repräsentiert jeweils eine virtuelle oder reale Datei auf dem Dateisystem und dient als entsprechender Informationsträger innerhalb der Applikation.

5.2 Schnittstellen

1. Interne Schnittstellen ergeben sich aus dem Klassendiagramm.
2. Externe Schnittstellen:
   a) Für den normalen Gebrauch haben wie das GUI. Beim GUI wird Wert auf das einfache verwalten von Tags und Dateien gelegt.

i. Das GUI bietet zwei Modi, einer der den herkömlichen Dateimanagern mit hierarchischer Ansicht entspricht und

ii. Einen Tagmodus, in dem sich Dateien anhand deren Tags durchsuchen und ordnen lassen.

b) Für scripting oder für solche Systeme ohne grafische Ausgabe haben wir eine CLI Version. Es wird besonders Wert auf das einfache Aufrufen von anderen Programmen (Scripts) gelegt.

i. Die Kommandos sollen in der Bedienung weitgehend mit dem standard Unix-Tools kompatibel sein (gleiche Parameternamen für gleiche Funktionen).

6 Datenmodell (grob)

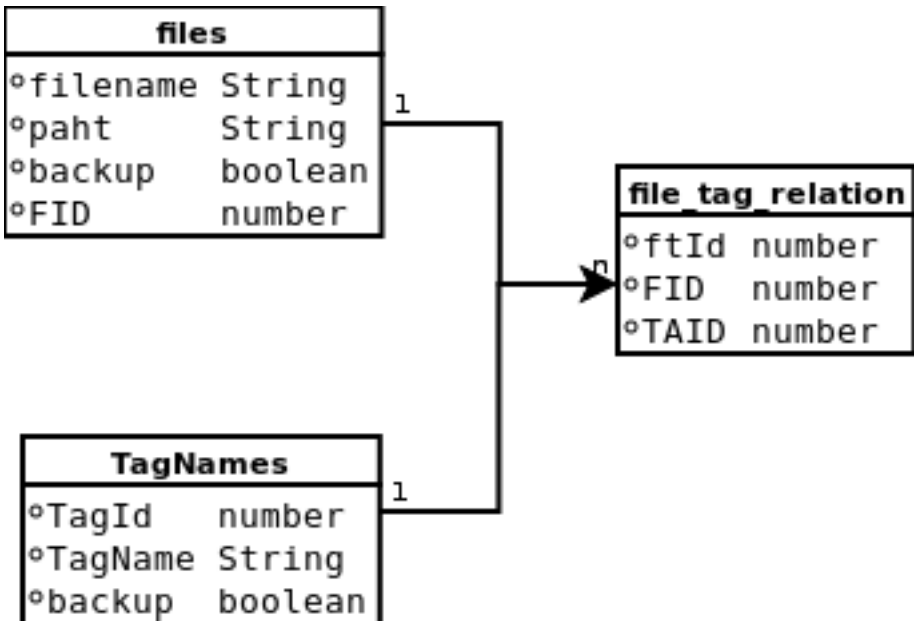

Figure 8: Datenmodell. Stellt die DB-Struktur dar.

7 Anforderungszuordnung

1. Main
2. DB
3. Utility
4. CLI
5. TagManager
6. FileManager
7. FileSystemListener
8. GUI
9. TagView
10. HierarchicalView
11. View
12. File

Nr.	Anforderungen	1	2	3	4	5	6	7	8	9	10	11	12
1	Das Programm starten												
2	GUI vorhanden												
3	CLI vorhanden												
4	Tag hinzufügen												
5	Datei auswählen												
6	Versionierung												
7	GUI: Versionierung für Tags												
8	Dateiänderungen werden erkannt												
9	Löschen wird erkannt												
10	Neue Dateien werden erkannt												
11	Hierarchische Anzeige												
12	Tag Anzeige												

8 Mittelbedarf

- Sachmittel
  - Programmierumgebung
  - Internetzugang
- Personal
  - Das Personal ist durch das Projekt vorgegeben.
  - Wir benötigen einen Projektleiter.
  - Da der Programmieraufwand aber recht gross ist, brauchen wir neben den zwei Vollzeitprogrammierern muss auch der Projektleiter nebenbei als Entwickler tätig sein.
- Ausbildung
  - Einige der Programmierer werden Python lernen müssen.
- Dienstleistungen
  - SVN Repository
  - Wiki
  - Kommunikation (Email)

9 Planung und Organisation

- erforderliches Projektteam (Zusammensetzung, Mitwirkung der Fachabteilungen)
- Meilensteine

10 Wirtschaftlichkeit

- Nutzen abschätzen
  - Der Nutzer ist sehr gross, weil man sich viel weniger mit dem Dateiverwalten ausseinandersetzen muss.
- Kosten abschätzen
  - Da wir eine kostenlose Programmiersprache und und Google Code verwenden haben wir keine direkte Kosten.
- Nutzen und Kosten gegenüberstellen
  - Grosser Nutzen und keine Kosten: Es lohnt sich.

11 Konsequenzen

Auswirkungen (organisatorisch, personell, baulich, Vorschriften/Weisungen)
Es stehen neue Wege zum Organisieren und Finden von Dateien zur Verfuegung. Das Organisieren von Projekten die sich ueber mehrere Ordner verteilen wird vereinfacht. Die Beteiligten sammeln Erfahrungen mit den im Projekt eingesetzten Technologien.

bei Nichtrealisierung
Schlechte Note.

bei verspäteter Realisierung (gegenüber Wunschtermin)
Schlechte Note

auf Schnitts.tellen zu anderen Systemen
Das Programm ist für den lokalen offline Einsatz vorgesehen. Onlinemodul falls wir genug Zeit haben.

Qualitätsverbesserungen
Einfacherer Umgang mit Dateien. Mehr Moeglichkeiten zur Verwaltung von Dateien auf dem lokalen Betriebssystem.

Risikobeurteilung
Da das Team viel neues lernen muss und das Projekt recht viele einzelne Teile beinhaltet ist das Risiko, dass es nicht gaenzlich fertiggestellt werden kann als beachtlich einzustufen. Wenn das Programm keine Vorteile gegenüber anderen Dateiverwaltungstools beinhaltet, gerät das Programm in Vergessenheit.

Ausweichmöglichkeiten Herkoemmliche Dateimanager (Die ohnehin unschlagbaren tools ls,cp,rsync,mv,rm,find,locate).

12 Antrag auf Freigabe der nächsten Projektphase

Wir bitten Sie um die Freigabe der nächsten Projektphase.