Berufsakademie Sachsen
Staatliche Studienakademie Leipzig

Pflichtenheft QuixSync

Projektarbeit
für Softwareprojekt im fünften Semester
in der Studienrichtung Informatik

Eingereicht von: Quentin Weber, 5000924
Philipp Ludwig, 5000964
Chris Sembritzki, 5000993
Seminargruppe: CS16-1

Inhaltsverzeichnis

1 Auftrag....4
1.1 Beteiligte Personen....4
1.2 Auftragsbeschreibung....4
1.3 Meilensteine....4
1.4 Dokumentation....4
2 Anforderungen....5
2.1 Pflichtanforderungen....5
2.2 Wunschanforderungen....5
2.3 künftige Anforderungen....5
2.4 Machbarkeit....6
3 Produkteinsatz....6
3.1 Anwendungsbereiche....6
3.2 Zielgruppen....6
3.3 Betriebsbedingung....6
4 Produktübersicht....7
4.1 Kurze Produktübersicht....7
4.2 UML-Diagramme....8
4.2.1 Ablauf einer kompletten Synchronisation....8
4.2.2 Indizierung....9
4.2.3 Vergleich....10
4.2.4 Synchronisierung....11
4.3 ER Diagramme....12
4.3.1 Speicherelement....12
4.3.2 File Size....12

5 Produktfunktionen 13
5.1 Analysieren 13
5.2 Vergleichen 13
5.3 Synchronisierung 14
5.4 FTP-Verbindungen 14
5.5 Daemon Betrieb 14
6 Benutzeroberfläche 15
7. Abbildungsverzeichnis 20

1 Auftrag

1.1 Beteiligte Personen

Auftragnehmer:

Projektleiter:

Quentin Weber

- Quentin.Weber@cs16-1.ba-leipzig.de

Projektteam:

Philipp Ludwig

- Philipp.Ludwig@cs16-1.ba-leipzig.de

Chris Sembritzki

- Chris-Thomas.Sembritzki@cs16-1.ba-leipzig.de

Auftraggeber:

Christian Heller

- Christian.Heller@ba-leipzig.de

1.2 Auftragsbeschreibung

Das Ziel des Projektes ist das Erstellen eines Tools zur Synchronisation von Verzeichnissen und Dateien. Dies soll lokal auf einem Rechner oder über das Netzwerk erfolgen. Dem Nutzer wird für diesen Zweck eine grafische Oberfläche zur Verfügung gestellt.

1.3 Meilensteine

Lastenheft/Thema:	Freitag, den 05.10.2018 um 12:00 Uhr
Pflichtenheft:	Freitag, den 19.10.2018 um 12:00 Uhr
Softwareprototyp:	Freitag, den 07.12.2018 um 12:00 Uhr
Präsentation:	Freitag, den 14.12.2018 um 08:00 Uhr

1.4 Dokumentation

Die Dokumentation und Versionierung erfolgt über GitHub (https://github.com/BuckUbel/QuixSync).

2 Anforderungen

Die Anforderungen werden in Form von User Stories dargestellt. Diese beschreiben einen spezifischen Anwendungsfall, welcher von der Anwendung ausgelöst werden soll.

2.1 Pflichtanforderungen

Als Anwender möchte ich einen konkreten Ordner mit einem anderen vergleichen. Als Ergebnis dieses Vergleiches sollen mir die Unterschiede aufgelistet werden.

Als Anwender möchte ich einen bestimmten Ordner an verschiedenen Stellen innerhalb des Dateisystems ablegen. Diese verschiedenen Ordner sollen den selben Informationsgehalt und Struktur besitzen.

Als Anwender möchte ich eine graphische Benutzeroberfläche für die Konfiguration und Bedienung des Programmes verwenden..

2.2 Wunschanforderungen

Als Anwender möchte ich nicht nur lokale, also auf dem aktuellen Rechner erreichbare, Ordner synchronisieren. Ein Abgleich zweier Ordner über FTP wäre wünschenswert.

Als Anwender brauche ich einen automatisierten Abgleich der zu synchronisierenden Ordner. Das Zeitintervall dafür kann selbst gewählt werden. Dies sorgt dafür, dass ein manuell angestoßener Abgleich nicht mehr zwingend durchgeführt werden muss, um die beiden Ordner auf dem aktuellen Stand zu halten.

2.3 künftige Anforderungen

Die nachfolgenden User Stories beschreiben Anforderungen, die in der Zukunft der Anwendung hinzugefügt werden könnten.

Als Anwender möchte ich eine auf UNIX-Systemen optimierte Dateisynchronisation durchführen.

2.4 Machbarkeit

Der angegebene Auftrag ist mit der Programmiersprache Java umsetzbar. Der Betrieb der zu entwickelnden Software ist hardwareseitig nicht sehr Ressourcenbeanspruchend und sollte damit auf jeder Konfiguration betrieben werden können.

Als reine Entwicklungszeit bis zum Einreichen des ersten Prototyps beim Auftraggeber werden xx Stunden veranschlagt.

3 Produkteinsatz

3.1 Anwendungsbereiche

Dieses Programm hat die Funktion zwei Verzeichnisse zu analysieren, zu vergleichen und schlussendlich zu synchronisieren. Die Synchronisation eines Ordners kann auch als Backup für eben jenes Verzeichnis dienen. Dies kann über im Dateisystem eingebundene Ordner erfolgen.

3.2 Zielgruppen

Dieses Programm ist besonders für Personen interessant, die häufig auf unterschiedlichen Rechnern arbeiten müssen und den aktuellen Stand eines Ordners auf allen Geräten zur Verfügung gestellt haben wollen.

3.3 Betriebsbedingung

Der Benutzer muss ein Desktop-Betriebssystem mit der aktuellen Version von Java 8 verwenden. Allerdings wird im Rahmen dieses Projektes die Synchronisierung ausschließlich auf Windows 10 getestet.

Soll eine FTP-Verbindung erstellt werden, muss eine Netzwerkverbindung zu dem entsprechenden FTP-Server realisiert werden können.

4 Produktübersicht

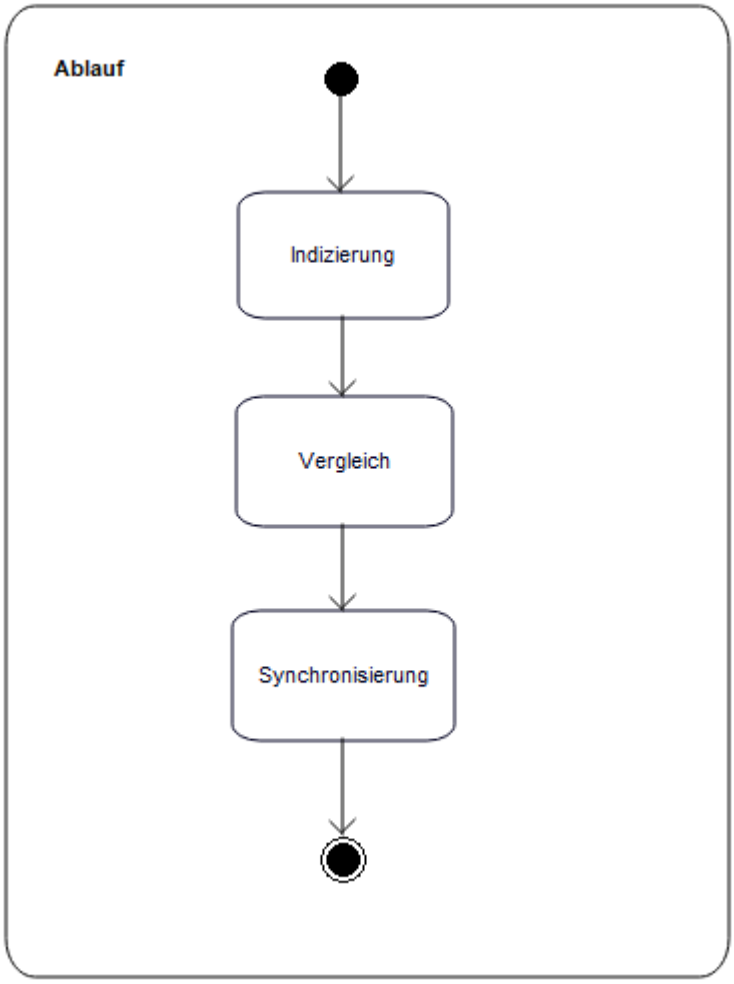
4.1 Kurze Produktübersicht

QuixSync wird über eine grafische Oberfläche bedient. Diese soll dem Nutzer folgende Funktionen zur Verfügung stellen:

- Auswählen zweier Verzeichnisse (Quellverzeichnis, Zielverzeichnis)
- Erstellen einer Indexdatei für ein Verzeichnis
- Vereinfachtes Präsentieren einer Indexdatei
- Erstellen einer Vergleichsdatei zwischen zwei Indexdateien
- Vereinfachtes Präsentieren einer Vergleichsdatei
- Starten einer weichen Synchronisation aufgrund einer Vergleichsdatei
- Starten einer harten Synchronisation aufgrund einer Vergleichsdatei
- Starten einer kompletten Synchronisation
  (Erstellen und Vergleich von Indexdateien mit inbegriffen)
- Setzen von Nutzer-spezifischen Einstellungen

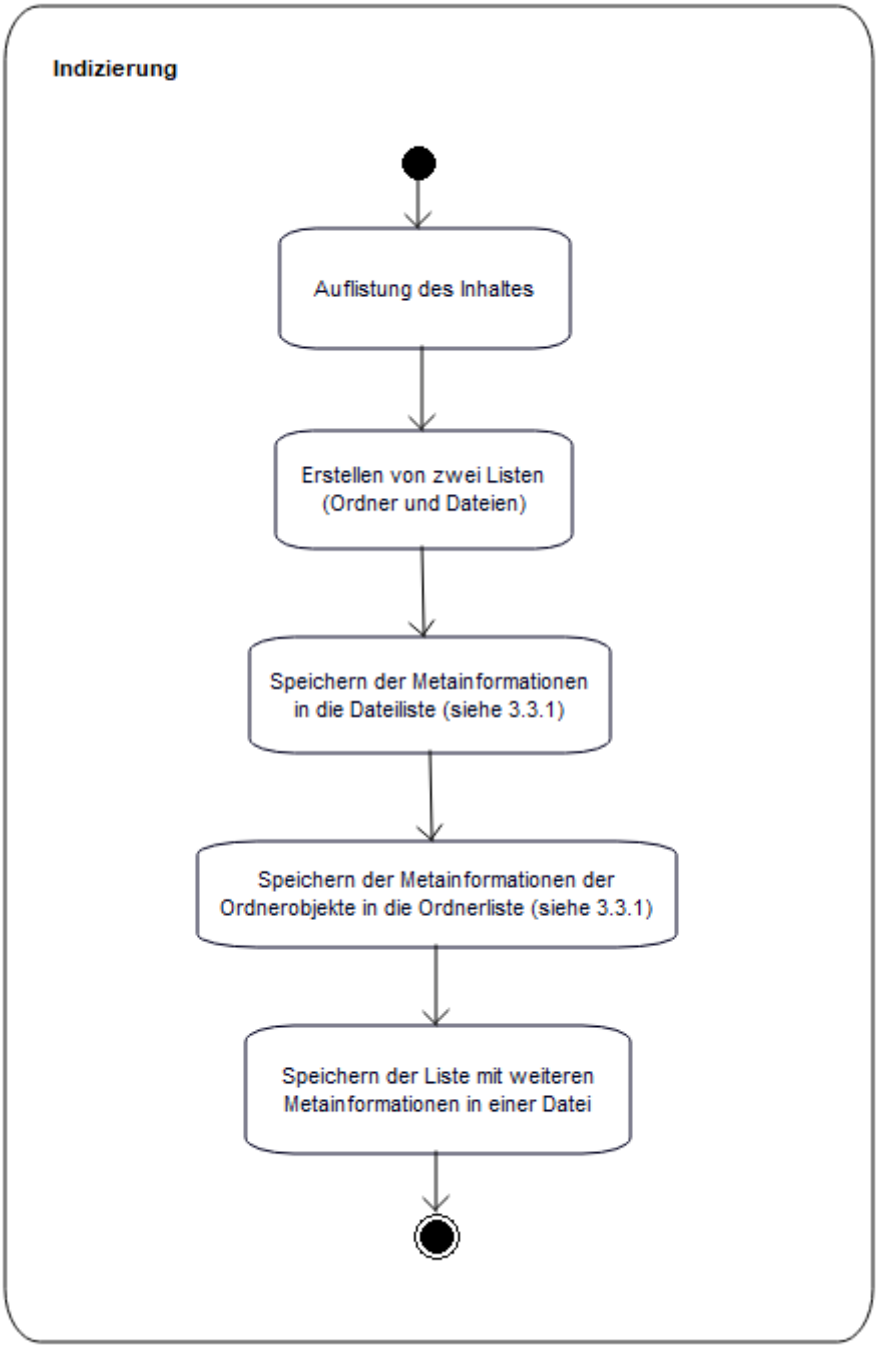
4.2 UML-Diagramme

4.2.1 Ablauf einer kompletten Synchronisation



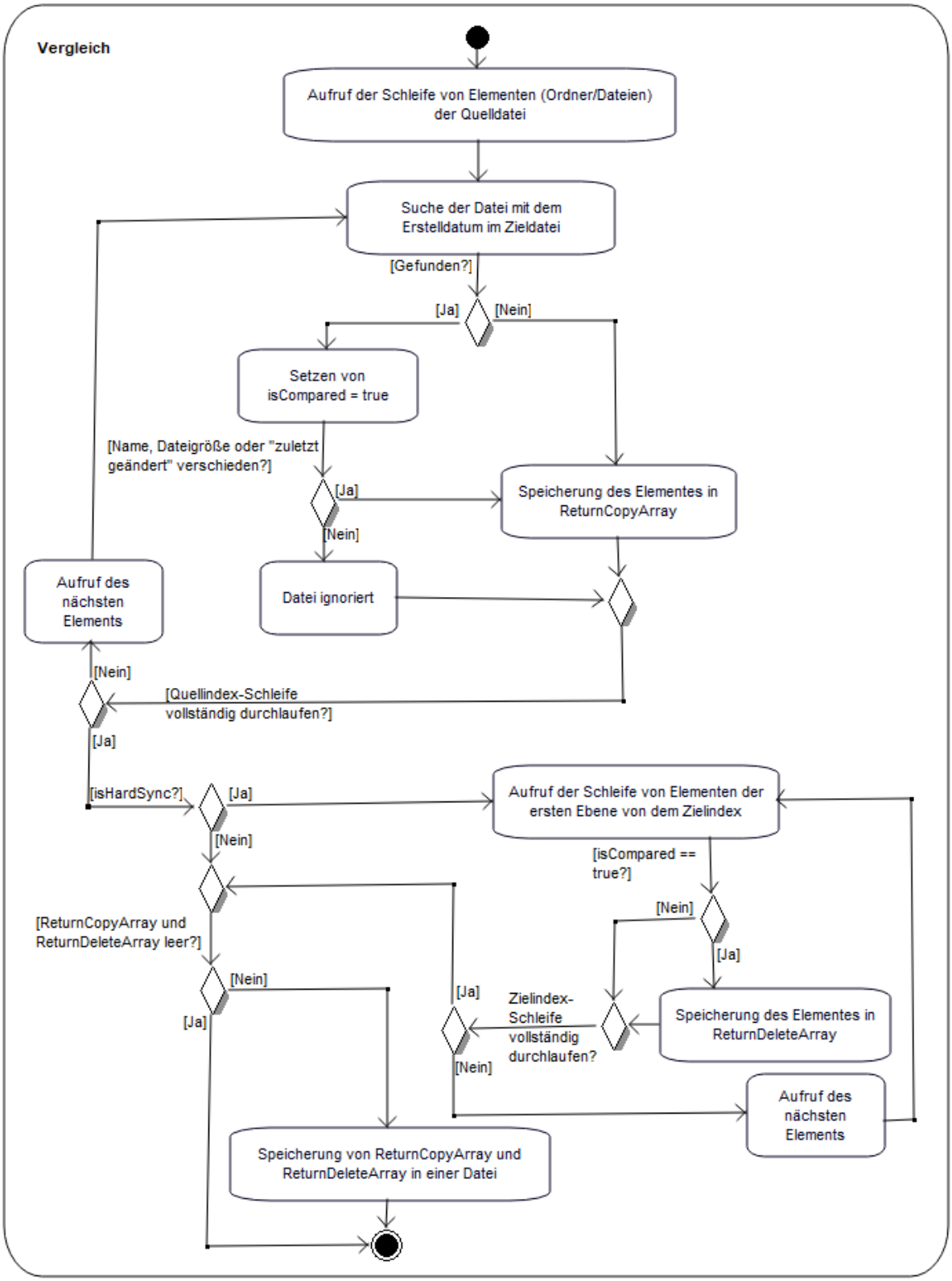
4.2.2 Indizierung

Parameter: Verzeichnis



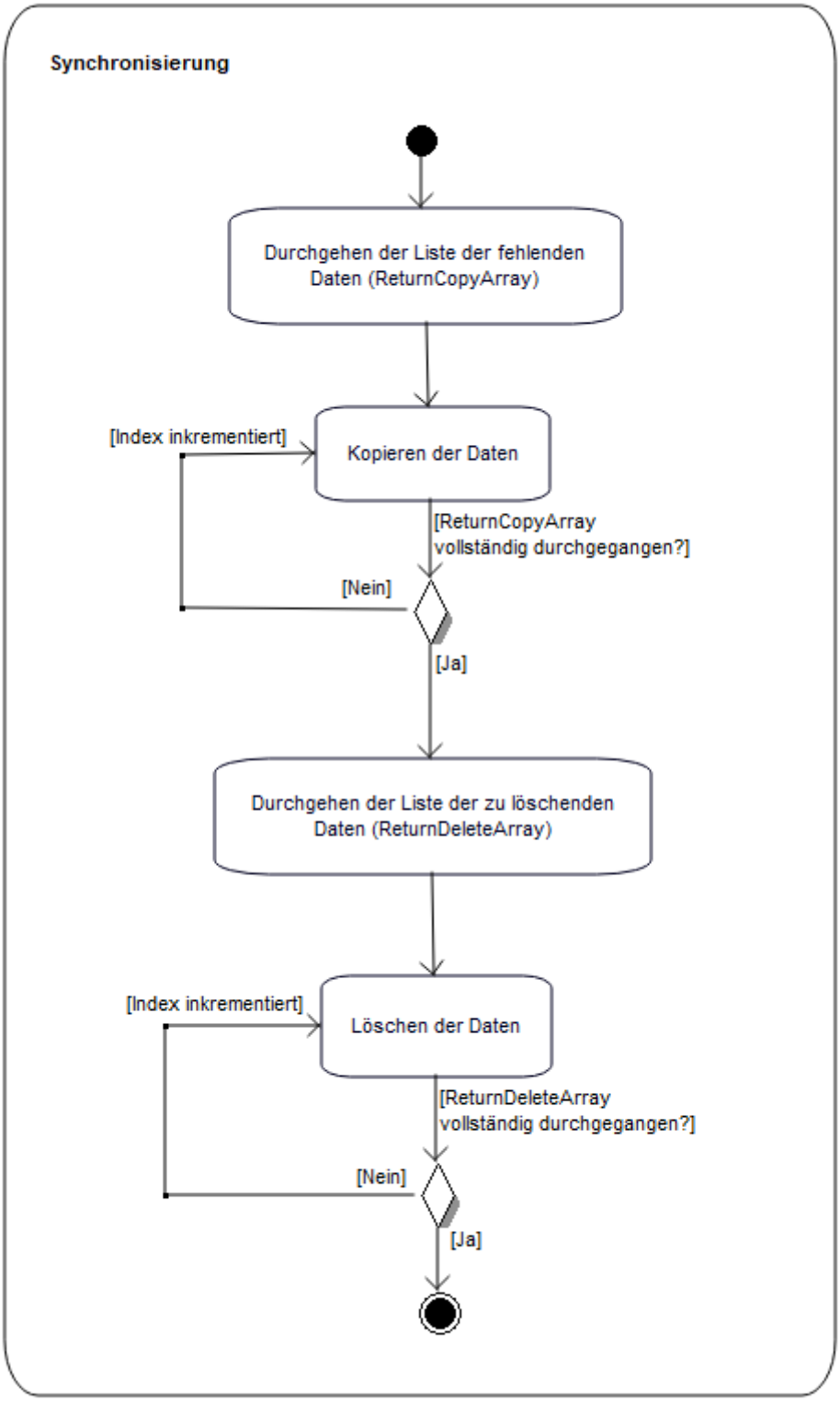
4.2.3 Vergleich

Parameter: zwei Indizierungsdateien, isHardSync



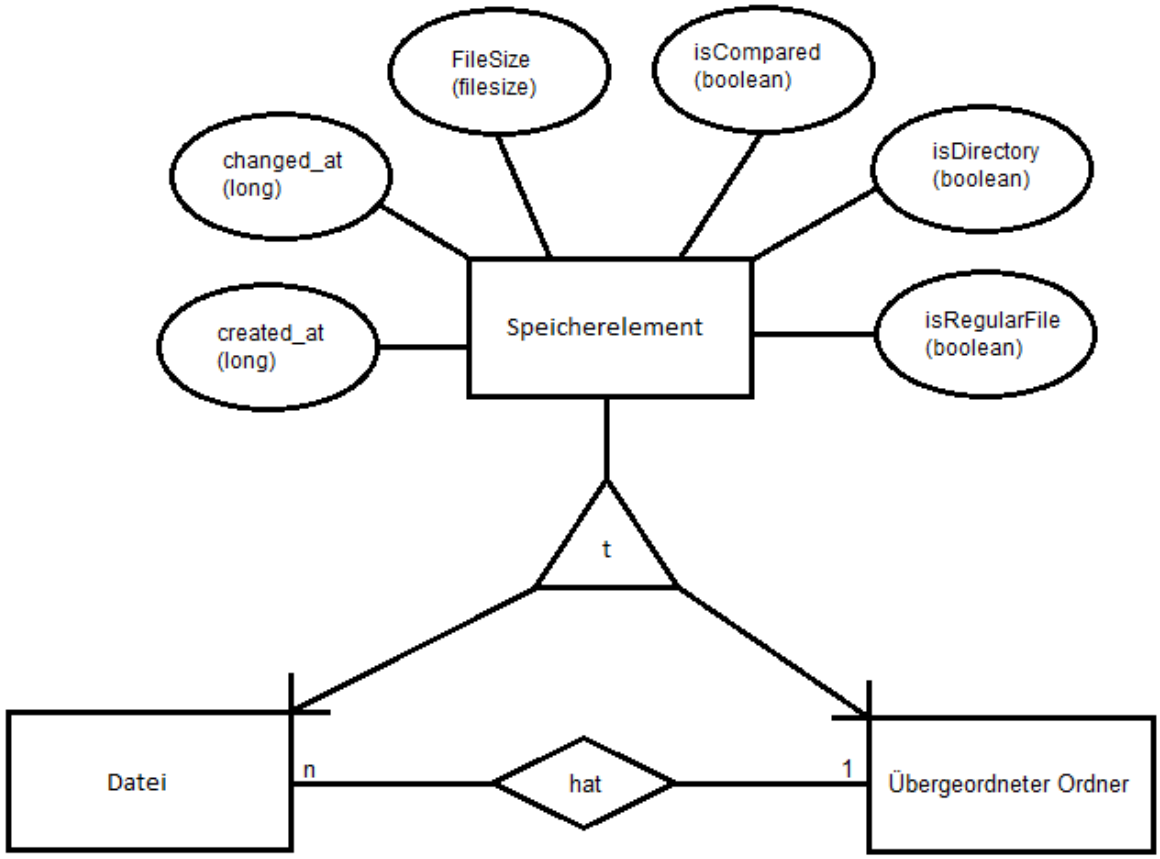
4.2.4 Synchronisierung

Parameter: Vergleichsdatei

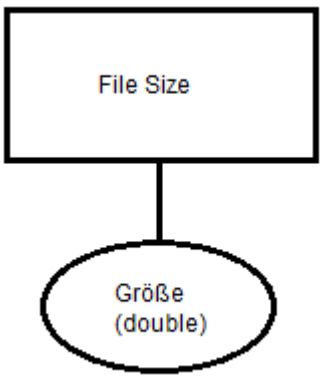


4.3 ER Diagramme

4.3.1 Speicherelement



4.3.2 File Size



5 Produktfunktionen

5.1 Analysieren

In diesem Punkt wird ein Verzeichnis indiziert. Das heißt man gibt ein Verzeichnis an und zu diesem wird eine Index-Datei hergestellt. Diese beinhaltet jede Datei und jeden Ordner. Zusätzlich werden auch Metadaten, wie Erstelldatum, Änderungsdatum und Dateigröße gesammelt.

Dazu wird durch eine rekursive Funktion der angegebene Ordner durchsucht und die entsprechenden beinhalteten Elemente werden in einer Liste gespeichert. Dabei werden zwei Listen geführt. Eine für die Dateien und eine für die Verzeichnisse. Somit kann eine Trennung dieser beiden Datentypen erfolgen. Diese zwei Listen werden nach der Befüllung in die Indexdatei geschrieben, gemeinsam mit der Dateianzahl und der Datenmenge.

5.2 Vergleichen

Für den Vergleich benötigt man nun zwei dieser Indexdateien. Diese stehen nun für das Ziel- und das Quellverzeichnis. Die Quell-Indexdatei wird für jedes einzelne Element durchgegangen. Dabei wird geschaut, ob dieses Element auch in der Ziel-Indexdatei zu finden ist. Als Grundlage für den Vergleich das Erstelldatum benutzt, denn ein Element kann jederzeit umbenannt werden, jedoch bleibt das Erstelldatum dabei gleich. Wenn eine Element gefunden wird, welches fehlt oder verändert wurde, wird dessen absoluter Pfad in einer neuen Liste gespeichert. Diese wird später dafür genutzt um diese Elemente in den Zielordner zu kopieren. Weiterhin bekommt das Element in der Ziel-Indexdatei ein Flag („isCompared“) dafür, dass es verglichen wurde.

Wenn alle Dateien verglichen wurden, wird in eine weiche Synchronisierung und eine harte Synchronisierung unterschieden. Bei der weichen Synchronisierung werden nur Elemente aus dem Quellverzeichnis in den Zielverzeichnis kopiert. Bei der harten Synchronisierung, werden Daten aus dem Zielordner gelöscht, wenn sie nicht auch im Quellordner vorkommen. Dafür muss eine weitere Schleife durchgeführt werden. In dieser werden alle Elemente in der Ziel-Indexdatei in der Quell-Indexdatei verglichen,

so lange sie noch nicht in der vorherigen Schleife verglichen wurde. (Flag „isCompared“). Wenn dies bei einem Element der Fall ist, wird der absolute Pfad dieses Objektes in eine weitere neue Liste gespeichert. Diese wird später genutzt um diese Elemente zu löschen, da sie nicht im Quellverzeichnis vorgekommen sind.

5.3 Synchronisierung

Die Synchronisierung nutzt nun diese beiden Listen. Die erste Liste mit den fehlenden Elementen in dem Zielverzeichnis, und die zu löschenden Elemente in dem Zielverzeichnis. So wird durch ein Schleife alle zu kopierenden Elemente durchgegangen und werden in das entsprechende Verzeichnis kopiert. Wenn die Schleife abgeschlossen ist, wird die nächste angestoßen, welche die Elemente im Zielverzeichnis löscht, die nicht im Quellverzeichnis vorkamen.

5.4 FTP-Verbindungen

Bei einer Synchronisierung über FTP verändern sich nur die Funktionen Analyse und Synchronisierung. Denn hierfür muss erst eine FTP-Verbindung etabliert werden und gegebenen Falls Anmeldedaten erfasst werden.

5.5 Daemon Betrieb

Der eventuelle einzubindende Daemon Betrieb soll das automatische Synchronisieren ermöglichen. Dafür soll ein Verzeichnis überprüft werden, welches bei einer Änderung mit einem anderen synchronisiert wird und die Änderung mit überträgt. Zwischen zwei Synchronisationen soll eine bestimmte Zeitspanne liegen, damit nicht immer wieder eine neue Synchronisierung angestoßen wird.

6 Benutzeroberfläche

Die nachfolgenden Abbildungen stellen grobe Vorstellungen der Anwendung dar. Einige Funktionen sind darin noch nicht so enthalten wie sie später tatsächlich verbaut werden. Jedoch kann man so schon einen Eindruck der Anwendung erhalten.

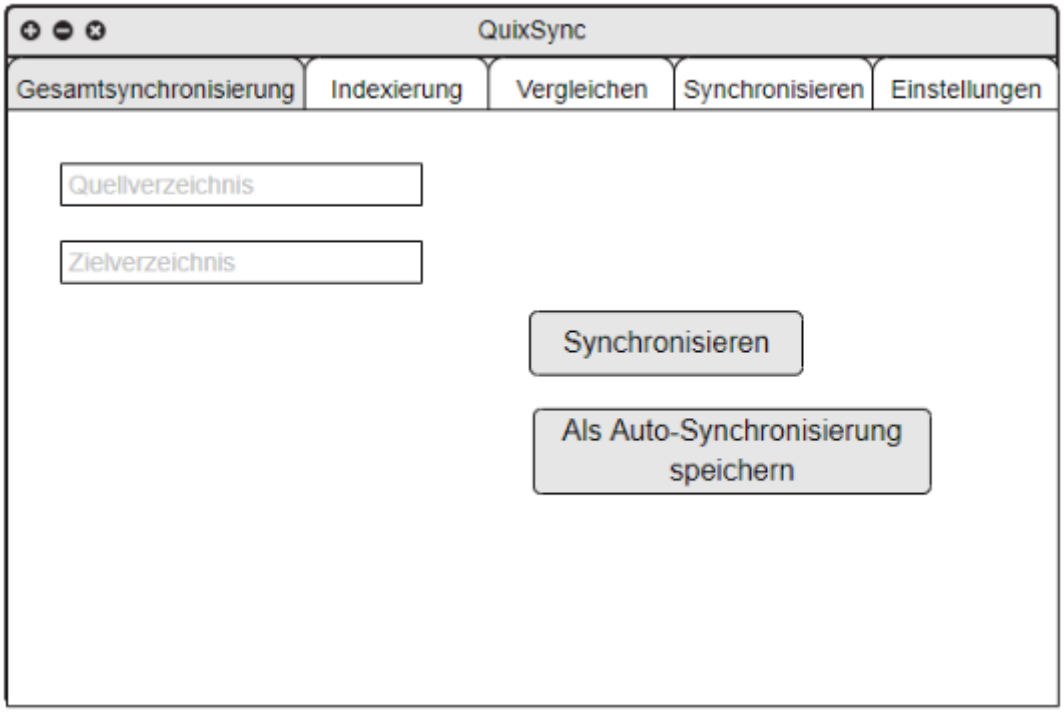


Abbildung 1 Hauptseite

Auf der Hauptseite der graphischen Oberfläche wird die Gesamtsynchronisierung angeboten, bei der alle notwendigen Schritte nacheinander ablaufen. Dazu müssen nur die beiden Verzeichnisse angegeben werden. Hier kann auch für den eventuellen Daemon-Betrieb eine Synchronisierung abgespeichert werden, welche dann automatisch abgerufen wird.

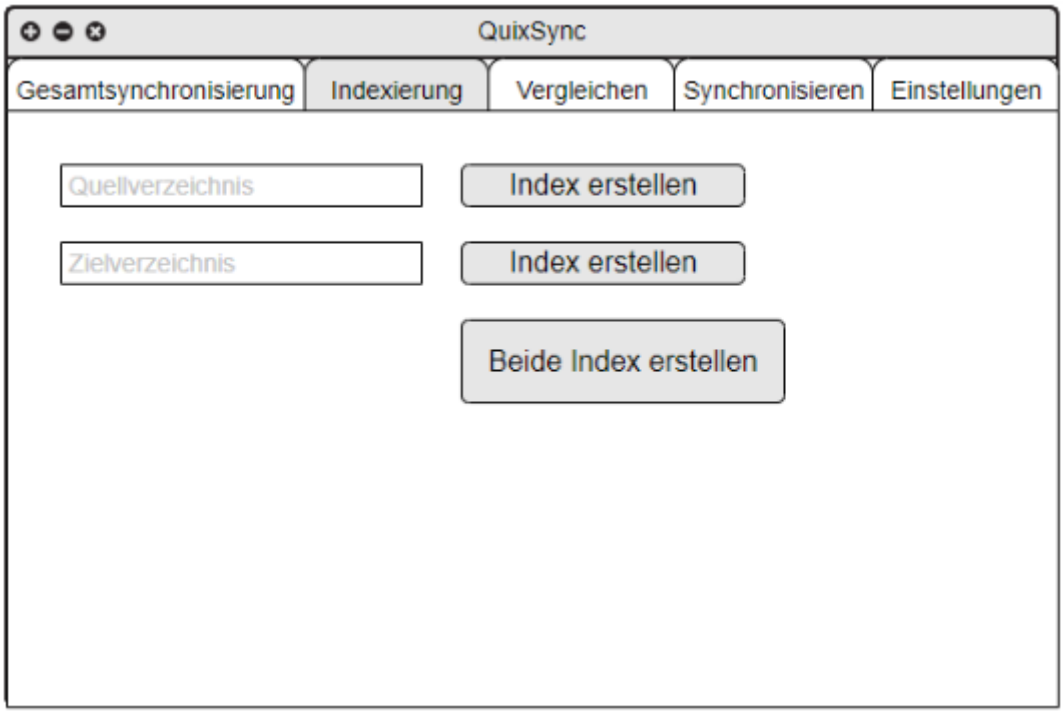


Abbildung 2 Indexierung

Auf dieser „Seite“ der Anwendung kann manuell ein Index erstellt werden, welcher in einem temporären Verzeichnis gespeichert wird.

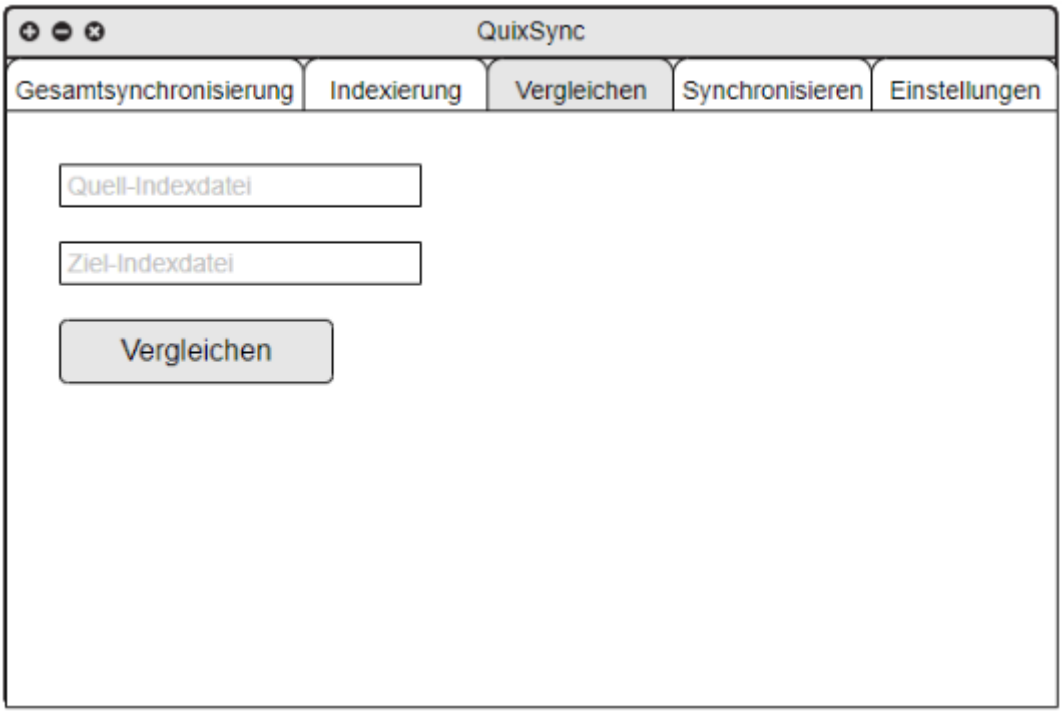


Abbildung 3 Vergleichen

Unter dem Tab „Vergleichen“ kann eine Vergleichsdatei erstellt werden, welche aus dem Vergleich der zwei Indexdateien resultiert wird. Diese soll im Anschluss auch in dieser Anzeige mit angezeigt werden können. Jedoch wurde diese Anzeige in dieser Vorschau noch nicht mit realisiert.

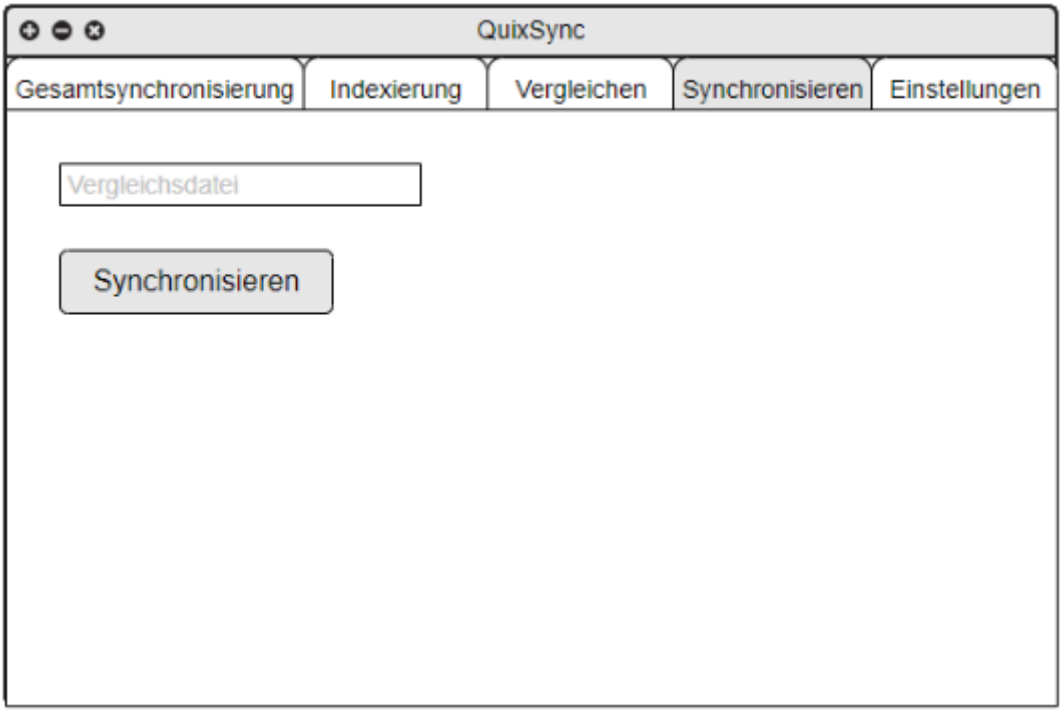


Abbildung 4 Synchronisierung

Hier wird eine Vergleichsdatei gefordert, anhand das Programm eine Synchronisierung starten kann.

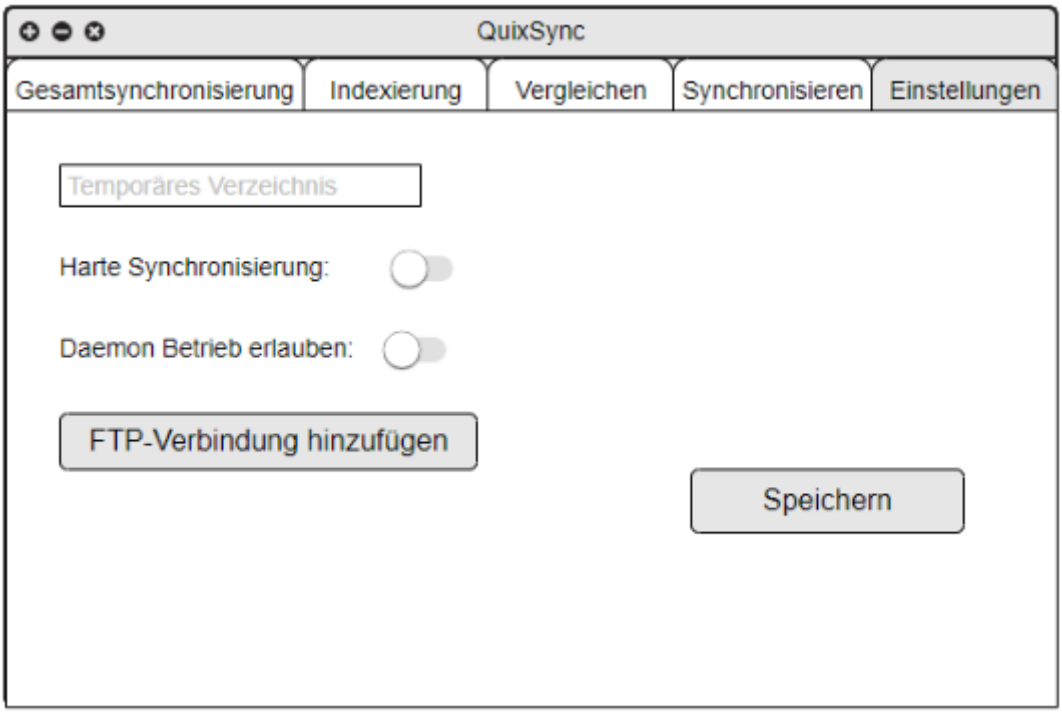


Abbildung 5 Einstellungen

In dem Tab „Einstellungen“ kann das temporäre Verzeichnis verändert werden. Ebenfalls kann hier zu einer harten Synchronisierung umgeschaltet werden.

7. Abbildungsverzeichnis

Abbildung 1 Hauptseite 15
Abbildung 2 Indexierung.....16
Abbildung 3 Vergleichen 17
Abbildung 4 Synchronisierung.....18
Abbildung 5 Einstellungen.....19