GEORG-SIMON-OHM
HOCHSCHULE NÜRNBERG

Projektarbeit

Systemprogrammierung mit Perl
Archivierung

Technische Hochschule Nürnberg

Georg Simon Ohm

Fakultät Informatik

Wintersemester 2014/15

Gruppenmitglieder/Matrikelnummer:	**Michel Angelo Ramunno**	**2428854**
	Patrick Vogt	**2606768**
	Michaela Fentze	**2230617**
	Muhammed Kasikci	**2428520**

Eingereicht bei:	**Prof. Dr. A. Hein**
Abgabetermin:	**22.01.2015**

Ehrenwörtliche Erklärung

Wir versichern, dass wir die Arbeit selbständig verfasst, nicht anderweitig für Prüfungszwecke vorgelegt, alle benutzten Quellen und Hilfsmittel angegeben sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet haben.

Inhalt

1. Motivation

Warum sollen Daten archiviert werden? Diese Frage lässt sich leicht beantworten. Es gibt gesetzliche Vorgaben, die verlangen, dass bestimmte Daten über einen längeren Zeitraum aufbewahrt werden. Außerdem kommt hinzu, dass bei einem Datenverlust die Daten aus dem Archiv wieder hergestellt werden können. Dies sind nur ein paar wenige von zahlreichen Gründen warum man Daten archivieren sollte, deshalb wurde eine Konsolen-Anwendung zur Archivierung entwickelt.

2. Projektgruppe und Zuständigkeiten

Name	Zuständigkeit
Patrick Vogt	Invoker (Aufruf von Create, Restore, Delete, List), List, Verbosity und Programmhilfe
Michel Angelo Ramunno	Create und Klassendiagramme
Michaela Fentze	Delete
Muhammed Kasikci	Restore und Profiling

3. User Guide

Die Anwendung wird über die Konsole gestartet. Die folgende System-Voraussetzung muss zur erfolgreichen Ausführung erfüllt sein: Perl 5.8 oder höher.
Nachfolgend werden die verschiedenen Aufrufmöglichkeiten der Anwendung beleuchtet.

```
> perl my_perl_archive.pl –h
```

Dieser Aufruf stellt die Programmhilfe auf der Konsole dar. Die Programmhilfe enthält eine detaillierte Beschreibung der Verwendung der Switches und Angaben über die Autoren.

```
> perl my_perl_archive.pl [–v] –c [Quellverzeichnis] [Zielverzeichnis]
```

Dieser Aufruf erstellt im Zielverzeichnis ein neues Archiv vom Quellverzeichnis. Das erstellte Archiv hat einen Verzeichnisnamen der Form SOURCE_yyyy_mm_dd_hh_ii_ss und ist eine 1:1-Kopie des Quellverzeichnisses.
Die Abbildungen von Hashes auf Verzeichnispfade kann in den Archivverzeichnissen in der Datei hashtable.txt eingesehen werden.
Der Switch -v ist optional und aktiviert bei Verwendung den Verbose-Mode. Durch Angabe einer Zahl zwischen 1 bis 9 wird zusätzlich ein Verbose-Level gesetzt. Das Level 1 aktiviert die Default-Verbose-Ausgabe, das heißt es werden zusätzliche Informationen zu den durchgeführten Aktionen ausgeben. Die Level 2 bis 8 sind bisher noch nicht vergeben.

```
> perl my_perl_archive.pl [-v] -c -s [Quellverzeichnis] [Zielverzeichnis]
```

Wert	Bedeutung
SOURCE	Hexadezimaler Hash
yyyy	vierstellige Jahreszahl
mm	Monat
dd	Tag
hh	Stunde
ii	Minute
ss	Sekunde

Bei diesem Aufruf handelt es sich um die „verschlankte Archivierung“. Hierbei wird zunächst ein neues Archiv erstellt. Anschließend wird überprüft, ob vorhergehende Archive vorhanden sind. Wenn dies der Fall ist, dann wird überprüft, ob Dateien vorhanden sind, die zum Vorgängerarchiv keinerlei Änderungen haben. Diese Dateien werden im Vorgängerarchiv durch Links zum aktuelleren Verzeichnis ersetzt.

```
> perl my_perl_archive.pl [-v] -s [Archivverzeichnis]
```

Dieser Aufruf verschlankt ein Archiv wenn vorhergehende Archive mit unveränderten Dateien vorhanden sind. Alle unveränderten Dateien werden wie beim vorher dargestellten Aufruf durch Links ersetzt.

```
> perl my_perl_archive.pl [-v] -r [-p] [Quellverzeichnis] [Zielverzeichnis] [Archivname] [Zeitstempel] [[Partial-Objekt]]
```

Dieser Aufruf stellt ein Archiv wieder her. Bei der Angabe des optionalen Switches -p kann auch nur ein Teil eines Archivs, also ein Unterverzeichnis oder eine einzelne Datei wiederhergestellt werden. Durch den Zeitstempel wird das zuletzt gültige Archiv gesucht.

```
> perl my_perl_archive.pl [-v] -d [Zu löschendes Objekt]
```

Dieser Aufruf löscht das angegebene Objekt. Dies kann entweder ein ganzes Archiv, ein Unterverzeichnis eines Archivs oder eine einzelne Datei sein.

```
> perl my_perl_archive.pl [-v] -l [Archivverzeichnis] [Zeitstempel]
```

Dieser Aufruf listet den Inhalt des, vom Zeitstempel ausgehend zuletzt gültigen Archivs auf. Bei der Auflistung wird zwischen Verzeichnissen, Links (Verweisen) und normalen Dateien unterschieden.

```
> perl my_perl_archive.pl -m [Archiv] [[Name einer Abbildungstabelle]]
```

Dieser Aufruf listet alle Abbildungen von Hashes auf Verzeichnispfade auf. Falls die Datei Abbildungstabelle umbenannt wurde, so kann man als zusätzlichen Parameter den Namen der Abbildungstabelle angeben. Der Name der Abbildungstabelle wird nur benötigt, falls die Datei umbenannt wurde anderenfalls wird der Standardname „hashtable.txt“ angenommen.

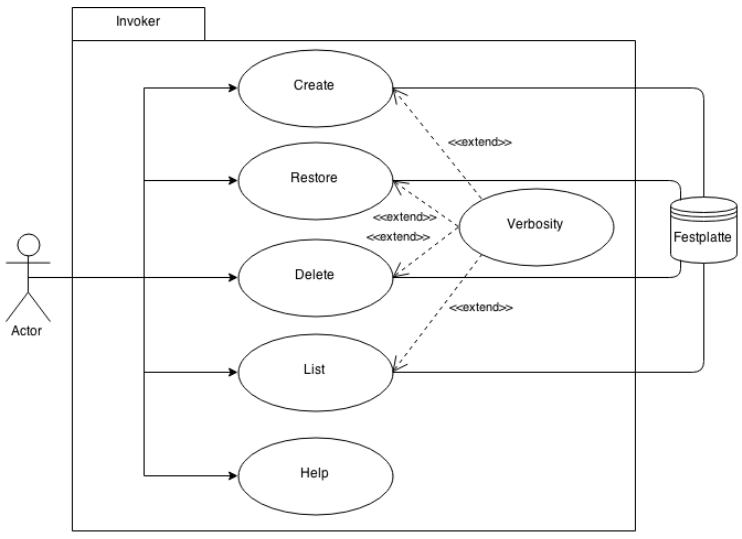
Zu den Switches werden auch noch ausführliche Schreibweisen angeboten. Nachfolgend nochmal alle Switches in einer Tabelle:

Switch kurz	Switch lang	Funktion
-h	--help	Ruft die Programmhilfe auf.
-c	--create	Erstellt ein neues Archiv.
-s	--slim	Verschlankt ein Archiv. Auch in Kombination mit -c anwendbar.
-r	--restore	Stellt ein Archiv wieder her.
-p	--partial	Stellt einen Unterordner eines Archivs oder eine einzelne Datei wieder her. Nur in Kombination mit -r anwendbar.
-d	--delete	Löscht ein Archiv, einen Unterordner eines Archivs oder eine einzelne Datei.
-l	--list	Listet den Inhalt eines Archivs zu einem gegebenen Zeitpunkt auf.
-v	--verbose	Aktiviert den Verbose-Mode.
-m	--mapping	Gibt alle Abbildungen von Hashes auf Verzeichnisse in STDOUT aus.

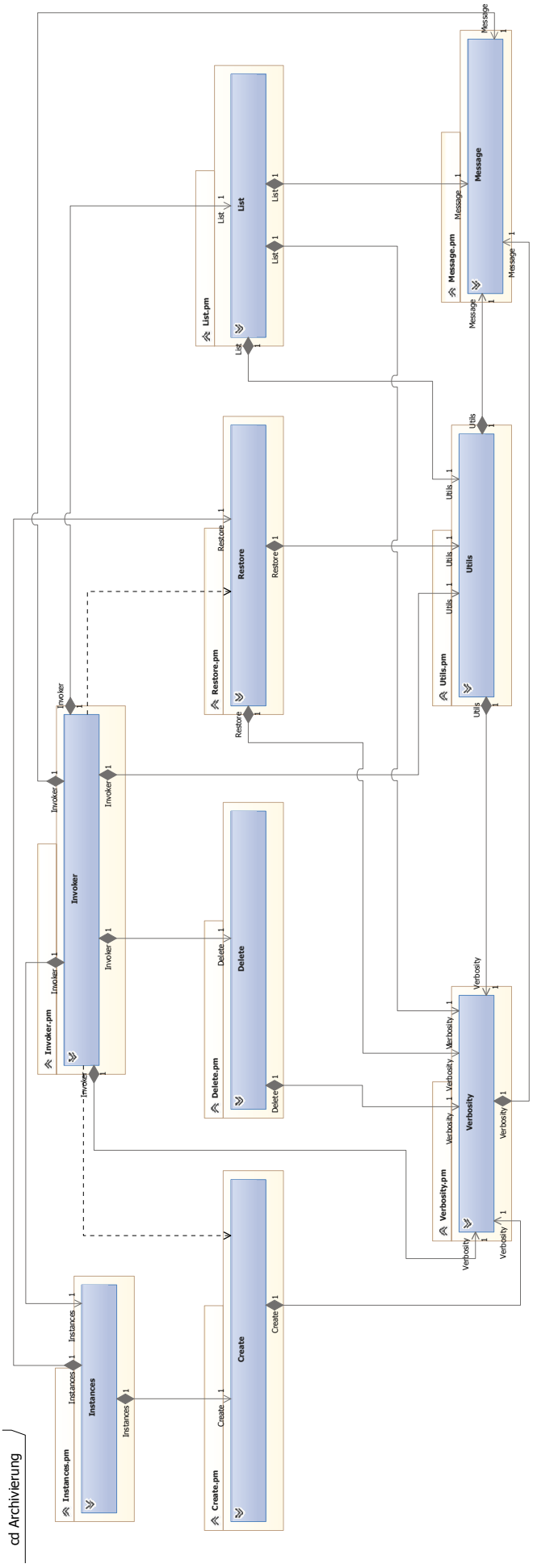
Folgende Kombinationen von Switches in Kurz- und Langform sind nicht erlaubt: cr, cd, cl, cp, cm, ch, rc, rd, rl, rs, rm, rh, dc, dr, dp, dl, ds, dm, dh, lc, lr, ld, lp, ls, lm, lh, mc, mr, md, ml, mh, hc, hr, hd, hl, hm, hv.

4. Software-Design

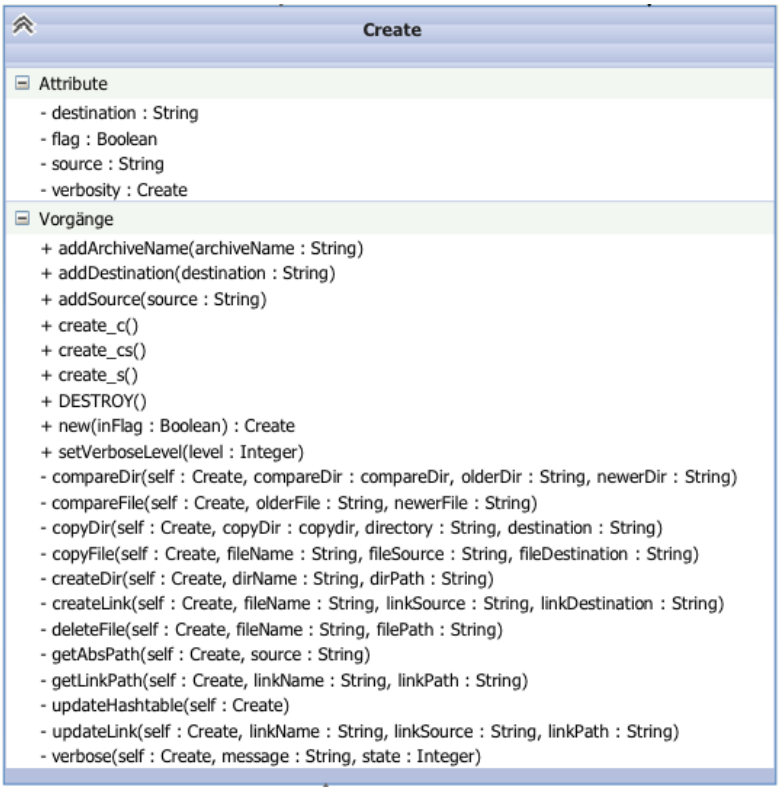
Die Anwendung wurde so konzipiert, dass nach der Analyse der Switches nicht entsprechend dem Switch die Methoden aus den entsprechenden Klassen Create, Restore, Delete oder List aufgerufen werden, sondern die Klasse Invoker die weitere Verarbeitung übernimmt.



Des Weiteren sollte erwähnt werden, dass bei der Entwicklung von diesem Archivierungs-Tool stets auf das Prinzip der Erweiterbarkeit geachtet wurde. Es ist somit denkbar, dass bei Bedarf von weiteren Funktionen einfach Switches ergänzt und Methoden dem Invoker hinzugefügt werden. Die Funktionalität kann dann in einer eigenen Klasse implementiert werden.


cd Archivierung
Message
Message.pm
Message
List
List.pm
List
Utils
Utils.pm
Utils
Restore
Restore.pm
Restore
Invoker
Invoker.pm
Invoker
Delete
Delete.pm
Delete
Verbosity
Verbosity.pm
Verbosity
Create
Create.pm
Create
Instances
Instances.pm
Instances

4.1. Create



Mit „*Create*“ kann ein neues Archiv angelegt werden. Dieses Archiv beinhaltet alle Unterverzeichnisse und Dateien eines Verzeichnisses in hierarchischer Struktur. Die Archivierung kann auf folgende Arten stattfinden:

- **Normale Archivierung:** Hierbei werden alle Unterverzeichnisse und Dateien in das Zielverzeichnis kopiert.
- **Verschlankte Archivierung:** Hierbei wird das neue Archiv wie bei der normalen Archivierung erzeugt. Zusätzlich werden vorhergehende Archive verschlankt.
- **Archiv-Verschlankung:** Bei der Archiv-Verschlankung werden alle Archive eines Verzeichnisses auf unveränderte Dateien untersucht. Hierbei wird immer ein Archiv mit dem vorhergehenden Archiv verglichen. Alle Dateien die sich nicht geändert haben werden im vorhergehenden Archiv durch einen Link zum aktuelleren Archiv ersetzt.

„*Create*" benötigt zur Erstellung eines Archivs ein Quellverzeichnis (das zu archivierende Verzeichnis) und ein Zielverzeichnis (Verzeichnis in dem alle Archive angelegt werden sollen).

Nachfolgend wird der Anwendungsfall „*Create*" beschrieben, der im Use Case Diagramm im Anhang betrachtet werden kann. Der Anwendungsfall veranschaulicht die strukturelle Vorgehensweise der Archiverstellung und Verschlankung.

Archiv erstellen mit Option -c:

Bei diesem Anwendungsfall handelt es sich um die normale Archivierung. Hierbei werden alle Dateien und Unterverzeichnisse des Quellverzeichnisses kopiert beziehungsweise erstellt.

Archiv erstellen mit Option -cs:

Bei diesem Anwendungsfall handelt es sich um die verschlankte Archivierung. Hierbei wird zunächst ein neues Archiv erstellt, wie in „Archiv erstellen mit Option -c“. Anschließend wird überprüft ob vorhergehende Archive vorhanden sind. Wenn dies der Fall ist, wird überprüft ob Dateien vorhanden sind, die zum Vorgängerarchiv keinerlei Änderungen haben. Diese Dateien werden im Vorgängerarchiv durch Links zum aktuelleren Archiv ersetzt.

Archiv erstellen mit Option -s:

Bei diesem Anwendungsfall handelt es sich um die Archiv-Verschlankung. Hierbei wird zunächst geprüft, ob vorhergehende Archive vorhanden sind. Ist ein vorhergehendes Archiv vorhanden, so wird dies auf unveränderte Dateien untersucht. Alle unveränderten Dateien werden wie in „Archiv erstellen mit Option -cs“ beschrieben durch Links ersetzt. Im Folgenden wird die Verwendung von Create näher betrachtet. Bei allen drei Archivierungsarten muss eine Instanz von `Create` mittels der Methode `new` erzeugt werden. Die Methode `new` bekommt optional einen Übergabeparameter, der die Ausgabe steuert. Bei Aufruf von `new` mit 1 als Parameter findet eine Ausgabe auf der Konsole statt. Alternativ kann die Ausgabe aber auch über die Methode `setVerboseLevel` aktiviert werden.

Um ein Archiv ohne Verschlankung zu erstellen muss folgendermaßen vorgegangenn werden:

1. Erzeugen einer neuen Instanz von `Create` mittels der Methode `new`
2. Hinzufügen des Quellverzeichnisses mittels der Methode `addSource`
3. Hinzufügen des Zielverzeichnisses mittels der Methode `addDestination`
4. Erzeugen des Archivs mittels der Methode `create_c`

Die Vorgehensweise bei der Archiverstellung mit Verschlankung ist:

1. Erzeugen einer neuen Instanz von `Create` mittels der Methode `new`
2. Hinzufügen des Quellverzeichnisses mittels der Methode `addSource`
3. Hinzufügen des Zielverzeichnisses mittels der Methode `addDestination`
4. Erzeugen des Archivs mittels der Methode `create_cs`

Bestehende Archive können wie folgt verschlankt werden:

1. Erzeugen einer neuen Instanz von `Create` mittels der Methode `new`
2. Hinzufügen des Zielverzeichnisses mittels der Methode `addDestination`
3. Hinzufügen des Archivnamens mittels der Methode `addArchiveName`
4. Verschlanken des Archivs mittels der Methode `create_s`

4.1. Klassenbeschreibung Create

Attribute der Klasse Create

source	In diesem Attribut befindet sich das Quellverzeichnis
destination	In diesem Attribut befindet sich das Zielverzeichnis
flag	Mit diesem Attribut wird die Option für die detaillierte Ausgabe gesteuert

verbosity	In diesem Attribut befindet sich eine Instanz der Klasse `Verbosity`, welche für die Ausgabe von Meldungen zuständig ist

Public Methoden der Klasse Create

new

Beschreibung:	Erzeugt ein neues Objekt der Klasse `Create`
Parameter:	$flag = Ansteuerung der Ausgabe von Programminformationen (optional)
Rückgabe:	Keine

addSource

Beschreibung:	Fügt das Quellverzeichnis hinzu
Parameter:	$source = Pfad zum Quellverzeichnis
Rückgabe:	Keine

addDestination

Beschreibung:	Fügt das Zielverzeichnis hinzu
Parameter:	$destination = Pfad zum Zielverzeichnis
Rückgabe:	Keine

addArchiveName

Beschreibung:	Fügt den Archivnamen hinzu, der für eine Verschlankung benötigt wird
Parameter:	$archiveName = Name des Archivs ohne date_time
Rückgabe:	Keine

setVerboseLevel

Beschreibung:	Setzt das Level der Verböse-Ausgabe
Parameter:	$level 0 = Keine Ausgabe 1 = Normale Ausgabe 2 .. 8 = reserviert 9 = Debug Ausgabe
Rückgabe:	Keine

create_cs

Beschreibung:	Erzeugt ein neues Archiv und verschlankt alle vorhergehenden Archive
Parameter:	Keine
Rückgabe:	Keine

create_c

Beschreibung:	Erzeugt ein neues Archiv
Parameter:	Keine
Rückgabe:	Keine

create_s

Beschreibung:	Verschlankt alle Archive im Archivverzeichnis, die den gleichen Namen haben.
Parameter:	Keine
Rückgabe:	Keine

DESTROY

Beschreibung:	Zerstört das Objekt
Parameter:	Keine
Rückgabe:	Keine

Private Methoden der Klasse Create

verbose

Beschreibung:	Erzeugt eine Ausgabe auf STDOUT, wenn das Flag dafür gesetzt wurde
Parameter:	$message = Ist die auszugebende Nachricht $state = Gibt den Status der Nachricht an {OK, WARNING, ERROR}
Rückgabe:	Keine

updateHashtable

Beschreibung:	Überprüft, ob es bereits einen Eintrag in der Hashtable für dieses Archiv gibt, falls nicht wird ein neuer Eintrag erstellt
Parameter:	$self = Instanz von Create
Rückgabe:	Keine

copyFile

Beschreibung:	Kopiert eine Datei vom Quellverzeichnis ins Zielverzeichnis
Parameter:	$self = Instanz von Create $fileName = Name der Datei, die gelöscht werden soll $fileSource = Verzeichnis, in dem sich die Datei befindet $fileDestination = Verzeichnis, in das die Datei kopiert werden soll
Rückgabe:	Keine

createDir

Beschreibung:	Erzeugt ein neues Verzeichnis
Parameter:	$self = Instanz von Create $dirName = Name des Verzeichnisses $dirPath = Pfad des Verzeichnisses
Rückgabe:	Keine

copyDir

Beschreibung:	Kopiert alle Dateien im aktuellen Verzeichnis und allen Unterverzeichnissen in das Zielverzeichnis
Parameter:	$directory = Aktuelles Unterverzeichnis $destination = Ziel-Unterverzeichnis
Rückgabe:	Keine

deleteFile

Beschreibung:	Löscht eine Datei im angegebenen Verzeichnis
Parameter:	$self = Instanz von Create $fileName = Name der Datei, die gelöscht werden soll $filePath = Verzeichnis, in dem sich die Datei befindet
Rückgabe:	Keine

createLink

Beschreibung:	Erzeugt einen neuen symbolischen Link auf eine Datei im angegebenen Verzeichnis
Parameter:	$self = Instanz von Create $fileName = Name der Datei, die verlinkt werden soll $linkSource = Verzeichnis, in dem sich die originale Datei befindet $linkDestination = Verzeichnis, in dem der Link erstellt werden soll
Rückgabe:	Keine

updateLink

Beschreibung:	Aktualisiert den symbolischen Link auf die Originaldatei
Parameter:	$self = Instanz von Create

	$linkName = Name der Link-Datei $linkSource = Verzeichnis, in dem sich die Originaldatei befindet $linkPath = Verzeichnis, in dem sich die Link-Datei befindet
Rückgabe:	Keine

getLinkPath

Beschreibung:	Liefert den Pfad der Originaldatei zurück
Parameter:	$self = Instanz von Create $linkName = Name der Link-Datei $linkPath = Verzeichnis, in dem sich die Link-Datei befindet
Rückgabe:	Pfad zur originalen Datei

getAbsPath

Beschreibung:	Liefert den absoluten Pfad des angegebenen Verzeichnisses zurück
Parameter:	$source = Pfad absolut oder relativ
Rückgabe:	Absoluter Pfad des Verzeichnisses

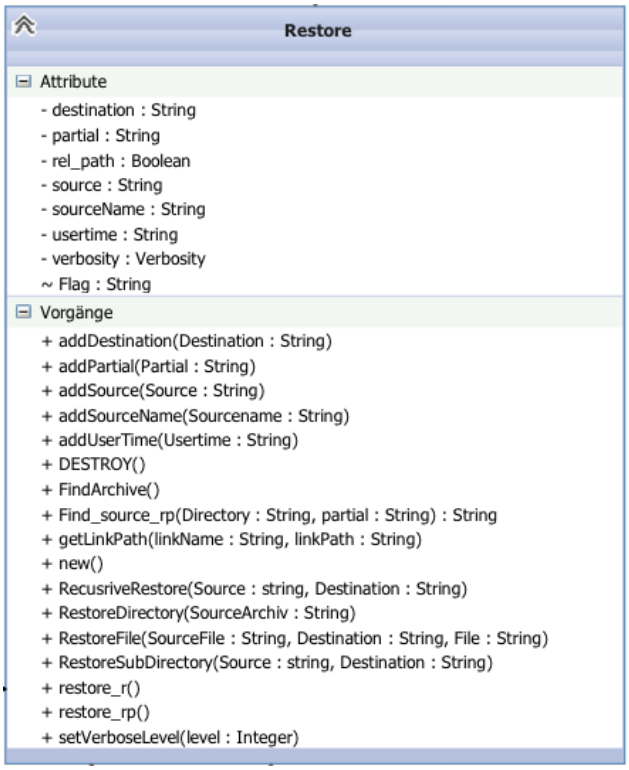
compareFile

Beschreibung:	Vergleicht zwei Dateien auf Gleichheit
Parameter:	$olderFile = ältere Datei $newerFile = neuere Datei
Rückgabe:	true = Dateien sind gleich false = Dateien sind unterschiedlich

compareDir

Beschreibung:	Vergleicht die Archive zur Verschlankung. Ist eine Datei unverändert wird diese durch einen Link auf das neue Verzeichnis ersetzt
Parameter:	$olderDir = älteres Archivverzeichnis $newerDir = neueres Archivverzeichnis
Rückgabe:	Keine

4.2. Restore



Um ein ganzes Archiv wieder herzustellen muss die Methode `restore_r()` aufgerufen werden. Diese Methode ruft als erstes die Methode `findLastValidArchiv()`, aus der Klasse `Utils`, auf. Diese Funktion gibt den Pfad zum letzten gültigen Archiv zurück, übergeben wird der Pfad zum Archiv, die Zeitangabe und der Name des Archivs. Nachdem die genau Adresse zum Archiv fest steht, wird der Wiederherstellungsprozess begonnen. Die Methode `RestoreDirectory()` rekonstruiert das Archiv in das angegebene Zielverzeichnis. Falls ein Verzeichnis mit dem selben Namen im Zielverzeichnis existiert wird dieser gelöscht und ein neues Verzeichnis mit dem selben Namen erstellt. Die Methode ruft die Funktion `RecursiveRestore()` auf. `RecursiveRestore()` wird bei jedem Unterverzeichnis rekursiv aufgerufen. Im Quellarchiv wird der Inhalt durchlaufen und geprüft ob es sich um eine Datei, Link oder Verzeichnis handelt.Falls es eine Datei ist wird sie in den neu erstellten Ordner rein kopiert. Bei einem Link wird mithilfe der `getLinkPath()`-Methode der absolute Pfad zur Originaldatei gefunden und dieser wird kopiert. Wenn ein Verzeichnis gefunden wird, ruft sich die Methode `RecursiveRestore()` rekursiv auf. Da dieses Verzeichnis selbst Unterverzeichnisse, Dateien oder Links beinhalten kann wird ein rekursiver Aufruf benötigt. Das Ende der Methode ist erreicht wenn die letzte Datei kopiert wurde.

Falls nur ein Unterverzeichnis oder eine einzelne Datei wieder hergestellt werden soll, muss die Funktion `restore_rp()` aufgerufen werden. Wie in `restore_r()` verwendet auch diese Methode die Funktion `findLastValidArchiv()` von `Utils` um den absoluten Pfad zum gewählten Archiv zu finden.

Die Methode nutzt den zusätzlichen Parameter partial. Dieses Attribut kann sowohl als relativer Pfad angegeben werden (z.B. "\Unterverzeichnis1\text23.txt" oder "Unterverzeichnis1\text23.txt") als auch der Name des Unterverzeichnisses oder der Datei (z.b. "text23.txt").

Im ersten Fall wird dem Zielverzeichnis und dem Quellverzeichnis der partial-Teil angehängt. Die neuen Pfade werden auf Existenz geprüft. Falls der Pfad zu einem Link führt wird die Originaldatei mit `getLinkPath()` gefunden. Falls in partial nur der Name angegeben ist, wird mithilfe der Methode `Find_source_rp()` das Quellarchiv rekursiv durchgegangen bis eine Datei oder ein Unterverzeichnis mit dem selben Namen gefunden wird. Hierbei ist es wichtig, dass die Endung der Datei (z.B. ".txt") angegeben wird. Nachdem die genauen Pfade gefunden sind, wird unterschieden ob der Quellpfad zu einer Datei, einem Link oder einem Unterverzeichnis führt.

Bei einem Link wird zuerst die Methode `getLinkPath()` aufgerufen um die Originaldatei zu finden. Der Pfad zur Originaldatei und zum Zielverzeichnis wird der Methode `RestoreFile()` übergeben. Falls eine Datei mit dem selben Namen im Zielverzeichnis existiert, wird diese gelöscht und die neu Datei wird rein kopiert. Ansonsten wird die neue Datei, ohne löschen, in das angegeben Verzeichnis kopiert. Bei einer Datei wird auch die `RestoreFile()` Methode aufgerufen.

Wenn der Pfad zu einem Verzeichnis führt, wird die Methode `RestoreSubDirectory()` aufgerufen. Dieser Methode wird der absolute Pfad zum Unterverzeichnis im Quellarchiv und im Zielarchiv mitgegeben. Diese beiden Parameter werden der Methode `RecursiveRestore()` übergeben welche das

Quellverzeichnis rekursiv durchläuft und wie in `restore_r()` beschrieben die Dateien und Ordner wiederherstellt. Der Programmablauf kann in den Aktivitätsdiagrammen im Anhang genauer betrachtet werden.

Die Vorgehensweise beim Wiederherstellen eines Archivs ist:

5. Erstellen eines RestoreWin-Objekts mit `RestoreWin->new()`
6. Hinzufügen des Pfads zu den Archiven mit `addSource()`
7. Hinzufügen des Zielverzeichnisses mit `addDestination()`
8. Hinzufügen eines Zeitstempels mit `addUserTime()`
9. Setzen des Verbose-Levels mit `setVerboseLevel()` [Optional]
10. Starten des Wiederherstellungsvorgangs mit `restore_r()`

Um ein Unterverzeichnis oder eine einzelne Datei wieder herzustellen muss folgendermaßen vorgegangen werden:

1. Schritte 1 bis 4 wie oben beschrieben
2. Hinzufügen des relativen Pfads zum wiederherzustellenden Objekts mit `addPartial()`
3. Setzen des Verbose-Levels mit `setVerboseLevel()` [Optional]
4. Starten des Wiederherstellungsvorgangs mit `restore_r()`

4.2.1.Klassenbeschreibung Restore

Attribute der Klasse Restore

source	Gibt den Pfad zu den Archiven an.
sourcename	Gibt den Namen des Archives an.
destination	Gibt den Pfad zum Zielverzeichnis an.
usertime	Gibt die Zeitangabe in dieser Format an yyyy_mm_dd_hh_ii_ss.
partial	Gibt den relativen Pfad zum Unterverzeichnis oder der Datei an.
rel_path	Gibt an ob es sich bei partial um einen relativen Pfad handelt.
verbosity	Ist ein Verbosity-Objekt, welches die Ausgabe unterstützt.
Flag	Gibt an ob eine Datei kopiert wurde oder nicht.

Methoden der Klasse Restore

new()
Beschreibung: Erzeugt ein neues Objekt der Klasse RestoreWin
Parameter: Keine
Rückgabe: Keine

addSource()
Beschreibung: Fügt das Quellverzeichnis hinzu.
Parameter: $Source = Das Quellverzeichnis
Rückgabe: Keine

addDestination()
Beschreibung: Fügt das Zielverzeichnis hinzu.
Parameter: $Destination = Das Zielverzeichnis

Rückgabe: Keine

addSourceName()
Beschreibung: Fügt den Archivnamen hinzu.
Parameter: $Sourcename = Der Name des Archives
Rückgabe: Keine

addUserTime()
Beschreibung: Fügt die vom Benutzer eingegeben Zeit hinzu
Parameter: $Usertime = Zeitangabe (Format yyyy_mm_dd_hh_ii_ss)
Rückgabe: Keine

addPartial()
Beschreibung: Fügt den relativen Pfad hinzu und bearbeitet ihn, falls nötig.
Parameter: $Partial = Unterverzeichnis oder Datei (siehe Beschreibung restore_rp)
Rückgabe: Keine

setVerboseLevel()
Beschreibung: Setzt den Verbose-Level.
Parameter: $level = Ausgabe (1 normale Ausgabe, 2..8 reserviert, 9 Debug-Ausgabe)
Rückgabe: Keine

restore_r()
Beschreibung: Hauptfunktion um ein ganzes Archiv wieder herzustellen.
Parameter: Keine
Rückgabe: Keine

restore_rp()
Beschreibung: Hauptfunktion um ein Unterverzeichnis oder eine Datei wieder herzustellen.
Parameter: Keine
Rückgabe: Keine

Find_source_rp()
Beschreibung: Hilfsfunktion um eine bestimmte Datei oder Verzeichnis zu finden.
Parameter: $Directory = Verzeichnis in dem gesucht werden soll
$partial = Name der Datei oder Unterverzeichnisses
Rückgabe: Absoluten Pfad zum Unterverzeichnis oder Datei.

RestoreDirectory()
Beschreibung: Hilfsfunktion um ein ganzes Verzeichnis wieder herzustellen
Parameter: $SourceArchiv = Genauer absoluter Pfad des Archives
Rückgabe: Keine

RecursivRestore()
Beschreibung: Hilfsfunktion um rekursiv ein Verzeichnis wieder herzustellen.
Parameter: $Source = Absoluter Pfad zur Quelleverzeichnis
$Destination = Absoluter Pfad zum Zielverzeichnis
Rückgabe: Keine

RestoreSubDirectory()
Beschreibung: Hilfsfunktion um ein Unterverzeichnis wieder herzustellen.
Parameter: $Source = Genauer absoluter Pfad zum Quellunterverzeichnis

	$Destination = Genauer absoluter Pfad zum Zielunterverzeichnis
Rückgabe:	Keine

RestoreFile()

Beschreibung:	Hilfsfunktion um eine Datei wieder herzustellen.
Parameter:	$SourceFile = Pfad zur Quelldatei $Destination = Pfad zum Zielverzeichnis $File = Dateiname
Rückgabe:	Keine

FindArchive()

Beschreibung:	Hilfsfunktion um das zu wiederherstellende Archiv zu finden.
Parameter:	Keine
Rückgabe:	Absoluten Pfad zum zu wiederherstellenden Archiv

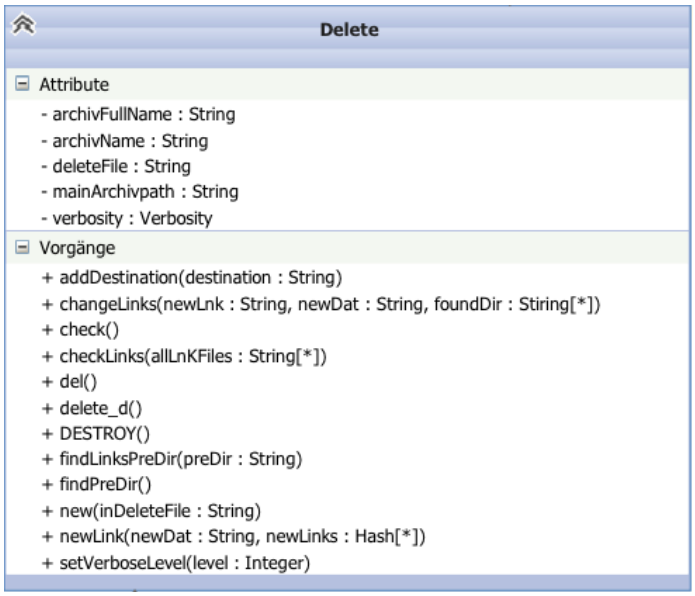
getLinkPath()

Beschreibung:	Liefert den Pfad der original Datei zurück.
Parameter:	$linkName = Name der Link-Datei $linkPath = absoluter Pfad zum Verzeichnis in dem sich der Link befindet
Rückgabe:	Absoluten Pfad zur Originaldatei

DESTROY()

Beschreibung:	Freigeben der Ressourcen
Parameter:	Keine
Rückgabe:	Keine

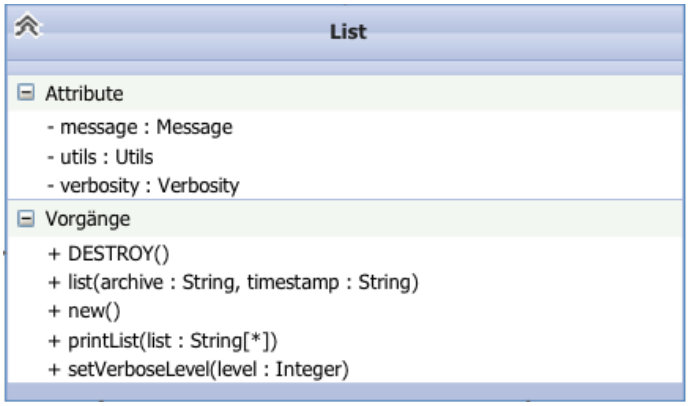
4.3. Delete



4.3.1. Klassenbeschreibung Delete

[Beschreibung von Michaela]

4.4. List



Die Methode `list(...)` erwartet als Parameter den Pfad zu einem Archiv und einen Timestamp (Zeitstempel) der From yyyy_mm_dd_hh_ii_ss. Aus dem Archivpfad wird der Archivname extrahiert, welcher zusammen mit dem Zeitstempel für die Ermittlung des zuletzt gültigen Archivs notwendig ist. Wenn das zuletzt gültige Archiv im Archivverzeichnis (eine Verzeichnishierarchie höher als das angegebene Archiv) gefunden wurde, werden die Inhalte des Archivs rekursiv in einem Array gespeichert. Dabei werden jedoch die Verzeichnisse . und .. ignoriert. Das Array wird dann der Methode `printList(...)` übergeben und Element für Element iteriert. Dabei wird geprüft, ob das Element ein Verzeichnis, ein Link (Verweis) oder eine Datei ist und, entsprechend um Informationen erweitert, ausgegeben.

4.4.1. Klassenbeschreibung List

Attribute der Klasse List

verbosity	Instanz der Klasse `Verbosity`
message	Instanz der Klasse `Message`
utils	Instanz der Klasse `Utils`

Methoden der Klasse List

new

Beschreibung:	Erzeugt ein neues Objekt der Klasse List
Parameter:	Keine
Rückgabe:	Keine

setVerboseLevel

Beschreibung:	Setzt das Verbose-Level
Paramter:	$level = Verbose-Level zwischen 0 und 9
Rückgabe:	Keine

list

Beschreibung:	Durchläuft rekursiv alle Verzeichnisse eines zuletzt gültigen Archivs und speichert die Inhalte der Unterverzeichnisse in einem Array
Parameter:	$archive = aufzulistendes Archiv $timestamp = Zeitstempel der Form yyyy_mm_dd_hh_ii_ss

Rückgabe: Keine

print_list
Beschreibung: Iteriert über die Inhalte einer „Verzeichnisliste“ und gibt diese mit entsprechenden Informationen aus
Parameter: $list = Verzeichnisliste
Rückgabe: Keine

DESTROY
Beschreibung: Gibt Ressourcen frei und zerstört das Objekt
Parameter: Keine
Rückgabe: Keine

5. Ergebnisse des Profiling

Nachfolgend werden Testfälle vom 18. Januar 2015 dargestellt. Es waren 3000 kleinere Text-Dateien (6 - 20 KB) auf fünf Ordner verteilt. Die Testfälle wurden auf einem Computer mit den folgenden Eigenschaften durchgeführt:

- Windows 7 Home Premium
- Pentium Dual-Core E5400; 2,70 GHz; 2 Kerne
- 4 GB RAM
- HDD Festplatte

In den Testfällen wurden unterschiedliche Szenerien berücksichtigt. Es folgt eine kurze Erklärung der verwendeten Abkürzungen:

- max. Shortcuts: Alle 3000 Dateien waren Links
- max. Files: 3000 Text-Dateien
- average: 1500 Text-Dateien und 1500 Links

Die Profiling-Ergebnisse der jeweiligen Klassen können durch die CPU-Auslastung und sonstige laufende Hintergrundprogramme, zum Zeitpunkt des Tests, beeinflusst worden sein. Hinzu kommt außerdem die zusätzliche Zeit, die der Profiler während der Tests benötigt.

5.1. Create

Methode	Zeit insgesamt	Zeit Create.pm	Zeit max. Modul
Create_c	6.67s	617ms	File/Copy -> 5.94s
Create_cs create one archive slim one archive (max. Shortcuts)	25.6s	8.12s	Win32/Shortcut.pm -> 10.9s
Create_s of two archives (max. Shortcuts)	56.0s	15.0s	Win32/Shortcut.pm -> 29.1s

Methode	Aufrufe insgesamt	Aufrufe Create.pm	Aufrufe max. Modul
Create_c	123592	27241	File/Copy -> 81027
Create_cs (max. Shortcuts)	408738	183343	Create.pm -> 183343
Create_s (max. Shortcuts)	1113774	582333	Create.pm -> 582333

Man kann erkennen, dass das Erstellen eines Archivs weitaus weniger Zeit in Anspruch nimmt, als ein Archiv zu verschlanken.

5.2. Restore

Methoden	Zeit insgesamt	Zeit RestoreWin.pm	Zeit max. Modul
Restore_R (max. Files)	8.69s	1.31s	File/Copy.pm -> 6.75s
Restore_R (max. Shortcuts)	15.8s	1.06s	Win32/Shortcut.pm -> 7.69s
Restore_R (average)	12.1s	1.18s	File/Copy.pm -> 7.09s
Restore_P Subdirectory (2500 Files)	6.58s	844ms	File/Copy.pm -> 5.26s
Restore_P Subdirectory (2500 Shortcuts)	12.9s	834ms	File/Copy.pm -> 6.25s
Restore_P Link	181ms	30.0ms	RestoreWin.pm -> 30ms
Restore_P File	186ms	32.0ms	RestoreWin.pm -> 32ms

Methoden	Aufrufe insgesamt	Aufrufe RestoreWin.pm	Aufrufe max. Modul
Restore_R (max. Files)	255929	21167	File/Copy.pm -> 90032
Restore_R (max. Shortcuts)	387960	60167	File/Copy.pm -> 90032
Restore_R (average)	321960	40667	File/Copy.pm -> 90032
Restore_P Subdirectory (2500 Files)	214478	18180	File/Copy.pm -> 75029
Restore_P Subdirectory (2500 Shortcuts)	324509	50680	File/Copy.pm -> 6.25s
Restore_P Link	4352	600	Term/ANSIColor.pm -> 1426
Restore_P File	4426	612	Term/ANSIColor.pm -> 1426

An den Zeilen Restore_P Link und Restore_P File kann man erkennen, dass der Unterschied zwischen dem Wiederherstellen einer Datei und dem Auffinden einer Originaldatei und anschließendem Wiederherstellen sehr gering ist. Die meiste Zeit wird außerhalb der Klasse RestoreWin.pm verbracht.

5.3. Delete

Methoden	Zeit insgesamt	Zeit del.pm	Zeit max. Modul
Delete (max. Shortcuts)	22.0s	7.07s	del.pm -> 7.07s
Delete (average)	5.97s	952ms	File/Path.pm -> 4.54s
Delete (max. Files)	7.33s	1.84s	File/Path.pm -> 5.01s

Methoden	Aufrufe insgesamt	Aufrufe del.pm	Aufrufe max. Modul
Delete (max. Shortcuts)	388443	81131	del.pm -> 81131
Delete (average)	91054	87	File/Spec/Win32.pm -> 51134
Delete (max. Files)	91065	94	File/Spec/Win32.pm -> 51134

Delete, wie auch Restore und Create, benötigen insgesamt wenig Zeit. Die hauptsächliche Arbeit (Kopieren, Verlinken, …) wird von anderen Klassen erledigt. An den Ergebnissen von Delete ist auffällig, dass sehr wenig Aufrufe innerhalb der Klasse del.pm benötigt werden.

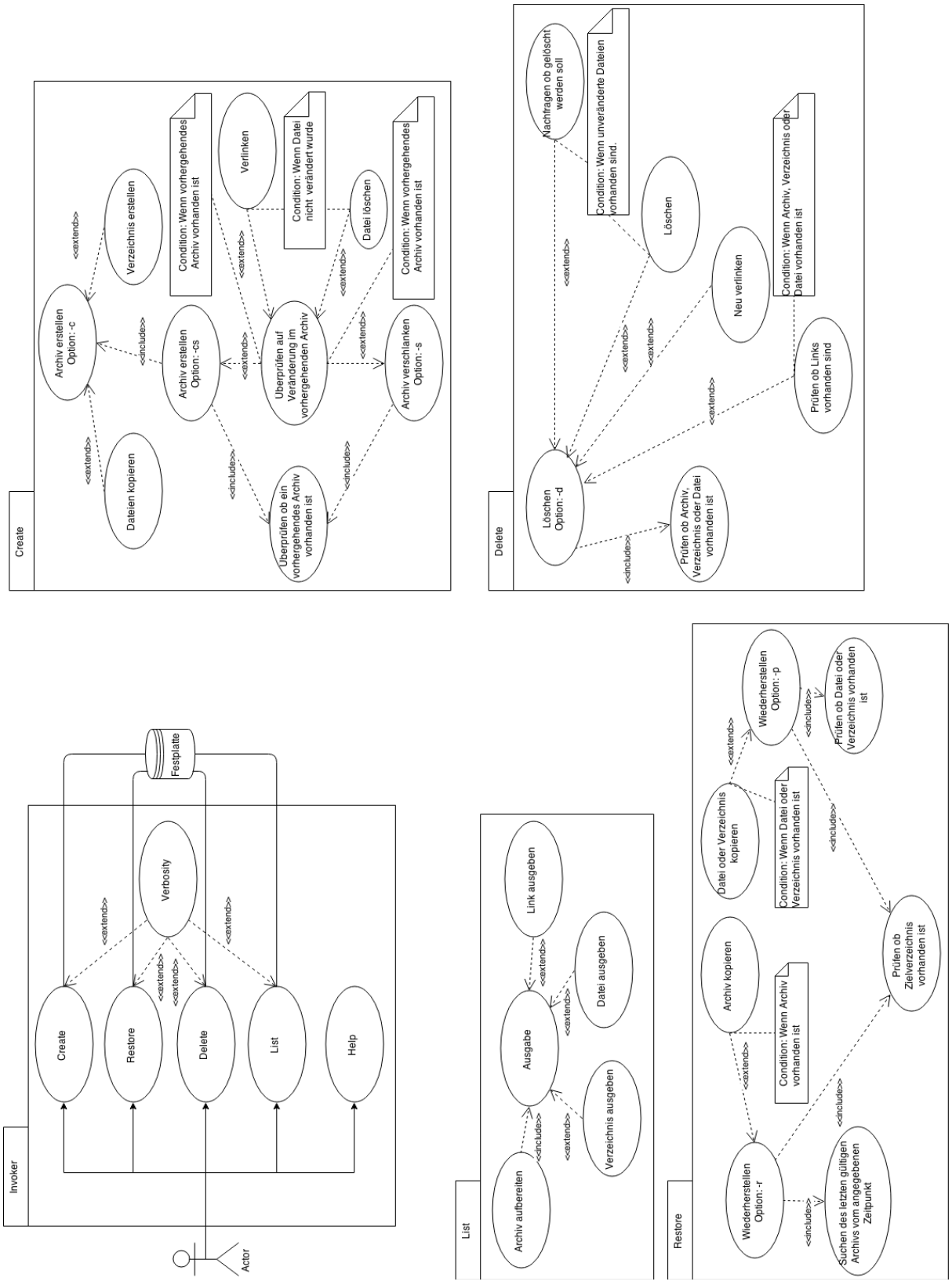
5.4. List

Methoden	Zeit insgesamt	Zeit List.pm	Zeit max. Modul
List	280ms	83.1ms	List.pm -> 83.1 ms

Methoden	Aufrufe insgesamt	Aufrufe List.pm	Aufrufe max. Modul
List	45171	15093	File/Find.pm -> 24432

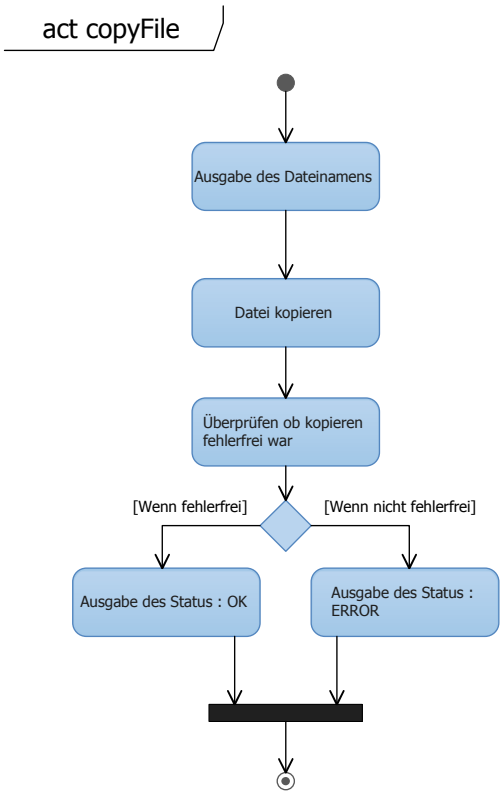
6. Anhang

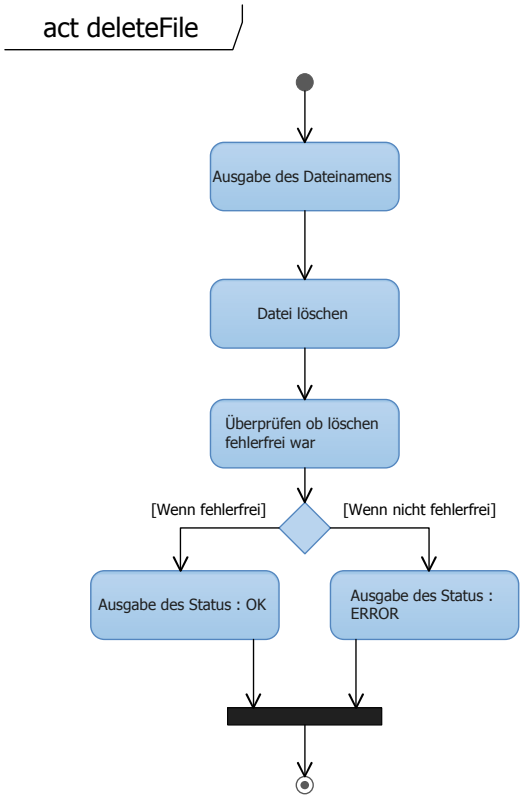
6.1. Use-Case-Diagramm

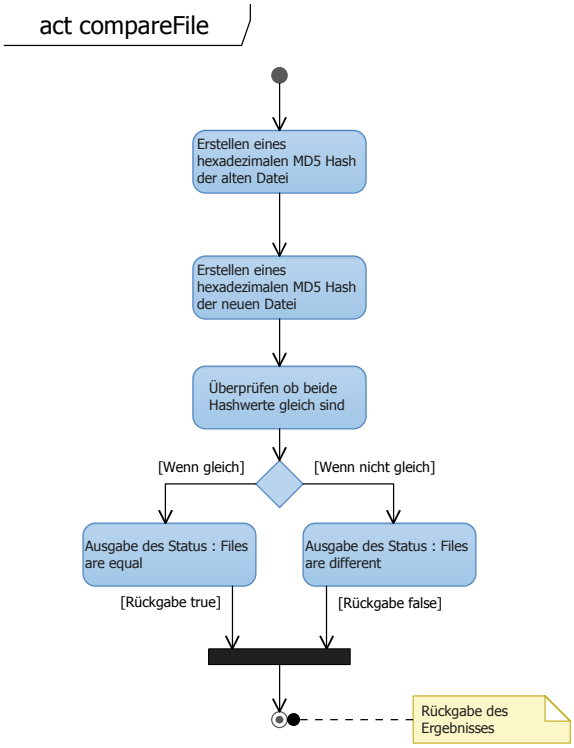


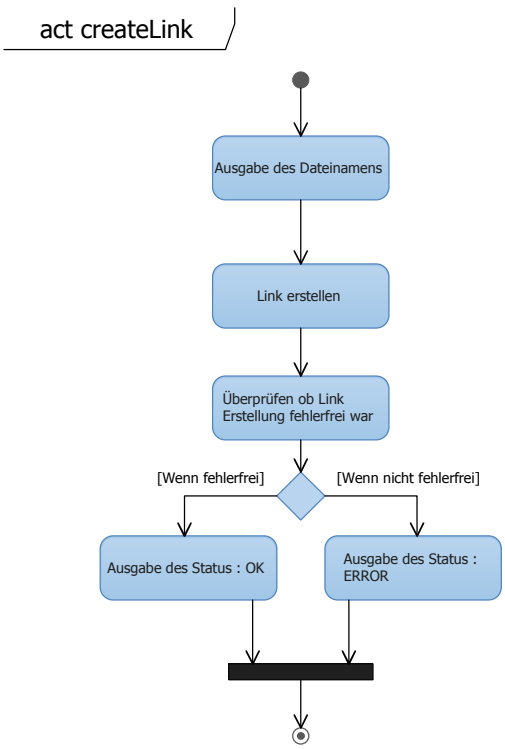
6.2. Klassendiagramm

6.3. Aktivitätsdiagramme zu Create

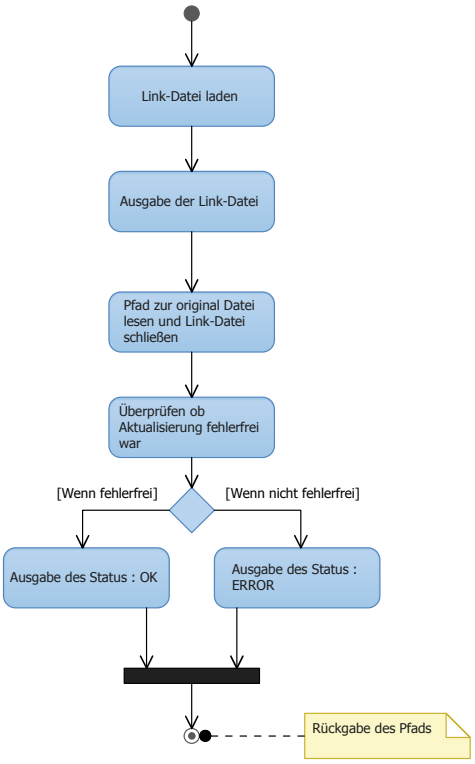




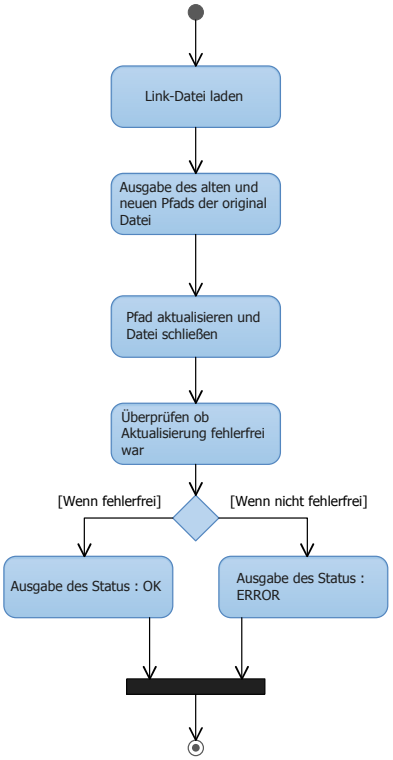




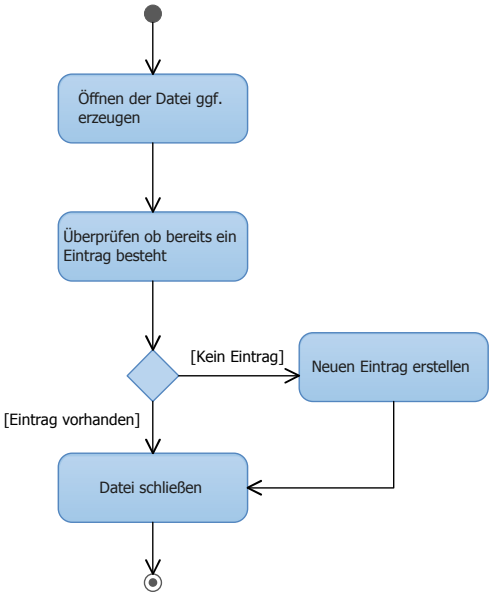
act getLinkPath



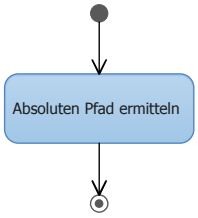
act updateLink

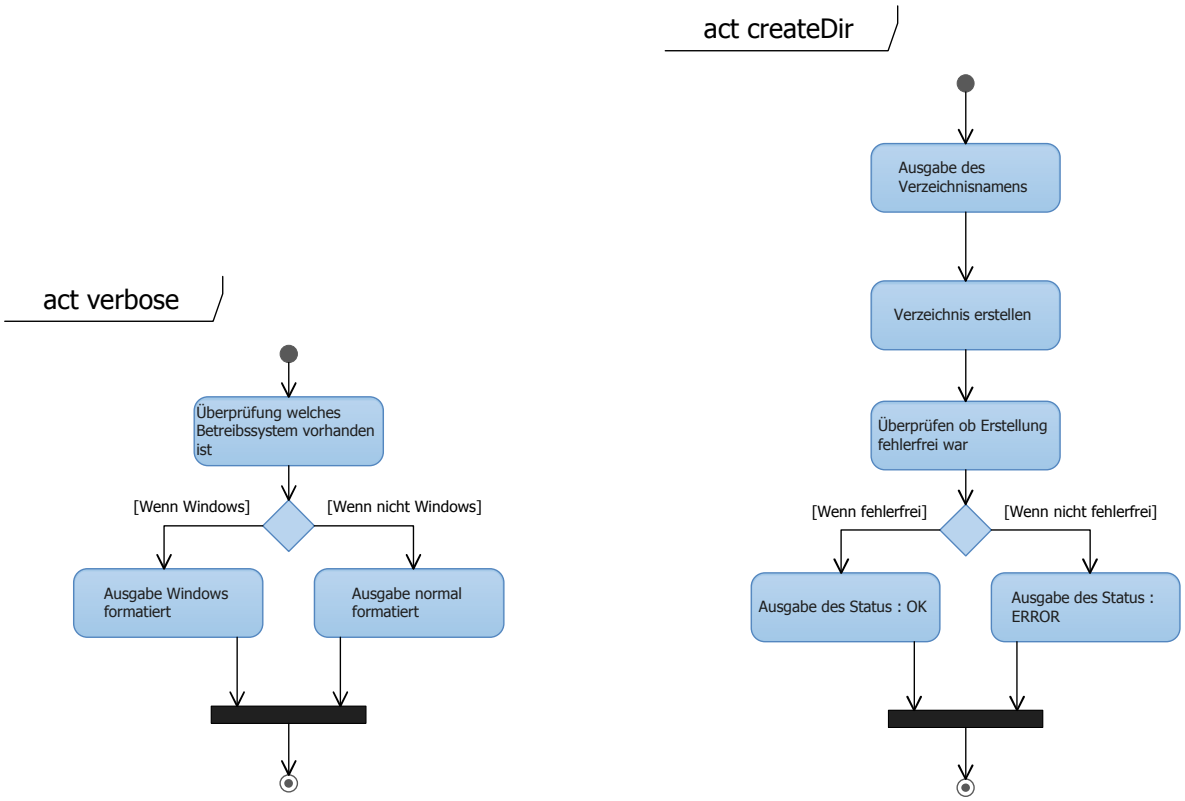


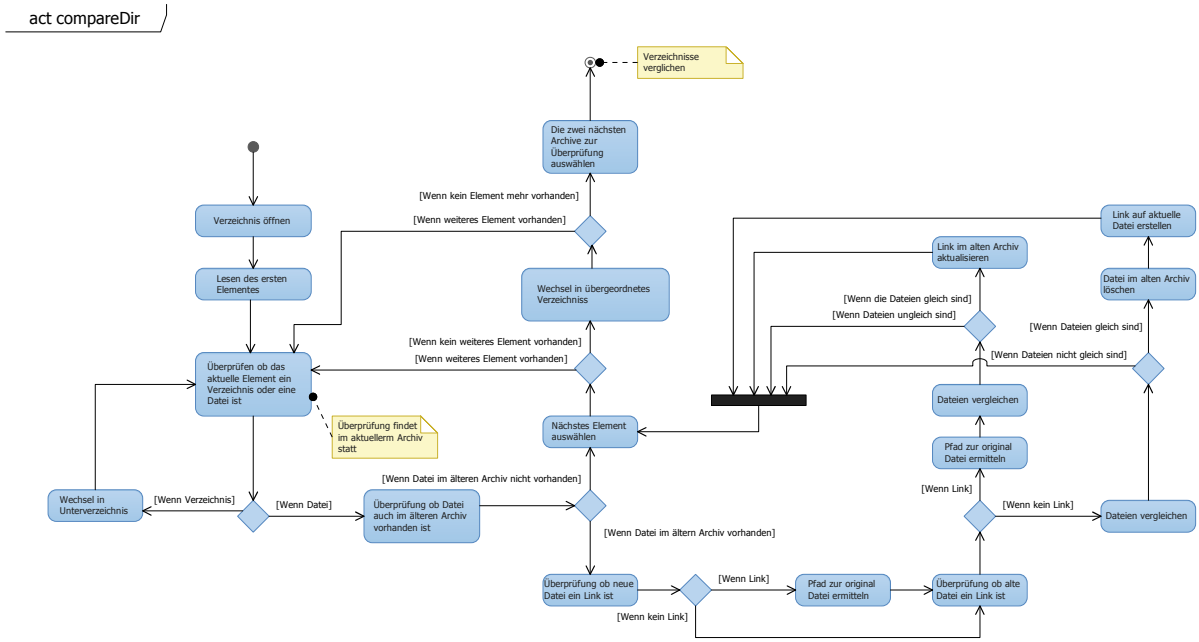
act updateHashtable



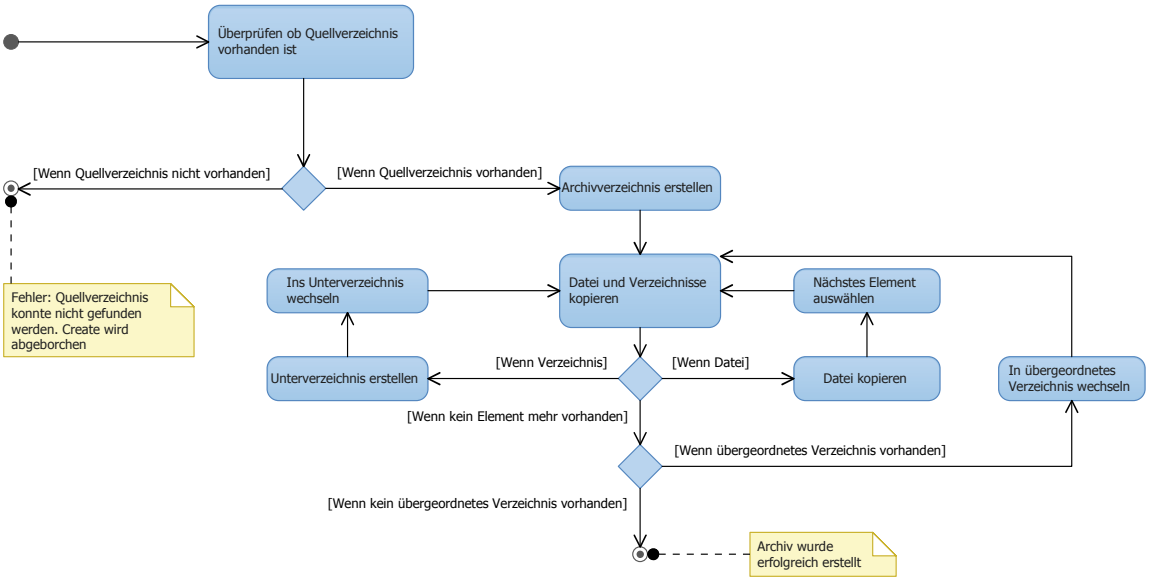
act getAbsPath



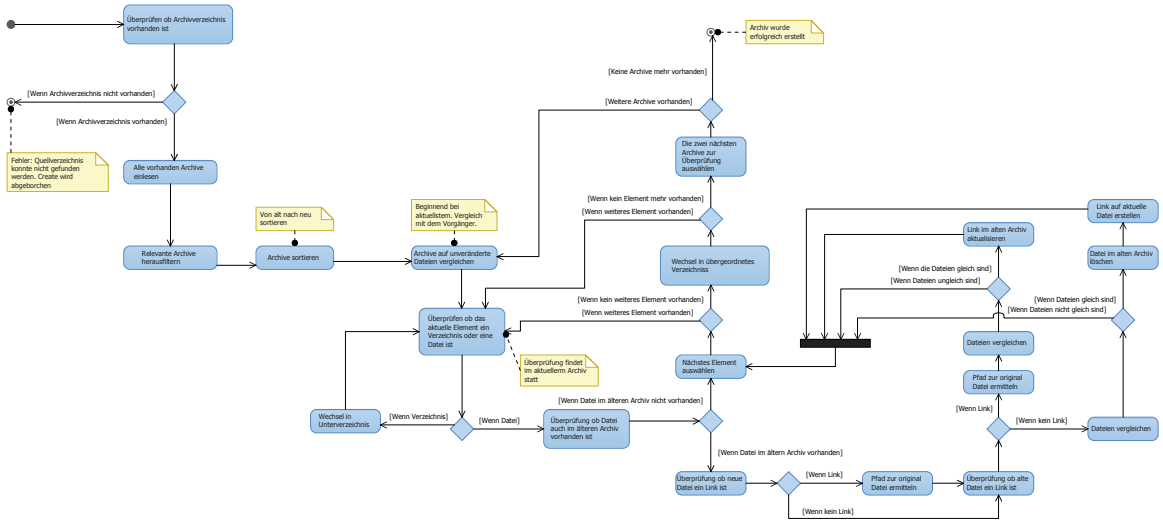

act createDir
Ausgabe des Verzeichnisnamens
Verzeichnis erstellen
Überprüfen ob Erstellung fehlerfrei war
[Wenn fehlerfrei]
[Wenn nicht fehlerfrei]
Ausgabe des Status : OK
Ausgabe des Status : ERROR
act verbose
Überprüfung welches Betreibssystem vorhanden ist
[Wenn Windows]
[Wenn nicht Windows]
Ausgabe Windows formatiert
Ausgabe normal formatiert


act compareDir
Verzeichnisse verglichen
Die zwei nächsten Archive zur Überprüfung auswählen
[Wenn kein Element mehr vorhanden]
[Wenn weiteres Element vorhanden]
Verzeichnis öffnen
Lesen des ersten Elementes
Wechsel in übergeordnetes Verzeichniss
Link auf aktuelle Datei erstellen
Link im alten Archiv aktualisieren
Datei im alten Archiv löschen
[Wenn die Dateien gleich sind]
[Wenn Dateien ungleich sind]
[Wenn Dateien gleich sind]
[Wenn Dateien nicht gleich sind]
[Wenn kein weiteres Element vorhanden]
[Wenn weiteres Element vorhanden]
Überprüfen ob das aktuelle Element ein Verzeichnis oder eine Datei ist
Überprüfung findet im aktuellerm Archiv statt
Nächstes Element auswählen
Dateien vergleichen
Pfad zur original Datei ermitteln
[Wenn Link]
[Wenn kein Link]
Dateien vergleichen
Wechsel in Unterverzeichnis
[Wenn Verzeichnis]
[Wenn Datei]
Überprüfung ob Datei auch im älteren Archiv vorhanden ist
[Wenn Datei im älteren Archiv nicht vorhanden]
[Wenn Datei im ältern Archiv vorhanden]
Überprüfung ob neue Datei ein Link ist
[Wenn Link]
Pfad zur original Datei ermitteln
Überprüfung ob alte Datei ein Link ist
[Wenn kein Link]

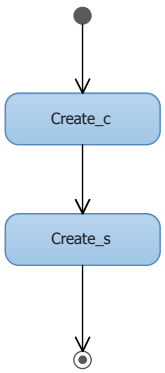
act Create_c



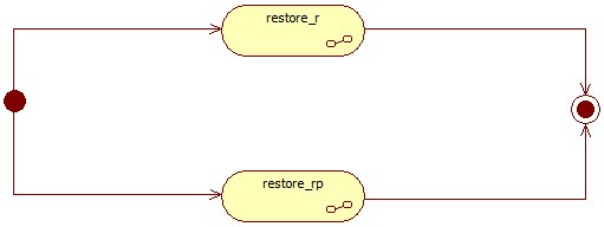
act Create_s

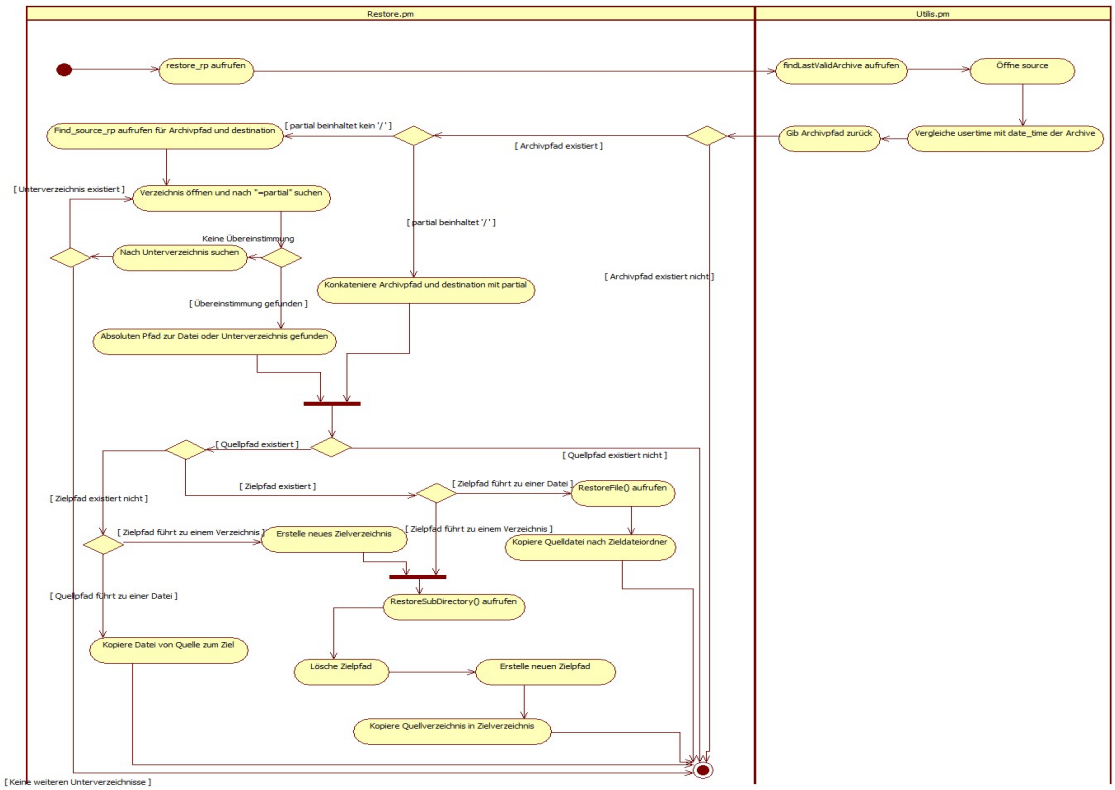


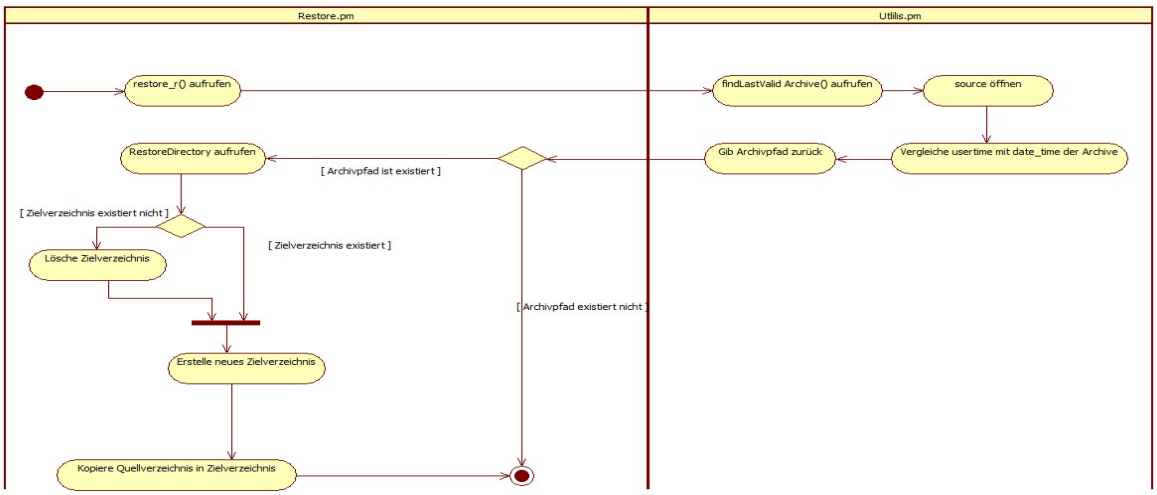
act Create_cs



6.4. Aktivitätsdiagramme zu Restore

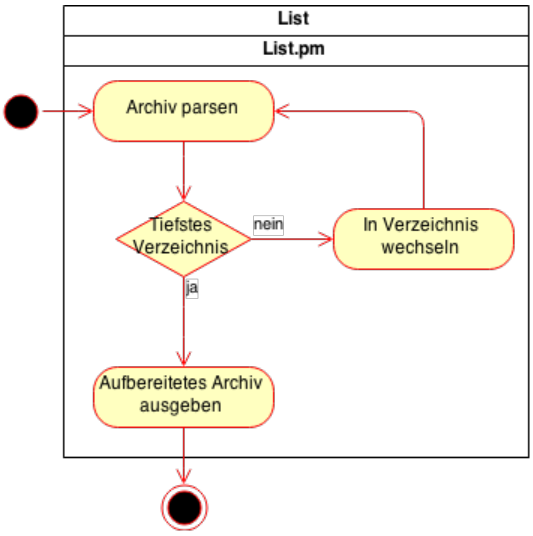






6.5. Aktivitätsdiagramme zu Delete

6.6. Aktivitätsdiagramm zu List



7. Quellenangabe

Prof. Dr.-Ing. Axel Hein (2014). Systemprogrammierung mit Perl - Projekt-Definition und Projekt-Planung. Fakultät Informatik, Technische Hochschule Nürnberg Georg-Simon-Ohm