Dokumentation: Archivierung

Patrick Vogt, Michel Angelo Ramunno, Michaela Fentze, Muhammed Kasikci

Systemprogrammierung mit Perl

Prof. Dr.-Ing. Axel Hein

Fakultät Informatik

Technische Hochschule Nürnberg

Wintersemester 2014/15

Samstag, 17. Januar 2015

Inhalt

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.

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

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.

Projektgruppe und Zuständigkeiten

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

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]
```

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.

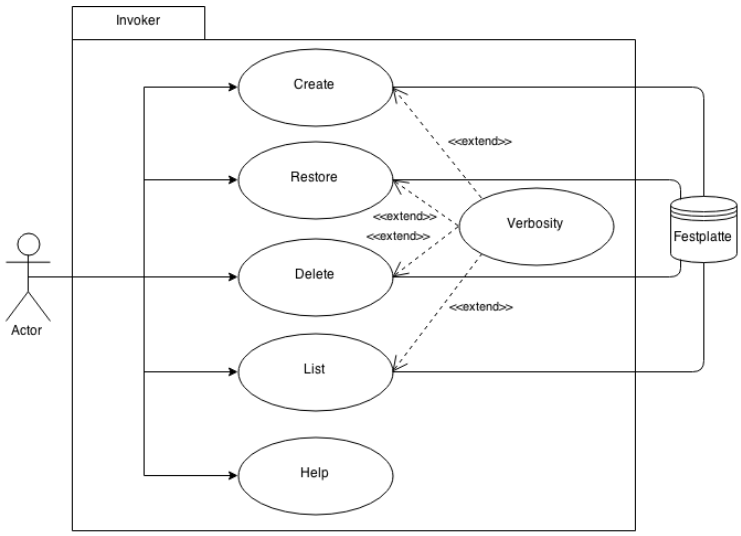
Zu den Switches werden auch noch ausführliche Schreibweisen angeboten. Nachfolgend nochmal alle Switches in einer Tabelle:

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.

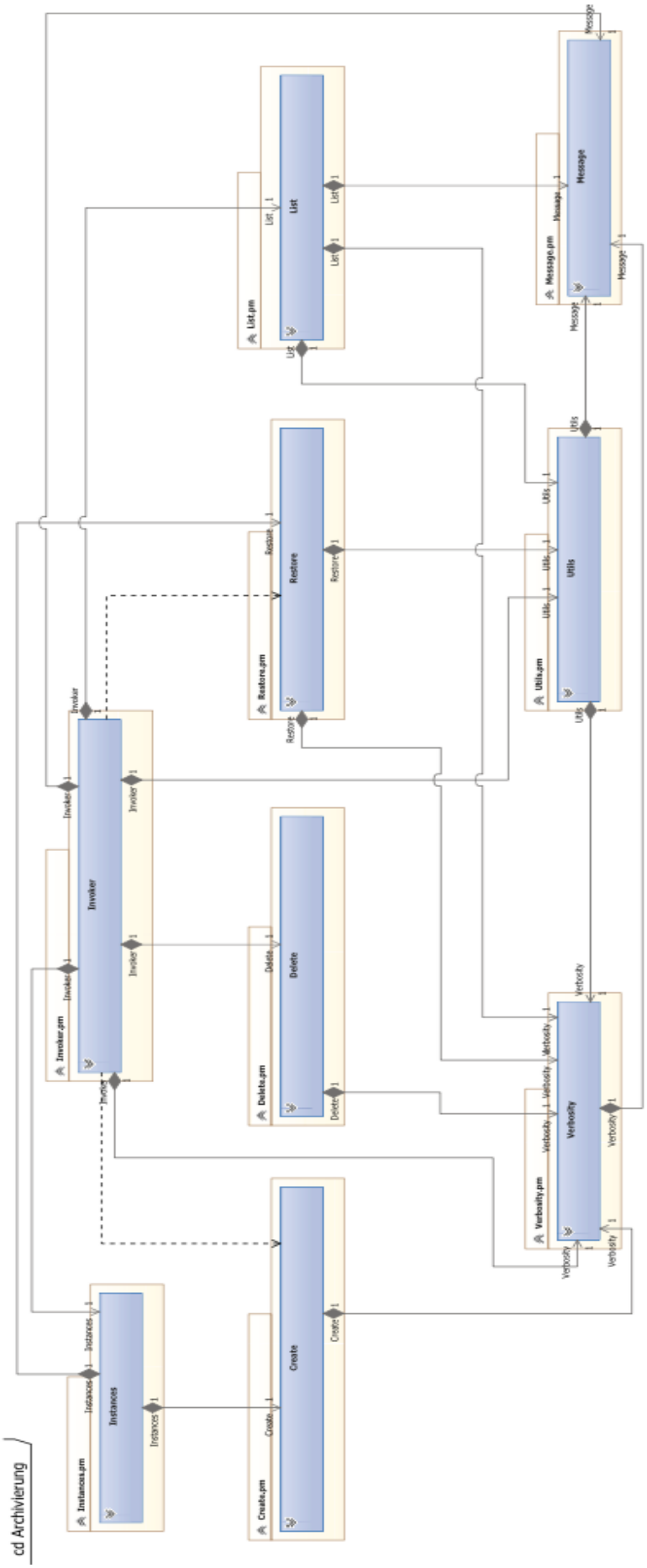
Folgende Kombinationen von Switches in Kurz- und Langform sind nicht erlaubt: cr, cd, cl, cp, cm, ch, rc, rd, rl, rs, rm, rh, dc, dr, dp, dl, ds, dm, dh, lc, lr, ld, lp, ls, lm, lh, mc, mr, md, ml, mh, hc, hr, hd, hl, hm, hv.

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.



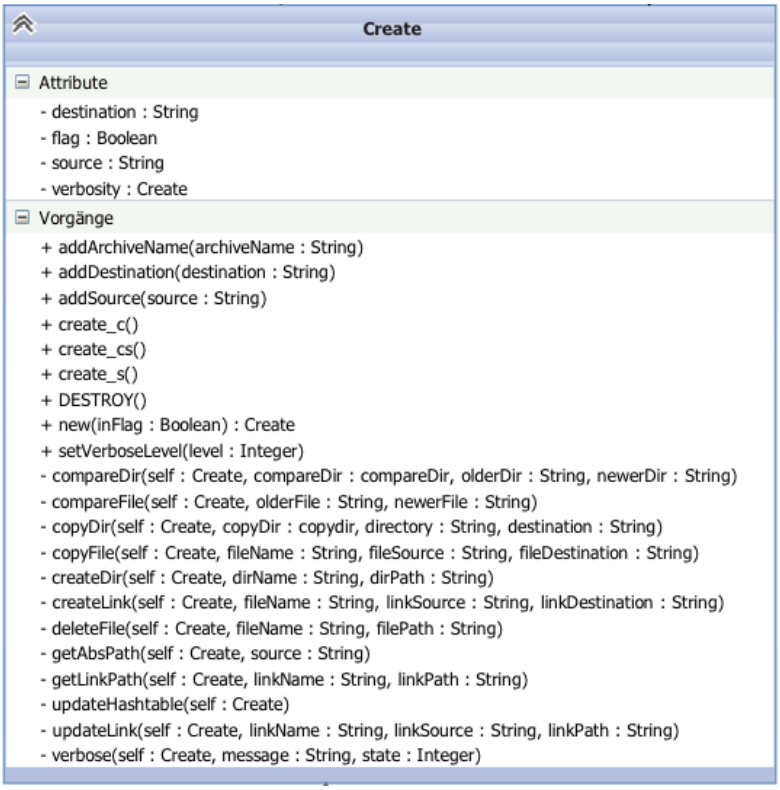
Des Weiteren sollte erwähnt werden, dass bei der Entwicklung von diesem Archivierungs-Tool stets auf das Prinzip der Erweiterbarkeit geachtet wurde. Es ist somit denkbar, dass bei Bedarf von weiteren Funktionen einfach Switches ergänzt und Methoden dem Invoker hinzugefügt werden. Die Funktionalität kann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


cd Archivierung
Message
List
Restore
Utils
Invoker
Delete
Verbosity
Instances
Create

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.

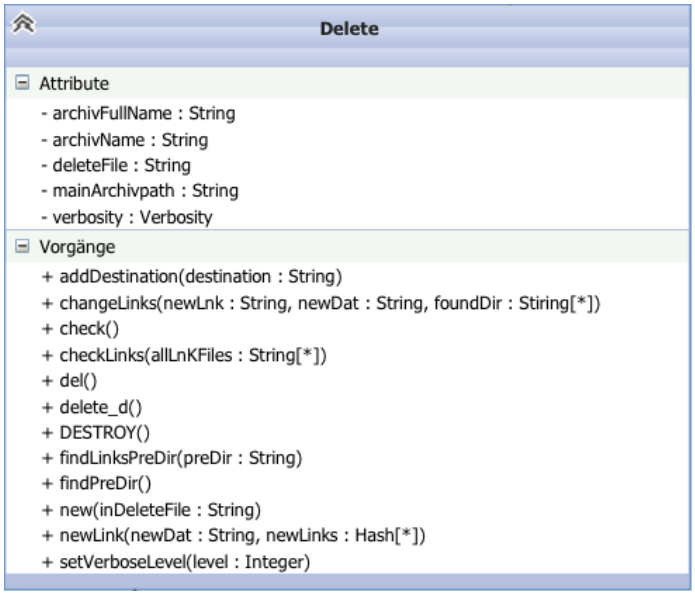


[Beschreibung von Michel Angelo]

Restore

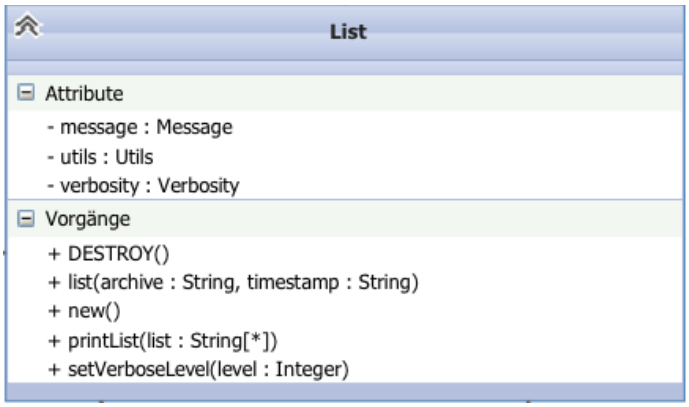
[Beschreibung von Muhammed]

Delete



[Beschreibung von Michaela]

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 mit Informationen ausgegeben.

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.

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.

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.

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.

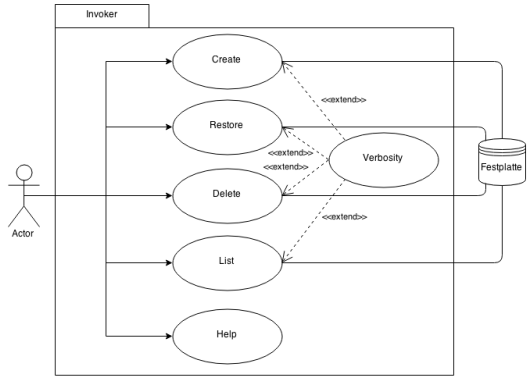
List

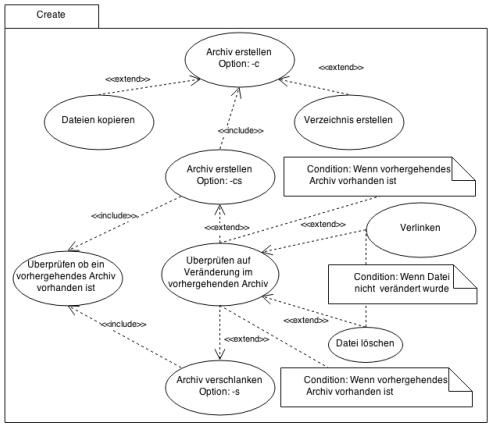
Methoden	Zeit insgesamt	Zeit List.pm	Zeit max. Modul
List	280ms	83.1ms	List.pm -> 83.1 ms

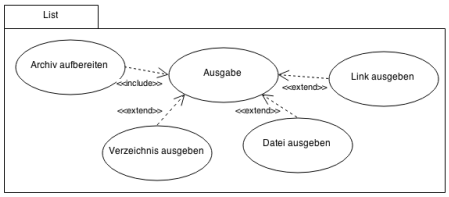
Methoden	Aufrufe insgesamt	Aufrufe List.pm	Aufrufe max. Modul
List	45171	15093	File/Find.pm -> 24432

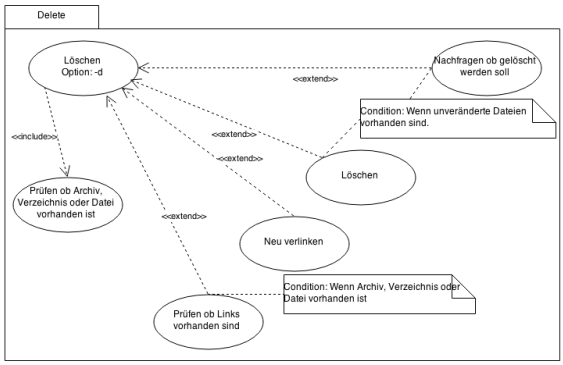
Anhang

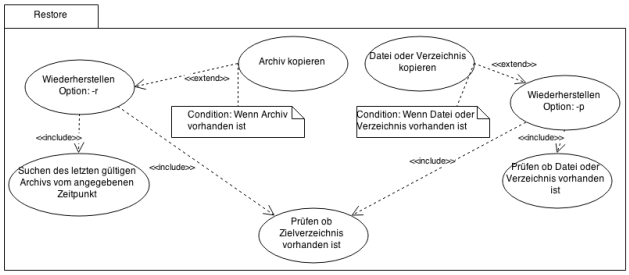
Use-Case-Diagramm



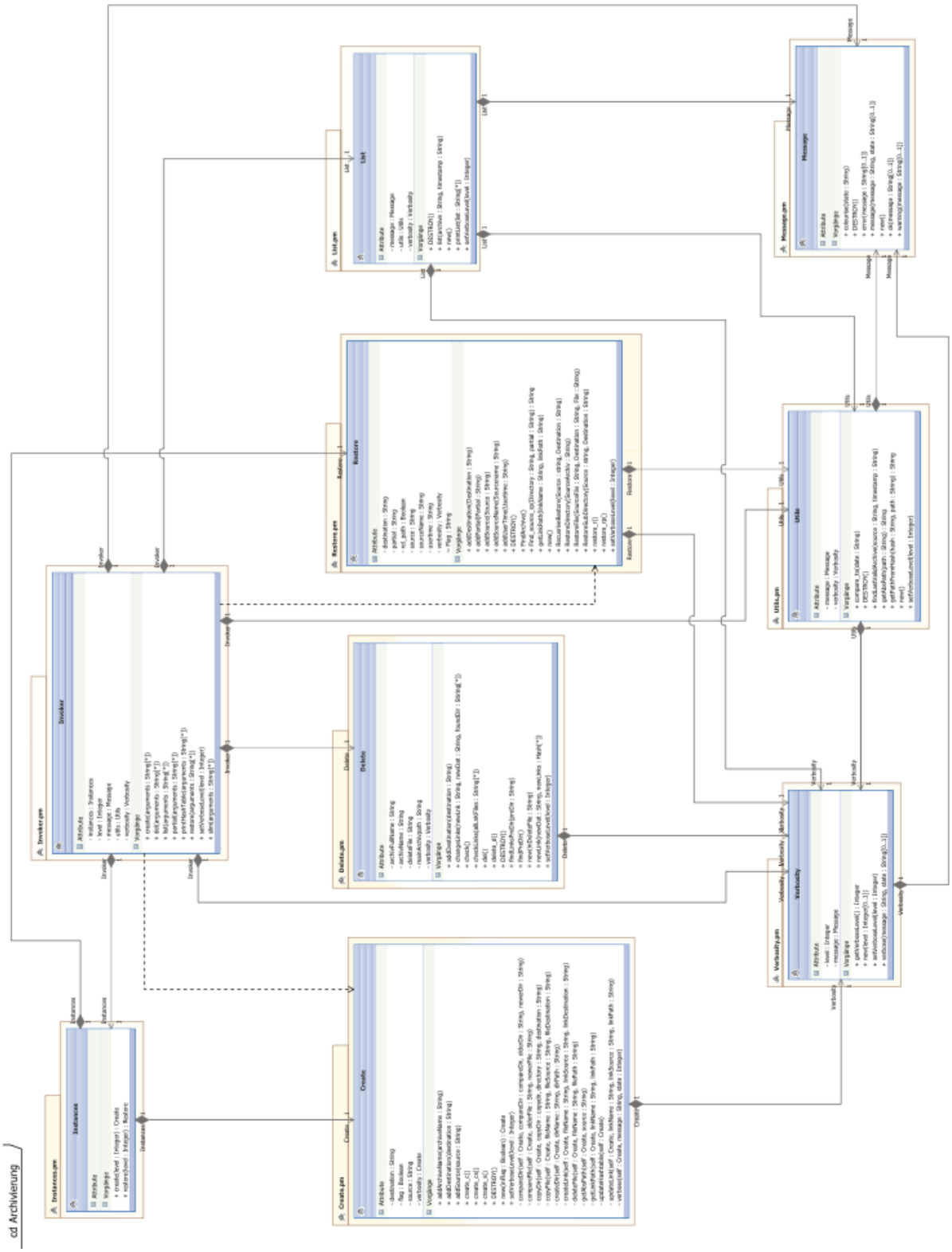








Klassendiagramm



Quellenangabe

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

[Weitere Quellenangaben]