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

Freitag, 16. 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.

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.

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

```
> 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 nicht sichtbaren 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]
```

Dieser Aufruf erzeugt zuerst wie vorhergehend schon beschrieben ein Archiv und verschlankt zusätzlich ältere Archive mit gleichen Inhalten. Das bedeutet aus gleichen Dateien erzeugt dieser Aufruf Verweise in den älteren Archiven auf die Dateien des neuen Archivs.

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

Dieser Aufruf verschlankt, wie oben Beschrieben, Archive im Archivverzeichnis mit gleichen Namen und Inhalten.

```
> perl my_perl_archive.pl [-v] -r [-p] [] [] [] [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]
```

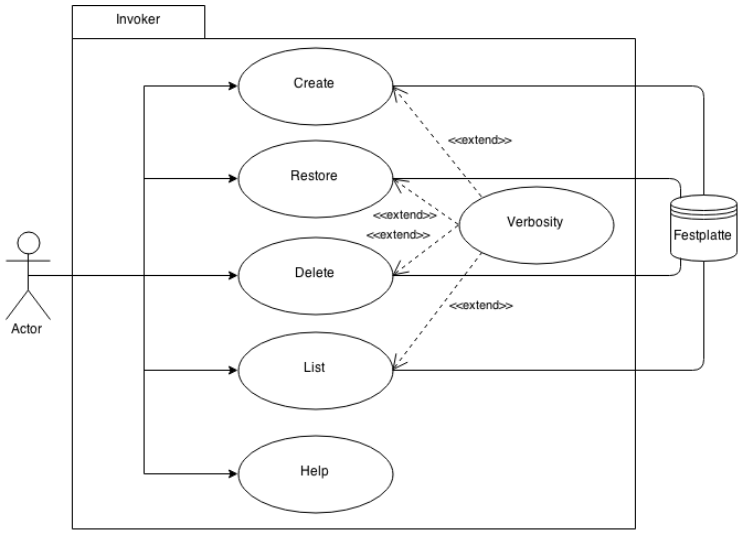
Dieser Aufruf listet den Inhalt des, vom Zeitstempel ausgehend zuletzt gültigen Archivs auf. Bei der Auflistung wird zwischen Verzeichnissen, Links (Verweisen) und normalen Dateien unterschieden.

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.

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.

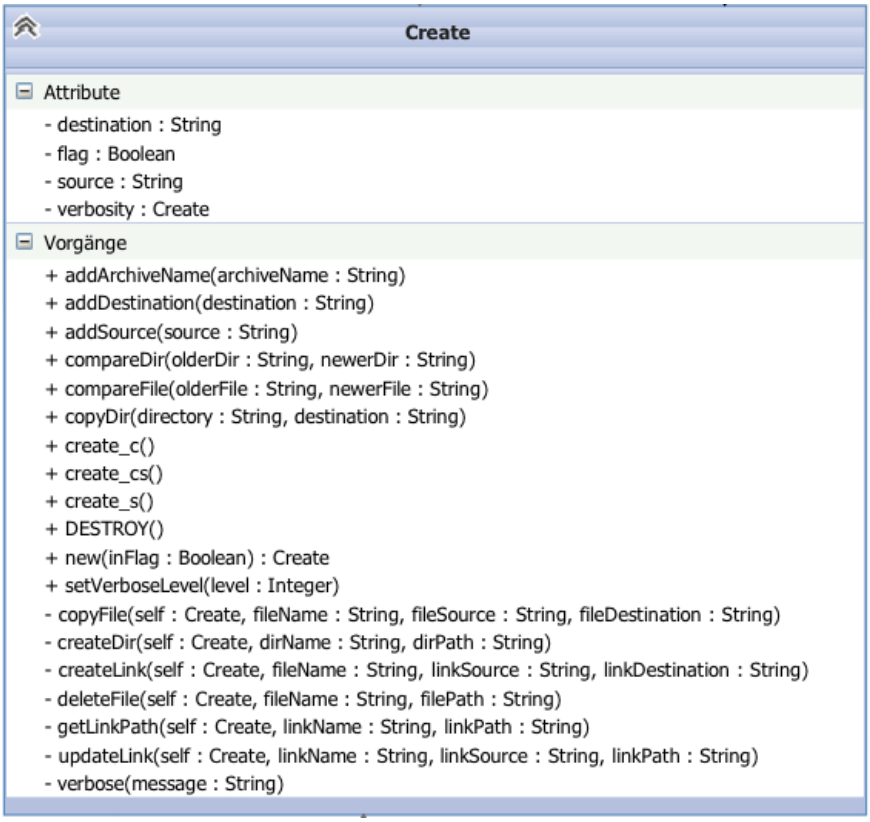


Des Weiteren sollte erwähnt werden, dass bei der Entwicklung von diesem Archivierungs-Tool stets auf das Prinzip der Erweiterbarkeit geachtet wurde. Es ist somit denkbar, dass bei Bedarf von weiteren Funktionen einfach Switches ergänzt und Methoden dem Invoker hinzugefügt werden. Die Funktionalität kann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

[KLASSENDIAGRAMM OHNE ATTRIBUTE UND METHODEN]

[Beschreibung des Klassendiagramms]

Create

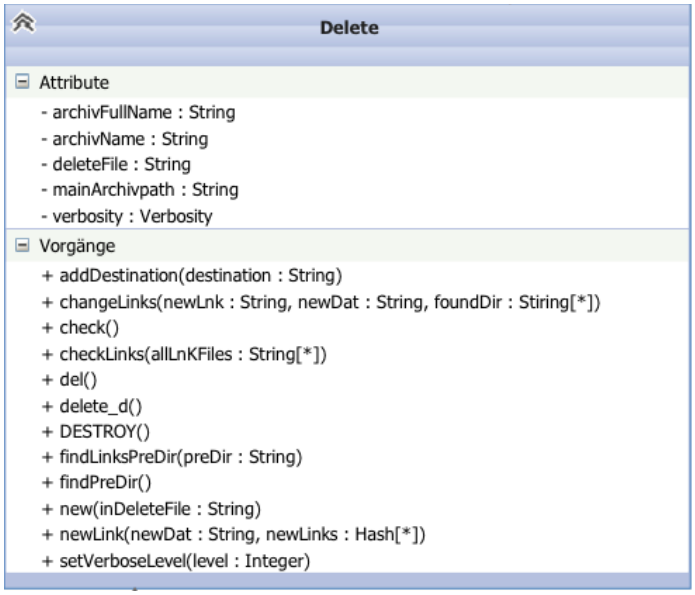


[Beschreibung von Michel Angelo]

Restore

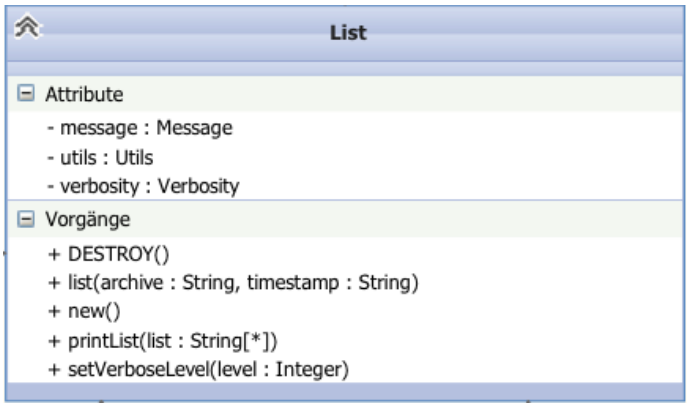
[Beschreibung von Muhammed]

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

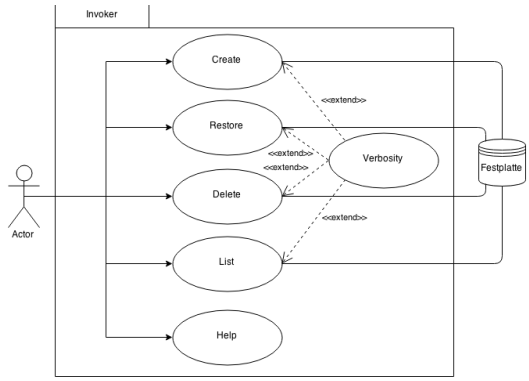
Element iteriert. Dabei wird geprüft, ob das Element ein Verzeichnis, ein Link (Verweis) oder eine Datei ist und entsprechend mit Informationen ausgegeben.

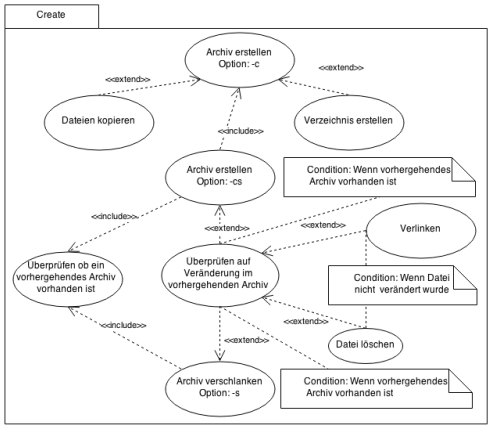
Aufrufhierarchie mit NYTProf

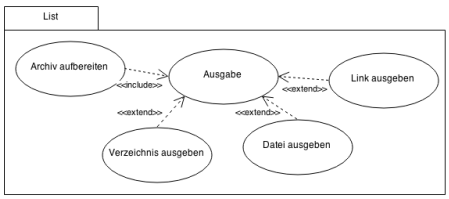
[Beschreibung von Muhammed]

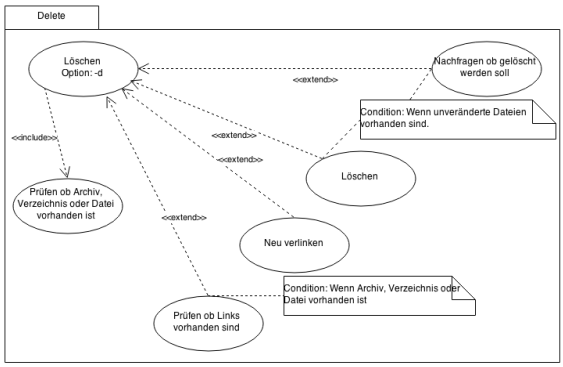
Anhang

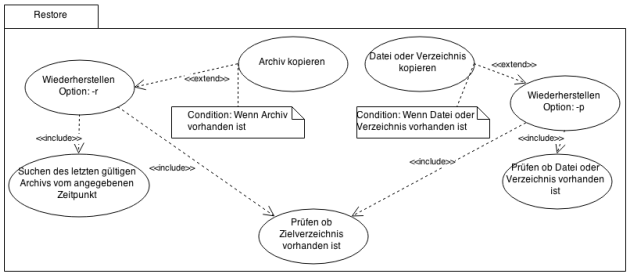
Use-Case-Diagramm



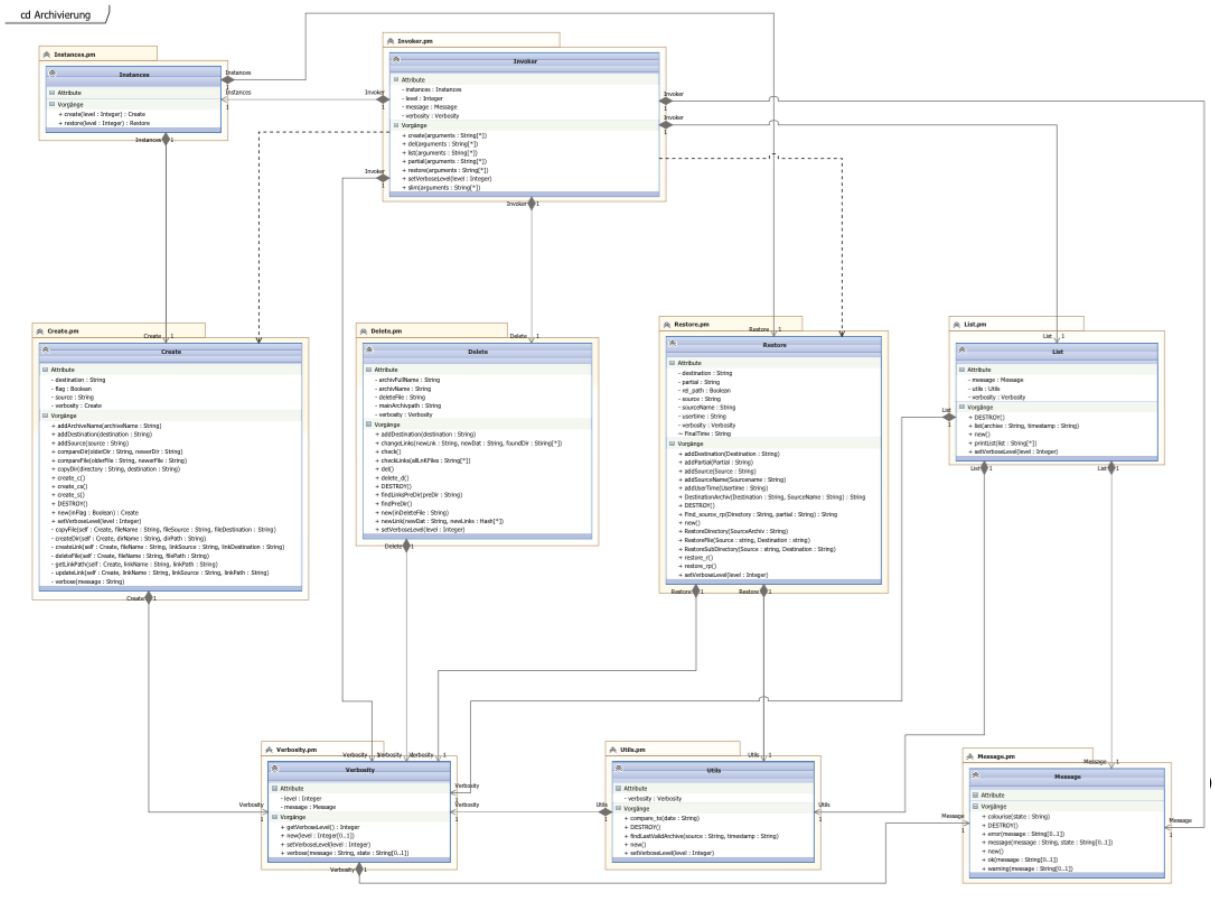








Klassendiagramm



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]