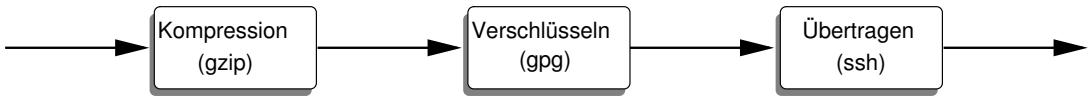
Prof. Dr. M. Jäger, FB MNI, FH Gießen-Friedberg

Programmierung mit Datenströmen: Was herauskommt, wenn man Pipelines verbiegt

Das altbekannte, vom UNIX-Betriebsystem stammende Konzept der Pipelines ist heute sogar zum Softwarearchitekturstil („pipes and filters“) avanciert. Welch eine Karriere! Grund genug, etwas mit den Pipelines herumzuspielen.

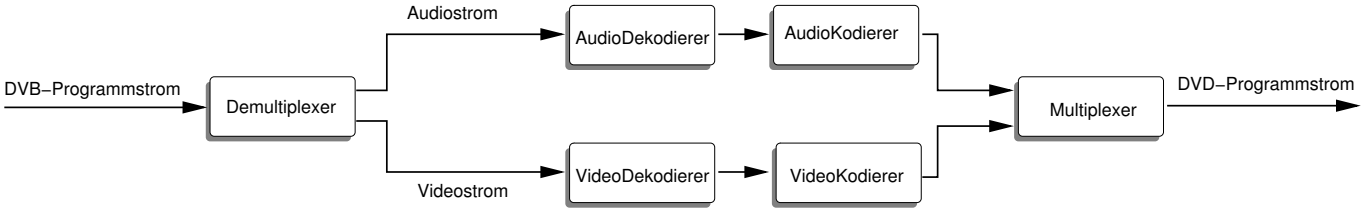
Eine Pipeline etabliert einen Datenstrom, der durch mehrere gleichzeitig aktivierte Transformationsprozesse, Filter, in Fließband-Manier verarbeitet wird. Zwischen zwei Filtern wird jeweils eine „Pipe“ als Kommunikationspuffer zur Entkopplung verwendet.

Ein Beispiel: Eine Pipeline, die Dateien erst komprimiert, dann kryptographisch verschlüsselt und schließlich auf einem entfernten Rechner abspeichert:



Filter können prinzipiell auch mehrere Eingabedatenströme zu einem Ausgabedatenstrom weiterverarbeiten oder aus einem Eingabedatenstrom mehrere Ausgabedatenströme erzeugen. Es gibt also Verzweigungen und Zusammenführungen von Datenströmen.

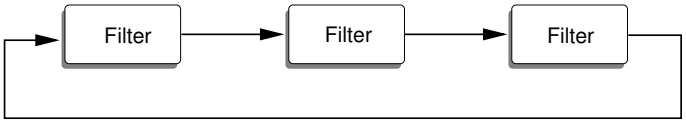
Beispiel: Eine Pipeline erzeugt aus einer digitalen Fernsehaufzeichnung (DVB-Programmstrom) ein für das Brennen einer Video-DVD geeignetes Format, wobei sowohl der Ton als auch das Bild umkodiert werden (vereinfacht):



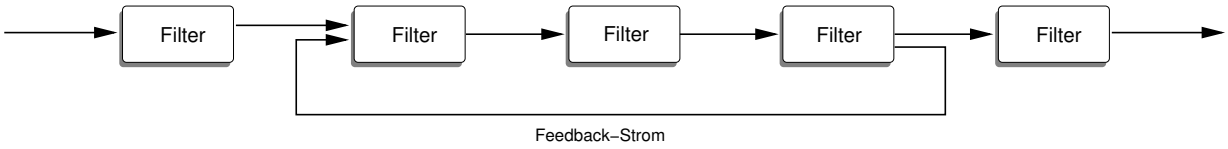
Ohne das Pipelinekonzept steigt der Zeitbedarf für die Verarbeitung eines Films durch die notwendigen Zwischenspeicherungen auf einer Festplatte auf ein Vielfaches.

Solcherlei Anwendungen sind bekannt und bewährt.

Normale Pipelines sind „gerade Fließbänder“ (siehe Bild oben). Was aber, wenn man die Pipeline kräftig verbiegt und einen Ring daraus formt?



Oder, etwas allgemeiner, in Kombination mit Verzweigungen und Zusammenführungen:



Ich nenne diese Gebilde „Ring-Pipelines“ oder „Feedback-Pipelines“.

Der geringe Aufwand der technischen Realisierung soll anhand einer POSIX-Konformen Shell gezeigt werden. Während die „klassische“, lineare Pipeline dort wie folgt aussieht:

```
filter1 | filter2 | filter3 | ...
```

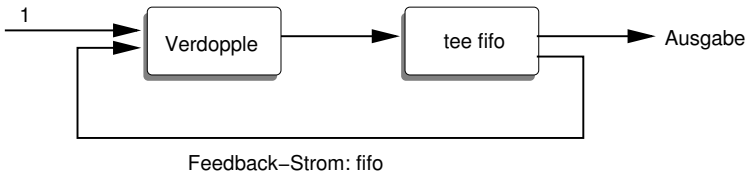
ist die ringförmige Anordnung mittels einer benannten Pipe (FIFO) einfach realisierbar. Nehmen wir an, der Ausgabedatenstrom von Filter 4 soll sowohl an Filter 5 weitergeleitet werden als auch an Filter 2 zurückgeführt werden (wie im Bild oben):

```
mkfifo f
(filter1;cat f) | filter2 | filter3 | filter4 | tee f | filter5
```

Der Standardfilter „tee“ dupliziert seinen Eingabestrom, die Daten werden über eine namenlose Pipe sowohl an Filter 5 als auch durch die benannte Pipe „f“ an Filter 2 weitergeleitet. In diesem Fall wird Filter 2 zuerst die Ausgabedaten von Filter 1 erhalten und anschließend die Feedback-Daten von Filter 4. Beliebige andere Konstellationen zur Zusammenführung beider Datenströme sind machbar.

Eine Ring-Pipeline kann, mit einem Initialwert versorgt, auf einfache Weise einen unendlichen Datenstrom erzeugen.

Ein einfache Anwendung: Eine Ring-Pipeline, die alle Zweierpotenzen (ab 2 aufwärts) ausgibt:



Der Shell-Code dazu demonstriert Eleganz und Einfachheit der Realisierung:

```
mkfifo f
(echo 1;cat f) | verdopple | tee f
```

Dabei könnte man den Filter zum Verdoppeln wie folgt definieren:

```
verdopple() { while read x; do expr 2 "*" $x; done; }
```

Die Konstellation ist leicht erweiterbar. Um etwa die Folgenelemente zu summieren, müsste man nur einen Addierer hinter den Verdoppler einfügen.

Gibt es für Ring-Pipelines nicht-triviale sinnvolle praktische Anwendungen?

Ja! Dazu ein Beispiel:

Während meines Forschungssemesters wurde auf dem FH-Homepage-Server vom DVZ sowohl der ssh-Zugang als auch der rsync-Dienst endgültig eingestellt. Die Situation ist typisch für die meisten Internet-Provider und führt zu folgender Problemstellung:

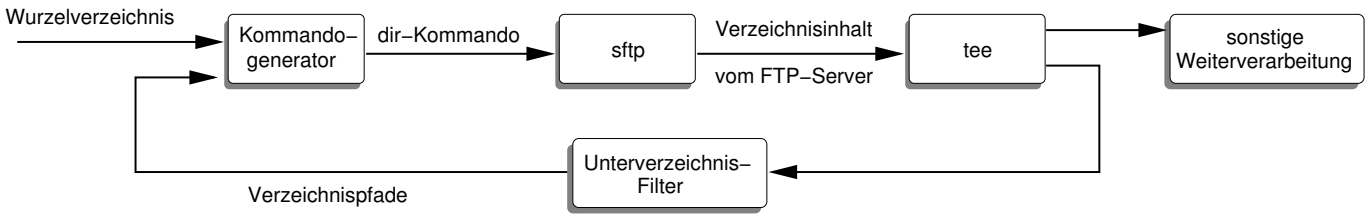
Wie kann eine vollautomatische Spiegelung von Verzeichnissen über das Internet erfolgen, wenn man nur mit primitiven Programmen wie „sftp“ auf nicht-interaktive Weise auf den WWW-Server zugreifen kann?

Positiv: sftp agiert auf Wunsch als Filter
Negativ: Weder FTP-Server noch sftp können rekursive Verzeichnislisten erstellen

Die Datenstrom-orientierte Lösung:

- Man versorgt sftp mit dem Wurzelverzeichnis der zu spiegelnden Hierarchie als Initialwert und veranlasst die Ausgabe einer nicht-rekursiven Verzeichnisliste
- In einer Ring-Pipeline werden daraus die Unterverzeichnisse selektiert und in die Pipeline „zurückgefüttert“. Dadurch wird der Unterverzeichnis-Baum komplett aufgebaut und kann in beliebiger Weise verarbeitet werden.

Das Datenstrom-basierte Konzept ist von der Architektur her einfacher und eleganter als der klassische Lösungsansatz, der auf einer Zerlegung der Client-Server-Kommunikation in einzelne Kommunikationsschritte basiert. Außerdem bietet der hohe Grad an Nebenläufigkeit bei der Datenstrom-Lösung die besten Voraussetzungen für eine gute Effizienz.



Fazit: Pipelines können mehr, als gemeinhin bekannt ist!