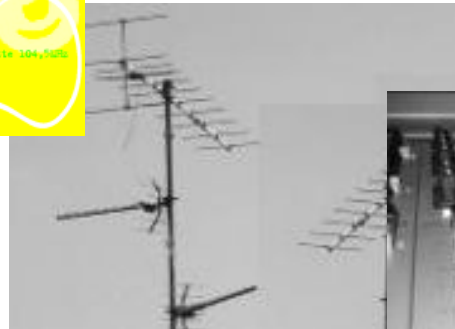
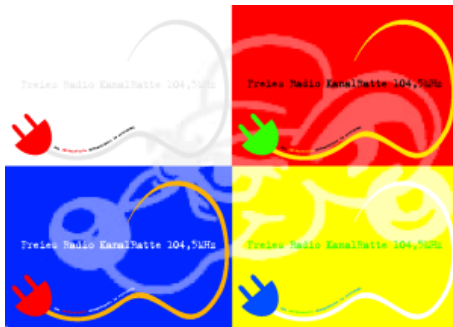




Kanal Rette  
Freies Radio Schopfheim



## Inhalt

- **Portrait: Freie Radios und Kanal Ratte**
- **Übersicht: Radio- und Studio-Technik**
- **Archivierung**
- **Sendeautomatisierung**
- **Streaming**
- **Anwenderprogramme**
- **Weitere Ideen und Ausblick**
- **verwendete Software und Informationsquellen**



## **Manuel Schneider**

- **24 Jahre**
- **Fachinformatiker (Systemintegration) seit 2003**
- **momentan Student für angewandte Informatik (FH)**
- **All-Things-Open Projektgruppe**
- **Freies Radio Kanal Ratte (Schopfheim)**
- **Kooperation mit RadioTux**

## Portrait: Freie Radios

### Herkunft / Zielgruppe

- Bürgerradio
- z. T. aus Piratensendern der 80er Jahre entstanden
- Anti-AKW-Bewegung
- oft alternativ / autonom
- für Jedermann offen

## Portrait: Freie Radios

### Organisationsform

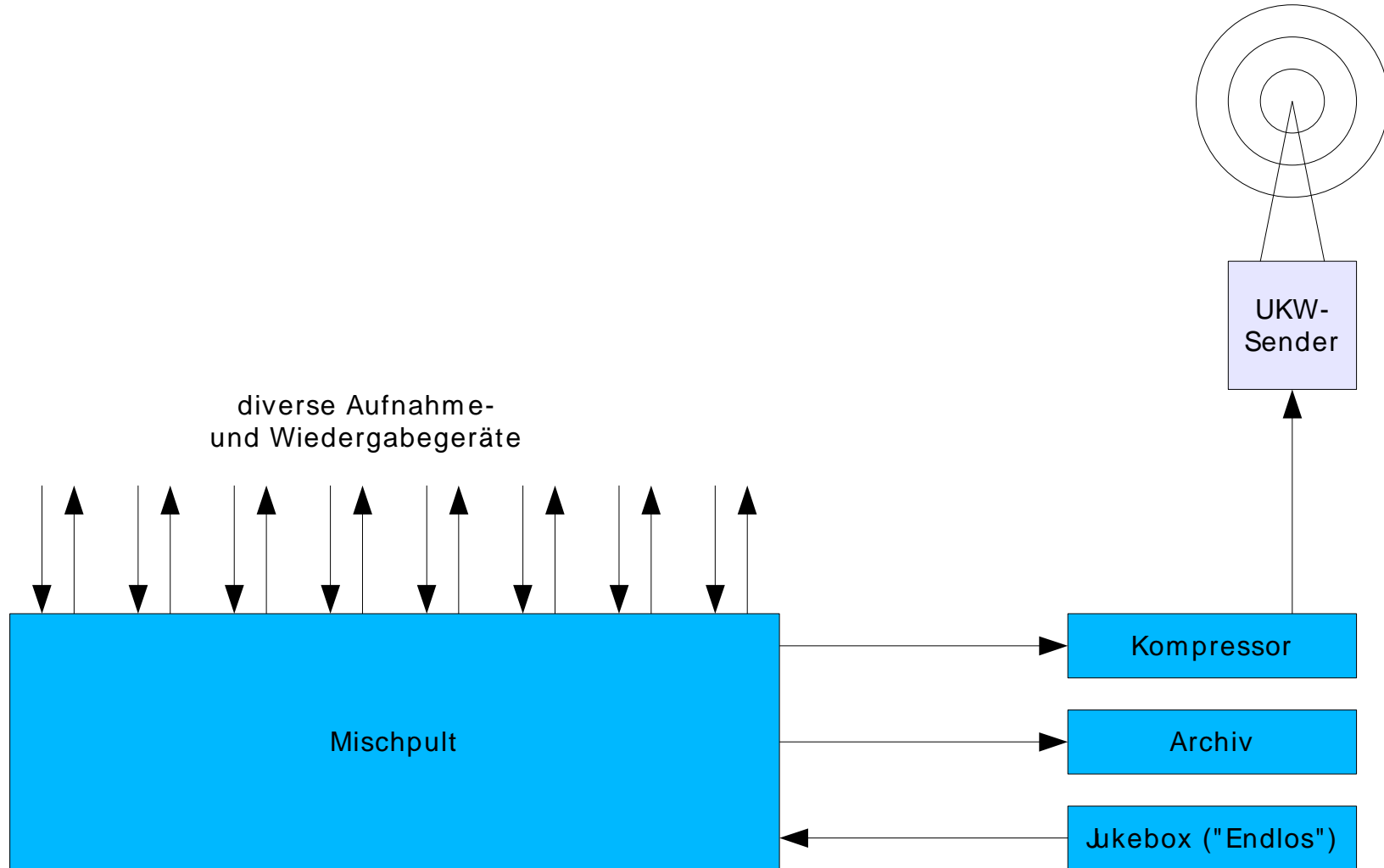
- meist Vereine
- meist basisdemokratisch
- selten auch GmbHs

## Portrait: Freie Radios

### Finanzierung

- Landesmedienanstalten
- Spenden
- Mitgliedsbeiträge
- nicht kommerziell, daher keine Werbung

## Übersicht: Radio- und Studio-Technik



## Übersicht: Radio- und Studio-Technik

### Anforderungen:

- **Sende-Archivierung (gesetzlich: sechs Wochen)**  
früher: Aufnahme auf Videobänder
- **Endlos-Band**  
früher: manuell bediente Kassetten
- **Sendeführung von anderen Sendern**  
früher: Kassetten per Post
- **Livestream**
- **PC für bequeme Aufnahme / Schnitt / Musikwiedergabe**

## Archivierung

### Setup

- **Server am Aufnahme-Ausgang des Mischpults**
- **qualitativ hochwertige Soundkarte**
- **Verzeichnisstruktur bildet Kalender ab**
- **Aufnahmeskript erstellt MP3-Datei (stündlich)**
- **Cron-Job**

## Archivierung

```
/usr/local/bin/aufnahme-rkr.sh  
# nimmt das mp3-file auf  
# Variablen setzen  
NAME=$(date +Kanal-Ratte-%H-Uhr-%d-%m-%Y)  
HEUTE=$(cat /usr/share/kanalratte/heute)  
ARCHIVPFAD=/mnt/data/sendearchiv  
export TERM=xterm  
# mp3 aufnehmen  
killall -gq arecord  
/usr/local/bin/sub/pegel-aufnahme.sh  
arecord -t wav -f cd -d 3605 -q | /usr/bin/lame -b128 -s -  
$ARCHIVPFAD/$HEUTE/$NAME.mp3 > /dev/null
```

```
/usr/local/bin/aufnahme-rdl.sh  
# nimmt das mp3-file auf  
# Variablen setzen  
NAME=$(date +Radio-Dreyeckland-%H-Uhr-%d-%m-%y)  
HEUTE=$(cat /usr/share/kanalratte/heute)  
ARCHIVPFAD=/mnt/data/sendearchiv  
export TERM=xterm  
# mp3 aufnehmen  
killall -gq arecord  
/usr/local/bin/sub/pegel-aufnahme.sh  
arecord -t wav -r 44100 -d 3605 -q | /usr/bin/lame -s 22.05 -b24 -s -  
$ARCHIVPFAD/$HEUTE/$NAME.mp3 > /dev/null
```

```
/usr/local/bin/sub/pegel-aufnahme.sh  
#!/bin/bash  
amixer sset ADC,0 127 > /dev/null  
amixer sset ADC,1 127 > /dev/null
```

## Archivierung

### Crontab (Auszug)

```
...
#### Sendearchivierung ####
# RKR
0 15-23 * * * /usr/local/bin/aufnahme-rkr.sh
0 0-2 * * * /usr/local/bin/aufnahme-rkr.sh
## RDL
0 3-14 * * * /usr/local/bin/aufnahme-rdl.sh
#
#### Wartungsarbeiten ####
1 0 * * * /usr/local/bin/variablen.sh
55 23 * * * /usr/local/bin/verzeichnis.sh
3 7 * * * /usr/local/bin/copymp3.sh
...
```

### `/usr/local/bin/verzeichnis.sh`

```
#!/bin/bash
# Verzeichnisse erzeugen und löschen
# Variablen setzen

HEUTE=$(cat /usr/share/kanalratte/heute)
MORGEN=$(cat /usr/share/kanalratte/morgen)
LDATUM=$(cat /usr/share/kanalratte/ldatum)
#CDATUM=$(cat /usr/share/kanalratte/cdatum)
WOCHENTAG=$(cat /usr/share/kanalratte/Wt_heute)
ARCHIV=/mnt/data/sendearchiv

# cp -r -u $ARCHIV_NEU/$CDATUM $ARCHIV_ALT
rm -rf $ARCHIV/$LDATUM
mkdir $ARCHIV/$MORGEN
cp $ARCHIV/liesmich.txt $ARCHIV/$HEUTE/$WOCHENTAG.txt
chmod 777 $ARCHIV/$MORGEN
```

## Archivierung

```
/usr/local/bin/variablen.sh
#!/bin/bash
# setzt Umgebungsvariablen
VDIR="/usr/share/kanalratte"

# heutiges Datum für verzeichnisse etc..
echo $(date +%Y-%m-%d) > $VDIR/heute
# morgiges datum
echo $(date +%Y-%m-%d --date "tomorrow") > $VDIR/morgen

# welcher tag muß gelöscht werden
echo $(date +%Y-%m-%d --date "43 days ago") > $VDIR/ldatum
# welcher tag soll kopiert werden
echo $(date +%Y-%m-%d --date "1 day ago") > $VDIR/gestern

# aktueller Wochentag
echo $(date +%A) > $VDIR/Wt_heute

#Datum letzter X-Tag
for wt in Monday Tuesday Wednesday Thursday Friday Saturday Sunday
do
  for i in 1 2 3 4 5 6 7
  do WOCHENTAG=$(date +%A --date "$i day ago")
  echo $(date +%Y-%m-%d --date "$i day ago") > $VDIR/$wt
  if test $WOCHENTAG = $wt
  then break
  fi
done
done
```

## Archivierung

```
/usr/local/bin/copymp3.sh
# kopiert mp3-file der Stunde 0
# Variablen setzen
NAME=$(date +Kanal-Ratte-00-Uhr-%d-%m-%Y)
HEUTE=$(cat /usr/share/kanalratte/heute)
GESTERN=$(cat /usr/share/kanalratte/gestern)
ARCHIVPFAD=/mnt/data/sendearchiv
mv $ARCHIVPFAD/$GESTERN/$NAME.mp3 $ARCHIVPFAD/$HEUTE/$NAME.mp3
```

## Sendeautomatisierung

### Setup

- **Archivrechner auf Eingang vom Mischpult legen**
- qualitativ hochwertige Soundkarte
- **Wiedergabeskript spielt gewünschte Sendung ab**
- **zusätzliches Vorproduktionsverzeichnis**
- **Cronjobs gemäss Sendepplan**

## Sendeautomatisierung

```
/usr/local/bin/sub/wdh_ciao.sh
# Benutzer prüfen (wegen manuellem Start), muss "archiv" sein
/usr/local/bin/sub/checkuser.sh

# Tag der Originalsendung
TAG=$(cat /usr/share/kanalratte/Saturday)
ARCHIVPFAD=/mnt/data/sendearchiv/$TAG

# Verzeichnis fuer bearbeitete oder ausgewaehlte Dateien - diese werden bevorzugt gespielt!
cd /mnt/data/musik/WDH/ciao

# Erstelle Liste aus Zusatzverzeichnis
ls -l *.mp3 > templiste.txt
# Hinzufügen der normalen Archivpfade
echo $ARCHIVPFAD/Kanal-Ratte-16-Uhr-*.mp3 >> templiste.txt
echo $ARCHIVPFAD/Kanal-Ratte-17-Uhr-*.mp3 >> templiste.txt

# Reduzieren auf n Files = n Stunden - dh. jede Datei enthält eine Stunde Programm!
head -n2 $ZUSATZPFAD/templiste.txt > $ZUSATZPFAD/liste.txt

# Herunterfahren des aktuellen Pegels und Stoppen der Wiedergabe
/usr/local/bin/sub/auspegeln.sh
/usr/local/bin/sub/meta-update.sh "Kanal Ratte - Ciao Italia"

# Starten der Wiedergabe
mpg123 -r 44100 -q -@ liste.txt | /usr/local/bin/sub/einpegeln-rkr.sh

# Fortsetzen mit Endlosbetrieb
/usr/local/bin/endlos.sh
```

## Sendeautomatisierung

### Crontab (Auszug)

```
...
#### Wiederholungen ####
# MONTAG:
0 15 * * 1      /usr/local/bin/endlos-start.sh
0 16 * * 1      /usr/local/bin/osp_mo16.sh
0 17 * * 1      /usr/local/bin/sub/meta-update.sh "Kanal Ratte - DonnaWetter"
0 18 * * 1      /usr/local/bin/sub/meta-update.sh "Kanal Ratte - Sound ohne Ende"
0 19 * * 1      /usr/local/bin/wdh_celentano.sh
0 20 * * 1      /usr/local/bin/wdh_mo20uhr.sh
0 20 * * 1      /usr/local/bin/sub/meta-update.sh "Kanal Ratte - Klassik"
0 22 * * 1      /usr/local/bin/wdh_musicwave.sh
0 0 * * 2      /usr/local/bin/wdh_revelation.sh
...
```

```
/usr/local/bin/sub/checkuser.sh
if [[ `id -nu` != "archiv" ]]
then
  echo Falscher Benutzer!
  su archiv
  exit 1
fi
```

## Sendeautomatisierung

```
/usr/local/bin/sub/auspegeln.sh
```

```
for ((i=`amixer sget DAC,0 | grep % | cut -d " " -f 4`; i > 20 ; i--))
do
  amixer sset DAC,0 $i > /dev/null
  amixer sset DAC,1 $i > /dev/null
  sleep 0.05
done
killall -gq mpg123
killall -gq ogg123
```

```
/usr/local/bin/sub/einpegeln-rkr.sh
```

```
for ((i=40; i <= 95 ; i++))
do
  amixer sset DAC,0 $i > /dev/null
  amixer sset DAC,1 $i > /dev/null
  sleep 0.06
done
```

```
/usr/local/bin/sub/einpegeln-rdl.sh
```

```
for ((i=61; i <= 121 ; i++))
do
  amixer sset DAC,0 $i > /dev/null
  amixer sset DAC,1 $i > /dev/null
  sleep 0.06
done
```

```
/usr/local/bin/sub/pegel-rkr.sh
```

```
amixer sset DAC,0 95 > /dev/null
amixer sset DAC,1 95 > /dev/null
```

```
/usr/local/bin/sub/pegel-rdl.sh
```

```
amixer sset DAC,0 121 > /dev/null
amixer sset DAC,1 121 > /dev/null
```

## Sendeautomatisierung

### Endlos-Band

- grosse Musiksammlung auf dem Server
- Jingles
- Zufallsskript
- Endlos-Skript
- Cron-Jobs

## Sendeautomatisierung

```
/usr/local/bin/endlos-start.sh
#Endlosbetrieb (mp3)
/usr/local/bin/sub/checkuser.sh
/usr/local/bin/sub/meta-update.sh "Kanal Ratte - Sound ohne Ende"
/usr/local/bin/endlos.sh
```

```
/usr/local/bin/sub/endlos.sh
#Endlosbetrieb (mp3)
/usr/local/bin/sub/checkuser.sh
/usr/local/bin/sub/auspegeln.sh
/usr/local/bin/sub/pegel-rkr.sh
```

```
cd /mnt/data/endlos/1
rm playlist 2> /dev/null

for j in playlist*
do
    mv $j playlist
    mpg123 -r 44100 -q -@ playlist > /dev/null
done

/usr/local/bin/sub/zufallsliste.sh
rm playlist 2> /dev/null
for j in playlist*
do
    mv $j playlist
    mpg123 -r 44100 -q -@ playlist > /dev/null
done
mpg123 -q -Z *.mp3
```

## Sendeautomatisierung

```
/usr/local/bin/sub/zufallsliste.sh
# Abspielen ALLER Titel eines Verzeichnisses
# in Zufallsreihenfolge mit JINGLES aller 4 Titel
JINGLEPFAD=/mnt/data/endlos/jingles
MUSIKPFAD=/mnt/data/endlos/l
ANZTITEL=0
ANZJINGLE=6

cd $MUSIKPFAD
rm playlist* neulist urliste *.ti

# Auflisten aller mp3-Dateien
cd $MUSIKPFAD
ls *.mp3 > urliste
split -1 urliste list

for i in list*
do mv $i $RANDOM.ls
done

for i in *.ls
do ((ANZTITEL=${ANZTITEL+1}))
mv $i $ANZTITEL.ti
done

cd $MUSIKPFAD
rm neulist
echo $JINGLEPFAD/RKR-Jingle_JeffBuckley.mp3 > neulist
j=0
JINGLE=0
for i in *.ti
do echo $(cat $i) >> neulist
((j=${j+1}))
if [ ${j%5} == 0 ]
then ((JINGLE=${JINGLE+1}))
JI=jingle$JINGLE.jg
echo $(cat $JI) >> neulist
if [ ${JINGLE%6} == 0 ]
then JINGLE=0
fi
fi
done

split -12 neulist playlist
```

## Sendeautomatisierung

### **Sendeführung von anderen Sendern**

- **Programm per Livestream übernehmen**
- **ausreichende Internetanbindung**
- **ausfallsicheres Skript mit Fallback zu Endlos**
- **Überwachungsskript zum Umschalten nach Ausfall**
- **Cron-Jobs**

## Sendeautomatisierung

```
/usr/local/bin/rdl-start-ansage.sh
# Start des RDL-livestreams mit Ansage
/usr/local/bin/sub/auspegeln.sh
/usr/local/bin/sub/pegel-rkr.sh
mpg123 -q /mnt/data/endlos/ansage_sender.mp3
/usr/local/bin/sub/meta-update.sh "03 - 15 Uhr: Radio Dreyeckland"
/usr/local/bin/rdl.sh

/usr/local/bin/sub/endlos.sh
# Livestream abspielen solange bis von extern Prozesstruktur getötet wird
while /bin/true
do
  /usr/local/bin/sub/auspegeln.sh
  # Prüfen, ob Quell-Stream läuft
  /usr/local/bin/sub/checkstream.sh http://mail:8000/rdl.ogg
  if [[ $? == 0 ]]
  then
    # Stream läuft, also aufschalten
    /usr/local/bin/sub/meta-update.sh "03 - 15 Uhr: Radio Dreyeckland"
    ogg123 -q http://mail:8000/rdl.ogg | /usr/local/bin/sub/einpegeln-rdl.sh
  else
    # Stream läuft nicht, Endlos aufschalten
    /usr/local/bin/endlos.sh | /usr/local/bin/sub/einpegeln-rkr.sh
  fi
done
```

## Sendeautomatisierung

```
/usr/local/bin/sub/checkstream.sh
# Stream mit curl abrufen:
# -f Fehlerausgabe aktiviert
# -s Fortschrittsanzeige deaktiviert
# -m max. eine Sekunde abrufen
# Ausgabe (Datenstrom) nach /dev/null verschieben
/usr/bin/curl -fs -m 1 $1 > /dev/null

# Fehlernummer des obigen Aufrufs auswerten:
if [ $? == 28 ]; then
    # Fehler 28 bedeutet, dass der Stream wegen Zeitüberschreitung abgebrochen wurde, dh. Stream
    läuft
    exit 0
else
    # Ansonsten ist ein anderer Fehler aufgetreten (üblicherweise 'nicht gefunden')
    exit 1
fi
```

## Sendeautomatisierung

```
/usr/local/bin/checkstream.sh
# Ueberpruefung ob RDL-Stream laeuft
if (pidof ogg123) > 0
  then
    exit 0
  else
    /usr/local/bin/sub/checkstream.sh http://mail:8000/rdl.ogg
    if [[ $? == 0 ]]
      then
        /usr/local/bin/rdl.sh
      else
        /usr/local/bin/checkendlos.sh
    fi
  fi
fi

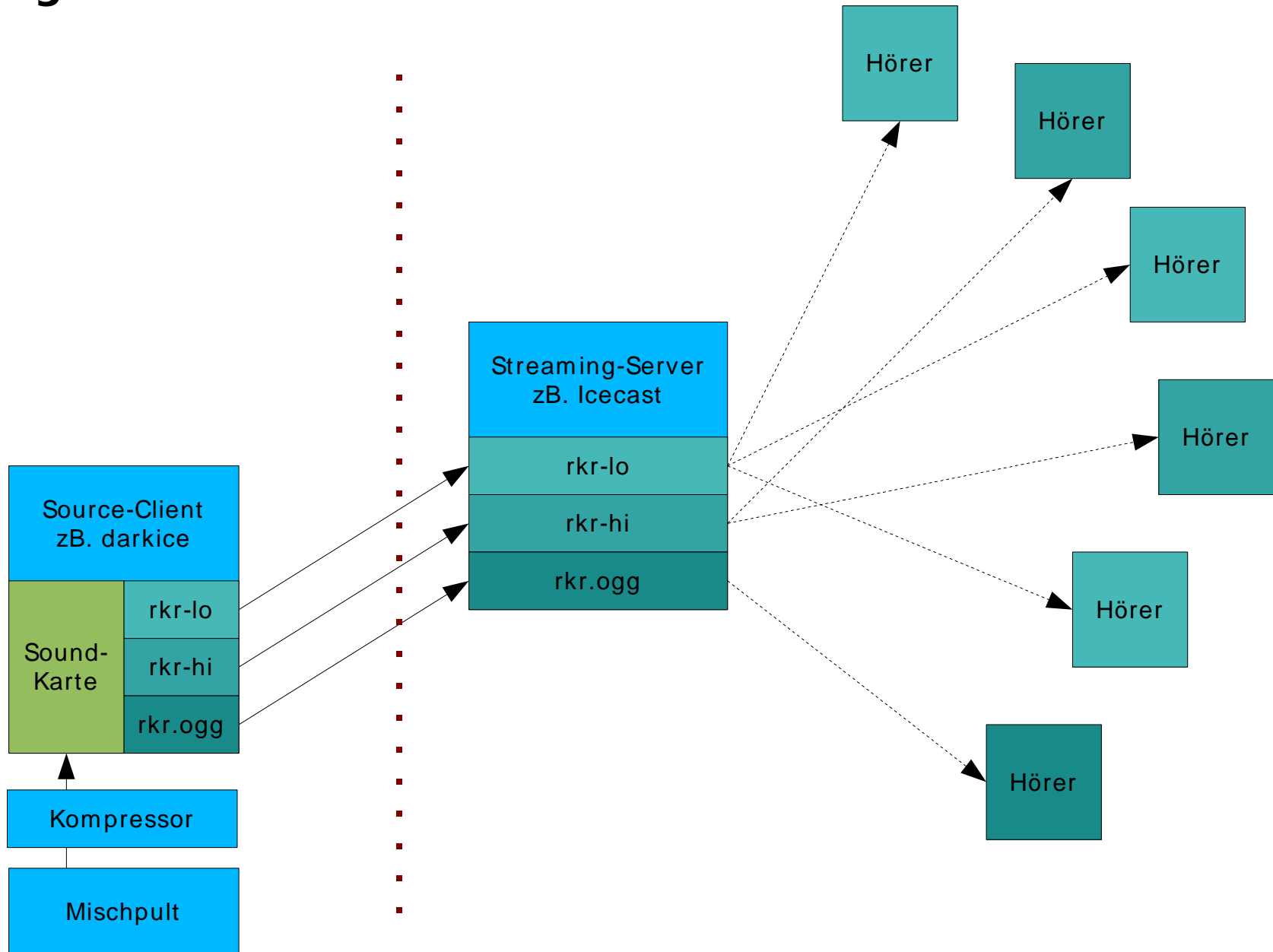
/usr/local/bin/checkendlos.sh
# Ueberpruefung ob Endlos laeuft
if (pidof mpg123) > 0
  then exit 0
  else /usr/local/bin/endlos.sh
fi
```

## Sendeautomatisierung

### Crontab (Auszug)

```
...
#### taeglich Kanal Ratte ####
# Endlos regelmässig prüfen:
2-59 0-2,15-17,21-23 * * * /usr/local/bin/checkendlos.sh
2-59 18-20 * * 0-5 /usr/local/bin/checkendlos.sh
#
#### taeglich Radio Dreyeckland ####
# Stream starten:
0 3 * * * /usr/local/bin/rdl-start-ansage.sh
# Stream regelmässig prüfen:
2,6,9,15,20,25,30,35,40,45,50,55,59 3-14 * * * /usr/local/bin/checkstream.sh
# Stream ab und zu neu starten (um Knackser nach langem Betrieb zu verhindern):
0 6,9,12 * * * /bin/killall ogg123
...
```

## Streaming



## Streaming

### **Livestream für Internethörer**

- **Server mit Soundkarte**
- **Source-Client als Encoder**
- **Streaming-Server mit guter Anbindung**

## Streaming

`/etc/icecast2/icecast.xml` (Auszug)

...

```
<mount>
  <mount-name>/rkr.ogg</mount-name>
  <password>SECRET</password>
  <max-listeners>2</max-listeners>
  <stream-name>Freies Radio Kanal Ratte</stream-name>
  <stream-description>Kanal Ratte, Freies Radio Schopfheim (104,5MHz)</stream-description>
  <stream-url>http://www.kanalrattefm.de/</stream-url>
  <genre>Various</genre>
  <public>1</public>
</mount>
```

...

## Streaming

`/etc/icecast2/icecast.xml` (Auszug)

```
...
[general]
duration          = 0          # duration of encoding, in seconds. 0 means forever
bufferSecs       = 5          # size of internal slip buffer, in seconds

[input]
device           = /dev/dsp    # hw:0,0
sampleRate       = 44100      # sample rate in Hz. try 11025, 22050 or 44100
bitsPerSample    = 16         # bits per sample. try 16
channel          = 2          # channels. 1 = mono, 2 = stereo

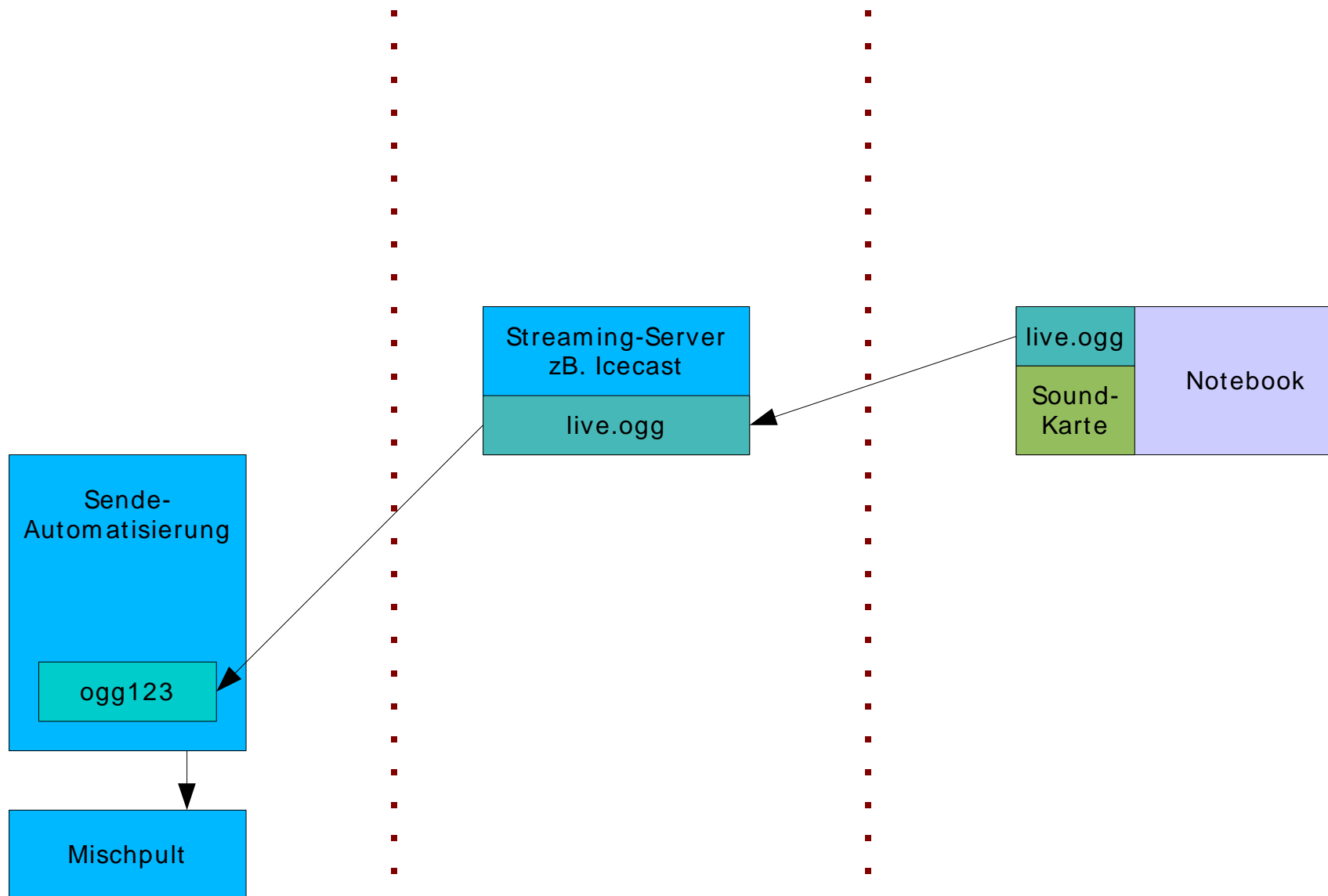
[icecast2-0]
bitrateMode      = vbr
format           = vorbis
quality          = 0.5
channel          = 2
server           = mail
port             = 8000
password         = SECRET
mountPoint       = rkr.ogg
name             = Freies Radio Kanal Ratte
description      = Livestream (Ogg Vorbis) von Kanal Ratte, Freies Radio Schopfheim (104,5MHz)
url              = http://www.kanalrattefm.de/
genre            = Various
public           = yes
...
```

## Streaming

### Livesendungen von unterwegs

- Notebook mit Soundkarte
- Source-Client als Encoder
- Internetanbindung vor Ort (mind. ISDN)
- Streaming-Server

## Streaming

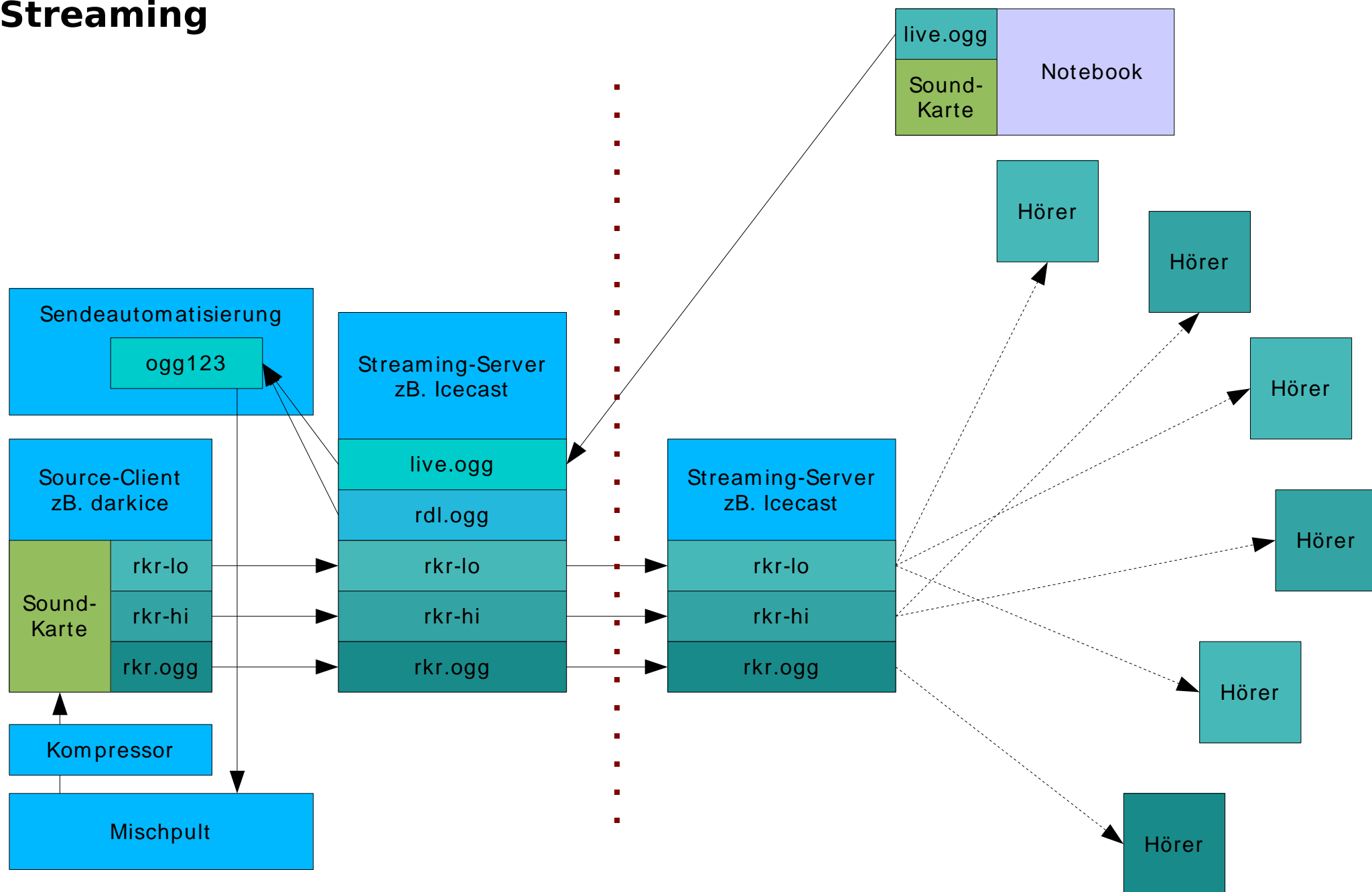


## Streaming

### Traffic sparen und weitere Tricks

- Relaying on Demand
- Burst on Connect (Puffer schneller füllen)
- Meta-Daten per Webinterface aktualisieren
- Fallback-Mountpoints

## Streaming



## Streaming

`/etc/icecast2/icecast.xml` (Auszug, Relayserver)

```
...
  <relay>
    <local-mount>/rkr.ogg</local-mount>
    <server>router.radio-kanalratte.de</server>
    <port>8000</port>
    <mount>/rkr.ogg</mount>
    <username>relay</username>
    <password>SECRET</password>
    <relay-shoutcast-metadata>1</relay-shoutcast-metadata>
    <on-demand>1</on-demand>
    <max-listeners>20</max-listeners>
    <public>1</public>
  </relay>
...
```

`/usr/local/bin/sub/meta-update.sh`

```
# Als Parameter übergebener String ($1) URL-tauglich machen
META=`echo $1 | sed -e 's/ /+/g' -e 's/&/%26/'`
```

```
# mit curl Webinterface mit Zugangsdaten aufrufen und String als Parameter übergeben
```

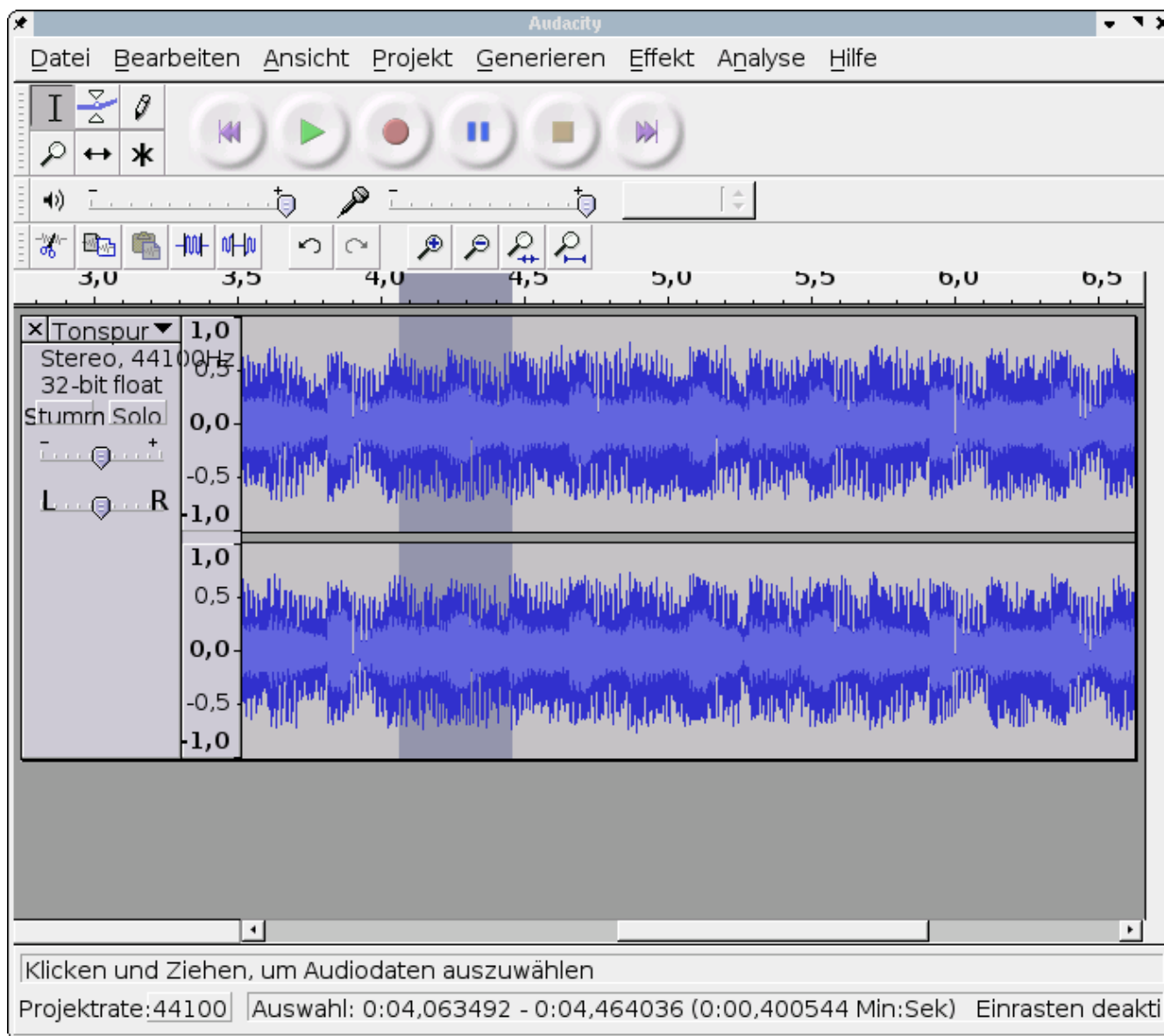
```
curl -fs -u U:PW "http://mail:8000/admin/metadata.xsl?song=$META&mount=%2Frkr.ogg&mode=upinfo" > /dev/null
curl -fs -u U:PW "http://mail:8000/admin/metadata.xsl?song=$META&mount=%2Frkr-lo&mode=upinfo" > /dev/null
curl -fs -u U:PW "http://mail:8000/admin/metadata.xsl?song=$META&mount=%2Frkr-hi&mode=upinfo" > /dev/null
```

## Anwenderprogramme

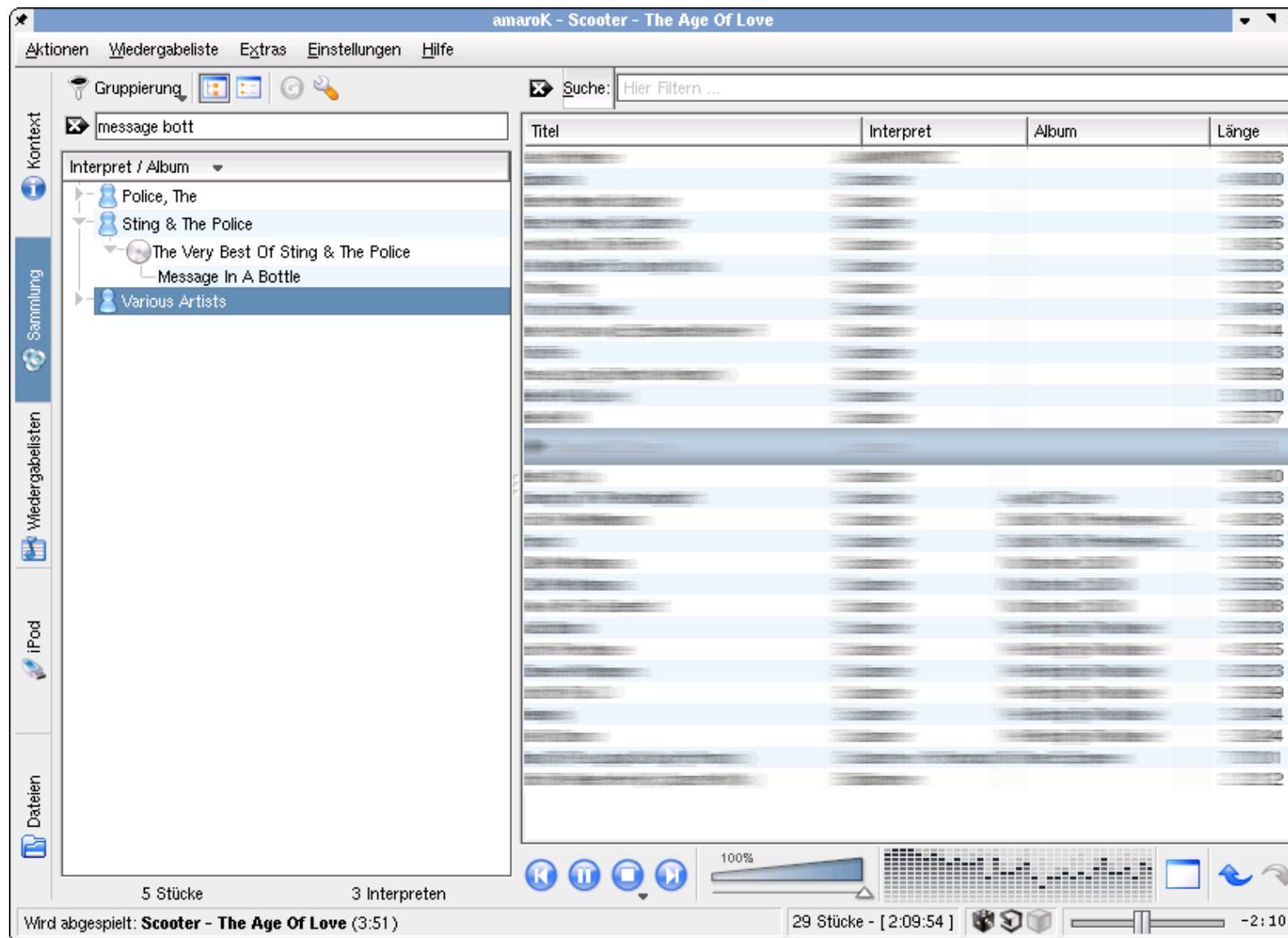
### Software

- |   |                  |
|---|------------------|
| - <b>Audio-Aufnahme und -Schnitt</b>    | <b>Audacity</b>  |
| - <b>Musik-Wiedergabe und -Sammlung</b> | <b>Amarok</b>    |
| - <b>Organisation und Kommunikation</b> | <b>MediaWiki</b> |

## Aufnahme, Schnitt und Musik



## Aufnahme, Schnitt und Musik



## Anwenderprogramme

### Benutzerverwaltung

- **MySQL-Datenbank**  
PAM-MySQL, NSS-MySQL
- **NFS-Server**

## Weitere Ideen und Ausblick

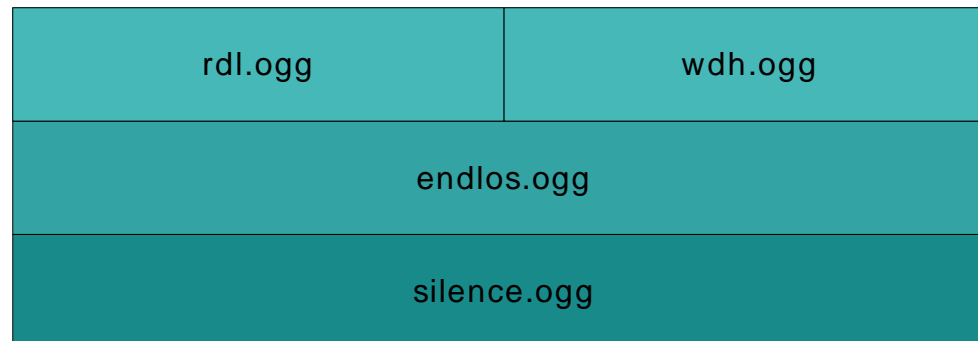
### **Sendeautomatisierung 1**

- **zentraler Perl-Daemon**
- **Steuerung durch MySQL-DB**
- **Overlay-Sendeplan**
- **webbasierte Verwaltung**

## Weitere Ideen und Ausblick

### Sendeautomatisierung 2

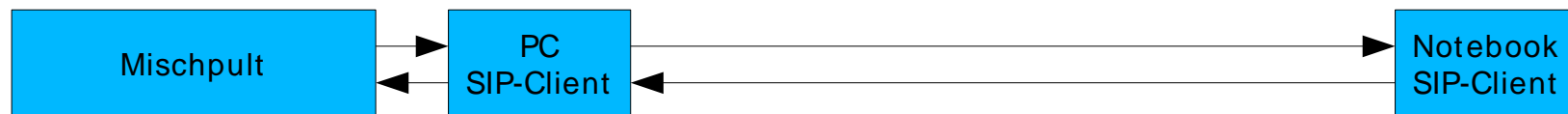
- alle Eingangskanäle als Streams
- Fallbacks nach Priorisierung



## Weitere Ideen und Ausblick

### Liveübertragung

- **Sendezuführung via SIP**
- **Ogg Speex-Codec oder Selberbau mit Ogg Vorbis**
- **minimale Latenz**
- **Duplex – Kommunikation on Air mit dem Studio**
- **Alternative: Eigenen Client entwickeln**



## verwendete Software und Informationsquellen

### **Sendearchivierung / -Automatisierung und Livestream**

- **bash**
- **alsa-Treiber / alsa-utils**
- **mpg123 / ogg123**
- **cron**
- **icecast**
- **darkice**
- **curl**

## verwendete Software und Informationsquellen

### Anwenderprogramme

- audacity
- Amarok
- KDE
- twinkle

## verwendete Software und Informationsquellen

### Benutzerverwaltung

- MySQL
- apache
- php
- pam\_mysql
- libnss\_mysql

## verwendete Software und Informationsquellen

### Weboberfläche

- **phpmyadmin**
- **mediawiki**
- **courier-mta**
- **pure-ftpd**
- **spamassassin**
- **clamassassin**
- **OTRS**

## Abschluss

**Vielen Dank für Ihre Aufmerksamkeit!**