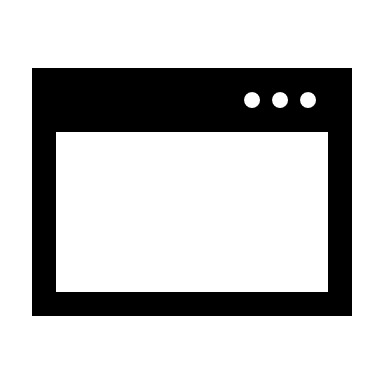
Det. Project

The goal of this project is to have data from a satellite signal receiver constantly analyzed. The result is displayed in a user friendly way which enables querying over all the results. The information extraction should be extensible due to the always changing requirements.

The following project plan/documentation shows an application consists of several microservices running on different targets. Those four core services ensure to achieve the stated goal as such as that it will also be scalable. The known data source provides a stream of audio & video data but the architecture will also allow usage of other sources. Data streams are analyzed and information is extracted in the following steps. The result of the analysis is stored and displayed properly.

Figure basic overview of the four core modules



**Data source**

**Info extraction**

**Info storage**

**Display**

# Data Source

Data source are X Sat Receiver.

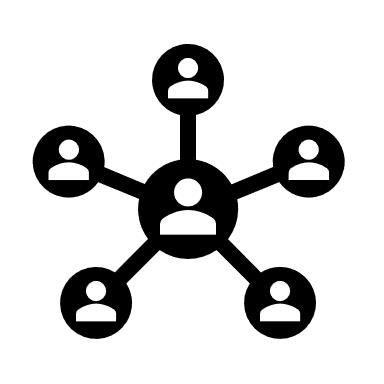
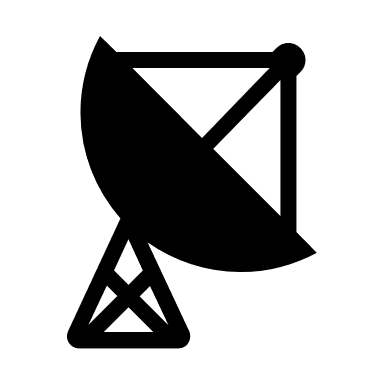
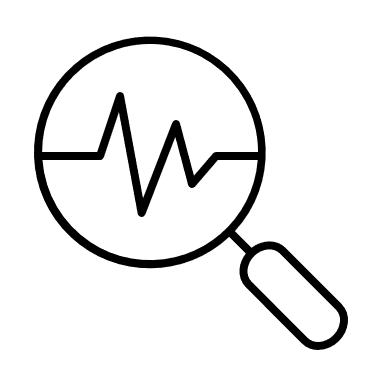
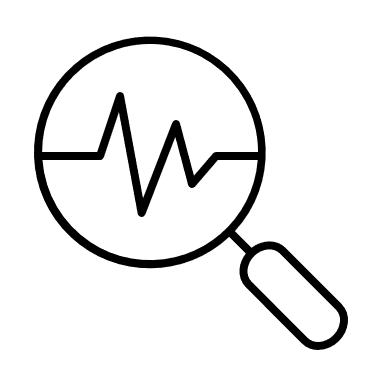
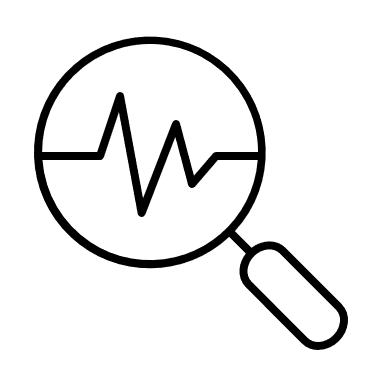
|  |  |  |
| --- | --- | --- |
| Name | Datasheet | Manual |
| Ericsson, Receiver 1 Matrix 9 Rx8200 | [Datasheet](file:///C:\Users\Daniel_Klauser\AppData\Roaming\Microsoft\Word\rx8200_BSbroadcast.pdf) | [Manual](file:///C:\Users\Daniel_Klauser\Desktop\Militär\971944.pdf) |
| Tandberg, Receiver TT 1260 | [Datasheet](file:///C:\Users\Daniel_Klauser\AppData\Roaming\Microsoft\Word\Tandberg-TT1260.pdf) | [Manual](file:///C:\Users\Daniel_Klauser\Desktop\Militär\TandbergTt1260UsersManual462521.843528215.pdf) |

They are able to provide data such as :

|  |  |  |
| --- | --- | --- |
| Format | Data | Device |
| MPEG-2 | Audio & Video | RX8200 |
| MPEG-4 | Audio & Video | RX8200 |
| MPEG-2 | Audio & Video | TT1260 |
|  |  |  |

Probably more, but I m not from the subject.

The Data source is connected to a/some device/s which contain services which receive and preprocesses the incoming data stream from the data source. If the signal is preprocessed the data packages are sent via pub sub message over TCP/IP.



**Data source**

**Data stream**

**Preprocessing**

**Publish**



Figure data-source schema

To clarify the task of the preprocessors :

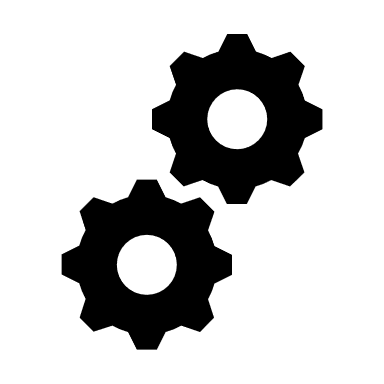
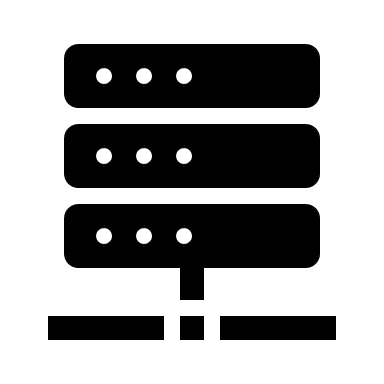
* Configurable via file
  + Keyword to publish
  + Internet Information
  + Output Data format
    - Ex. Video of 10 Sec.
    - Single Frame.
    - Audio Signal 5 min.
* Accessing data stream
* Split audio & video
* Block noise (avoid unnecessary calculation)
* Publishing configured output

The preprocessed data maybe has to be stored somewhere to be accessed by the info extraction. Therefore an intermediate database has to be used.

|  |  |
| --- | --- |
| Technology | Adv./Disadv. |
| Apache Kafka | Horizontal Skalierbar, Pub-Sub (DB and PubSub) |
| ActiveMQ | Pub-Sub with bigger package size |
| MQTT - Publish | Only lightweight messages, needs another db |

## Problem with data conversion

State of the art information extraction are up to now not able to process images and audio together. Therefore those must be separated. The separation is probably not possible in memory, therefore separation must be saved in files.



**MPEG**

**Video**

**Audio**

**preprocessor**

**ActiveMQ**

**Librosa // ffmpeg**

Figure 3 preprocessor workflow

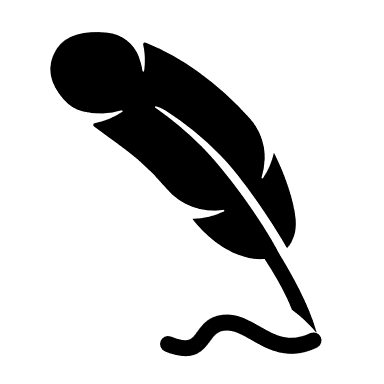
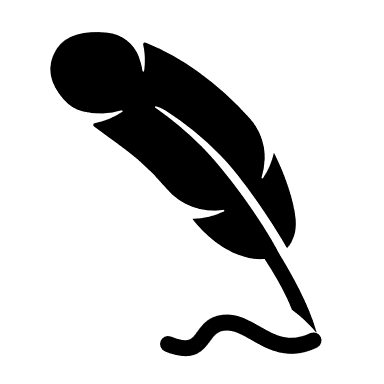
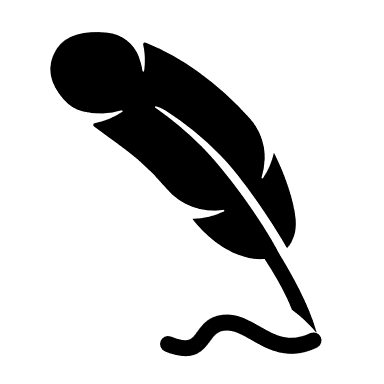
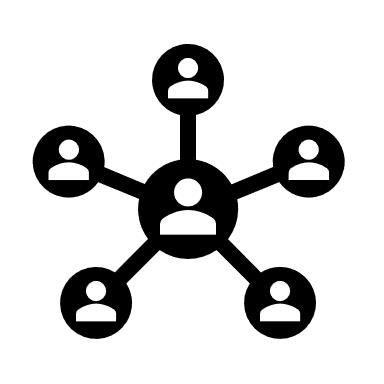
# Information Extraction

Firstly the information extractor handles data on subscription. The information extraction itself has the task to get useful features from received data.

Figure information extraction schema

**Subscribe**

**Info Extraction**



**Database write**

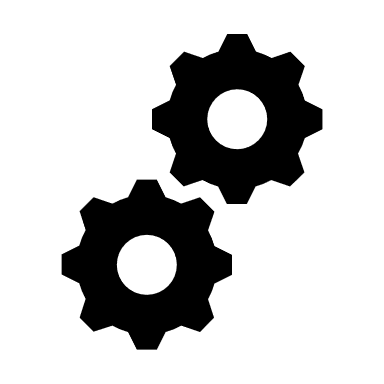


**Info Storage**

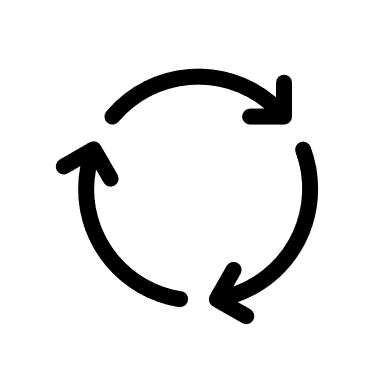
One can think of several possibilities how to extract information from audio & video data. Most recent technologies allow a thorough information extraction in exchange for computational cost. The computational cost is not be underestimated, this is why they should be encapsuled and receive the data via message provider.



**ActiveMQ**



**Collect/decode**



**inference**



**infoExtractor**

{

“name”:nameOfInfoExtractor,

“time” : unixTimeStamp,

“result” : resultInference,

“decodedData”: blob

“typeOfBlob”: videoAudioOrElse

}



**S3**

**Postgres**

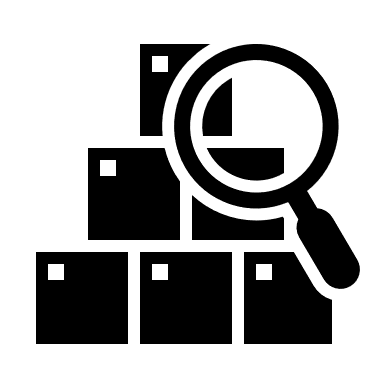
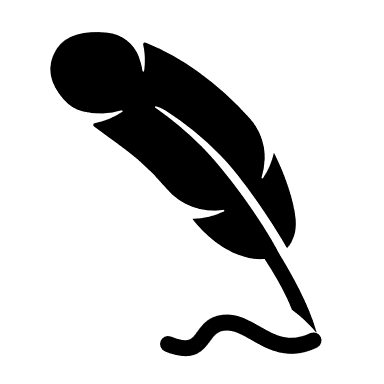
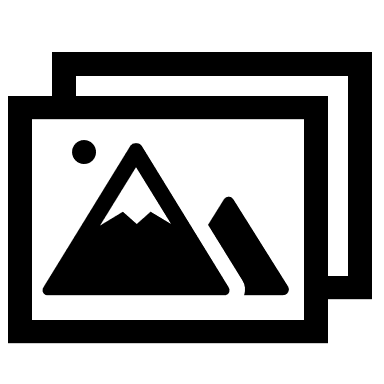
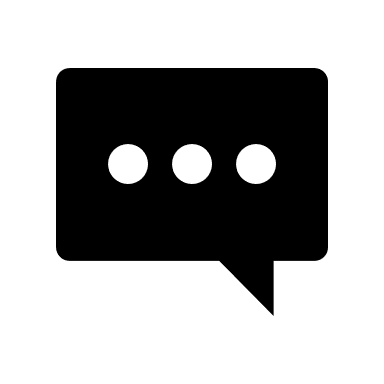
# Info Storage

The database requires storage for all kinds of data formats such as video, single frames or audio data. The asynchronous access should of course be provided. The time of creation and the creator of the data (info extractor) should always be part of the database entry.

Figure storage schema

**Info Storage**

**Read**



**Write**

|  |  |
| --- | --- |
| Technology | Adv./Disadv. |
| S3 Bucket | Rest API |
| Hadoop HDF5 Cluster | Massiv Skalierbar |
| PostgreSQL | Free, supports blob, expertise is given |
| Filesystem | Less initial overhead, permissions may block later |

# Display

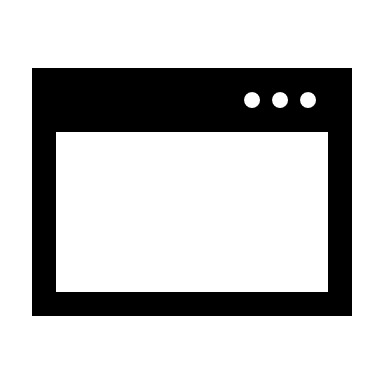
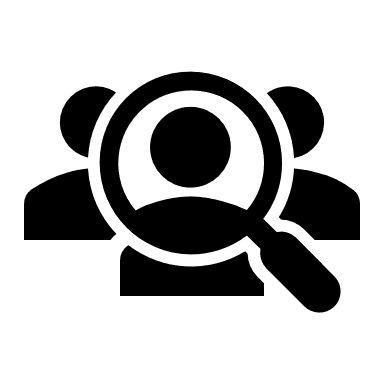
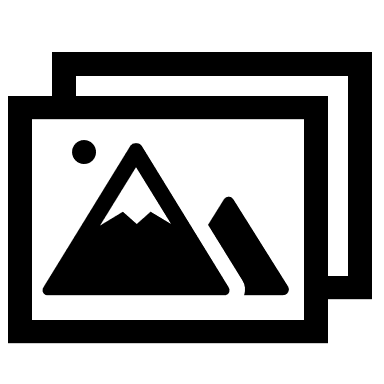
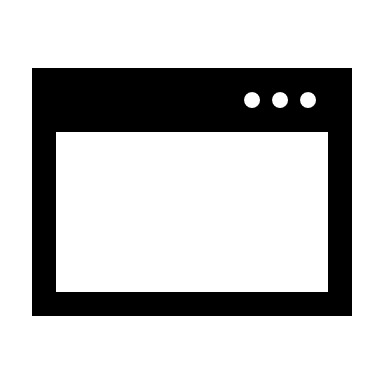
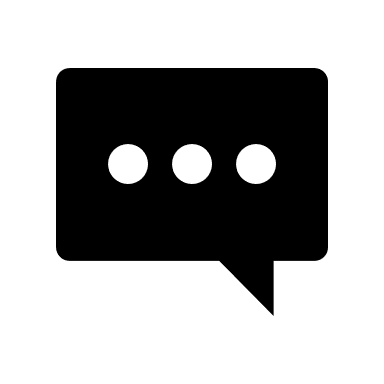
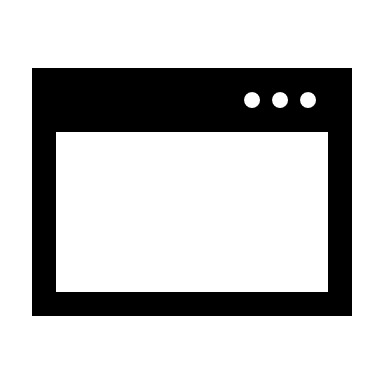
The front-end consists of a simple search function as such as a specific result displaying of the running info extractors. The search function should therefore address general info extractor such as audio-to-text or scene understanding information.

Figure user interface schema

**Info Storage**



**Display**



Of course the displayed audio or video should be played within the UI and can be downloaded as wished.

|  |  |
| --- | --- |
| Technology | Adv./Disadv. |
| Frontend : React | In use CNO |
| Backend : Spring Boot | Java |
| Frontend : Angular (similar to react) |  |
| Backend : node.js/express & flask | Easy integration in frontend, in use CNO |
|  |  |

# Work Packages

|  |  |  |  |
| --- | --- | --- | --- |
| WP # | Work | Est. Req. Time | Critical |
| 0 | Setup  ☐Create GIT  ☐Create 4 different projects (data source, info extraction, database api, display)  ☐Create Work packages | 1 Week |  |
| 1 | Data Source  ☐ensures grabbing of data from the receiver | 3 Weeks |  |
| 2 | Prototype  ☐ensure base classes of all modules  ☐ensure unit tests (&mocks)  ☐ensure documentation setup  ☐ensure linter setup  ☐ensure connection of services  ☐ensure pub sub works with large data – otherwise another data storage is needed as pub sub reference. (take stored data from WP1 or from internet)  ☐ | 3 Weeks |  |
| 3 | Data Source preprocessor  ☐implements configurable grabber from WP 1  ☐implements data conveyor & splitter  ☐publish data as stated in WP2  ☐implement noise blocker (not prio) | 2 Weeks |  |
| 4 | Database setup  ☐select database  ☐setup Blob database  ☐create database “store” interface infoExtraction side (infoExtractorName, time, result, analyzed data (audio video whatever - blob))  ☐plans UI | 2 Weeks |  |
| 5 | Info Extraction  ☐use basic network to fill database  ☐create config for dummy audio analysis  ☐create config for dummy video analysis  ☐create config for dummy frame analysis | 2 Weeks |  |
| 6 | Display  ☐creates a minimum UI  ☐searchbar  ☐video & audio player & display image  ☐implements simple queries on db | 2 Weeks |  |