What we have and what can be done.

# What we have

The system is based on a three-layer-architecture. The modularity should provide a high level of adaptability.

## Connection to App

The system receives POST requests from the app, sends the request to the database and sends a POST response to the app. For more info see JavaDoc.

## Database

* Delay, Jitter and Loss- data, Interfaces and services
  + is being cached
  + Initialization for x Days in the past is possible
  + We can delete old data
  + Chunking is available
  + Multithread capable
  + Many tasks are multithreaded for better performance
* MongoDB

## Fetch data

So far the following things are queried via SOAP:

* Delay, Jitter, Loss data from HADES
* Throughput data from BWCTL
* Utilization data from RRD

The responses are parsed with JDOM.

More detailed information can be found in the JavaDoc.

# What can be done

## Connection to App

* Currently, the request and response mechanism is fully implemented for Delay Jitter Loss only
* For other requests, dummy-responses are sent
* More powerful (but bigger/slower/not as easy to use) JSON parser? Currently [minimal-json](https://github.com/ralfstx/minimal-json) by Ralf Sternberg is used

## Database

* Auto update (getting data for a not complete chunk)
* Implementing other features like Throughput data, Utilization etc.
* Deletion isn’t thread safe
* Evaluate data

## Fetch data

1. There is no system how requests are handled. This could be problematic with an increasing number of requests from multiple users. A simple solution would be a queue of requests.
2. Currently, interfaces for utilization is divided into input and output, and returned separately. These could be grouped into pairs.