REPORT REVIEWING MOPIDY SOFTWARE ARCHITECTURE

- Nidhi S Tiwari

- Mopidy is an extensible, hackable music server which allows users to connect their choice of web client to the multiple cloud services like Spotify, SoundCloud and Google Play Music. The users can add new music easily and control the music from any phone, tablet, or computer operating on any platform from android, linux and iOS.

- There are more than 10 extensions for Mopidy. With the help of extensions, music from cloud service, like Spotify, SoundCloud, and Google Play Music can be easily added. Through Vanilla Mopidy, music from your local disk and radio streams could also be enjoyed.

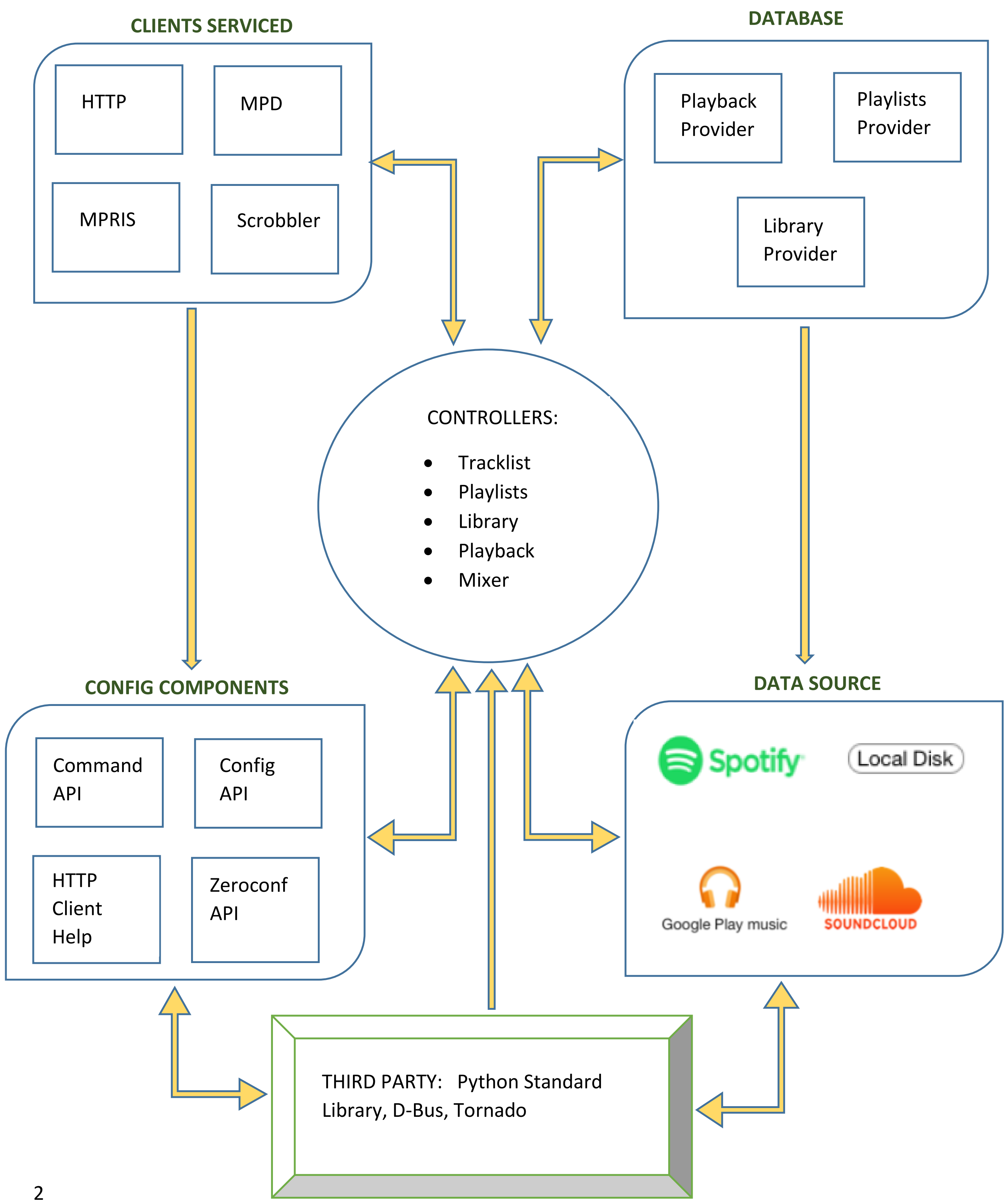
- Mopidy is a server written in Python and has network connectivity, audio output and supports multiple operating systems and devices like Linux, Mac and it's most interesting feature is that it is hackable and very convenient for creating your own hacks. In one project, a Raspberry Pi was embedded in an old cassette player.

- Additional flexibility of mopidy is through its ability of combine multiple frontends and backends (through a core) and also facilitate use of multiple web clients like MPD (Music Player Daemon) and HTTP. Additional frontends for controlling Mopidy can be installed from extensions.

- Mopidy is a completely open source project and GitHub is the main platform in which the Mopidy is developed. Like any project hosted in GitHub, the developers come from everywhere as long as they are interested in Mopidy and are willing to contribute.

- As an open source project, users can easily download the source code and build the program from GitHub. Not only personal users, there also some software, such as Parity, use the code from Mopidy project with their own UI.

MODULE ORGANISATION:



Description of the modules as shown in the above diagram:

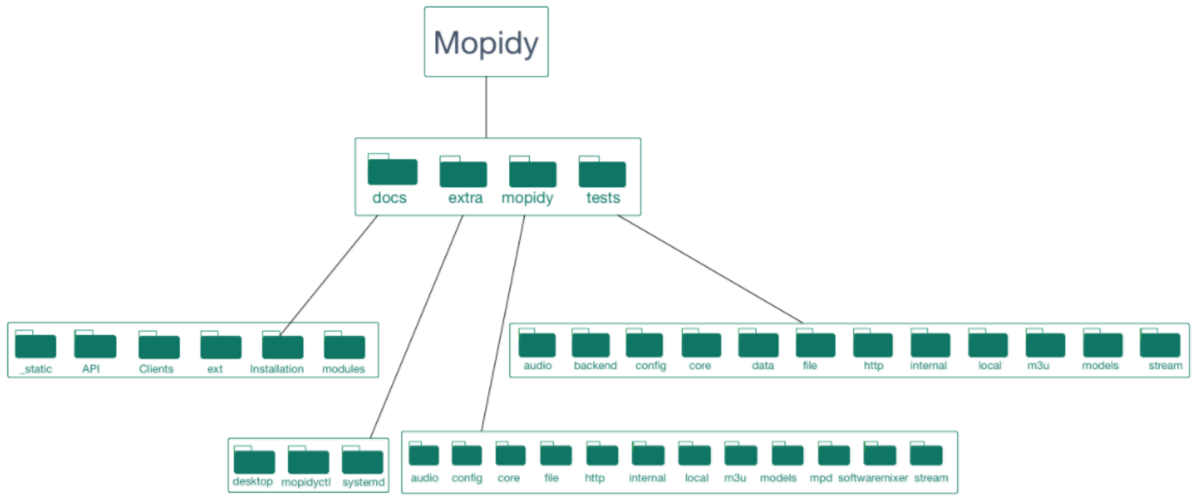
- Client: These are the web clients directly providing the resource for user and can implement server for protocols such as HTTP, MPD as well as MPRIS.
- Controllers: The domain layer contains several controllers for different functionalities like keeping track of the tracklist and etc.
- Database & data source: These are a set of providers having different functionalities to provide various music sources.
- Config components: The utility layer contains many modules used by other layers.
- Third party: The third party layer has several platforms which are imported to achieve a set of functionalities.
- Core: The core controls the playlists, tracklists, library. It is lightweight and also links the front-end and the backend.

The above depicted modules depend on each other and the arrows depict that dependency of these modules is facilitated.

- Design Model:
  - Design constraints that apply when designing the system's software elements helps increase the system's overall technical coherence and makes it easier to understand, operate, and maintain. It can reduce risk and duplication of effort by identifying standard approaches to be used when solving certain types of problems as well.
- Configuration:
  - You can customize the volumes, separate the bass form other volume components. The default mixer runs along with the audio module.
  - For the initialization, Mopidy has a lot of config values you can tweak, but you only need to change a few to get up and running.
  - When you have created the configuration file, open it in a text editor, and add the config values you want to change.
  - For termination and restart of operation you type in commands to the server terminal
- Standardised Testing:
  - Mopidy has standard test cases for each module in their framework. There is a separate subdirectory called tests where there are folders containing unit tests for each module, backend and front end as well as web clients like HTTP and MPD.
  - Developers could see detailed test in this file and configure additional modules that they write with parallel testing accordingly.

Source Code organization:

Since it is an open source project, it is important to keep the code structure so that other developers can easily



- Under the root directory, Mopidy is divided into four subdirectories, docs, extra, mopidy and tests respectively.
- The docs folder contains manuals and other important information. The manuals are also separated to different folders according to their content.
- The mopidy subdirectory contains the source code for the actual implementation of the Mopidy server.
- Mopidy is well organized by dividing its functionalities into different modules. The structure of the mopidy subdirectory is also based on the modulation of the whole system's design. Each module has its own directory to store its source code. This makes it easier for developers to analyse each module.

Conclusion

- I had some trouble understanding all the jargon used while perusing through the Mopidy documentation and wanted to break it down and simplify it for the purposes of this report.
- Mopidy is a server that communicates with multiple backends, simultaneously, to provide a combined and streamlined music playing service. It can also communicate with multiple frontends like Spotify, Soundcloud, Google Cloud to play music collected from all these sources in a single place. It is a one stop place where you can curate your playlists from different sources into a single place. It's audio convenience bundled with a developer's fantasy. However, it does come with its fair share of bugs.
- The number of functionalities and customizations involved and its fairly lightweight size shows us its efficiency and optimality. This and its organization is by far its most impressive feature.

-------- END --------