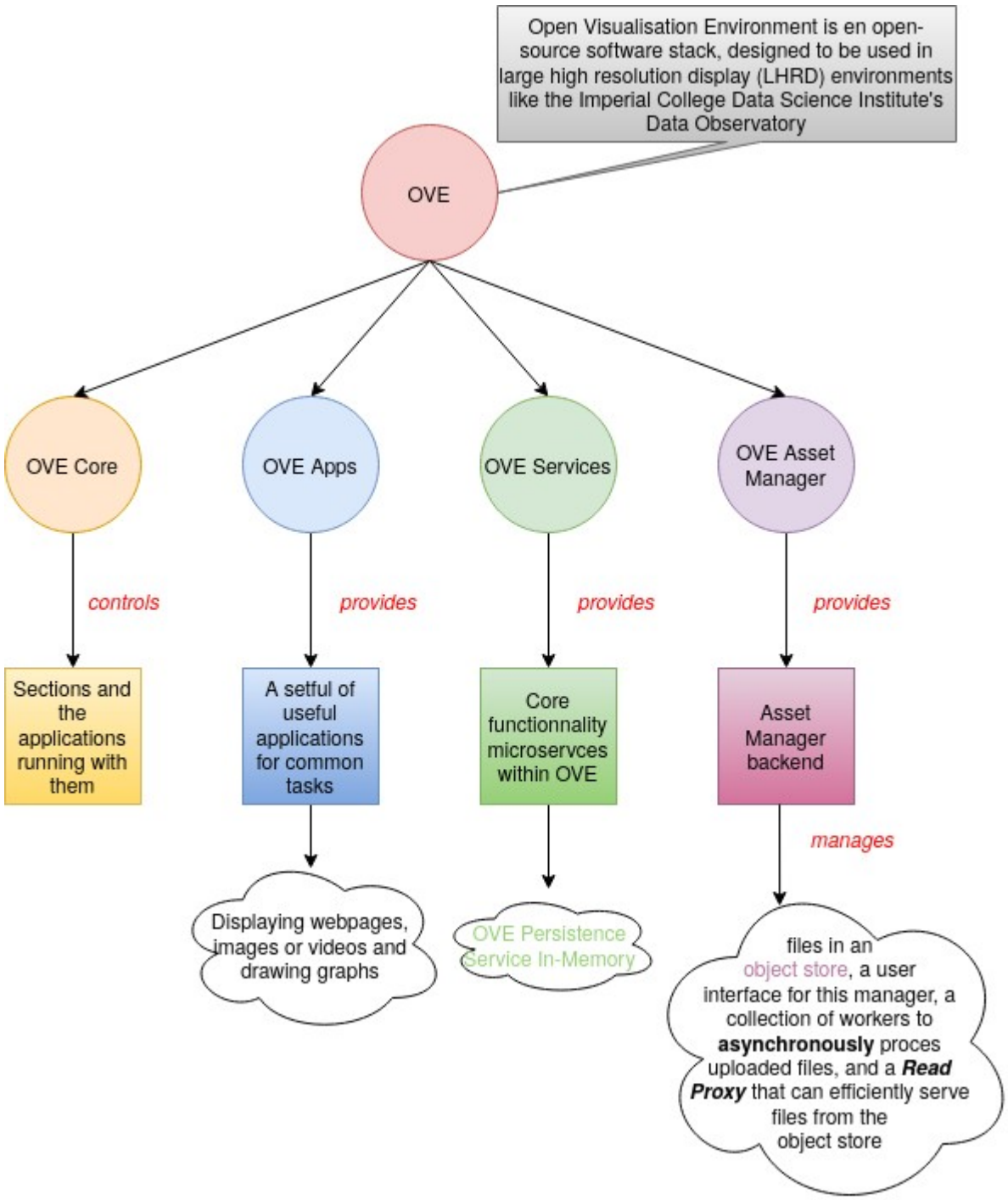
Notes about OVE

https://github.com/ove/

Open Visualisation Environment (OVE) is an open-source software stack, designed to be used in large high resolution display (LHRD) environments like the Imperial College Data Science Institute's Data Observatory.

Uses an implementing a microservices architecture that allows the distributed execution of applications using web technologies



OVE Core : controls sections and the applications running within them

OVE Apps : provides a set of useful applications for common tasks such as displaying webpages, images or videos and drawing graphs

OVE Services : provides core functionality microservices within OVE such as the Persistence Service, which is used to compute absolute positions in pixels from positions expressed relative to a grid or as a percentage of the total space size.

OVE Persistence Service – In-Memory : This service provides persistence for the OVE framework using an in-memory storage implementation. This is also the most straightforward persistence service implementation and is not highly available.

OVE Asset Manager : provides an Asset Manager backend that manages files in an object store, a user interface for this manager, a collection of workers to asynchronously process uploaded files (e.g., to create tiles from an image, or expand a zipped archived), and a Read Proxy that can efficiently serve files from the object store.

Components :

- The **Backend** exposes a full RESTful API for operations on projects, assets and files. This service can work independently from all the other services.
- The **Read proxy** exposes an HTTP endpoint to read individual files off the object store. The AM Read Proxy is authenticated by the same rules as the Backend service.
- The **UI** exposes a User Interface for most of the operations performed by the Backend service.

Workers can be scheduled to asynchronously perform a task performed on a file, such as converting it to a new file format.

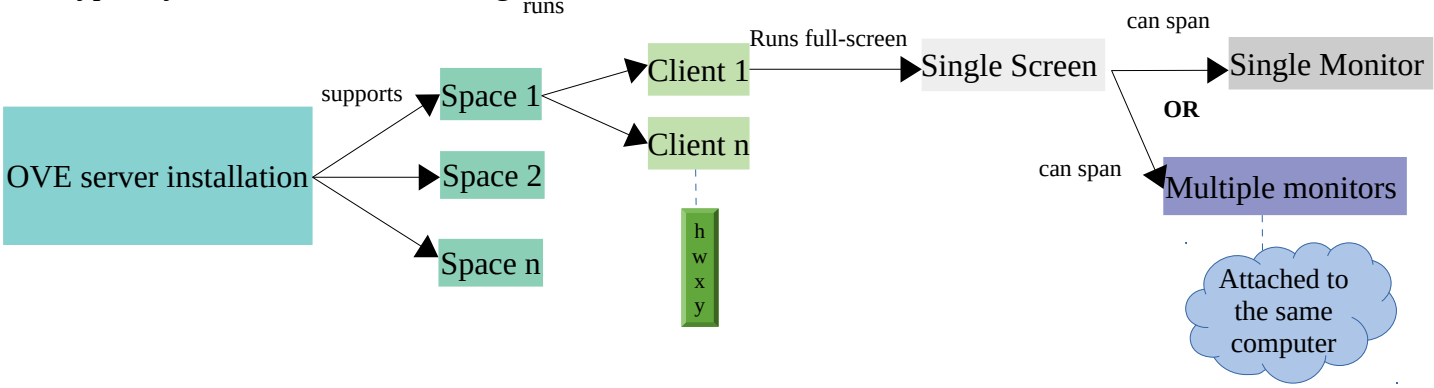
An **Asset** is a collection of one or more **files** that can be treated as a single unit and versioned, processed and displayed together. Each asset has associated metadata (e.g. processing state, name, description, tags, etc.).

OVE also provides user interfaces and software development kits that can be used to design and develop projects.

OVE also provides user interfaces and software development kits that can be used to design and develop projects.

A `space` is a collection of monitors, which may be attached to different computers, that together form a single display. OVE is designed to be used in Large High Resolution Display environments; but, it is also suitable for use on much smaller displays with a single or a few monitors.

In each space, OVE runs within a number of `clients`. An OVE `client` is a browser window and typically runs full-screen on a single `screen`.



A screen can span a single monitor or can span multiple monitors that are attached to the same computer.

The Spaces.json file is used to define the spaces of OVE and describe the arrangement of clients within each space (these terms are explained in the Basic Concepts page).

The order in which the `clients` are defined determines the `oveViewID` of each, and thus the URL that should be opened in each browser window.

The default `Spaces.json` file defines two spaces (`LocalFour` and `LocalNine`), corresponding to `2x2` and `3x3` arrangements of `clients`. Each `client` in these spaces has a resolution of `1440x808`, so they have a total resolution of `2880x1616` and `4320x2424` respectively.

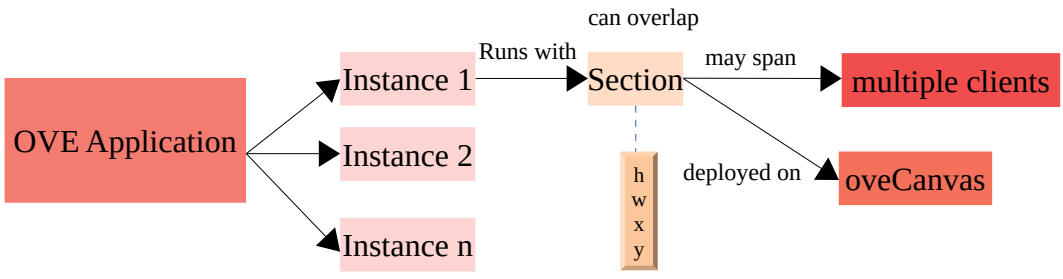
The OVE object : The OVE object (`window.ove`) provides a number of useful functions and data structures to handle state, to interpret geometry and to communicate via WebSockets. It also provides a context (`window.ove.context`) to hold the application's local variables.

OVE application → Client-Server architecture

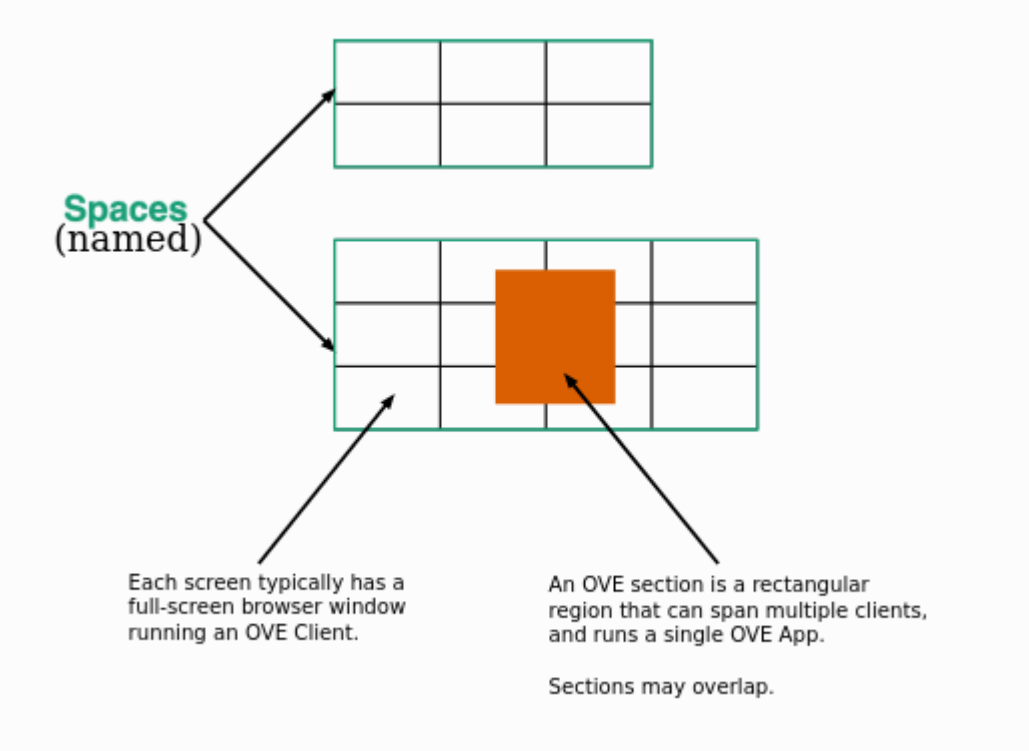
Application structure

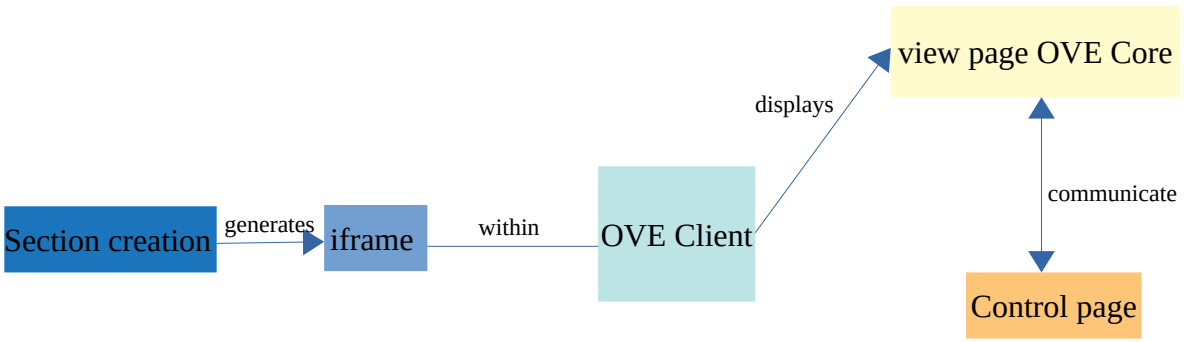
An application typically has the structure:

```
├── README.md
├── package.json
└── src
    ├── client
    │   ├── common
    │   │   ├── <app-name>.css
    │   │   └── <app-name>.js
    │   ├── constants
    │   │   └── <app-name>.js
    │   ├── control
    │   │   ├── <app-name>.css
    │   │   └── <app-name>.js
    │   ├── index.html
    │   └── view
    │       ├── <app-name>.css
    │       └── <app-name>.js
    ├── config.json
    ├── data
    ├── swagger-extensions.yaml
    └── index.js
```



An `oveCanvas` can also have `groups` of one or more `sections` and `groups`.





OVE core accepts registration of **peer nodes** using the http://OVE_CORE_HOST:PORT/peers API method. Once registered OVE peers will cross-post messages that are broadcasted using WebSockets. Peer nodes of OVE serve two purposes, ***high availability*** (HA) or ***distribution of workload***.

OVE Applications

- Alignment : helps align the monitors in an OVE installation
- Audio : supports the playing of audio files within the OVE Framework
- Charts : supports visualisation of charts using the OVE framework
- Controller : a unified controller for all OVE apps (supports common pan and zoom operations)
- HTML : supports displaying HTML web pages using the OVE framework
- Images : supports displaying HTML web pages using the OVE framework
- Maps : supports visualisation of dynamic maps using the OVE framework

- Networks : supports visualisation of networks with node-link diagrams using the OVE framework
- PDF : supports displaying PDF documents using the OVE framework
- QR Code : displays a QR code using the OVE framework.
- Replicator : replicating content of an OVE installation
- SVG : supports rendering SVG using the OVE framework
- Videos : supports playing videos using the OVE framework
- WebRTC : supports videoconferencing and screen sharing using the OVE framework
- Whiteboard : creates a whiteboard that can be used within the OVE framework

Note that the launcher can only be used to create sections that lie entirely within a space: if your chosen goemetry does not lie entirely within a space, you will be shown a form validation error.