Configuration files

Patterns

All the configuration files on this IOC project follows JSON data representation pattern. On this document is described all 3 different configuration file patterns found on this project.

Rooms configuration file pattern

The purpose of this file is to map the location of the switches being monitored by the IOC to the Sirius room where each one is located

- Rooms register (Register that wraps all other registers -> Pattern start)

  ```
  {
          “rooms”: [<List of room objects>]
  }
  ```

- Room object

  ```
  {
          “id”:<room ID>,
          “name”:<room Name>,
          “switches”: [<List of switch objects>]
  }
  ```

- Switch object

  ```
  {
          “ip”:<switch IP>
  }
  ```

Switches configuration file pattern for the IOC with REST API login authentication

This file has the purpose of mapping all ports to be monitored on each switch, the PV name for each device that is being monitored and the information for connecting and login in each switch.

- Switches register (Register that wraps all other registers -> Pattern start)

```
{
        “switches”: [<List of switch objects>]
}
```

- Switch object

```
{
        “login_data”:
        {
                “ip”:<switch IP>,
                “port”:<service port>,
                “username”:<switch authentication username>
                “password”:<switch authentication password>
        },
        {
        “devices”:[<List of device objects>]
        }
}
```

- Device object

```
{
        “name”:<device name/part of PV name respective to device name >
        “port”:<switch port which device is attached>
}
```

Switches configuration file pattern for the IOC without REST API login authentication

The configuration file for either the IOC code that performs the REST API login authentication and the one that doesn’t are almost the same. The difference between the patterns is the absence of the “username” and “password” elements from the “devices” section of each “switch” register.

- Switches register (Register that wraps all other registers -> Pattern start)

```
{
        “switches”: [<List of switch objects>]
}
```

- Switch object

```
{
        “login_data”:
        {
                “ip”: <switch IP>,
                “port”:<service port>,
        },
        {
        “devices”:[<List of device objects>]
        }
}
```

- Device object

```
{
        “name”:<device name/part of PV name respective to device name >
        “port”:<switch port which device is attached>
}
```

Updating files

The procedure to update the configuration files from this project is exactly the same as inserting manually new data into a JSON register.

Considering that the actual IOC purpose is monitoring power at ports from Sirius network infrastructure switches, and that this infrastructure is being assembled and has it’s own architecture pattern it’s really important to know which files should be updated on certain situations.

The first case is when devices are connected or disconnected from switches. In this case the file “switches.config” shall be updated, so it’s possible to start/stop monitoring the PoE status of that device. A second case is when any switch is connected on the infrastructure. In this case the both files, “rooms.config” and “switches.config”, might be updated. The “rooms.config” file is updated for mapping in which Sirius room the switch can be found and the “switches.config” is updated for making possible to the IOC to connect to it. The procedure for removing a switch from the infrastructure is the same for adding a new one, but instead of adding à new record to the file you should remove the record referring to the switch.

	Connecting Switch	Removing Switch	Connecting Device	Disconnecting Device
switches.config	X	X	X	X
rooms.config	X	X		

Table1 : Operation X Files to be updated

It is very important to mention that the file ”switches.config” has 3 copies inside all the directories from the complete poemonitor project. This is because either the IOC and the GUI are standalone versions, so this replicated files make possible to run any of this applications without cloning all the git repository. Considering that, when updating the “switches.config” file, all 3 files should be updated to ensure that all applications are up-to-date.