The Journal of Open Source Software

Autosumit GUI: A ReactJS based visual interface to monitor experiments in a High Performance Computing environment

Wilmer V. Uruchi Ticona[*1], **Miguel Castrillo**[1], **and Daniel Beltrán**[1]

1 Barcelona Supercomputing Center **2** Institution Name **3** Independent Researcher

DOI: DOIunavailable

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Reviewers:

- @Pending Reviewers

Submitted: N/A
Published: N/A

Summary

`Autosubmit GUI` is a front-end software developed using *Javascript* and *ReactJS* that aims to show complex information from the execution of the workflow of experiments (managed by `Aut osubmit`) in a *High Performance Computing (HPC)*. `Autosubmit` is a Python-based workflow manager that handles complex tasks involving different sub steps (e.g. scientific computational experiments) which ma be executed in one or different computing systems (platforms), from *High Performance Computers* to small clusters or workstations. This workflow manager is able to orchestrate the tasks (jobs) that constitute the workflow while respecting their dependencies and handling errors.

This front-end software consumes information served by an API (`Autosubmit API`) that collects information from the execution of the workflow of experiments. An experiment consists of jobs that are executed on the selected platform following a established sequence. The execution of an experiment and its jobs generates a high amount of information that needs to be visualized later by users. `Autosubmit API` summarizes this information and presents it as API requests. `Autosubmit GUI` consumes these API requests and shows the information available in a condensed, comprehensive, and dinamic way. `Autosubmit GUI` uses three highly popular and useful libraries: `FancyTree`, `vis.js`, and `react-google-charts`; among other web resources that facilitate the visualization of information and information updates.

Statement of need

The number of jobs in an experiment workflow (managed by `Autosubmit`) ranges from one job to several thousands, where the jobs can vary in the kind of task they perform, from data retrieval to complex climate simulation. The result of the execution of an experiment workflow is usually large amounts of information scattered in a file system. Originally, this information was accessed through the file system or by using terminal commands (provided by `Autosubmit`) to get the information as a `text`, `pdf` file, or terminal output. The *Computational Earth Science* team decided that a better way to access this information without risking unreliability nor loss of performance was needed. A web application was chosen as the solution architecture that would enable us to deploy this service easily and with minimum requirements for our users. Furthermore, this choice would allow developers to reuse existing technology, in the form of open-source visualization libraries, to allow `Autosubmit GUI` to show information in a graphical way. The software has been designed and is developed following a efficiency first premise, since the amount of information we need to show is considerable and we aim to maintain minor loading times.

The main experiment workflow representation generated by the users of `Autosubmit` was a graph generated through graphviz in a `pdf` file. Consequently, that was the first representation we tried to mimic in `Autosubmit GUI`. For this purpose we chose `vis.js`, a proven javascript

*Custom footnotes for e.g. denoting who the corresponding author is can be included like this.

library that provides graph representations for the provided data. This library also provides a rich API through which we can manipulate it and dinamically update it. In our implementation we present several ways to access the `vis.js` objects and adapt them to our necessities in concordancen with the requirements of *ReactJS*, we avoid unnecessary refreshes of the graph component by storing it in the React internal state. We believe that this can serve as a guide to those willing to work with graph representations in a web environment.

Later, `Autosubmit GUI` was required to implement an experiment workflow representation that showed more information at first glance, and that arranged the jobs in a hierarchical way. As a result, we decided to implement a tree view based on `FancyTree`, a Javascript library. This library provides a rich API that allowed us to implement a structured visualization of the experiment workflow. Again, we adapted this library to work smoothly with *ReactJS* and that is updated efficiently through the internal component infrastructure. It is also a good example on how to use the tools provided by this library. Moreover, statistics are shown using `react-google-charts`.

Finally, a word on our programming language and framework choice. *Javascript* was the obvious, and perhaps required, choice in this respect. This programming language has taken over the web and is becoming an standard, for better or worse. For our front-end purpose, it has all the tools that we needed and gives us access to a variety of useful extensiones. The main vision of `Autosubmit GUI` was to developed a graphical interface that showed as much information as possible at first glance; moreover, this interface should be scalable, with more requirements of information taking an area of the browser window. It was decided that the interface should be divided in modules that would be displayed in a tiled design. The interface should be able to update the modules (tiles) individually and independently without affecting general performance. *ReactJS* provides the tools to accomplish this purpose, where we can translate our idea of modules to components. So, it was the chosen framework.

Main features

Main features.

Citations

If you want to cite a software repository URL (e.g. something on GitHub without a preferred citation) then you can do it with the example BibTeX entry below for Smith (2020).

For a quick reference, the following citation commands can be used.

Acknowledgements

We acknowledge contributions from Francisco Doblas-Reyes, Kim Serradell and all our Earth Science users.

References

No referencess

Smith, A. (2020). Fidgit: An ungodly union of GitHub and figshare. In *GitHub repository*. GitHub. https://github.com/arfon/fidgit