REPORT ON OpenDreamKit DELIVERABLE D4.15

Exploratory support for live notebook collaboration

BENJAMIN RAGAN-KELLEY, VIDAR TONAAS FAUSKE

Due on	31/08/2018 (M36)
Delivered on	31/08/2018
Lead	Simula Research Laboratory (Simula)
Progress on and finalization of this deliverable has been tracked publicly at: `https://github.com/OpenDreamKit/OpenDreamKit/issues/89`	

CONTENTS

1. INTRODUCTION

The Jupyter Notebook is a web application that enables the creation and sharing of executable documents containing live code, equations, visualizations and explanatory text. One of the main uses of Jupyter is as an interactive computing environment, building interactive notebook documents (notebooks for short). **T4.2** is focused on improving the process of collaborating on Jupyter notebooks in the various ways researchers and educators use them. D4.6 improved collaborating on notebooks via traditional version-control systems.

This deliverable aims to improve the process of "live collaboration" on notebooks, where two authors on different computers are editing the same notebook at the same time, and they are kept in sync over the Internet.

There have been prior efforts to enable live collaboration on notebooks, notably Google Colaboratory and CoCalc (formerly SageMathCloud). These efforts add live collaboration on notebooks to an existing, fully hosted cloud service (COCALC can also be downloaded and run as free, open source software, but not integrated into existing hosted notebook servers).

Our goal is to learn from these examples and build, in collaboration with the Jupyter community, an official live collaboration implementation into the JupyterLab notebook application, so that it can be available to all Jupyter users.

The target usage scenario for this effort is to enable live collaboration on notebooks whenever two or more users have access to the same hosted notebook server, e.g. hosted with JupyterHub. Example applications include researchers co-authoring an analysis, or an instructor helping a student through examples. This should require no additional infrastructure, and not rely on any commercial hosted service. In particular, it should be able to be used anywhere that Jupyter is already used, without opting into other systems. This last point will be the primary differentiator for the official Jupyter implementation, as opposed to those currently existing in hosted services. We also aim to keep the additional runtime dependencies for the server component minimal: a Python runtime, already needed to run the Jupyter notebook implementation, and a nodejs

runtime, already needed to run certain aspects of the JupyterLab application. No databases or external services will be required.

Existing real-time collaboration implementations have had to heavily modify or entirely re-implement the Jupyter notebook client application in order to achieve real-time collaboration. This can lead to a maintenance burden, attempting to keep modifications working with updates to the Jupyter software as the two projects diverge. Delivering a working real-time collaboration implementation in JupyterLab itself should allow projects similar to (or possibly including) CoCalc to have real-time collaboration with less effort, even if they choose to keep their own implementation. The basis of both the server- and client-side should be reusable both inside and outside JupyteLab, enabling real-time collaboration on documents other than notebooks and outside the Jupyter ecosystem.

While the focus of this implementation is to sync two users with access to the same notebook server, decoupling the real-time server from the notebook server could allow collaboration on documents shared across implementations.

2. Prototyping

The Jupyter Community has developed a first prototype of live collaboration in JupyterLab, prior to OpenDreamKit involvement. This implementation relied on the Google real-time documents API and storing notebook documents in Google Drive. By relying on an existing implementation of a real-time server and storage, development could focus on making JupyterLab's internal models and APIs suitable for real-time collaboration. In 2017, Google deprecated this API and it can no longer be used, but the lessons learned from this prototype have been useful in guiding JupyterLab design to best fit a real-time collaboration model.

OpenDreamKit participation began with the next stage, designing a full real-time collaboration prototype that does not rely on any external hosting provider, such as Google. This work was carried out in close collaboration with the Jupyter developer community. There are many participants in this effort, and OpenDreamKit has contributed approximately three person-months to the design and implementation of real-time collaboration in JupyterLab.

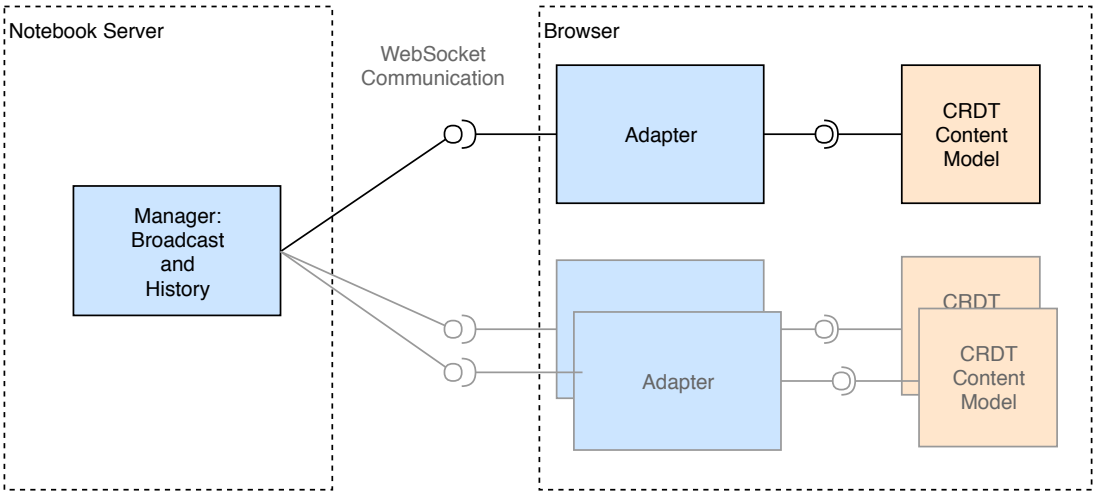

FIGURE 1. A schematic overview of the currently planned real time collaboration components. Light blue components should be implemented in JupyterLab, while the beige components (CRDT models) should be implemented in PhosphorJS.

The current road map specifies an implementation build on top of *conflict-free replicated data types* (CRDTs). The implementation outlines the following components: a client side implementation of the CRDTs; a client side interface for broadcasting these changes to other

consumers; a wire protocol for broadcasting changes and requesting history over the network; and a notebook server extension for broadcasting changes. See Figure 1 for a schematic overview.

So far, the wire format for communicating and requesting changes and history has been agreed upon, and proof of concept implementations for the transmission and receipt of the wire format have been completed. Work is ongoing in the PhosphorJS project to implement efficient client side CRDTs. The CRDT implementation will not be restricted to Jupyter notebooks. Any document-editing in JupyterLab, or built on the same APIs will be able to support real-time collaboration via this system, extending our possible impact beyond users of Jupyter notebooks.

3. Challenges and status

The biggest challenge for OpenDreamKit participation in JupyterLab is that it is a large collaboration, driven by many interests, often where one task must wait for another to be completed. Real-time collaboration is of interest to many involved, but relies on significant changes of underlying data structures in JupyterLab before implementation can proceed.

While we have not yet delivered a fully working production implementation of real-time collaboration, we have laid the groundwork and assembled prototypes and an implementation plan in collaboration with the JupyterLab team, and we are optimistic that our remaining resources will enable us to help deliver a working implementation integrated into JupyterLab by the completion of **T4.2**.

Disclaimer: this report, together with its annexes and the reports for the earlier deliverables, is self contained for auditing and reviewing purposes. Hyperlinks to external resources are meant as a convenience for casual readers wishing to follow our progress; such links have been checked for correctness at the time of submission of the deliverable, but there is no guarantee implied that they will remain valid.