Project Jupyter: Computational Narratives as the Engine of Collaborative Data Science

Note: this is the full text of the grant proposal that was funded by the Helmsley Trust, the Gordon and Betty Moore Foundation and the Alfred P. Sloan Foundation on April 2015, as described on these two announcements from UC Berkeley and Cal Poly, and press releases from the Helmsley Trust and the Moore Foundation. A PDF version of this document can be found here.

Fernando Perez
Lawrence Berkeley National Lab &
UC Berkeley

Brian E. Granger
Cal Poly San Luis Obispo

Statement of the problem and audience

Computers are good at consuming, producing and processing data. Humans, on the other hand, process the world through narratives. Thus, in order for data, and the computations that process and visualize that data, to be useful for humans, they must be embedded into a narrative - a *computational narrative* - that tells a story for a particular audience and context. There are three fundamental aspects of these computational narratives that frame the problem we seek to address.

First, a single computational narrative needs to span a wide range of contexts and audiences. For example, a biomedical researcher might originally perform statistical analyses and visualizations for a highly technical paper to be published in an academic journal. Over time, however, that same individual will give talks to other researchers, or even non-technical audiences. Eventually, it may even be important to enable non-coding lab scientists to perform that same statistical analyses and visualizations on data from new samples using a simplified graphical user interface. Across all of these different audiences and contexts, core aspects of the computational narrative remain invariant.

Second, these computational narratives need to be reproducible. That is, other people - including the same scientist six months later - need to be able to understand exactly what was done (code, data and narrative) and be able to reliably reproduce the work in order to build new ideas off it. Reproducibility has long been one of the foundations of the scientific method, but the rise of data science brings new challenges to scientific reproducibility, while simultaneously extending these questions to other domains like policy making, government or journalism.

Third, computational narratives are created in collaboration. Multiple individuals need the ability to work together at the same time, on the code, data and narrative. Collaboration is present in nearly all contexts where computational narratives are created: between two postdocs and a professor in the same research group; between the writers, editors and visual designers of an online news site; between the data scientists and business strategists at a large internet company; or between a teacher and students in a university classroom.

Given this background, the core problem we are trying to solve is *the collaborative creation of reproducible computational narratives that can be used across a wide range of audiences and contexts*. We propose to accomplish this through Project Jupyter (formerly IPython), a set of open-source software tools for interactive and exploratory computing. These software projects support scientific computing and data science across a wide range of programming languages (Python, Julia, R, etc.) and already provide basic reproducibility and collaboration features. This grant aims at making major progress atop this foundation. The main application offered by Project Jupyter is the Jupyter Notebook, a web-based interactive computing platform that allows users to author computational narratives that combine live code, equations, narrative text, interactive user interfaces and other rich media. These documents provide a complete record of a computation that can be converted to a number of formats (HTML, PDF, etc.) and shared with others through email, Dropbox, GitHub, etc. They can also be published online thanks to our Jupyter Notebook Viewer, a free service we operate that allows anyone on the web to view a notebook as a regular web page.

Related work and collaborations

In this section, we describe related work in two areas: interactive computing and online collaboration software. After that, we detail the various organizations with which we have significant collaborations.

There are a number of interactive computing environments that have similarities to our work with Project Jupyter. The largest group of products, by number of users, are the traditional commercial interactive computing environments: Matlab[1], Mathematica[2], SAS[3], SPSS[4] and Microsoft Excel. While these products are extremely popular, their proprietary nature and expensive licensing fees make them unattractive for open and reproducible scientific research and data science.

[1] Matlab, MathWorks, 2014 <http://www.mathworks.com/products/matlab>
[2] Mathematica, Wolfram, 2014 <http://www.wolfram.com/mathematica>
[3] SAS, 2014 <http://www.sas.com>
[4] SPSS, IBM, 2014 <http://www-01.ibm.com/software/analytics/spss>

On the open source side, there are the popular Sage[5] and RStudio[6] projects and the newer Spyder IDE[7], Beaker Notebook[8] , Zeppelin Project[9], SageMathCloud[10] and Wakari[11](Wakari is a proprietary project based on open-source tools). A number of these projects (Sage, Spyder, Beaker, SageMathCloud and Wakari) rely on and provide integration with the Jupyter/IPython architecture.

In the area of online collaboration software, there are two gold standards. First, Google Drive[12] has, quite literally, invented modern online collaboration by offering a productive environment that allows multiple, distributed users to simultaneously edit documents, spreadsheets, and slide presentations. For many organizations, these *real-time collaboration* capabilities of Google Drive have transformed how distributed teams get work done together. Second, for code and data, git[13] and GitHub[14] have played a similar transformative role in distributed collaboration. The git project is an open source distributed version control system that programmers use to track and share changes in complex software. GitHub is a commercial (but free for public usage) collaboration platform built around git that has become invaluable for companies, open source projects and scientists alike. SageMathCloud and Wakari expose the Jupyter Notebook online and provide some collaboration features. While there are other online code and document collaboration platforms (Bitbucket, Office 365, Hackpad, Etherpad, etc.) all of these are largely inspired by Google Drive and git/GitHub.

Over the past few years, we have spent significant amounts of time and effort investing in relationships with other individuals and organizations that have overlapping missions, impact areas, user groups and technologies as Project Jupyter. In the area of *academic research and education*, we have ongoing collaborations with individuals and departments at Stanford, UW, NYU, MIT, Harvard, Bryn Mawr, U. Southampton, U. Sheffield and Simula Research Lab (Norway). In the area of *open science*, we coordinate efforts with the Center for Open Science (Brian Nosek and Jeff Spies) and Software Carpentry (Greg Wilson). In traditional *journalism*, we have relationships with staff at 538, BuzzFeed and the New York Times focused around data-driven journalism. In *open source software*, we collaborate closely with the core developers of all the major scientific computing and data science projects in Python

[5] Sage Math Project, 2014 <http://www.sagemath.org>
[6] RStudio, 2014 <http://www.rstudio.com>
[7] Spyder IDE, 2014 <https://code.google.com/p/spyderlib>
[8] Beaker Notebook, 2014 <http://beakernotebook.com>
[9] Zeppelin Project, 2014 <http://zeppelin-project.org>
[10] "Account - SageMathCloud." 2013. 30 Jan. 2015 <https://cloud.sagemath.com/>
[11] "Wakari - Web-based Python Data Analysis." 2012. 30 Jan. 2015 <https://wakari.io/>
[12] Google Drive, Google, 2014 <https://www.google.com/drive>
[13] The git Project, 2014 <http://git-scm.com>
[14] GitHub, 2014 <https://github.com>

(NumPy, SciPy, Pandas, Matplotlib, Scikit-Learn, etc.), Julia (core developers) and R (rOpenSci).

We also work closely with a number of companies that are building products based on the Jupyter Notebook, contribute code and financial resources to the project and serve as advisors on a wide range of technical and strategic topics. Because these collaborations are so important for the ongoing sustainability of Project Jupyter, we wish to highlight a few of these.

GitHub is an online collaboration and hosting site for software projects and code. For us, GitHub is significant because many users of the Jupyter Notebook host and share their computational narratives, as Jupyter Notebook documents, on GitHub. Our own Notebook Viewer service renders notebooks stored on GitHub as static HTML pages, which can be shared with anyone in the world without their installing anything. We are currently working with Arfon Smith and Tim Clem of GitHub to explore other integration points between GitHub and Project Jupyter.

Rackspace is a commercial cloud hosting company that supports a wide range of open source projects, including Project Jupyter. For the last year, Rackspace has provided significant hosting resources for our popular Notebook Viewer service. This includes one of Rackspace's engineers, Kyle Kelley, building and maintaining this deployment and making significant contributions to the Jupyter codebase. In the summer of 2014, Kyle, with Rackspace's explicit support, began a much more ambitious effort to offer cloud-hosted Jupyter Notebooks where users could instantly try a live Jupyter Notebook to run Python, R and Julia code. Thanks to this work, we were able to embed a live demo of the Jupyter Notebook in an article about the project that was published in November of 2014 in Nature[15].

Microsoft has been collaborating with the PIs of this grant since 2008, when Dr. Brian Granger added support for Microsoft's job scheduler to IPython's parallel computing framework. Since then, Microsoft has added IPython integration to the popular Python Tools for Visual Studio, demonstrated the Jupyter Notebook running in the Microsoft cloud (Azure) and donated $100,000 to the project through NumFOCUS. We are currently working closely with Microsoft to identify future areas of collaboration.

Bloomberg is one of the largest financial data and news companies in the world. Two individuals at Bloomberg (Jason Grout and Sylvain Corlay) have been approved to contribute to Project Jupyter in an official capacity. Both are regular contributors and have been critical in the design of the project's interactive widget architecture. This collaboration has led to our working with Chris Colbert and others at Bloomberg to begin designing the next generation of web-based user interfaces. We will use the

[15] "IPython interactive demo : Nature News & Comment." 2014. 30 Jan. 2015 <http://www.nature.com/news/ipython-interactive-demo-7.21492>

open-source phosphor.js JavaScript library developed at Bloomberg, needed for this grant's deliverables. Bloomberg also has an official open source program and policies, including hosting of "Open Source Days" at their headquarters in NYC and London.

Google approached us in 2014 regarding a prototype of a project, called coLaboratory, that integrates the Jupyter Notebook into Google Drive. Most importantly, this prototype demonstrated that building real-time collaboration into the Jupyter Notebook would be possible. However, this prototype also revealed the incredible technical challenges of doing so. In late 2014, Google donated $100,000 to Fernando Perez at UC Berkeley to hire a postdoc (Matthias Bussonnier) that would begin to extract the Google Drive integration from coLaboratory into Project Jupyter itself. We want to emphasize two aspects of this collaboration. First, the technical challenges in building real-time collaboration into the Jupyter Notebook are so significant that we could not do this without close collaboration with Google. Second, the effort required to implement this in Project Jupyter requires resources that extend *far* beyond those provided to us by Google.

O'Reilly Media is a major publisher of technology focused books and online content as well as the organizer of the most significant conferences in the data science and open source spaces (Strata, Hadoop World, OSCON, etc.). In 2014, we began working with the CTO of O'Reilly, Andrew Odewahn, to explore ways of integrating the Jupyter architecture into their publishing platform, to enable both authors and readers of O'Reilly content to experience books as live, computational entities. O'Reilly already has multiple books that include code examples as Jupyter Notebooks.

Organizational background

Mission and Background

Project Jupyter's mission is to create open source tools for interactive scientific computing and data science in research, education and industry, with an emphasis on usability, collaboration and reproducibility.

Today's Jupyter evolved from the IPython project, created in 2001 as an interactive Python shell by Dr. Fernando Perez. Dr. Brian Granger joined the IPython project in 2004, and the two of us have led the project since then. The core development team has grown to roughly a dozen active contributors and a "long tail" of community contributors currently numbering over 400, who participate with various degrees of regularity.

For the first decade, IPython focused strictly on scientific and interactive computing in the Python language, providing a rich interactive shell well suited to the workflow of everyday research, as well as tools for parallel computing. It was part of an

organic ecosystem of open-source projects for scientific computing in Python, informally known as the "SciPy Stack".

Around 2010, IPython evolved from providing only a terminal-based interactive shell into a generic architecture for interactive computing and computational narratives in any programming language. This design allowed us to build the web-based Notebook described in this proposal. This expansion beyond Python led to a renaming of all the non-Python specific parts to *Project Jupyter*. Today, this architecture supports over 20 different programming languages, with most implementations having been created by third-parties.

Core problems and constituency

While scientists have always used computers as a research tool, they use them differently than industrial software engineers: in science, the computer is a kind of "abstract microscope" that enables the scientist to peek into data and models that represent or summarize the real world. Software engineers tend to write programs to solve reasonably well-defined and independently specified problems, and their deliverable is a software artifact: a standalone application, library or system.

While standalone software libraries exist in science (say the building of a library to solve differential equations), we target a more common scenario: the iterative exploration of a problem via computation and the interactive study of intermediate results. In this kind of computational work, scientists evolve their codes iteratively, executing small test programs or fragments and using the results of each iteration as insight that informs the next step. The computations inform their understanding of their scientific questions, and those questions shape the process of computing. The nature of this process means that, for scientists, an interactive computing system is of paramount importance: the ability to execute very small fragments of code (possibly a single line) and immediately see the results is at the heart of their workflow.

Furthermore, the purpose of computation in science is precisely to advance science itself. In the famous words of R. Hamming, *"the purpose of computing is insight, not numbers."* For this reason, computation in science is ultimately in service of a result that needs to be woven into the bigger narrative of the questions under study: that result will be part of a paper, will support or contest a theory, will advance our understanding of a domain. And those insights are communicated in papers, books and lectures: *narratives* of various formats.

The problem the Jupyter project tackles is precisely this intersection: creating tools to support in the best possible ways the computational workflow of scientific inquiry, *and* providing the environment to create the proper narrative around that central act of computation. We refer to this as *Literate Computing*, in contrast to

Knuth's concept of *Literate Programming*, where the emphasis is on narrating algorithms and programs. In a Literate Computing environment, the author weaves human language with live code and the results of the code, and it is the combination of all that produces a computational narrative.

We consider this problem while acknowledging that science is, by definition, an open, collaborative enterprise founded on the principle of independent validation of all knowledge. This means that *supporting collaboration and reproducibility* are central guiding principles of the project.

Finally, while all the above has been cast in the context of scientific research, the rise of ubiquitous data science means that these same questions are now not only the purview of physicists or biologists. Today policy makers, journalists, business analysts, financial model builders, all work with the same tools and challenges: their data may come from a population census or the stock market, and instead of an academic paper they may be writing a blog post or a sales report for a client, but ultimately the process is similar. They need to extract insight from data using computational tools, and they need to communicate that insight to an audience in the form of a narrative that resonates with that audience.

So today, Project Jupyter serves not only the academic and scientific communities, but also a much broader constituency of data scientists in research, education, industry and journalism. Given the importance of computing across modern society, we see uses of our tools that range from high school education in programming to the nation's supercomputing facilities and the leaders of the tech industry mentioned above.

Basically, anyone who needs to execute code an interactive programming environment can be legitimately considered as served by our project. As computation and data analysis become pervasively woven into the fabric of society, our constituency continues to broaden. The challenge for our organization is to maintain a focused research agenda where we provide a coherent vision of the future in interactive computation, a clean set of abstractions and tools, and a sustainable community model. These things, combined, should serve as the foundation on which others can then build the solutions they need in their specific contexts. The purpose of this proposal is to advance the state of the art in those core questions.

Project organization

Project Jupyter is organized around an open-source model that allows for individual Contributors to join the effort based on their personal interest, resources and availability. Along side this open community of Contributors, the project has a thin

layer of formal organizational structure and governance. A summary of that structure and governance follows[16]:

- A large, public, open and inclusive community of *Contributors* participate in the creation of Project Jupyter's software. Some contribute code, others documentation, ideas or bug fixes. Nearly all technical decisions are made through the informal consensus of this open community.
- Through a record of sustained activity, *Contributors* can be nominated to have more rights and responsibilities in the development of specific parts of the project. This is done by providing them with write privileges (known as "commit rights") in the code repositories of the organization, hosted on GitHub. We currently have roughly 20 people in this capacity.
- *Contributors* who demonstrate significant leadership and contributions for longer than 1 year can be nominated to the Jupyter *Steering Committee*. The *Steering Committee* is the formal governing body for the project and is ultimately responsible its the technical, strategic and communal health. We emphasize, however, that the *Steering Committee* delegates essentially all technical decisions to the open community.
- From a legal perspective, Project Jupyter is part of the NumFOCUS Foundation, a 501(c)3 organization dedicated to support research, development and education in open source scientific computing and data science. NumFOCUS provides legal structure and fiscal sponsorship for small amounts of community-focused project funds.
- The project also has *Institutional Partners*: companies, universities and other legal entities who have at least one employee on the *Steering Committee*. By raising money (donations, grants, for-profit business models) and employing project *Contributors* and *Steering Committee* members these *Institutional Partners* are the main source of financial support for the project.
- We emphasize, however, that *Contributor* or *Steering Committee* status is always based on the technical participation of individuals, rather than the financial weight of the *Institutional Partners*; it is impossible to "buy your way" onto the *Steering Committee*. The current *Institutional Partners* are UC Berkeley, Cal Poly, Rackspace and Continuum Analytics.
- The PIs on this grant are project *Contributors*, *Steering Council* members and employees of the UC Berkeley (Fernando Perez) and Cal Poly (Brian Granger) Institutional Partners. Through their seniority and long time (14 and 10 years

[16] Our detailed governance policies are publicly available here: https://github.com/ipython/ipython/wiki/IPEP-29:-Project-Governance.

respectively) commitment, leadership and contributions, they effectively lead the Steering Council and project.
- Project Jupyter deliberately has no full time employees through NumFOCUS; all full time staff positions are handled through Institutional Partners. UC Berkeley currently has 1 full time software engineer and two postdocs (in addition to F. Perez). Cal Poly currently has one full time software engineer, who is currently paid as an independent contractor through funds from Microsoft/NumFOCUS (in addition to B. Granger). Other Steering Council members are employed by Rackspace (Kyle Kelley) and Continuum Analytics (Damian Avila).

Project activities and highlights

The main project activities, supported by a combination of open source volunteers, funded researchers and industry partners, are:

- The creation of open source software, hosted on the Github site under the *ipython* and *jupyter* organizations. We currently host 46 public repositories.
- Hosting online services powered by our software, currently supported by Rackspace:
    - The Notebook Viewer (http://nbviewer.ipython.org): renders the URL of any notebook as a static web page, enabling effortless sharing of notebooks. This service gets currently ~ 800,000 page views per month, from ~ 200,000 visitors.
    - TryJupyter/tmpnb (http://try.jupyter.org): an ephemeral, anonymous live Jupyter Notebook. This lets anyone log into a Notebook server and experiment with the provided example notebooks or type their own code.
    - A live demo for the Nature Journal: since November 2014, as a companion for an article about IPython published by Nature, we have hosted an instance of the ephemeral notebook service that lives in the Nature.com domain. This has served over 20,000 live sessions and broke readership records for simultaneous users on the Nature site.
- A public chat room (https://gitter.im/ipython/ipython/help) where our project developers help members of the public with general questions.
- A public mailing list where more long-form discussions take place.
- Weekly development meetings publicly broadcast via Google+ Hangouts and archived on YouTube. This improves our community engagement and the transparency of our process. Multiple other open source projects have adopted this model since we introduced it in 2013.

Estimated user base. It is very hard to get accurate user counts for an open source project that can be downloaded freely from multiple sources. But *we estimate at least 2 million users for IPython*. This is a rough number, but if anything, a conservative undercount. We justify this number as follows:

- Estimates of Linux users (not datacenter servers) range from 20M for Ubuntu to ~70M across Linux Distributions[17].
- The Debian Linux distribution tracks package installations with the 'popcon' tool. This shows IPython to be regularly installed in ~ 5% of Debian systems[18].
- If we use Debian as a baseline, and estimate total Linux user counts at ~50M (rough average of the above two numbers), we get about 2.5M installs of IPython on Linux.
- This doesn't count many other sources IPython can be installed from, such as Github, Python's package repository, the Continuum Anaconda distribution, Enthought Canopy, etc. Nor does it count the increasing number of server-side hosted deployments we see more and more of.

Other highlights. A few other relevant achievements of the project over the last few years:

- Very large user base.
- Three books have been published devoted to IPython [19] [20] [21], and several more cover IPython in detail (one or more chapters). Two books have also been written either entirely as Notebooks or containing extensive Notebook collections[22] [23], and we know several more are being written.
- Courses at top universities in the US and abroad use Jupyter Notebooks as core educational technology. We know of over a dozen at UC Berkeley, Cal Poly, U. Santa Clara, Harvard, Columbia, U. Claude Bernard Lyon (France), and more.

---

[17] http://en.wikipedia.org/wiki/Ubuntu_(operating_system)#Installed_base
http://en.wikipedia.org/wiki/Linux_adoption#Measuring_desktop_adoption

[18] https://qa.debian.org/popcon.php?package=ipython

[19] "Learning IPython for Interactive Computing and Data ..." 2014. 30 Jan. 2015 <https://www.packtpub.com/big-data-and-business-intelligence/learning-ipython-interactive-computing-and-data-visualization>,

[20] "IPython Notebook Essentials | Packt." 2014. 30 Jan. 2015 <https://www.packtpub.com/application-development/ipython-notebook-essentials>

[21] Rossant, Cyrille. *IPython Interactive Computing and Visualization Cookbook*. Packt Publishing Ltd, 2014.

[22] "Python for Signal Processing - Featuring IPython Notebooks." 2013. 30 Jan. 2015 <http://www.springer.com/engineering/signals/book/978-3-319-01341-1>

[23] "Mining the Social Web, 2nd Edition - O'Reilly Media." 2013. 30 Jan. 2015 <http://shop.oreilly.com/product/0636920030195.do>

- There have been 18 academic (peer-reviewed or preprints) articles [24] that provide IPython Notebooks to support reproducibility.
- There are independent implementations of the Jupyter protocol that provide kernels in over 25 different programming languages.
- Google Research created and released the CoLaboratory system for integration of Notebooks with Google Drive as an app in the Chrome web store. This effort led to a funded collaboration with our team.
- Professors Lorena Barba (George Washington U), Ian Hawke (U. Southampton) and Carlos Jerez (U. Pontificia Católica de Chile) taught in 2014 a MOOC on numerical computing whose teaching materials consist entirely of IPython Notebooks[25].

There are many more teaching materials, conference talks, blog posts and projects using our architecture and tools, that we can not fit in this space. We refer the reader to the Notebook Gallery and list of Projects using IPython in our wiki for a comprehensive list (https://github.com/ipython/ipython/wiki).

Approach and strategy

This project is structured in a "3+1" format, with three main focus areas of research and development and one extra topic of ongoing work. The three focus areas are *Interactive Computing*, *Computational Narratives* and *Collaboration*. The problem of *Sustainability* will require ongoing attention but is conceptually distinct from the first three, as it doesn't focus on specific research questions or deliverables.

Interactive Computing

At the heart of the entire Jupyter architecture lies the idea of *interactive computing*: humans executing small pieces of code in various programming languages, and immediately seeing the results of their computation. Interactive computing is central to data science because scientific problems benefit from an exploratory process where the results of each computation inform the next step and guide the formation of insights about the problem at hand. In this *Interactive Computing* focus area, we will create new tools and abstractions that improve the reproducibility of interactive computations and widen their usage in different contexts and audiences.

[24] "A gallery of interesting IPython Notebooks · ipython ... - GitHub." 2013. 30 Jan. 2015 <https://github.com/ipython/ipython/wiki/A-gallery-of-interesting-IPython-Notebooks>

[25] "Practical Numerical Methods with Python | GW Open edX." 2014. 30 Jan. 2015 <http://openedx.seas.gwu.edu/courses/GW/MAE6286/2014_fall/about>

Notebooks as interactive applications

The Jupyter Notebook has an architecture for interactive widgets that bind data and code in the backend (Python, Julia, R, etc.) to interactive user interfaces running in the browser. This enables users to quickly explore code and data by manipulating user interface controls (buttons, sliders, etc.). This architecture is being used to create custom user interfaces in the notebook using powerful JavaScript libraries such as d3.js. These widgets open the door for non-technical "consumers" of notebooks to explore data in a notebook, without coding. We will extend the widget architecture into a full-blown application framework.

First, we will enhance our widgets to handle more complex hierarchies of objects with events, parent/child relationships, etc. Second, we will enable these widgets to be deployed in contexts outside the traditional notebook, where a live computational kernel is not running, such as nbviewer or a static HTML page. Third, we will create a system that allows users to bundle, share and deploy sets of widgets as independent "apps." This will allow users and developers to leverage the notebook for highly customized, but still data and code driven, user interfaces that can be used with non-technical audiences.

Modular, reusable UI/UX

The Notebook user interface (UI) is the main way that users interface with our project on a daily basis. In order to create a more humane and productive environment for users, we will invest significant effort on improving the UI and user experience (UX).

First, we will create a more modular set of UI components to enable users and third party developers to build purpose specific UIs with custom components, such as file browsers, debuggers, variable inspectors, documentation panes, etc. This work will be done in collaboration with Bloomberg (see above for the details of this collaboration). Second, by hiring a full time UI/UX person, we will begin to perform studies of the usability of our various UIs. This will include qualitative user testing and quantitative A/B testing. Third, we will add a set of richer actions that can be performed on single or multiple notebook documents. These include multicell operations (cut/copy/paste), structural operations that allow different sections and subsections to be collapsed/expanded and moved atomically, and an improved dashboard for working with directories of files and notebooks.

Software engineering with notebooks

The Jupyter Notebook emerged out of a need to capture otherwise transient interactive computations in a form that could be reproduced and shared with others. However, as single notebooks grow in size and complexity, they become difficult to work with from the software engineering perspective (testing, documentation, reuse, modularity, etc.). In this grant we will improve the software engineering aspects of notebook based workflows.

First, we will improve the ability of users to transition from a single large notebook, to a smaller notebook that calls code contained in external modules that can be tested and documented separately. Second, we will work to enable notebooks themselves to be treated as importable Python packages. This will enable code in one notebook to be used in other notebooks or projects more easily. Third, we will build tools that allow notebooks to be tested in similar ways as traditional software projects are. Fourth, we will build tools that can verify that a notebook is reproducible; that it gives the same results when run again. This verification will be performed by rerunning the original notebook, comparing the output of the rerun notebook with that of the original, and then creating a human readable "reproducibility report" that summarizes the differences, if any. This will also enable us to develop quantitative metrics that measure the degree of reproducibility.

Capstone

As a capstone to the *Interactive Computing* focus area, we will create a prototype of a spreadsheet app/widget that is integrated with the notebook architecture. This prototype will validate our improvements to widgets, our UI/UX modularization work and our work towards "application" oriented usages of our architecture.

Computational Narratives

Jupyter Notebooks enable users to create and share code and data driven narratives. These narratives end up being used in a wide variety of *contexts*: academic publications, blog posts, books, traditional journalism articles, technical documentation, government reports, grant applications, industry research and commercial products. Furthermore, a *single* notebook could be communicated in a number of different *formats* (PDF, live demo, web page, slideshow) to different *audiences*. Today, users encounter significant pain along this path. In this *Computational Narratives* focus area, we will build tools to enable the notebook to be used more easily across different contexts, formats and audiences.

nbconvert

nbconvert is the command line tool that Jupyter offers for converting notebooks to different formats (currently LaTeX/PDF, HTML, Markdown and reveal.js). We will target the following areas of work on nbconvert in this grant.

First, we will create an EPUB exporter for nbconvert. Because EPUB is based on HTML/CSS/JavaScript, it is a much better format than LaTeX/PDF for representing the rich content found in notebooks. Most importantly, EPUB is an ideal format for interfacing with publishers in the academic and technical spaces. EPUB support will help our ongoing collaborations with O'Reilly Media and Nature. Second, the nbconvert software needs significant refactoring and improvements to its command line and programming APIs to enable users and developers to customize it more easily. This includes improved documentation and examples. Third, we will explore the feasibility of exporting notebooks to Microsoft Word. This will be done through our ongoing collaboration with the data, development and machine learning teams at Microsoft, which is described above.

Element filtering

Notebooks contain code (source code and output) and markdown (narrative) cells. When a notebook is used across different contexts and audiences, it is useful to filter what types of content is visible. For example, when a notebook is read by non-technical users, it is helpful to hide all of the source code, but show the narrative text, visualizations and widgets. To address these needs, we will create a system that allows the notebook content to be selectively filtered, based on the intended context and audience.

First, we will enable cells to be tagged with user selected labels ("homework", "testing", etc.). Users will add labels to cells using an appropriate user interface and the labels will be stored in the notebook metadata. Second, we will create an interactive query syntax that allows content to be selectively hidden and shown based on the cell type, cell labels, widgets, input/output, etc. This query syntax will be integrated into the live notebook, nbconvert and nbviewer.

Documentation

Documentation is one of the primary ways that users interact with open source software. Existing documentation for Jupyter and other open source projects is written as static web pages generated by markup languages such as Markdown. Developers are forced to manually copy and paste code samples into this format. This leads to documentation that easily falls out of dates and cannot be tested. For users, these static web pages are not integrated with the notebook, can't be searched and most

importantly, can't be run as live code. Furthermore, all of the documentation for each open source project is hosted on different places on the web.

To address these issues, we will create a notebook based documentation system for our own and other open source software projects. This documentation system will allow developers to write documentation as notebooks and package them with their own project as live code. Upon installation of these packages, users will be able to browse and search all of the documentation from within the notebook. Most importantly, users will be able to run the documentation as live code. To test this approach, we will write our own documentation using this system.

Capstone

As a capstone to this *Computational Narratives* focus, we will test our deliverables in the context of collaborations with publishers, both traditional and web-oriented. These publishers include O'Reilly, Nature, GitHub, BuzzFeed and 538.

Collaboration

Since the Jupyter Notebook was released in 2011, better support for collaborative workflows has been our users' most common request. This is for good reason. In our modern, web-enabled companies, universities, research labs and non-profits, data science and scientific computing are carried out by distributed teams whose work and contributions are tightly coupled. For static content, this is enabled by technologies such as email, video chat, online comment/review systems, GitHub and Google Docs. Today, the Jupyter notebook has almost no support or these types for synchronous and asynchronous collaborations, which limits the impact and usefulness of the notebook in collaboration rich contexts such as education and scientific research.

Real time collaboration

In this grant, we will add real-time collaboration capabilities to the notebook that are modelled on the abstractions and architecture of Google Drive/Docs. This will allow multiple users to share notebooks with each other online, and edit those notebooks together in real time. To this collaborative editing system we will add user presence, commenting and cloud based document storage.

As described above, this work is extremely technical and will require major rewrites of significant portions of our architecture. Furthermore, there are significant security issues to work through. Because of the difficulty and scope of this work, we are working directly with Google Research to help us design the underlying architectures and implement them in our software (see above for the details of this collaboration).

The initial implementation of these features will rely on open Google APIs (Drive API, Real Time API), however, we plan on building abstractions and APIs that will allow us to plug into a number of different collaborative backends (Firebase, etc.).

JupyterHub

The basic Jupyter Notebook is a single user web application that most users install and run on their own laptop or desktop. JupyterHub is a multiuser version of the notebook server than can be run on a central server(s) or in the cloud. JupyterHub eases the installation and deployment of the notebook to large numbers of users and opens the door for novel collaboration possibilities, However, the version of JupyterHub that exists today has very limited sharing capabilities. In this grant we will improve the collaboration capabilities of JupyterHub in the following ways.

First, we will define richer and finer grained sharing semantics that allow users to share notebooks with other individuals or groups. This will include user interfaces that make this easy to do across different storage and deployment backends. Second, we will work to create tools that ease the deployment of JupyterHub in different contexts. Third, we will allow users of JupyterHub to "publish" notebook to other users of JupyterHub or the public.

Capstone

The individual tasks in this *Collaboration* focus area involve creating new, but semi-separate, collaboration capabilities for the different sub-projects (Jupyter Notebook, JupyterHub). As a capstone, we will begin to integrate these different collaboration approaches to create an integrated, ubiquitous system for notebook based collaboration.

Sustainability

As the scale of Jupyter's usage and development expands, it is important to create for us to create a sustainable technical project, community and organization. This focus area is conceptually different from the above described 3 main technical areas and will involve an ongoing set of activities throughout the project period.

People

People are the backbone of our sustainability plan. While the project has had over 400 contributors, most of the major work has been done by a few key individuals. Thus, our first goal is to expand the set of these key contributors.

First, we will set up a robust training program that leverages senior project staff to manage and train new undergraduates, graduate students and postdocs to work on the project at Cal Poly and UC Berkeley. New data science programs at both of these universities will be used as the needed source of initial human capital for these efforts. We have an excellent track record of training students; two of our most senior Contributors and Steering Council members were previously undergraduate students of Brian Granger.

Second, we will send these newly trained individuals out into academia and industry where they can expand our network of collaborators, contributors and Institutional Partners even further. We know of multiple companies that are currently interested in hiring project newly trained Contributors and Steering Council members. To initiate this "sending out" this grant will fund two of our current Steering Council members (Min Ragan-Kelley and Thomas Kluyver) to move from UC Berkeley to institutions in Norway and the UK. We plan on working with them to build their own, independent, European based funding sources in the future. Their move will also create two new Institutional Partners: Simula Research Lab and the University of Southampton or Sheffield (which of these two will join is being determined, but one of them will).

Finally, a key concern of the project, inscribed in a larger societal discussion of the problem, is improving the diversity of our community. We address it in detail in the required Appendix.

Events

Jupyter related events will enable us to build a more sustainable community of users, developers and collaborators.

First, we will continue to have week long developer meetings twice per year. These meetings bring together 5-15 core developers and designers to review the project's progress, discuss major technical and architectural issues and plan the future roadmap of the project. Because our core developers are geographically distributed, these in person meetings are critically important for us to build a cohesive developer community and project.

Second, for the first time, we plan on organizing JupyterCon, an annual conference to bring together all of Jupyter's developers, users, collaborators and Institutional Partners. This conference will be a 2-3 day event in the Bay Area or New York that has time for talks, coding sprints, brainstorming, etc. As the project grows in size, JupyterCon will be an important way for us to bring our community together in a focused event. JupyterCon will also enable Institutional Partners to provide input to the project; eventually we forsee Insitutional Partners as becoming an advisory board for

the project that works alongside the Steering Council. This centralized conference will also be used to seed other, smaller community organized outreach event in cities throughout the world (JupyterDays).

Third, we will continue to disseminate the results of our work to an ever wider range of communities in academia and industry. This will include talks at academic and industry focused conferences and workshops and the publication of articles about our work in academic journals.

Year by year output

Our approach to building software

We want to clarify our approach to building software and describe how that relates to the deliverables of this grant. The approach described here has emerged from our own experience in building open source software over the last 14 years as well as a careful study and application of the methods described by Eric Ries in his book, the *The Lean Startup*, as well as the books and courses of Steve Blank.

First, for each deliverable, we always begin by creating an Minimum Viable Product (**MVP**). The MVP is an initial implementation of that deliverable that provides the absolute minimal set of features we hypothesize will be useful to our users. The MVP always has a very limited scope and lacks features present in the final version.

Second, we immediately release the MVP to our users and begin watching how they respond. Our goal in this phase is to collect as much information as possible to **validate** our hypotheses about the deliverable. During this stage, we also identify the individuals and organizations who are stakeholders and collaborators in building that particular deliverable.

Third, we then incorporate the information gathered through validation to build the final version of the deliverable that has the exact set of features required by users. The final version typically has much larger scope than the MVP and is built with the collaborators identified in the validation stage.

Given this background, *we expect the funding provided in this grant will be sufficient for us to complete the MVP and validation stages of each deliverable*. For some of the smaller deliverables, we also expect to start building the final versions. However, we expect the final versions of all deliverables to be out of scope of this grant for two reasons. First, the validation stage is completely unpredictable. Features are used in unexpected ways, new groups of users emerge, other developers extend and reuse our work in innovative ways, and new collaborators and stakeholders emerge. Second, the

validation phase typically expands the scope of the deliverable far beyond our original plans and budget.

However, we want to emphasize that the validated MVPs produced through this grant's activities will be highly functional and have a deep impact on our users. To set the scale appropriately, we consider most of our current software, including the Jupyter Notebook, to be at the validated MVP stage.

Year-by-year plan of deliverables

The following table details our year-by-year plan of deliverables in the core focus areas of *Interactive Computing*, *Computational Narratives* and *Collaboration*. The numbers in the table represent the number of full time staff technical staff working on that deliverable at UC Berkeley, Cal Poly, Simula Research Lab and the University of Southampton. This table only includes our eight software engineers and postdocs, as well as ½ FTE on Y1 for a technical writing consultant. We expect the UI/UX designer, Project Manager and 2 PIs to work across all deliverables each year.

Deliverable	Year 1	Year 2	Year 3
NB as Apps ⊥	0.5	1	1
UI/UX ⊥	1.5	2	2
NB Software Eng. ‖	1		
nbconvert ‖		1	1
Element Filtering ⊥	1	0.5	0.5
Documentation ‖	1.5	1	
Real Time Collab ⊥	2	2.5	2.5
JupyterHub ⊥	1		1

Measuring effectiveness

We will measure the effectiveness of our work through the following metrics. All target numbers are three year totals unless otherwise specified.

- Number of talks given at conferences and workshops we have never attended before (target=10).
- Ethnic and gender diversity of active Jupyter developers in our broader community (target=3 from currently underrepresented groups. This would be a very significant increase, given our core development team has roughly a dozen regularly active members).
- Traffic on project web site and web services (nbviewer and tmpnb) (target=4x current traffic).
- Number of deployments of JupyterHub by 3rd parties in research, education, industry (target=direct knowledge of 25).
- The number of universities, companies and research labs that adopt the Jupyter Notebook and related technologies at an institutional level (target=6).
- Institutions and projects that use our building blocks as infrastructure to create other software and products (target=12).
- The number of books, academic publications, education course materials, journalism articles that use the notebook as a primary or secondary mechanism to deliver content (target=100).
- The number of new collaborations with large companies in the data science space (target=6).
- The number of companies providing funding for the project (target=12 companies).
- The number of new Institutional Partners of the project (target=4)
- The number of undergraduate, graduate and postdoc students that are trained through the grant activities and placed in related jobs (target=12).
- Number of deliverables for which the MVP and validation stages are completed (target=all).
- The MVP and validation stages increase the impact, scope and reached users of our deliverables so significantly that we have to do additional fundraising to complete final versions.

Budget justification

The main expenses for this grant are for the salaries, travel and supplies for full time project staff who will work at UC Berkeley, Cal Poly, Simula Research Lab and the University of Southampton or Sheffield. Because of this, our budget justification is organized around these staff positions. The amounts allocated for travel are based on our actual travel numbers over the past two years. For project PIs, this amounts to approximately 10 week long trips per year and for other project staff this amounts to

1-3 week long trips per year. First year supplies are higher to enable us to purchase computers and monitors for staff.

Project PIs (Dr. Fernando Perez and Dr. Brian Granger). Core activities of the project PIs include managing and training staff, creating the strategic direction of the project, building relationships with Institutional Partners and key collaborators, giving talks at conferences and workshops, fundraising, community building, hiring project staff, etc. To enable both PIs to focus on Project Jupyter, the budget funds significant fractions of the PIs' time. Dr. Perez will lead the UC Berkeley team and bridge the project with related activities in data science at the Berkeley Institute for Data Science and Lawrence Berkeley National Laboratory, where he holds a Staff Scientist appointment. Dr. Brian Granger will lead the team at Cal Poly, where he is an Associate Professor of Physics and Data Science. This position gives him access to highly motivated and talented students that can be hired and trained to work on the project.

Existing full time staff (Dr. Min Ragan-Kelley, Dr. Thomas Kluyver, Dr. Matthias Bussonnier and Jon Frederic). These individuals are core Contributors and Steering Council members who form the technical backbone of the project; without their full time work on the project, technical activity would completely grind to a halt. Dr. Ragan-Kelley will be working as a postdoc at Simula Research Lab. Dr. Kluyver will be working as a postdoc at the University of Southampton. Dr. Bussonnier will be a postdoc at UC Berkeley. Mr. Frederic will be a senior software engineer at Cal Poly.

Project Manager. This grant will scale our full time project staff from 6 to 14. We feel it is critical for us to hire a new full time Project Manager (at UC Berkeley) to help the PIs manage the increased project scope and technical staff. This role will enable us to segment the deliverables into smaller pieces that can be tackled in parallel by more independent teams, while still keeping the project wide vision and approach consistent.

UX/UI Designer. Significant amounts of our current and proposed work involves visual and interactive design, as well as frontend web development. We currently have no designers on the team; this has created a significant bottleneck for us, even with our current scope. To enable us to tackle the ambitious user interface work of this grant, we propose to hire a new full time user interface/experience (UI/UX) designer who will work with the different teams to design, build and test these user interfaces.

Software engineering. The deliverables of this grant will expand the scope, complexity and effort of the project significantly. The popularity of the project has also put incredible pressure on our existing developers to give talks at conferences and support users online. To enable us to tackle these challenges, we propose to hire four new full time software engineers: two postdocs at UC Berkeley and two software engineers at Cal Poly.

Administrative help. Currently the project PIs have to do all project administration: filing reimbursements, managing the budget, travel and event planning, etc. To handle the increased budget and activities of this grant we propose to hire two new project administrators (full time at Cal Poly and ½ time at UC Berkeley). These administrators will free the PIs, Project Manager and technical staff to focus on the deliverable of the grant. The full time administrator at Cal Poly will also be the lead on planning various project events: core developer meetings, JupyterCon and JupyterDays.

Consultants. During the first year we will hire two consultants. First, we will hire a web/design firm to overhaul our entire web presence, which has seen nearly zero development in years, is a poor representation of the project and lacks key usability aspects. Second, we will hire a technical writer to go over our entire body of documentation and help us triage, rewrite, clarify and organize it. Beyond the first year, our full time technical staff will take over the maintenance of these resources, however, their current state is so bad that we need extra help catching up.

Students. This grant will fund 4 summer undergraduate students each year at Cal Poly. These students are a critical part of our long term sustainability plan. Our goal is to train these students to work on Project Jupyter in different capacities (software engineering, technical writing, design, etc.) and then send them out to jobs at current and future Institutional Partners. Dr. Granger's position at Cal Poly gives him access to talented students in the new Data Science and Computer Science degree programs.

JupyterCon/Days. We plan on organizing a new Jupyter focused conference each year (JupyterCon). This will be a 2-3 day event that will bring together contributors, steering council members, institutional partners, third party developers and users. Over the long term, we plan on this conference becoming self supporting through industry sponsorship and registration fees. The grant budget includes seed money we know will be required to get this conference off the ground. Some of this seed money will also be used to seed single day Jupyter events that are organized by the larger community (JupyterDays).

Core Developer Workshops. Over the past two years, we have bought our core project Contributors together twice a year for a week long in person meeting. During these meetings, we review the project's progress and set a roadmap for the following six months. Given the distributed nature of our project, these in person meetings are absolutely critical in building our team and setting the strategic vision of the project. The grant budget includes funding for our core Contributors to attend these events.

Other sources of support

Through NumFOCUS, Project Jupyter currently has funding from Microsoft, which donated $100,000 in 2013 to NumFOCUS for general project work. This money has been used to hire Steering Council member (Jonathan Frederic) as an independent contractor working at Cal Poly with PI Brian Granger.

The Institutional Partner, Rackspace, is donating significant cloud hosting resources and the time of Kyle Kelley, a Rackspace employee and Steering Council member.

The PIs of this grant, through the Institutional Partners UC Berkeley and Cal Poly have the following Jupyter related funding:

- A $1.15M grant from the Alfred P. Sloan Foundation during 2013 and 2014 for the creation of nbconvert, interactive widgets and a prototype of JupyterHub. We expect this grant's funds to run out in March 2015.
- A $100,000 grant from Google in the fall of 2014 to UC Berkeley to hire Steering Council member (Matthias Bussonnier) as a postdoc to begin integrating the Jupyter Notebook and Google Drive.
- A $100,000 grant from the Simons Foundation to F. Perez at UC Berkeley, that supports the integration of the Jupyter Notebook into a system for data sharing in neuroimaging. This grant is in collaboration with the Stanford Center for Cognitive and Neurobiological Imaging, led by prof. Brian Wandell.

We also plan to raise additional funding through UC Berkeley and Cal Poly in the coming years. This additional funding will be used primarily to fund new work not funded by this grant. Examples include:

- A replacement or extension to Markdown syntax that handles the full complexity of academic and book publishing.
- Features found in traditional Interactive Development Environments (IDEs) (IntelliJ, Microsoft Visual Studio and Eclipse) such as an interactive debugger, variable inspector, refactoring tools, etc.
- Technologies that enable the Jupyter Notebook and JupyterHub to scale to internet sized user groups.
- Developing tmpnb and nbviewer into full blown platforms for sharing, indexing, reviewing/commenting and searching notebook based content.
- Robust security for the Jupyter Notebook that addresses deployments on the open internet and in highly secure settings.
- A federated architecture for live Jupyter Notebooks in the cloud that spans multiple cloud vendors (Amazon, Microsoft, Rackspace) and geographic regions.
- Internationalization of the Jupyter Notebook, JupyterHub and Documentation.

- Accessibility support across all our software.
- Improvements to our parallel computing framework. There is significant interest in this area from federal agencies (NIH, NSF and DOE) in this area. Problems such as the use of our tools in High Performance Computing (parallel supercomputers) environments, or in domain-specific contexts like genome sequence analysis, present challenges that go far beyond the scope of the current proposal. We have already been approached by multiple scientists interested in pushing forward with ideas based on our architecture in directions like these and others.
- Implementations of interactive widgets for languages other than Python, such as R.

We have already received preliminary interest from the following companies in funding these and other initiatives: Google, Microsoft, Bloomberg, Continuum, Quantopian, West Health, IBM. In the coming 3 years, much of the time and effort of the project leaders and PIs of this grant (Fernando Perez and Brian Granger) will focus on this fundraising work. Some of the industry money that we raise will also be used to build final versions of the deliverables of this grant that end up with a larger scope than currently envisioned.