Sustaining scientific infrastructures: transitioning from grants to peer production (work-in-progress)

James Howison

University of Texas at Austin

Abstract

Science now relies on mid-level infrastructure, including shared instruments, cell lines, supercomputing resources, data sets, and software components. These are beyond the facilities and services traditionally provided by individual universities; funding agencies such as the NSF often support their initial creation but their long-term sustainability is a challenge and commercialization is only rarely an option. A promising model, though, is broad-based community support through peer production, often inspired by the organization of open source software projects. Such transitions, though, are not automatic or easy, just as commercialization is not. In this research I am studying successful and unsuccessful attempts to transition, building theory and practical guidance for scientists and funding agencies. In this work-in-progress paper, I present the motivation and background for the study and provide motivation through preliminary description of my first case study.

Keywords: cyberinfrastructure, peer production, organizational studies of science, scientific software, information work

Citation: Editor will add citation with page numbers in proceedings and DOI.

Contact: Editor will add e-mail address.

1 Introduction

How ought we sustain long-term, mid-level, scientific infrastructure, like scientific software, data sets, and shared collections of scientific samples and artifacts?

Given the limits of commercialization for very specialized scientific infrastructures, an increasingly common answer in practice is to build broad-based community-support, tapping into institutional funds, overhead, and portions of domain grants to sustain infrastructure collectively. While we know that work building and sustaining infrastructure is complex and often invisible, including to our systems of academic credit (Edwards et al., 2013; Ribes & Finholt, 2007; Star & Ruhleder, 1996), we know very little about the work of transitioning initially grant funded projects to long term models of community support. This is in marked contrast to the sizable literature on commercialization, usually called “technology transfer.” In this project I draw on the literatures of peer production and online communities to investigate the process of transitioning from grant funding to open models of technology-enabled collaboration to sustain scientific infrastructures, and to advance theory in the organization of science as well as peer production. In this research-in-progress paper I lay out the justification for the research, examine relevant literature, and describe the research plan, illustrating, where possible, with preliminary description of my first case.

Just as infrastructure underlies economic development, science too is supported by infrastructures (Edwards et al., 2013; NSF, 2012). Infrastructures range from the abstract, such as theories and other conceptual schemes and standards, to the social, such as invisible colleges and academic societies, and to the organizational, such as offices of sponsored programs as well as, of course, to the solidly material, such as desks, chairs and fire extinguishers. In between lie mixtures of ideas and artifacts, most obviously the infrastructure of scholarly communication, publishers, books and digital libraries. In the middle too is scientific software, a mixture of the highly conceptual made concrete. Such software includes analysis software, simulation software, and middleware that stiches together ideas and data to make science possible (Bietz, Baumer, & Lee, 2010; Howison & Herbsleb, 2013; Segal, 2009).

From where do infrastructures arise and where lies their future? Where and how are these infrastructures created and sustained? The discourse on technology transfer explores one path forward for the results of scientific practice (e.g., Bozeman, 2000; Siegel, Waldman, & Link, 2003; Teece, 1986). That path forward is a transition from the domain of scientific research to the general economy through commercialization. Here the source of long-term sustenance is familiar: a technology, technique, or artifact is sustainable if it is able to generate income sufficient to support its ongoing costs. Once a commercial transition is successful, the technology continues to be available to support scientific research (as well as far beyond). From the perspective of science, the usefulness of these technologies to the general economy produces resources that might be said to cross-subsidize the usefulness of these technologies to science.

Yet once we limit our gaze to those infrastructures that particularly support science we encounter technologies that are quite specific and may well have limited relevance outside science, rendering them poor candidates for commercialization and the positive cycle of cross-subsidization. Some are provided by institutions of scientific research, especially Universities, supported by their range of resources, from endowments, tuition, state government funds and indirect cost recovery on research grants ("overhead"). Thus while labs, furniture and building maintenance are scientific infrastructures, they do not "earn their keep" through technology transfer but through the overall operations of institutions that house scientific research.

In stark contrast are very large scientific infrastructures that are directly supported by congressional appropriations, such as the science elements of space-travel, Antarctic research bases, or mega-projects such as supercolliders. Important enough to national priorities (including defense and short-term economic development) to command direct appropriations from national governments, these infrastructures live or die (sometimes quite suddenly) at the level of national politics.

Falling in the middle are the technologies that are the focus of this research: technologies that are vital to scientific research, yet not large enough for direct appropriations, nor falling into those facilities and services usually provided by Universities, nor obviously candidates for technology transfer via commercialization. This is not a new category; indeed it includes elements such as shared collections of scientific samples and artifacts, shared instruments, and non-profit scientific publishing that have been developed and maintained by scientists for hundreds of years. Yet it is a growing category and an important one (Atkins, 2003); Scientific software is an excellent example of this middle ground of science infrastructure.

2 A motivating pilot study: Enzo

In pilot work for this project I have researched the Enzo software project (O'Shea et al., 2004; The Enzo Collaboration et al., 2013). Enzo provides simulation facilities for examining star and galaxy formation, including the ability to simulate the formation of the very first objects in the universe. This software is derived from cutting edge science practice; it is something of a side effect of pursing a mainline scientific domain research question. Yet quite the side effect; now having supported science through two decades and hundreds of papers by many authors. Such software actualizes the discoveries made with it, embodying them, storing them, and making them available for further research in a form different from, but just as useful as, knowledge abstractions and equations (Edwards, 2010; Ince, Hatton, & Graham-Cumming, 2012). In this way, scientific software is similar to other kinds of scientific infrastructure: infrastructures make science work better, but they themselves take a great deal of work (Lee, Dourish, & Mark, 2006).

Unlike equations (or at least more obviously than equations), infrastructures require ongoing work to keep them scientifically useful. This is particularly true of software where work is needed not only to respond to advances in the scientific frontier but to changing underlying technologies and, perhaps least well appreciated, changes in the software stack on which it is built. Keeping software scientifically useful (avoiding "bit-rot" in software engineering parlance) is hard work (Bietz, Ferro, & Lee, 2012; Howison & Herbsleb, 2013; McCullough, McGeary, & Harrison, 2006). How is this work to be resourced for the long-term benefit of science?

Enzo is not an obvious candidate for commercialization in the short term because understanding star formation is not an economically profitable enterprise, unlike some scientific software that began with grants, such as the SAS statistical package, or one likely to see widespread non-scientific use (Dalle & Rousseau, 2004). Nor is it a candidate for direct congressional appropriations, being neither large enough, nor unique enough, having several competitor projects. Nor is maintenance of Enzo obviously a

traditional indirect cost of research, being closely tied into the process of scientific work itself and being useful across many different institutions.

My pilot research shows that Enzo was supported for many years as many emergent scientific infrastructures are: through the enterprising activities of a lead PI in skillfully using resources from institutionally diverse sources, including startup research funds, portions of domain grant funds, redirected overhead, and the elastic time-commitment of the lab's scientists and graduate students themselves (the 20 hours mentioned in student stipends being something of a fiction), giving another spin to “invisible work”. Finally, the work was also supported by small grants targeted at Enzo itself, written to add new cutting edge scientific features. This model of infrastructure support is resilient and pervasive. It is also tiring, frustrating and limiting.

In particular the work that is hard to justify under a model of PI support is also possibly the work that is potentially most valuable to science: generalization and coordination. And that is the work needed to make, and keep, the software shareable and useful to large numbers of scientists, such as providing documentation, considering parameters outside the immediate interest of the sponsoring lab and the constant drip of keeping up to date with software dependencies. Such work brings the software from a single lab and renders it scientific infrastructure, providing a platform of shared capabilities that not only makes the science of a field more efficient, but also more mutually intelligible, driving forward scientific discovery. Without this work, software (if it comes to be used outside a specific lab at all) fractures, growing further apart, ossifying; losing the opportunity of supporting coherence in a scientific field.

In the case of Enzo, this generalizing and coordinating work was limited for a long time. In this period the software was widely used, but developed further in a fragmented fashion. At the top level there was the original distribution and a small number of labs actively developing their own forked versions. But even within these labs individuals very had their own personal forks, following their own research questions and frontier. Thus much work was done on the code but little was shared; the originating lab made releases but did not actively seek external contributions and did not integrate code from the outside.

This period continued until the watershed of transition to community-support. The major users, many of whom had once been post-docs in the supporting lab, came together in what they referred to as “the week of code.” Here they merged the lines of code that had diverged over time, shaking out their practices of collaboration, including technological infrastructure of code sharing and merging, as well as collective decision-making, providing the base for a joint community-supported project. The transition to community support raised many difficult questions for the participants: Who should fund the generalizing, coordination, and stewardship work? What source code technologies would be used? What publication should users of the software now cite? Would it be fair to the community if particular participants received ongoing grant money, while others did not? What was the ongoing role of the project founder and how could he (and the community) manage the surprisingly emotionally taxing process of the PI “letting go” of the software, yielding to community input and shared leadership.

Enzo, then, has undergone the process of interest in this project: Enzo is a shared project that not only reduces the time and effort needed to sustain the code, but provides growing coherence to a field of research inquiry. It is an example of the vision of cyberinfrastructure: because software is information it has low costs of reproduction and high potential for re-use and recombination meaning that small initial funding investments can lead to coalescence into widely used software platforms, resulting in widespread, long-lived, impact in the form of better science (Atkins, 2003; NSF, 2012; C. A. Stewart, Almes, & Wheeler, 2010).

3 Literature review

As scholars we must ask: what body of literature and theory captures the essence of the process that Enzo underwent? In particular, to what might we point to explain the success of Enzo in the face of failure (or even lack of attempt) in many other emergent infrastructures? As I will explain below, while there are excellent foundations for such work, there is not yet a scholarly literature addressing these questions directly.

We know a growing amount about infrastructure (including information infrastructures) (e.g., Star & Ruhleder, 1996), we know a good deal about the specifics of the academic environment, including and academic career paths (e.g., Owen-Smith & Powell, 2001), and the importance of specific kinds of reputation (e.g., Birnholtz, 2008). We know an increasing amount about technology-supported scientific

collaboration (e.g., Olson, Zimmerman, Bos, & Wulf, 2008), including emerging literature on mass-collaboration through citizen science (e.g., Wiggins & Crowston, 2011), and the nature of software work in science (e.g., Bietz et al., 2010). We also know a good deal about peer production outside science (e.g., Benkler, 2002), including the organization of successful open source software projects (e.g., Crowston, Wei, Howison, & Wiggins, 2012) and contribution and social capital in online communities (e.g., Ellison, Steinfield, & Lampe, 2011; Kraut & Resnick, 2012). From the literature on entrepreneurship and technology transfer, we know a good deal about starting ventures (e.g., Bozeman, 2000), and from literature in management we know a good deal about organizational change, including its difficulty (e.g., Armenakis & Bedeian, 1999; Feldman, 2000), although surprisingly little is known about change in the specific context of science, other than promising, but passing, mentions in ethnographies (e.g., Knorr-Cetina, 1999).

My project, then, seeks to bring this research together, building cross-disciplinary knowledge relevant to practical problems in Science Policy (Jackson, Steinhardt, & Buyuktur, 2013), as well as deepening theory in these areas by exploring its usefulness and completeness in a novel area.

Below I present my overall research design. First, however, I present an extended justification for the research by drawing on the bodies of literature above to examining the differences between software development occurring in a grant funded environment and that occurring in a peer production environment, emphasizing the challenges that a project in transition must overcome.

3.1 Differences between grant-funding and peer production

The way in which a project attracts and retains its resources has a strong impact on its overall organization, impacting the motivations of participants, project governance, collaboration technologies, and contribution processes.

Grant-funded projects obtain their resources from science funders, government agencies like the NSF or non-profit foundations like the Sloan foundation. Grant money is akin to investment capital: it is made available with the hope of amplified future returns, but rather than financial profit returns are in more and better science. Agencies, guided by peer review, set aside a portion of their funds aimed at supporting science in general and invest them in supporting software work relevant to science. Grants are transfers of funds to projects that are then converted to software work by providing rewarding opportunities for potential participants. This is particularly clear when projects pay staff or students directly for software work but of similar importance are opportunities for activities resulting in scientifically valuable reputation, such as being among the authors of “software papers” (Bietz et al., 2010; Howison & Herbsleb, 2011).

Open source peer production attracts resources differently. Despite the common association, peer-production ought to be distinguished from “open source” per se. Being open source is a characteristic of the code, while peer production is a characteristic of how it is produced (Benkler, 2002; von Hippel & von Krogh, 2003). It is possible to be open source but not to be produced through peer production, indeed many grant-funded and commercial projects are. On the other hand, virtually all peer production is open source. Participants in peer production are very rarely paid directly for their software work: they have different and diverse motivations. The literature on motivation to participate in open source has identified a set of non-monetary motivations, from the use value of the software itself, an opportunity to build reputation for career advancement, an opportunity for learning, to a chance to express a communitarian ideology and to work in teams (e.g., Crowston et al., 2012; Roberts, Hann, & Slaughter, 2006; K. J. Stewart & Gosain, 2006; Wagstrom, Herbsleb, Kraut, & Mockus, 2010). In science, it seems likely that use value for conducting science and scientific reputation, as expressed in publications, is a key non-monetary motivation for software work (Bietz et al., 2012; Howison & Herbsleb, 2011). Either way, resources (in the form of direct labor) are attracted to projects that provide circumstances in which participant’s motivations can be satisfied (Benkler, 2002; Howison & Crowston, 2014; Ke & Zhang, 2010; Michlmayr, 2003).

3.1.1 Governance

The source of resources affects the governance of projects, by which I mean how decisions are made, such as what features to implement. Grant-funded projects have a formal hierarchical structure, derived from the accountability requirements of funding agencies: PIs are ultimately responsible for the project. Certainly the traditions of science mean that management is not dictatorial, reflecting traditions of “gentle persuasion” in accomplishing joint work (Knorr-Cetina, 1999). On the other hand, relationships within

labs, between PIs, post-docs and students, can be quite hierarchical, especially in some fields (Stephan, 2012). While project leaders listen and argument is likely based on intellectual merits, decisions flow from the top.

By contrast, decision-making in peer-production is distributed and shared (Fielding, 1999; Shah, 2006). Since participants are not in an employment relationship they decide for themselves what to build and how to build it. Even in projects that have a "benevolent dictator," such as Linux, Perl or Python governance is ultimately shared. This is because the "dictator" retains only the ability to choose between the work that is offered by participants, not to tell participants what to do.

There are also differences in the relationship to user communities. In a grant-funding situation it is clear that users have legitimacy to make requests of the development team, because the grant is predicated on the value of assisting users in order to advance science. In true peer-production, however, the legitimacy of users is more nuanced, reflecting how user interests intersect with those of the developers and the extent to which users are likely to provide additional resources to the project (either as developers or supporting other users). To the extent that developers are motivated by use-value and not reputation, users who are not contributing resources to the project may have little legitimacy (Terry, Kay, & Lafreniere, 2010).

3.1.2 Collaboration infrastructures

Grant-funded software projects expect to build and deliver software to their user communities, whereas peer production software projects expect to attract those who are going to build the software. This underlies the choice of collaboration infrastructure, by which I mean technologies such as source code control, mailing lists, forums, issue trackers as well as social features such as whether a team is co-located or not. There has been more careful study of collaboration infrastructures in peer production environments (e.g., Crowston & Howison, 2005; de Souza, Froehlich, & Dourish, 2005; Ducheneaut, 2005; Robbins, 2002; Shaikh & Cornford, 2003) than of those used in grant-funded cyberinfrastructure development projects (as opposed to the domain-science focused "collaboratories" studied by Olson et al. (2008). The typical setup of peer production projects is that everything, source code repositories, communication venues and issue trackers, is set to open. Moreover, practices have developed to ensure that those around the world have time to weigh-in on discussions, such as waiting 24 hours for a vote to complete.

While future empirical work would be needed to speak with more certainty, in the course of my work on software in science, I have examined the websites of many scientific software projects (and interviewed their producers). It is almost universal to make the application and source code available, and some (though not all) projects have user mailing lists or forums. Very few have publicly accessible developer mailing lists or source code repositories. Tracker systems are often framed as "consulting" tickets and users cannot see other users' tickets by default. Finally, when grant-funded developers are usually co-located, discussion takes place in face-to-face meetings, if they are at distance, teleconferences are typical, requiring invitations.

The result of these differences is that open source peer production exhibits "actionable transparency" (Colfer & Baldwin, 2010), while grant funded projects typically do not. Potential contributors, or users seeking customization, can see enough of the project that the user can take action. They can see not only the source code, but how the project works, how contributions are expected, discussed and decided upon. In this way, potential contributors in peer production are able to judge whether and how to act, creating a bias towards self-help and contribution; a quite different situation than in grant-funded projects.

3.1.3 Contribution processes

The contribution process—the manner in which work is planned and executed—differs between grant-funded projects and peer production. Grant-funded projects have relative certainty about the resources they will have available going forward because they are able to employ developers on contracts. Certainty facilitates using project management techniques to create a task breakdown and roadmap, sequencing tasks to achieve goals (shorter or longer sequences depending on how influenced a project is by agile software methods). A work breakdown can allow for long-term planned interdependence between participants.

Peer production projects, in contrast, have little certainty about the availability of future resources. While projects may create overall roadmaps they are much more tentative. In fact empirical results, both participant observation and archival reconstructions of work, from studies of community-based open

source show that the majority of tasks undertaken are short and rarely have more than one programmer involved (Howison & Crowston, 2014). Rather than proceeding through planned interdependence, projects are more likely to defer complex tasks that would require risky interdependence. Through a process we have characterized as "collaboration through open superposition," small motivationally independent layers build on existing code over time. Surprisingly, through the modular services these layers provide, small, layered, contributions can eventually make previously complex tasks simple enough that they can be accomplished without long-term interdependent planning. This process is illustrated in Figure 2.

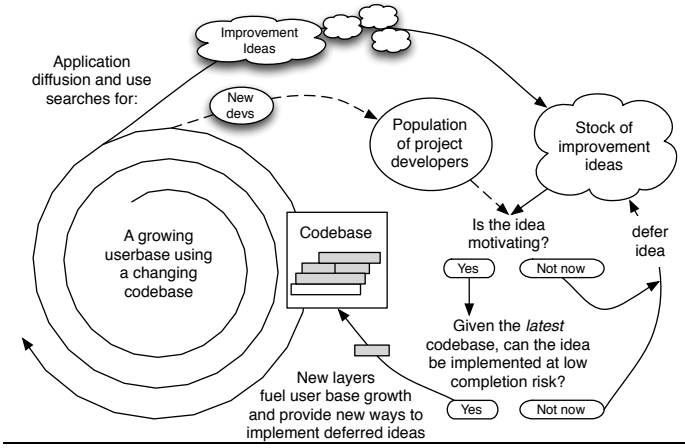

Figure 2: Collaboration through Open Superposition (source: Howison & Crowston, 2014)

3.1.4 Service center vs. Base for community

In summary, grant-funded projects establish themselves as service centers: they centralize work among grant-funded participants and attempt to reduce the amount of work users have to do. The argument that grant-funded projects make for long-term support is that of a public goods provider: projects have to serve a community well enough that the community agrees that continuing to fund the project is more worthwhile than funding other projects or domain science directly. As long as the grant funds are available this can be a sustainable model. Yet for many projects it is impossible to achieve this level of importance in the three to five years offered by typical initial grants.

In contrast, open source peer production implies that the project establishes itself as a base for community, creating the circumstances by which users can make contributions. The core members of the project act less like employees executing work, and more like stewards encouraging others and building community (Davis, Schoorman, & Donaldson, 1997). The actions needed to effect change can be counter-intuitive. For example, consider a project receiving a bug report. Assuming that the project has sufficient time, a service center would move to resolve the bug themselves, whereas an open community steward would work to make it as easy as possible for the user to solve the bug themselves before assisting directly. This is what Raymond (1998) meant by leaving "low-hanging fruit," leaving rough edges on a project to encourage outside participation.

4 Research Design

My research plan is to undertake in-depth case studies of transition (or attempted transitions) from grant funding to peer production, resulting in accessible and systematic narratives and a theoretical framework that will highlight key conditions and factors for successful transitions. The overall research question is: How did the project build a peer production community and why did it work?

Answering this involves three specific research questions:

1. What actions were taken to change the project?
2. How did routines in the project change as a result?
3. What conditions are relevant to the success of those actions in causing change?

I have chosen a qualitative approach and an in-depth multiple case study of a moderate number of cases for specific reasons. A qualitative approach is appropriate when the research task is to discover relevant factors and contextual conditions, rather than understand their magnitude (Eisenhardt, 1989; Eisenhardt & Graebner, 2007; Yin, 1994). A qualitative approach is also appropriate when looking to build theory rather than test it and creates results that are well suited for practitioners to relate to and contextualize their own experience (e.g., Greenhalgh, 2007).

The cases are being selected according to two criteria: theoretical sampling and participant engagement. Theoretical sampling ensures variety across a set of conditions anticipated to be relevant (Weick, 2007). The goal is not empirical representativeness as in random sampling but representativeness of relevant contextual conditions. My definition of sustainability is having sufficient resources available to undertake the work that is necessary to maintain scientific usefulness. In Howison and Herbsleb (2014) we outlined six different kinds of work: requirements, development, integration, releasing, user support and synchronization work (work that matches changes in other components). We argued that different ecosystem positions have different requirements for these different kinds of work. The ecosystem position of a piece of software has two aspects, the number of users and the diversity of usage contexts, especially the diversity of other components that a piece of software is used with. Projects that have more users, for example, must undertake more user support work, while projects with more diverse use contexts will have to undertake more synchronization work. The dimensions are not entirely independent, since very large numbers of users implies greater diversity in use contexts.

Figure 2 shows user numbers on the vertical and use context diversity on the horizontal and locates the projects that I will study initially. Since peer production closely links the motivations of participants and the type of work they are willing to do, it is reasonable to expect interesting differences in conditions and key factors for transition across these dimensions. Accordingly, I am selecting cases to cover the landscape of ecosystem position: number of users (high/low) and diversity of use contexts (high/low). For each situation I will study a minimum of two cases, which will help to discipline the analysis through replication to be sure that findings are not idiosyncratic to a single project (Eisenhardt, 1991).

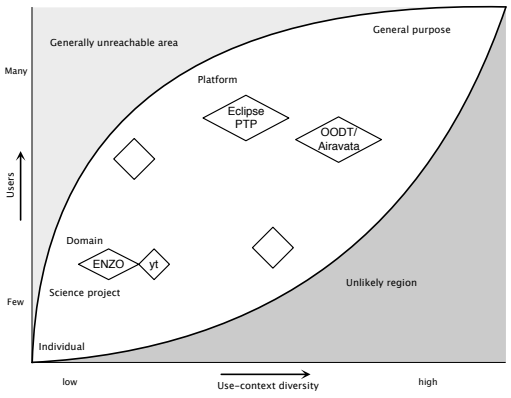

Figure 2: Dimensions of ecosystem position for theoretical sampling. Empty diamonds show future cases, to be drawn from NSF SI2 funded projects.

Participant engagement is the second criteria for case selection. Constructing rich narratives of change requires insight into the frame of mind, intentions and knowledge of participants over time. The strategy of artifact-supported interviews, described below, can be time-consuming for participants. Further the research requires access to multiple interviewees from a project and its user community.

My projects for initial study are Enzo, yt, Eclipse PTP, IRODS, and Apache OODT & Airavata. The projects range from astronomy (Enzo and yt), supercomputing (Eclipse PTP), data management middleware (IRODS) and scientific workflow management (Airavata). These initial projects were all grant funded by the NSF and have either undergone a successful transition (Enzo, yt, OODT and Airavata) or are currently seeking to transition (Eclipse PTP and IRODS). IRODS is particular interesting because they are explicitly considering models other than community-support.

A key challenge in studies of this kind is identifying and obtaining access to projects that did not transition to community support, despite intending to. These situations can be awkward, since people do not like to revisit what didn't work, especially in contexts that reflect on a person's profession. Nonetheless, engaged participants are strongly curious as to why their plans didn't work out and will participate if rapport has been established. To this end I intend to select a panel of projects from the NSFs SI2 program to complement my initial cases and to follow their efforts over time. In this way I will come in contact with participants as they contemplate change, allowing me to follow as changes unfold, both successfully and unsuccessfully.

4.1 Data collection approach

I am collecting data for my case studies through semi-structured interviews, supported by data from project archives and analysis of project repositories. While it is impossible to know in advance, each case will likely involve 15-25 interviews, across participants, users, and funders. In addition I am drawing on publications where appropriate. The overall objective is to put together a narrative that describes a project and its routines over time, as it transitions from initial grant funding and introduces peer production.

Semi-structured interviews are following an interview protocol tailored to the particular interviewee's involvement with a project and based on the organizational features identified in the literature review above. For example, for a project founder, the interview would begin by understanding when, why and how the project came into being. For a user of a project's software, the interview would seek to understand the user's scientific workflow, how the user came to know of and to use the project's software. In each case I am bolstering the interviewee's memory by using artifacts from published papers and project repositories, such as a CVS checkin or email list thread. These artifacts serve a dual purpose: they help to situate the interviewee in the relevant time period and they provide a logical structure for the interviewee to explain.

A project's repositories provide a second major data source. For example, as has been done in studying open source (Howison & Crowston, 2014), it is possible to reconstruct some of the activity in a project by pulling together evidence from the source code repository, bug-trackers, release notes and other project venues. I can also identify whether participants were grant funded at a particular time and see, for example, what type of work was done by "insiders" and what, if any, was done by "outsiders," producing time series that can help to identify important project events (Ducheneaut, 2005).

4.2 Data analysis

Analyzing qualitative data requires techniques to discipline the analyst's imagination and subject interpretations to validity checking (Weick, 1989). First I create memos that describe my interpretations of the interviews, focusing particularly on four elements that give structure to my analysis. The first is the analytic framework described above: motivations of participants, project governance, collaboration technologies, and contribution processes. The second is narrative cohesion, relying on the necessary logic of sequence: that some actions must logically precede others (e.g., writing code must proceed its review, or a grant must be awarded before it funds participants) (see Pentland, 1999). As simple as this seems, the necessary structure of narrative helps a great deal in disciplining the analysis. Third, a focus on changing routines creates a systematic ground for comparison before and after changes (Becker, Lazaric, Nelson, & Winter, 2005; Feldman, 2000; Pentland & Feldman, 2005). Further, multiple interviews give multiple sides of a story, allowing cohesion checks across interviews and time-stamped repository data.

Second, I will discipline my interpretations by returning to interviewees to engage in what is known as "member checking." This allows interviewees to hear how our interpretative work is unfolding and to provide clarifications, feedback, corrections or suggestions on how to gain addition insight into relevant issues. The responsibility of interpretation remains with the author, and care will be taken to consider issues of confidentiality between interviewees, where it is requested.

The expected outcomes of my research plan are:

1. Accessible and systematic case study narratives of transitions, and attempted transitions, to peer production.
2. A theoretical framework that explains when transitions to community support are appropriate (and when they are not), together with identifying actions which are effective for transitions between initial grant funding and peer production.

5 Conclusion

In this work-in-progress paper, I have argued that there is a class of infrastructure relevant to science that is important to sustain long-term but falls outside the traditional discourse of discovery through science funding and long-term sustenance through technology transfer to the commercial sector or congressional appropriation. These infrastructures, of which scientific software is a key example, are candidates for long-term community-support. Community support draws together resources from diverse sources and continues a project outside of direct core grant support, in a manner inspired by open source software development.

But, as I hope I have argued successfully, grant funded research and peer production are quite different in their organization, requiring intentional organizational change to effect successful transitions. If we can further understand this process of change, including when it is and is not appropriate, we can advance the science of science policy, the study of the organization of science, provide additional context for theories of peer production, and provide actionable insights for cyberinfrastructure projects.

Information technologies increasingly make new ways of working possible, including significant opportunities to draw community energy into great, sustainable, projects. Yet there is no magic in releasing something as "open source." Without a firm understanding of how to manage and effect organizational change in science, the opportunities to advance our infrastructures of knowledge will be missed.

References

Armenakis, A. A., & Bedeian, A. G. (1999). Organizational Change: A Review of Theory and Research in the 1990s. *Journal of Management*, *25*(3), 293–315. doi:10.1177/014920639902500303

Atkins, D. (2003). *Report of the National Science Foundation Blue-Ribbon Advisory Panel on Cyberinfrastructure*. Retrieved from http://www.nsf.gov/od/oci/reports/toc.jsp

Becker, M. C., Lazaric, N., Nelson, R. R., & Winter, S. G. (2005). Applying organizational routines in understanding organizational change. *Industrial and Corporate Change*, *14*(5), -791.

Benkler, Y. (2002). Coase's Penguin, or, Linux and The Nature of the Firm. *Yale Law Journal*, *112*, 369–446.

Bietz, M. J., Baumer, E. P., & Lee, C. P. (2010). Synergizing in Cyberinfrastructure Development. *Computer Supported Cooperative Work*, *19*(3-4), 245–281. doi:10.1007/s10606-010-9114-y

Bietz, M. J., Ferro, T., & Lee, C. P. (2012). Sustaining the development of cyberinfrastructure: an organization adapting to change. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work* (pp. 901–910). New York, NY, USA: ACM. doi:10.1145/2145204.2145339

Birnholtz, J. (2008). When Authorship Isn't Enough: Lessons from CERN on the Implications of Formal and Informal Credit Attribution Mechanisms in Collaborative Research. *Journal of Electronic Publishing*, *11*(1).

Bozeman, B. (2000). Technology transfer and public policy: a review of research and theory. *Research Policy*, *29*(4), 627–655.

Colfer, L., & Baldwin, C. Y. (2010). *The Mirroring Hypothesis: Theory, Evidence and Exceptions* (Working Paper No. 10-058). Harvard Business School Finance.

Crowston, K., & Howison, J. (2005). The social structure of Open Source Software development teams. *First Monday*, *10*(2). Retrieved from http://firstmonday.org/issues/issue10_2/crowston/index.html

Crowston, K., Wei, K., Howison, J., & Wiggins, A. (2012). Free (Libre) Open Source Software Development: What We Know and What We Do Not Know. *ACM Computing Surveys*, *44*(2), Article 7.

Dalle, J.-M., & Rousseau, G. (2004). Toward Collaborative Open-Source Technology Transfer. In *Proceedings of the ICSE 4th Workshop on Open Source*.

Davis, J. H., Schoorman, F. D., & Donaldson, L. (1997). Toward a Stewardship Theory of Management. *The Academy of Management Review*, *22*(1), 20–47. doi:10.2307/259223

De Souza, C., Froehlich, J., & Dourish, P. (2005). Seeking the source: Software source code as a social and technical artifact. In *Proceedings of GROUP '05* (p. -206).

Ducheneaut, N. (2005). Socialization in an Open Source Software Community: A Socio-Technical Analysis. *Computer Supported Cooperative Work (CSCW)*, *14*(4), -368.

Edwards, P. N. (2010). *A vast machine computer models, climate data, and the politics of global warming*. Cambridge, Mass.: MIT Press. Retrieved from http://site.ebrary.com/id/10424687

Edwards, P. N., Jackson, S. J., Chalmers, M. K., Bowker, G. C., Borgman, C. L., Ribes, D., … Calvert, S. (2013). *Knowledge Infrastructures: Intellectual Frameworks and Research Challenges* (Working Paper). Retrieved from http://deepblue.lib.umich.edu/handle/2027.42/97552

Eisenhardt, K. M. (1989). Building Theories from Case Study Research. *The Academy of Management Review*, *14*(4), 550.

Eisenhardt, K. M. (1991). Better Stories and Better Constructs: The Case for Rigor and Comparative Logic. *The Academy of Management Review*, *16*(3), 620–627. doi:10.2307/258921

Eisenhardt, K. M., & Graebner, M. E. (2007). Theory Building from Cases: Opportunities and Challenges. *The Academy of Management Journal*, *50*(1), 25–32. doi:10.2307/20159839

Ellison, N. B., Steinfield, C., & Lampe, C. (2011). Connection strategies: Social capital implications of Facebook-enabled communication practices. *New Media & Society*, 1461444810385389.

Feldman, M. S. (2000). Organizational routines as a source of continuous change. *Organization Science*, *11*(6), -629.

Fielding, R. T. (1999). Shared leadership in the Apache project. *Association for Computing Machinery. Communications of the ACM*, *42*(4). Retrieved from http://proquest.umi.com/pqdweb?did=894760711\\&Fmt=7\\&clientId=3739\\&RQT=309\\&VName=PQD

Greenhalgh, A. M. (2007). Case Method Teaching as Science and Art A Metaphoric Approach and Curricular Application. *Journal of Management Education*, *31*(2), 181–194. doi:10.1177/1052562906291306

Howison, J., & Crowston, K. (2014). Collaboration through open superposition: A theory of the open source way. *MIS Quarterly*, *38*(1), 29–50.

Howison, J., & Herbsleb, J. D. (2014). *The sustainability of scientific software production.* Working Paper, University of Texas at Austin.

Howison, J., & Herbsleb, J. D. (2011). Scientific software production and collaboration. In *Proceedings of the Conference on Computer Supported Cooperative Work (CSCW)*. Hangzhou, China.

Howison, J., & Herbsleb, J. D. (2013). Incentives and integration in scientific software production. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work* (pp. 459–470). San Antonio, TX. doi:10.1145/2441776.2441828

Ince, D. C., Hatton, L., & Graham-Cumming, J. (2012). The case for open computer programs. *Nature*, *482*(7386), 485–488. doi:10.1038/nature10836

Jackson, S. J., Steinhardt, S. B., & Buyuktur, A. (2013). Why CSCW needs science policy (and vice versa). In *Proceedings of the 2013 conference on Computer supported cooperative work* (pp. 1113–1124). ACM.

Ke, W., & Zhang, P. (2010). Extrinsic Motivations in Open Source Software Development Efforts and The Moderating Effects of Satisfaction of Needs. *Journal of the Association for Information Systems*, *11*(12), Article 5.

Knorr-Cetina, K. (1999). *Epistemic Communities*. Cambridge, MA: Harvard Education Press.

Kraut, R. E., & Resnick, P. (2012). *Building Successful Online Communities: Evidence-Based Social Design*. The MIT Press.

Lee, C. P., Dourish, P., & Mark, G. (2006). The human infrastructure of cyberinfrastructure. In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work* (pp. 483–492). New York, NY, USA: ACM. doi:10.1145/1180875.1180950

McCullough, B. D., McGeary, K. A., & Harrison, T. D. (2006). Lessons from the JMCB Archive. *Journal of Money, Credit, and Banking*, *38*(4), 1093–1107.

Michlmayr, M. (2003). Quality and the Reliance on Individuals in Free Software Projects. In *Proceedings of the ICSE 3rd Workshop on Open Source*.

NSF. (2012). *A Vision and Strategy for Software for Science, Engineering, and Education: Cyberinfrastructure Framework for the 21st Century (CIF21)* (Dear Colleague Letter No. 12-113). The US National Science Foundation. Retrieved from http://www.nsf.gov/pubs/2012/nsf12113/nsf12113.htm

Olson, G. M., Zimmerman, A., Bos, N., & Wulf, W. (2008). *Scientific Collaboration on the Internet*. Cambridge, MA: The MIT Press.

O'Shea, B. W., Bryan, G., Bordner, J., Norman, M. L., Abel, T., Harkness, R., & Kritsuk, A. (2004). Introducing Enzo, an AMR Cosmology Application. *arXiv:astro-ph/0403044*. Retrieved from http://arxiv.org/abs/astro-ph/0403044

Owen-Smith, J., & Powell, W. W. (2001). Careers and contradictions: Faculty responses to the transformation of knowledge and its uses in the life sciences. *Research in the Sociology of Work*, *10*, 109–140.

Pentland, B. T. (1999). Building process theory with narrative: From description to explanation. *Academy of Management Review*, *24*(4), 711–724.

Pentland, B. T., & Feldman, M. S. (2005). Organizational routines as a unit of analysis. *Industrial and Corporate Change*, *14*(5), -815.

Raymond, E. S. (1998). The Cathedral and the Bazaar. *First Monday*, *3*(3). Retrieved from http://www.firstmonday.org/issues/issue3_3/raymond/index.html

Ribes, D., & Finholt, T. A. (2007). Planning infrastructure for the long-term: Learning from cases in the natural sciences. In *Proceedings of the Third International Conference on e-Social Science*. Retrieved from http://davidribes.com/docs/RibesFinholt-LearningFromEScience(submitted).pdf

Robbins, J. E. (2002). Adopting OSS Methods by Adopting OSS Tools. In *Proceedings of the ICSE 2nd Workshop on Open Source*. Retrieved from http://opensource.ucc.ie/icse2002/Robbins.pdf

Roberts, J. A., Hann, I.-H., & Slaughter, S. A. (2006). Understanding the Motivations, Participation, and Performance of Open Source Software Developers: A Longitudinal Study of the Apache Projects. *Management Science*, *52*(7), 999.

Segal, J. (2009). Software Development Cultures and Cooperation Problems: A Field Study of the Early Stages of Development of Software for a Scientific Community. *Computer Supported Cooperative Work (CSCW)*, *18*(5), -606. doi:10.1007/s10606-009-9096-9

Shah, S. K. (2006). Motivation, governance, and the viability of hybrid forms in open source software development. *Management Science*, *52*(7), 1000–1014.

Shaikh, M., & Cornford, T. (2003). Version Management Tools: CVS to BK in the Linux Kernel. In *Proceedings of the ICSE 3rd Workshop on Open Source*.

Siegel, D. S., Waldman, D., & Link, A. (2003). Assessing the impact of organizational practices on the relative productivity of university technology transfer offices: an exploratory study. *Research Policy*, *32*(1), 27–48.

Star, S. L., & Ruhleder, K. (1996). Steps Toward an Ecology of Infrastructure: Design and Access for Large Information Spaces. *Information Systems Research*, *7*(1), 111 –134. doi:10.1287/isre.7.1.111

Stephan, P. (2012). *How Economics Shapes Science* (1st ed.). Harvard University Press.

Stewart, C. A., Almes, G. T., & Wheeler, B. C. (Eds.). (2010). NSF Cyberinfrastructure Software Sustainability and Reusability Workshop Report. Retrieved from http://hdl.handle.net/2022/6701

Stewart, K. J., & Gosain, S. (2006). The Impact of Ideology on Effectiveness in Open Source Software Development Teams. *MIS Quarterly*, *30*(2), 291–314.

Teece, D. J. (1986). Profiting from technological innovation: Implications for integration, collaboration, licensing and public policy. *Research Policy*, *15*(6), 285–305.

Terry, M., Kay, M., & Lafreniere, B. (2010). Perceptions and practices of usability in the free/open source software (FoSS) community. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 999–1008). New York, NY, USA: ACM. doi:10.1145/1753326.1753476

The Enzo Collaboration, Bryan, G. L., Norman, M. L., O'Shea, B. W., Abel, T., Wise, J. H., … Li, Y. (2013). *Enzo: An Adaptive Mesh Refinement Code for Astrophysics* (arXiv e-print No. 1307.2265). Retrieved from http://arxiv.org/abs/1307.2265

Von Hippel, E., & von Krogh, G. (2003). Open Source Software and the `Private-Collective' Innovation Model: Issues for Organization Science. *Organization Science*, *14*(2), 209–223.

Wagstrom, P., Herbsleb, J. D., Kraut, R. E., & Mockus, A. (2010, August 6). *The Impact of Commercial Organizations on Volunteer Participation in an Online Community*. Presented at the Academy of Management Conference (OCIS Division), Montréal, Canada. Retrieved from http://www.casos.cs.cmu.edu/publications/papers/2010The%20ImpactOfCommericial.pdf

Weick, K. E. (1989). Theory construction as disciplined imagination. *Academy of Management Review*, *14*(4), 516–531.

Weick, K. E. (2007). The generative properties of richness. *Academy of Management Journal*, *50*(1), 14–19.

Wiggins, A., & Crowston, K. (2011). From Conservation to Crowdsourcing: A Typology of Citizen Science. In *2011 44th Hawaii International Conference on System Sciences (HICSS)* (pp. 1–10). doi:10.1109/HICSS.2011.207

Yin, R. (1994). *Case study research: Design and methods* (2nd ed.). Newbury Park, CA.: Sage.