SIEPR Discussion Paper No. 07-22

Dynamics of Innovation in an "Open Source" Collaboration Environment

Lurking, Laboring and Launching FLOSS Projects on SourceForge

By
Paul A. David
Stanford University
And

Francesco Rullani
Copenhagen Business School

December 2007

Stanford Institute for Economic Policy Research
Stanford University
Stanford, CA 94305
(650) 725-1874

Dynamics of Innovation in an "Open Source" Collaboration Environment:

Lurking, Laboring and Launching FLOSS Projects on SourceForge

Paul A. David
Stanford University & Oxford Internet Institute
pad@stanford.edu

Francesco Rullani
Copenhagen Business School
fr.ino@cbs.dk

Version 8: 9 December, 2007

Accepted for publication in *INDUSTRIAL AND CORPORATE CHANGE*

Abstract

A systems analysis perspective is adopted to examine the critical properties of the Free/Libre/Open Source Software (FLOSS) mode of innovation, as reflected on the SourceForge platform (*SF.net*). This approach re-scales March's (1991) framework and applies it to characterize the "innovation system" of a "distributed organization" of interacting agents in a virtual collaboration environment, rather than to innovation within a firm. March (1991) views the process of innovation at the organizational level as the coupling of sub-processes of exploration and exploitation. Correspondingly, the innovation system of the virtual collaboration environment represented by *SF.net* is an emergent property of two "coupled" processes: one involves the interactions among agents searching the locale for information and knowledge resources to use in designing novel software products (i.e., *exploration*), and the other involves the mobilization of individuals' capabilities for application in the software development projects that become established on the platform (i.e., *exploitation*). The micro-dynamics of this system are studied empirically by constructing transition probability matrices representing the movements of 222,835 *SF.net* users among 7 different activity states, which range from "lurking" (not contributing or contributing to projects without becoming a member) to "laboring" (joining one or more projects as members), and to "launching" (founding one or more projects) within each successive 6-month interval. The estimated probabilities are found to form first-order Markov chains describing *ergodic* processes. This makes it possible the computation of the equilibrium distribution of agents among the states, thereby suppressing transient effects and revealing persisting patterns of project-joining and project-launching. The latter show the FLOSS innovation system on *SF.net* to be highly dissipative: a very large proportion of the registered "developers" fail to become even minimally active on the platform. There is nevertheless an active core of mobile project-joiners, and a (still smaller) core of project founders who persist in creating new projects. The structure of these groups' interactions (as displayed within the 3-year period examined) is investigated in detail, and it is shown that it would be sufficient to sustain both the exploration and exploitation phases of the platform's global dynamics.

Corresponding author: *Francesco Rullani, INO, Dept. of Innovation and Organizational Economics, Copenhagen Business School, Kilevej 14A, 3rd floor, office 3.46, 2000 Frederiksberg, Denmark. Ph: +45 3815 2837, Fax: +45 3815 2540, email: fr.ino@cbs.dk, homepage:https://mail.sssup.it/~rullani*.

Acknowledgements

We are grateful to the staff of *SF.net* staff for providing a copy of the *SF.net* archive, and to Paola Giuri, Salvatore Torrisi, Matteo Ploner, Gaia Rocchetti for providing the "cleaned" SFnetDataset that was created by their project (with Rullani's participation). Interviews with *SF.net* developers were of a significant help in interpreting the data from the archive. Research for this paper has had the benefit of discussions with Laurent Daudet, Giacomo Pasini, Alessio Moneta, Alessandro Sapio, Randolph Bruno, Markus Becker and Toke Reichstein and from the comments by an anonymous referee. In addition, this version has been improved by other comments on earlier presentations, made at the OWLS-Paris Workshop (April, 2006), the OSSWatch Conference on Sustainability and Open Source Software in Oxford (April, 2006), the OSS2006 conference in Como (June, 2006), the International Schumpeter Society Conference in Nice (June, 2006) and the INO seminar at Copenhagen Business School (October, 2006). David gratefully acknowledges the financial support of his research by the National Science Foundation IIS/CISE Digital Technology and Society Program (under award IIS 0326529 to Stanford University) and the Stanford Institute for Economic Policy Research's KNIIP Program. Rullani gratefully acknowledges the financial support his work received from the CE, DGXII, FP V Project "Economic Change: the micro foundations of institutional and organisational change: *EconChange*"; from Fondazione IRI in the context of the program "Programma di perfezionamento all'estero in discipline manageriali"; and from the Laboratory of Economics and Management of the Sant'Anna School of Advanced Studies in Pisa, Italy.

Keywords: innovation systems, collaborative development environments, industrial districts, exploration and exploitation dynamics, open source software, FLOSS, SourceForge, project-joining, project-founding, Markov chain analysis.

1. Introduction: Motivation, Overview and Organization of the Study

A systems analysis perspective is adopted in this study of the critical properties of the Free/Libre/Open Source Software (FLOSS) mode of innovation, as that manifests itself on the SourceForge platform (*SF.net*) – a collaborative development environment that provides infrastructure facilities to host open source software development projects. We assess the capabilities of the ensemble of agents in that virtual locale, employing March's (1991) view of the process of innovation at the organizational level as the coupling of sub-processes of "exploration" and "exploitation." The conceptual novelty of the present approach, however, scales up March's well-known framework from the original context of a corporate entity, and applies it empirically to study the system of agents interaction on *SF.net* as a "distributed organization." The "innovation system" that is an emergent property of such environments involves the search for information and knowledge resources (in human capabilities) pertaining to opportunities for the creation of novel software products (i.e., *exploration*), as well as for the mobilization and utilization of the software development resources that are accessible to the ensemble of projects situated already in this representative "virtual industrial district." (*exploitation*).

The result of the analysis is that the FLOSS model exhibits the tendency to increase both the involvement of developers in the projects (exploitation of existing knowledge basis) and new project creation (exploration of new trajectories). These results assure a certain degree of sustainability to the model in terms it innovative performance, but have to be looked at from the perspective of another property emerging from the analysis: *dissipation*. The overall process is in some respects highly dissipative, inasmuch as the emergence of a flow of viable new projects entails the mobilization of the attention and efforts of many agents. This should not be really surprising, because, like biological evolution, social and economic innovation are strongly dissipative processes..

1.1 Motivation: the need for a "systems perspective" on the FLOSS mode of innovation

When a FLOSS project releases its code under an open source license, this enables developers who have not worked on that software to examine it, modify it, discover the source of its performance defects, and repair them, and (under some conditions that depend upon the nature of the license[1]) redistribute both the original and the versions that carry their modifications. Other developers may then enter the production process, cooperating with the project's founder and its members to further enlarge and enhance the reliability and functionality of the code basis. The cooperative process is open and "fluid": non-members of the project may contribute by submitting bug reports, patches and requests for particular features; project members are free to leave, just as new developers can be added at each moment to the project's team. Every developer decides how and how much to contribute. When authoritative structures of governance emerge, they typically formalize and render explicit the informal processes that were already at work in the network of collaborators (Mateos Garcia and Steinmueller, 2003). The multiplicity of projects, each having its own organizational history, generates a variety of "experiments" in different governance structures.

Similarly, decentralized mechanisms are at work in the relationships among distinct projects. Precisely because the development process is undertaken on an open and voluntary basis, coordination is not assured. On the supply-side, projects may overlap in some of the features they provide for users and developers.[2] On the demand side, projects may compete for resources – i.e. developers with specific competences or simply developers who are willing to report bugs, or able to submit "patches" that fix a problem. Moreover, new projects may be launched at every moment by anyone, and may then attract other

[1] On the distinctions between different Open Source licenses see, among others, Lerner and Tirole (2005), and Giuri *et al*. (2002).

[2] Consider, for example, one of the most relevant projects on *SF.net*: *phpMyAdmin* (awarded with the "2006 Community Choice Awards"). The homepage of the project describes it as "phpMyAdmin is a tool written in PHP intended to handle the administration of MySQL over the Web. Currently it can create and drop databases, create/drop/alter tables, delete/edit/add fields, execute any SQL statement, manage keys on fields." But other *SF.net* projects have very similar features: *remote mysql manager* is described as "PHP code to manage a MySQL db over the web, Add/Drop databases and tables; Insert/Update/Delete records. Enter your own SQL query. No config files/scripts. Small and simple." Similar duplicative software offerings can be found among other categories of projects hosted by *SF.net*. On a similar point see also Klincewicz (2005).

developers, forming a team around them; or they may remain individual undertakings that fulfill the specific needs of their founders.

The foregoing description of the FLOSS development process from a "bird's-eye perspective" makes it evident that to expose the capabilities of the FLOSS mode of innovation there is a need for "systems-level" analysis. At the macro level, the literature on "open source" software development has focused attention on the organizational attributes and performance achievements of projects belonging to the upper tail of the distributions of size (in numbers of contributing developers, lines of code) or activity (frequency of releases, email messaging rates).[3] The larger ecology of FLOSS projects has been only recently been taken into account by considering projects belonging to the whole support of project size and productivity distributions (e.g. Lerner and Tirole, 2005; Comino *et al*., 2005; Giuri et al. 2006a). At the "micro" level, many scholars have tried to asses what motivates developers to participate in the activities of the FLOSS community (e.g. Shah, 2006; Bagozzi and Dholakia, 2006). The two levels are seldom linked, however, which tends to leave unstudied the ways in which the macro-properties of the system emerge from the interactions among developers at the micro-level. The present work moves a step in the direction of such a systemic assessment of the FLOSS mode of innovation.

1.2 Crucial innovation capabilities of the FLOSS mode of innovation

In setting out this line of inquiry it is necessary to begin by asking what specific properties should be examined in order to judge the *innovation capabilities* of the FLOSS mode of development. The descriptive account already offered makes it clear that the "ecology of projects" we are dealing with can be conceived as a *system* composed by different elements interacting with one another. Thus, the innovation capabilities of the FLOSS development mode may be better understood by analogy to other systems of innovation that feature self-organized interactions among a multiplicity of entities that form "production projects" around fairly narrowly specified goods. The literature on the economics of "industrial districts" is useful in this regard. Indeed, an explicit parallel between the Italian industrial district and the FLOSS "mode of production," based on stylized descriptions of the two, has been drawn by Maggioni (2004).

Like software FLOSS projects, specialized firms can be successful and grow in terms of numbers of employees, or fail, so that entrepreneurs can decide to abandon their enterprises and re-enter an inactive state, can apply to be hired by other existing firms, or start a completely new enterprise. A closer examination of the commonalities between the two systems would not be without interest, but here it is only necessary to extent it serves our purposes to use the resemblance between our ecology of software projects and the Silicon Valley-like localizations of industry as a way of indicating how the *innovation capabilities* of the aggregation of production activities in a virtual environment may be conceptualized from a systems-analytical perspective.[4]

A first point is that the capacity to grow and sustain innovation in industrial clusters is mainly identified with the ability of the locale to activate entrepreneurial innovation. The Silicon Valley experience, the recent history of Italy's Northeastern region and other similar examples show that the vigor of "entrepreneurial spirits" constitutes one of the fundamental factors accounting for the spectacular industrial development and economic growth experienced by these regions (Saxenian, 1994; Becattini, 2001). According to Braunerhjelm and Feldman (2006: p. 7): "One prominent feature of cluster formation seems to be the importance of entrepreneurship as an endogenous process, and the emergence and sustainability of clusters seem to critically interact with entrepreneurial activities." Why is the creation of new ventures so crucial? As Castilla *et al*. (2000: p.223) argue, "Part of the importance of these spin-offs is that most organizations resist changing their core technologies and structures […]. Thus, upgrading of a regional economy occurs especially through new organizations rather than through transformation of existing ones. […] Any region whose institutions or networks resist spin-offs or new entrants may face stagnation."

---

[3] Among the pioneering studies that set this pattern, see, e.g. Mockus et al. 2000; Krishnamurthy, 2002; von Krogh, Spaeth and Lakhani, 2003; Lee and Cole, 2003.

[4] There are also striking parallels to be drawn between the open flows of information and mobility of developers among FLOSS projects in the collaboration environment created by SourceForge, and the detailed picture of the circulation of information and of expertise carried by the movements of engineers and entrepreneurs among firms in a high-tech industrial district or "cluster", as may be seen from Lécuyer's (2005) richly documented account of "the making of Silicon Valley" in the era before the 1970's.

Entrepreneurs are crucial for clusters and networks because they are the main drivers of experimentation and novelty. This perspective is also adopted by Olsson and Frey (2002: p. 70) in their representation of the entrepreneur: "We present a formal model where the body of contemporary technology is regarded as a set in a metric 'technology space'. In this technology space, ideas are separated by 'technological distance' and the ideas contained in the technology set form an infinite, bounded, closed and connected set. The entrepreneur, who expands the technology set by convex, binary combinations of existing ideas, plays the key role in this setup." Entrepreneurs are then conceived as actors expanding the boundaries of the established knowledge basis.

From a system perspective, and especially when referring to self-organized systems, this last point has to be framed more broadly, so that the whole set of involved actors are considered. As Anderson (1999: p. 222) maintains: "When we observe complex aggregate structures, such as multinational corporations or the economic web of Silicon Valley, we need not search for complex building blocks. A defining feature of complexity is that self-organization is a natural consequence of interactions between simple agents". In other words, founders of new ventures should not be considered as stand-alone actors, but need to be conceived as embedded in a social structure that connects all the agents into one emerging organization. In this context, the exploration that many founders undertake entering unknown regions of the technology landscape represents the exploration of that wider organization.

The same process is discussed by March's (1991) well-known article on exploration and exploitation in organizational learning, which identified a series of situations in which exploration, i.e. "things captured by terms such as search, variation, risk taking, experimentation, play, flexibility, discovery, innovation" (ibid.: p. 71), becomes the key factor in organizational learning and performance. One such case is represented by ecologies of projects (ibid: p.84):

> "Multiple, independent projects may have an advantage over a single, coordination effort [because the latter] can be expected to do better (on average) than those that are more loosely coupled, [but] they also probably can be expected to become more reliable, less likely to deviate significantly from their mean performance. The price of reliability, however, is a smaller chance of primacy among competitors."

In other words, one of the main reason why the high rate of formation of new ventures exhibited by certain regions and areas is correlated with more rapid growth is that those new ventures represent the mechanism through which the system explores promising parts of the technology-space and product market.

The conclusion emerging from this brief review of the literature is that to evaluate the innovation capabilities of a system such as the FLOSS mode of innovation, it will be necessary to focus on the rate of new venture creation -- conceived of as a source of *exploration* for the whole system. But, as March points out, exploration is just one of the processes an organization or a system needs to undertake to sustain its growth. On the one hand, it has to be able to produce novelty, exploring the new ideas and developing new technologies. On the other hand, and at the same time, it has to be able to allocate the resources it has access to *exploitation*, i.e. "such things as refinement, choice, production, efficiency, selection, implementation, execution." (March, 1991: p. 71). At the level of the industrial system this duality becomes visible when we consider the roles that different entities play in the technological advancement of an industry. In such a framework one can observe firms acting as explorers, i.e. searching "at the boundaries" of the established technological paradigm and thus representing the group of pioneers possibly inducing a paradigm shift, and firms moving further along the existing technological trajectory, i.e. exploiting the specificities of the established paradigm.[5]. Exploitation, in this case, means "investing" in the existing technology, while exploration takes the form of a shift towards a new set of perceptions, problems, trade-offs and solutions.

Not being tied to antecedent organizational structures and cognitive schemes, new firms can internalize more efficiently and effectively the new perspectives embodied in a new paradigm. Even when start-ups do not represent the source of a paradigm shift of the whole system, they are likely to become the actors through which the change is realized. This is a quite general point that has been emphasized in studies

[5] "We shall define a 'technological paradigm' broadly in accordance with the epistemological definition as an 'outlook', a set of procedures, a definition of the 'relevant' problems and of the specific knowledge related to their solution. We shall argue also that each 'technological paradigm' defines its own concept of 'progress' based on its specific technological and economic trade-offs. Then, we will call a 'technological trajectory' the direction of advance within a technological paradigm." Dosi, 1982, p. 148.

of the emergence of new industrial technologies. In this context Dosi (1982: p. 1) noted that even if "often in this century the production of major technological advances has been the result of organized R&D efforts as opposed to the 'inventiveness' of individuals….[T]his period of emergence of new technologies is actually characterized by newly emerging firms, even in cases when the major technological advances were originally produced in established firms and institutions. "

The relationship between exploitation and exploration is thus far more complex than it may appear to be at first sight. March (1991: p.71-72) explicitly suggests the problems a system faces when trying to balance exploration and exploitation:

> "Adaptive systems that engage in exploration to the exclusion of exploitation are likely to find that they suffer the costs of experimentation without gaining many of its benefits. They exhibit too many undeveloped new ideas and too little distinctive competence. Conversely, systems that engage in exploitation to the exclusion of exploration are likely to find themselves trapped in suboptimal stable equilibria. As a result, maintaining an appropriate balance between exploration and exploitation is a primary factor in system survival and prosperity….Finding an appropriate balance is made particularly difficult by the fact that the same issues occur at levels of a nested system – at the individual level, the organizational level, and the social system level."

Thus, on the one hand, both exploitation and exploration can be conceptualized as processes whose premises have to be renewed in each period of time and whose outcome is to a certain extent related to the invested *resources*; and, on the other hand, both processes have a *nested structure*, as they occur at the individual, the organizational and the system level, and those levels are mutually interdependent.

1.3 Overview and organization of the study

Examination of the empirical evidence relating to these two capabilities and their interactions can shed new light on the *sustainability* of the FLOSS development system as a mode of innovation. For the purposes of this study we are able to draw upon micro-level data pertaining to the activities undertaken during the period between the beginning of September 2000 and December 2002 by the entire cohort of 222,835 individuals who had registered on *SF.net* during the 14 months from September 1, 2000 through October 26, 2001. Using a statistical approach based on Markov chain models of the movements of developers among seven defined "activity levels", it is found that the FLOSS mode of innovation exhibits the tendency to increase both the involvement of developers in the existing projects and new projects creation. For reasons to be discussed in the following section, we take these two classes of activities as manifestations of organizational-level exploitation of existing technologies and exploration of new trajectories, respectively.

SF.net will thus be seen to be more than simply a site that attracts curious visitors or transient developers, and more than a platform that provides convenient facilities for those registrants who bring to it their project ideas, already written code, and already organized co-developers, all of which existed before they had any significant experience of interacting with others in that environment. Indeed, the environment generates a population of new projects launched by cohorts of registrants well after they arrived, and the size distribution attained by those projects resembles that of the projects that are in a sense brought to the platform by the newly arriving developers.

While our findings are consistent with a certain degree of sustainability in innovative performance, they are qualified by an obtrusive feature of the micro-dynamics of the FLOSS development process on the SourceForge platform: *dissipation*. The system is highly dissipative in at least two respects: firstly, only a minor fraction of those who register on the platform ever become minimally active contributors to its projects, let alone project members; secondly, only a small proportion of the active core of developers emerge as founders of new projects. In addition, a very high proportion of the projects launched on the platform, some two-thirds in all, had just one single developer in their project group at the end of the period of observations.

These results emerge from a study that is organized as follows. Section 2 re-phrases the research questions posed by the introduction in an explicit, operational form. It briefly takes note also of the relevant antecedent literature, highlighting some of the relevant conceptual frameworks and noting the respects in which the approach adopted here is able to escape from the limitations imposed on earlier empirical research.

In section 3 we provide a full account of the dataset employed in this study, its sources, accuracy and appropriateness for the research questions that we wish to pursue. The methodology of Markov chain analysis is described in section 4, where we describe the "activity" levels that are recognized in the seven "states" of the model for which transition probability estimates are computed from the micro-level observations of developers' behaviors. This section also discusses the conditions for establishing whether the resulting transition probability matrix describes a stationary and ergodic process, because the properties of the latter will be used in the analysis to remove the transient disturbances and thereby expose the underlying dynamics of forces operating in the *SF.net* environment. The estimated transition probabilities are presented in Section 5, where the implications of these results are discussed first in regard to what they reveal about the dissipative nature of the FLOSS innovation system, and secondly in regard to the stationarity and ergodicity of the Markov chain.

Section 6 begins the more detailed data analysis and discussion of the results obtained with our Markov model, focusing on project-joining and project-founding by the active population of the cohort of developers under examination. It presents a comparison of their initially observed distribution among the activity states and the corresponding limiting distribution to which the Markov chain describing the system converges. Section 7 explores the effect of heterogeneities within sub-groups of the projects' founders, showing how the propensity to launch new projects tends to be conditioned by the individual's prior actions as a founder. Section 8 moves the analysis from the examination of Markov chains defined for current activity states to the investigation of expected "career histories" showing the number of cumulated projects founded and joined by each developer over the observed period. The conclusion of the paper in section 9 summarizes the findings in terms of the light they shed on the emergent properties of the FLOSS mode of innovation as represented by the activity on the *SF.net* platform, and comments in particular on their broad implications for the question of the sustainability of the FLOSS system – rather than the issues of sustainability of particular projects. It briefly considers ways in which future research could transcend the limitations of this exploratory implementation of the Markovian approach to analyzing the population dynamics of ecologies of FLOSS developers and their projects.

2. Research questions, empirical setting and methodology

In order to articulate a suitable systems-level approach to assessing the "sustainability" of the FLOSS mode of innovation, the preceding introductory discussion drew upon the seminal ideas presented by March (1991) and other related contributions to the organizational science literature. The present study internalizes those contributions by adopting the following research strategies. First, the focus of analysis is placed on identifying and characterizing the performance of each of two different phases of the innovative process – exploration and exploitation, and on the balance achieved in the coupling between those two sub-processes. Secondly, the framework of analysis for both sub-processes serves to explicitly connect the micro- and macro-structures of the system. Thirdly, the conceptualization of "exploration" at the system level is associated empirically with the mobilization of resources around novel enterprises, whereas the observable micro-level activity of resources at the disposal of already existing enterprises is associated to the phase of "exploitation". Lastly, the resulting competition for resources between the two processes should be explicitly taken into account.

2.1 Operationalizing exploration, exploitation and the sustainability of innovation

How are we to operationally define the concepts of exploration and exploitation in the context of FLOSS development, so that an empirical study can proceed in accord with the foregoing conceptual approach? The systems perspective and the importance of explicitly connecting the micro- and macro-structures of the system suggest that we should begin by focusing on the activities of the constituent agents or actors (the developers) and move upwards to the level of their common enterprises (the projects), from which perspective the developers appear as mobile resources for which the different projects must compete if they are to survive and grow. Applying the analogy of the industrial district, when developers join existing teams they can be thought of as resources the community devotes to the "exploitation" phase, moving forward the knowledge basis of the community along an already established trajectory. On the other hand, and again referring to the district analogy, when new ventures appear in the form of newly created FLOSS projects, these represent the allocation of developers' resources to exploration "at the boundaries" of the knowledge basis already mastered by the ensemble of projects established in the virtual locale.

It is worth pointing out that the difference between FLOSS projects and firms reinforces this set of correspondences. Firms organize the relationship among a group of individuals and develop different products over time, but, by contrast, FLOSS projects self-organize around the implementation of one or another particular product idea that will be elaborated, extended and maintained by a fluid group of developers over the course of its life cycle. Within the framework as set out by March (1991), the activities of exploitation and exploration within an industry are seen as being carried on by each of the firms that constitute the industry, whereas, in almost all cases in the open source software world, developers willing to create an innovative product will launch a new project, and seek to mobilize a team of developers that will work to realize their idea.

The first implication to which this leads is that system-level, or community-level *exploitation* activities can be regarded as those associated with the mobilization and application of resources in existing projects, because an established project usually is tied to the development of a specific existing software product. When moving to consider community-level *exploration* activities, however, the argument cannot proceed so directly, because as has been noted previously, new FLOSS projects on *SF.net* are not necessarily *technologically* new. Some projects closely resemble existing ones at the point at which they acquire a distinct organizational identity. This occurs, for example, when new projects originate from the "forking" of existing projects, and the "spin-offs" start with basically the same code bases. It may be tempting to think of the exploration phase of software innovation as associated simply with the search for technical novelties in computer program architectures and the particular ways in which the latter are implemented in the code. Given such a view, the ubiquitous reuse of code, as well as of the phenomenon of "forking" in reaction to rejection of a proposed alteration in a project's design, likely lead one to conclude that the FLOSS development mode is very limited in its innovative capabilities.[6] But, it would be a mistake to limit the view new projects' exploratory significance to their possible broadening of the technological design space, and hence dismiss a "forked" project as uninteresting because a substantial proportion of the new entity's code-base contains the code of the initial project.

The potential for economically important software novelty may reside not in the technical modifications of the code *per se*, but in the latter having enabled the project to explore and possibly fill a new and growing market niche.[7] Innovation is not co-extensive with the introduction of technical novelties, and the reapplication of existing product designs and production methods in new market contexts was recognized by Schumpeter as entrepreneurial acts of innovation. Furthermore, even when new FLOSS projects initially resemble existing ones closely in both technology design and product concept, they tend to involve a substantially different group of people, with different skill sets and different conceptualizations of their product's future evolution. As Rosenkopf and Nerkar (2001) show, if organizations innovate through boundary-spanning, their exploration has a stronger impact on the evolution of the technology when the spanned boundary does not cross different technological domains but overcomes organizational boundaries. Knowledge coming from actors "outside the firm" is a source of diversity that can be exploited to escape the trap of "local search" and induce wider exploration even when the underlying technology is similar. The

[6] Klincewicz (2005) has concluded, on just such grounds, that the technological innovativeness of FLOSS projects is very low: "Decision making processes in many OSS projects are highly formalized, which additionally discourages new concepts. It is easier to fork the code, than convince project decision makers to implement certain ideas. An example could be Samba TNG (The Next Generation), forked from Samba (file sharing and printer server for Linux environment, able to communicate with Windows computers). When innovative architectural suggestions were opposed by Samba project leaders, their proponent had no other choice than to launch an independent project in 1999 (Weber 2004: 169). After 6 years of development, Samba TNG remains a niche project, not included in known Linux distributions nor supported by commercial vendors." (Ibid.:p.19).

[7] Recalling the example from the preceding footnote, in which Samba TNG, the "forked" version of the Samba project arose from the misalignment of the developers' visions of the projects, the difference between the two projects, indeed the very meaning of a fork, originated in the disparate architectural concepts that people wished to implement within Samba, and which implied different trajectories of exploration in the functional/product space. In this regard, whether the new (forked) project proved successful or ended up as a comparative failure, as a dormant occupant of a narrow product niche is irrelevant when referring to the capability of new projects to represent community-level exploration. Exploration is an inherently risky business.

[9] The role of diversity in enhancing the payoffs from search activities has been treated also from the perspective of cognitive psychology, where the value of exposure to different problem- framing and solution skills is emphasized (e.g., recently by Page, 2007). An earlier, search-theoretic approach to R&D economics focused on the implications of the

same logic applies generally to new ventures whether they take the form of new firms or new communities assembled around FLOSS projects. In all these cases people with diverse experiences, cognitive frameworks and sets of potential solutions gather together and share their knowledge to form new combinations.[9] The very fact of a project being *organizationally new*, connecting some previously non-interacting agents, tends to creates greater room for exploration even when the recombinant novelties emerge along an already established trajectory. Therefore, new projects, most plausibly are among the primary sites where new ideas actually will be implemented and elaborated. In short, the second proposition at which we have arrived holds that if the system's principal locus of "exploration" is to be found anywhere, then new project formation activity would be the obvious and best place to look for it.

The preceding considerations suggest that our analysis of the innovation capabilities of the FLOSS system should be directed to answering the following research questions:

1) *On exploitation*: Does the FLOSS system exhibit the tendency to increase the proportion of developers laboring in existing projects?

2) *On exploration*: Does the FLOSS system exhibit the tendency to increase the proportion of developers launching new projects?

3) *On sustainability*: What do the answers to the foregoing questions permit one to say about the sustainability of the FLOSS innovation mode that is displayed in the SourceForge ecology?

2.2 Empirical setting: the SourceForge environment

To observe an actual ecology of FLOSS projects the present study draws upon data pertaining to the individuals and projects that were observed on SourceForge.net (*SF.net*, at http://sourceforge.net) during the period from September 2000 to December 2002. *SF.net* is an on-line platform managed by SourceForge Inc., where FLOSS developers can meet and coherently organize their activities on different projects. *SF.net* is just one of the possible platforms developers can use to coordinate their work, being *FreshMeat* and *Savannah* notable sites that also host community mode open source software projects. Indeed, *SF.net* does not host the very largest and emblematic projects engaged in what Dalle and David (2005) label "*C*-mode" (for community-mode) production of FLOSS, many of which, like the Linux kernel or Apache, maintain their own sites. Nevertheless, as of 10th January 2003, the date of the last observations used in this study[10], 73 development teams on *SF.net* – including well-known projects like Freenet, BZFlag, Python, and JBoss[11] – had membership counts in the range from 30 to 102. These share many of the challenges of coordination, contributor recruitment and governance encountered by the largest *C*-mode entities, and, in that regard (as well as in the comparative salience of their products) are readily distinguished from the mass of FLOSS projects that were led and staffed by one or two individuals.[12] Moreover, *SF.net* is far and away the most densely populated in numbers of project groups and members: at May 9th, 2007, it hosts 147,905 projects and has 1,579,588 registered uses, against the 42,815 and 2,757 projects and the 383,282 and 50,267 registered users of *FreshMeat* and *Savannah*, respectively.

---

positive relationship between the second moment of a variate's distribution and the first moment of the corresponding extreme value distribution; this suggested external search as a means of increasing the variance of the underlying distribution of opportunities from which selections could be made for development research (see, e.g., David 1974).

[10] The whole set of available data span the period from the beginning of November 1999 to January the 10th, 2003. Our analysis focuses on the period from September 2000 to December 2002 because the specific variable we need are defined just in that interval.

[11] Notice that some projects host some features outside *SF.net*. For example, Python (at http://www.python.org/) and JBoss (at http://www.jboss.org/).

[12] The latter class of FLOSS projects are labeled *I-mode* ("independent") by Dalle and David (2005), who point out that the disclosure of source code and terms of licensing, rather than their organization structures and development methods, differentiate these projects from their counterparts among the proprietary software companies. FLOSS projects at that end of the size range can be created *ex post*, simply by a decision to distribute pre-existing conventional copyright-protected software packages under one or another of the licenses that comply with the "open source" definition.

Following the foundation of *SF.net* late in 1999, SourceForge Inc., collected and organized a data archive containing a chronological record of the activities of developers and projects hosted on the platform. On *SF.net* individuals can launch their own FLOSS projects, posting the relative code and benefiting from the tools the platform offers to manage the cooperative development of software (forums, mailing lists, code repositories, and so on). Other registered members can then join these founders and create ecology of teams working on FLOSS projects. Or they can simply remain "lurkers" and follow the activity of the hosted projects, only seldom sending some kind of contribution to the teams. As it is clear, *SF.net* constitutes an important microcosm of the FLOSS universe, in which a great variety of organizations are found to co-exist, some closely resembling one another, and others being very different in size, organizations structure, and purpose. Thus, it represents a perfect setting to undertake the system-level analysis described in the previous sections.

The portion of the dataset on which we rely contains observations on the activities undertaken during the period between the beginning of September 2000 and December 2002 by a cohort of 222,835 individuals that registered on *SF.net* during the 14 months from September 1, 2000 through October 26, 2001. We computed the mean probability that each one of these developers moved from a passive "lurking" state to more active states. Two main "active" states are considered, each one proxying one of the two processes we are interested in: exploration and exploitation. In particular, we detect if the developer has joined any project team (so that she contributes to the exploitation process undertaken by the community) or has founded new projects (exploring "at the boundaries" of the knowledge basis already mastered by the community) in a certain period of time. Since the probability of movements between a lurking state, a "member" state, or a "founder" state can be conceived as a 'transition probability', the properties of these transitions can be studied applying the statistical apparatus developed for the analysis of Markov processes. In what follows the estimation of transition probability matrices and the properties of Markov chains will be employed both descriptively and as a means of exposing the persisting micro-level patterns of project-joining and project-founding, and their relationship to the macro-level innovation capabilities of the FLOSS collaboration environment represented on SourceForge.

2.3 Role differentiation among developers: the background of related FLOSS community studies

The aim of the present study is to describe the innovation-generating potential of the FLOSS mode of development as these are reflected in the coupled processes of project-founding and project-joining. To our knowledge, this is the first attempt to infer system-level capabilities from the investigation of these two processes in the "open source" context. Moreover, our study helps to shed some more light on the phenomenon of FLOSS project founding, a process that appears comparatively neglected in the empirical literature. Save for a few exceptions (e.g., Giuri et al. 2006b, and Rullani 2006b), investigations of the organization and conduct of community-based peer production have not focused on projects foundation and instead focused exclusively upon already established projects, which then makes it natural to concentrate attention upon the processes through which FLOSS developers having variegated motivations and skills become attracted to one or another such "community," and come to take up the variety of tasks defined within those organizational entities. Few such studies have moved beyond snapshot views of the activities within projects, whereas the present analysis employs a statistical methodology that is explicitly dynamic in identifying distinct roles played by developers over the course of many months. .

Furthermore, rather than imposing preconceptions about the sequence of role-changing steps that the typical FLOSS developer follows in the progressive occupancy of different project roles, the approach adopted here provides a framework that uses the available micro-level data on the population of developers on *SF.net* to characterize patterns in the expected rates and directions of their movement among specified "development activity states" over successive 6-month time periods. This contrasts with the more usual, project-centered empirical approach that presupposes a set of differentiated roles between which individuals migrate step-wise – moving from peripheral and infrequent actions to regular attachment to a project and thence towards forms of contribution and administrative responsibility successively more and more exacting in their experience and skill-requirements, and therefore closer to the supposed "core" of the team working on the project in question. The most widely known schema for this structure is the so-called "onion model" (Crowston and Howison, 2005), which represents agents in the stylized FLOSS production community as being positioned in successive layers radiating outward form a "core" of initiators, release managers, maintainers, "core developers", co-developers, active users, specialized bug-patchers, and lurkers at the edges of the project who follow the threads in the email forum discussions, watch the evolution of the codes

in particular sub-projects, and eventually begin to engage with the community on a more regular basis or simply remain passive. Much of the research devoted to measuring the relative numbers of the members occupying these differentiated roles has taken the form of careful and quite detailed case studies of specific projects (e.g. Mockus et al., 2000), while the early empirical efforts to implement the "onion model" were confined to static, cross-section analyses (e.g., Crowston and Howison 2005) that precluded observation of developers' moving backwards and forwards between non-adjacent layers of the "onion", or concurrently occupying the same role in more than one project, or taking quite different roles in each of several projects.

A line of investigation that is more immediately related to the present study has been concerned explicitly with the sequential dynamics of FLOSS "project joining." Pioneering research on this subject was carried out by von Krogh, Spaeth and Lakhani (2003) in a case study of Freenet, a project developing a peer-to-peer system (which as was noted above is among the biggest projects hosted on *SF.net* in the period of analysis). By examining the chronological record of individuals' appearances in email subscriber lists and forum records of bug reports and patch submissions, as well as of the tasks performed as members of the project, the authors identify an implicit "joining script" defined by the type, frequency and intensity of individual's engagement; they then measure the distribution of durations that were required for contributors to make the passage from the periphery to the core of the project's code development activities. Jensen and Scacchi (2005) also have studied processes of "recruitment and role migration," focusing on the transitions from the role of end-user toward involvement in code development in a number of projects that were selected as separate case studies. In contrast to the notion that there is "a" joining script, they find a variety of paths, some of which do not conform to the conventional conceptualization of step-by-step progressions from periphery toward the project's core.

In a still more recent case study focusing on GNOME,[13] Herraiz *et al*. (2006) also ask whether the conventional dynamic interpretation of the onion model is a good representation of the joining process, and seek answers by estimating the distribution of transit times between developers' first instances of performing characteristic tasks in the onion's successive layers. Although it is typical for individuals in the GNOME community to have contacted the project by email before beginning to commit code[14], conventional suppositions regarding the next step in the sequence are violated by most of the GNOME developers' behaviors: for example, they often commit to the project's CVS before sending their first bug report. In general, this study finds striking evidence of the existence of significant heterogeneity in the developers' population with regard to the degree of directness and speed with which individuals move into core development work, and tries to put forward hypotheses on the possible variable that could accounting for the diversity.[15] A different aspect of the internal micro-dynamics is studied by Christley and Madey (2007). The authors define roles on the basis of the activities undertaken by developers registered to *SF.net*, and examine temporal changes in the distribution of the occupants of positions associated with those tasks. They find that aside from very broadly defined categories such as "Software Developer" and "Handyperson", the relative frequencies of specific task-positions exhibits high rates of decay, and conclude that FLOSS projects typically are too small to sustain permanent task-specialization. Rather than dealing with the problems posed by trying to find "stable" organizational task-associated roles that could be identified across a wide array of projects, the approach pursued here does not attempt to utilize the detailed information available in the *SF.net* archive about group members. Instead it focuses on analysis of a dataset that permits tracking the roles occupied by individuals in regard to project participation, project joining and project launching in successive time-intervals.

---

[13] GNOME is a large, well known project developing components of a FLOSS desktop environment, which enjoys sponsorship from several companies through its foundation http://foundation.gnome.org.

[14] Every time a developer changes the code basis of a project, CVS commits are produced. Thus, their number gives an idea of developers' rate of activity in the production of code and of their role into the project.

[15] The striking results reported by Herraiz *et al* (2006) is that almost two thirds of the entire small cadre of developers that was identified from the CVS as having moved within the shortest time-interval from joining the project into core development work with "commit" privileges were discovered to have been professional programmers sent to help the project by its sponsoring companies. On this issue the reader can see Dahlander and Wallin (2006).

3. The Data

3.1 Gathering data from SourceForge: the SFnetDataset

The statistical analysis is this paper is based upon an edited dataset (referred to here as the *SFnetDataset*) covering the *SourceForge* cohort of 222,835 individuals who registered on *SF.net* during the 14 months from September 1, 2000 through October 26, 2001. There are other datasets relating to *SF.net*. For example, the FlossMole project[16] gathers together and gives coherency to different data sources (obtained primarily from such platforms as *SF.net* or *Freshmeat*) describing FLOSS projects and developers' activities. Another recent initiative, undertaken by a group of researchers of the Notre Dame University, has released a series of monthly dumps of the *SF.net* data archives under a special license (see http://www.nd.edu/~oss/Data/data.html).

Several previous studies have drawn data from the *SF.net* site and its repositories to answer a variety of questions concerning the phenomena of open source software producers and production. The following list is not presented as inclusive, but suggests the richness and multi-dimensional nature of this information source. Krishnamurthy (2002), in a pioneering ecological study, examined a sample of project groups from the platform, using several indicators to select 100 "mature" projects to study their membership size and the activity levels reflected in their email forum discussions and commits to their respective CVS repositories. More recently, Madey *et al.* (2004) have mapped the network of developers' collaborations as members of project groups, Fershtman and Gandal (2004) sought to identify the determinants of variations in projects' average output of code per contributor-member, Lerner and Tirole (2005) examined the determinants of the choice of a specific open source license, while Lerner, Pathak and Tirole (2006) have drawn upon SourceForge and project web sites to create a dataset of large projects with which to study the contribution to FLOSS made by "developers" whose email address is a ".com" domain. Comino *et al*. (2005) have statistically identified factors determining *SF.net* projects' attained stages of development. Still more recently, Giuri *et al*. (2006a) have exploited information from the *SFnetDataset* (the same archival data as is used here) in order to study the diversity of skills, experience and the division of labor among the members of *SF.Net* projects extant during 2000-2002; Robles and Gonzalez-Barahona (2006) have utilized time stamp and other information collected on first registration to discover the geographical distribution of the more than 1,180,000 individuals who were listed as registered on *SF.net* in November 2005.

3.2 Accuracy of the data, dataset cleaning methods, and reliability in the present context

As is the case with many valuable deposits, there are non-negligible "refining" costs entailed in exploiting the material extracted from *SF.net.* Some errors have been found in the procedures followed by *SF.net* maintainers in building summary statistics and storing the data (see Hunt and Johnson, 2002). Moreover, even when the collected numbers are corrected, instances of developers' improper use of the facilities provided by *SF.net* somewhat degrade the accuracy of the resulting statistics (see Howison *et al*., 2005). To prepare the *SFnetDataset* on which this study is based, the original material from the archive was first "cleaned" by application of a variety of *ad hoc* procedures, and a systematic algorithm for inferring founders' identities from partial information has been implemented. Where automated procedures were not able to correct suspect data entries or identify a project founder, each problematic item was corrected by hand, comparing the available information with other sources of data (other records in the database, the *SF.net* website). In general, some assumptions about specific features of the operation of the *SF.net* data system were unavoidable, while some features of the data prevent the correction of some specific errors.[17] For example, when work on the dataset was first undertaken there was no way to unify different *user_id*'s that actually belonged to the same individual. Fortunately, a subsequent estimate of the impact of this and of other problems has shown that the resulting residual errors constitute a small fraction of the large number of developers involved in most of the processes that are examined by the present study. Moreover, even when this percentage I is not insignificant, most of these imperfections are basically random with respect to the history of each individual, and thus their net effect in biasing the estimates should be negligible – although

[16] FLOSSmole (formerly OSSmole, conveniently accessed at http://ossmole.sourceforge.net/) is a project developing tools for gathering data (metrics) about the development of free/libre/open source projects, archiving data donations from other research groups, and also publishing the resulting analyses about FLOSS projects. See Howison *et al*., 2005.

[17] A further discussion on the data can be found in Giuri *et al.* (2006a), that project being the original context in which the dataset was prepared.

they contribute to the noise in the data. A series of *ad hoc* controls on specific sub-samples and the close analysis of the aggregate results confirm that any remaining suspicious data can have at most a minor impact on the results.[18]

A number of previous studies have pointed to a different class of "perils and pitfalls" of working with *SF.net*. Most cautions that have been issued along these lines relate not to errors in the data itself, but to mistakes in analysis arising from casual acceptance of the information without an adequate understanding of the way in which it had been generated, or collected. Howison and Crowston (2004) and Rainer and Gale (2005) both warn against accepting observations based on *SF.net* as representative of the FLOSS community at large, in particular noting the high number of one-person and/or inactive projects hosted by the platform and the absence of many of the largest projects, which have their own web-sites. As has been noted previously, however, this study treats *SF.net* and the behaviors of the individual registrants not as representatives or exemplars of the universe of distributed collective production communities, but on their own terms, as reflecting the experience of a quantitatively important (but nonetheless particular) collaborative development environment. There is another "pitfall" to be borne in mind, which results from the platform's "open-ness": projects that have been largely developed externally to the *SF.net* environment, even those whose code is hosted elsewhere, may be posted on *SF.net* in order to give them greater visibility (Comino *et al*., 2005; Howison and Crowston, 2004). Where such is the case, a project that actually was important and very much alive (elsewhere), nonetheless, might appear on *SF.net* to be both small and dormant. This however means that the project's relationship with the platform must be very weak, and the role of the project in the whole ecology is likely to be marginal. Our data detect precisely these characteristics. Thus, since we focus on the "window" that *SF.net* offers to look at FLOSS development, this problem is to a certain extent irrelevant in our analysis.[19]

4. Markov chain analysis

Given the nature of the *SFnetDataset*, we need a statistical framework that is able to capture parsimoniously the main features of the micro-level processes that are of interest, namely, resources mobilization and new projects foundation. The application of Markov chain analysis (see e.g., Amemiya 1985, Hamilton 1994) is quite suitable for this purpose. In economics and managerial studies this is not a new approach, it has been applied widely in studying a variety of dynamical systems – including those governing industrial populations (e.g., Ezcurra *et al*., 2006), income distribution (e.g., McCall, 1971; Shorrocks, 1976), growth and regional inequality (e.g., Quah, 1993; Fingleton, 1997; Kremer et al., 2001), finance and accounting (e.g., Cyert *et al*., 1962; Konings and Roodhooft, 1997), and innovation (e.g., Cefis, 2003). In applying this framework for our present purposes, the appropriate first step is the delineation of a manageably compact "state space" describing an exhaustive set of activity-states into which individuals observed in the *SF.net* environment during any interval of time can be classified.

4.1 Defining the "activities" state space

Individuals registering on *SF.net* participate in the activities of this "community" in several basic roles: they may "lurk" without interacting with others or with existing projects in any of the several ways that the infrastructure of the platform would record. Alternatively, they may send patches, bug reports or request specific features to one or more of the established project; enter one or more projects and be listed as

[18] In some cases, we also "simulated" what would have happened if some of the most important estimates were different (e.g. we smoothed the skewed distributions of some rows) and find that the results are robust. As a final note on the limitation of the present analysis notice that we do not control for environmental changes that could have modified the conditions in which the individual histories of the users developed. When such changes were identifiable, their impact on the analysis was evaluated and always found negligible. For example, we found there were changes in the criteria that the *SF.net* staff applied in approving projects. Although that could affect the estimated likelihood of founding a new project over time, we have analyzed the data in such a way as to minimize the influence of such a changes: the limiting distribution in section 7 is estimated using only the cohorts for which that change did not occur, and the subsequent analysis (section 8) of the different behavior of the two subset of developers **C1** and **C2** is undertaken for all cohorts, but compares populations that are equally affected by this phenomenon because they were formed by approximately the same proportion of individuals in each cohort. Additional *ad hoc* robustness checks, such as the study of the cycle of new projects foundation, similarly show that that the result are not sensitive to those changes.

[19] Note also that the assessment of this latter problem by Comino *et al*., 2005 concludes, similarly, that in the context of *SF.net* as a whole the absence of data on projects that have their own websites elsewhere is of limited importance.

"group members;" and initiate new projects. A scale of activities is implicit in this, ranging from inactive to increasingly "active" behaviors, and resources mobilization can be conceived broadly as a transition from the lowest to the highest levels of developers' involvements into the community activity. As a practical matter, however, it is useful to recognize only three discrete levels in the "mobilization" of resource inputs: (i) contribution of bug reports, patches, and feature requests while remaining external to the projects, (ii) participation as internal members of projects, and (iii) the "founding" of new projects.

Consequently, we distinguish and capture all of these activity/roles by assigning every registrant to one and only one among the following set of "activity-states" in each of every 30-day period of her or his experience in *SF.net:*

0=non member and non founder[20]*, inactive*[21]*;*
1=non member and non founder, active;
2=member[22] *of 1 project and non founder of any project (both active and inactive);*
3=member of more than 1 projects and non founder of any project (both active and inactive);
4=founder of 1 project and member of 1 project (both active and inactive);
5=founder of 1 project and member of more than 1 projects (both active and inactive);
6=founder of more than 1 project and member of more than 1 projects (both active and inactive).

With the states *s=1, 2, …, 6* belonging to space ***s*** thus defined, observations on the frequency of the movements of individuals from state *i* in "epoch" T_t to state *j* in the next temporally adjacent "epoch", T_{t+1} , can be used to calculate average state dependent transition probabilities, and the results may then be organized to form a transition probability matrix. Treating these estimated mean probabilities as constant, the system is a homogeneous Markov chain with transition matrix $\boldsymbol{P}(t) = \boldsymbol{P}$.

4.2 Conditions for ergodicity of Markov chains, and their present relevance

A Markov chain is called *ergodic* if the distribution of the system among its states at step *r*, $\boldsymbol{p}(r)$, converges to a limiting (or equilibrium) distribution, $\boldsymbol{p}$, which is independent of the initial distribution $\boldsymbol{p}(0)$.[23] By analysis of the transition matrix $\boldsymbol{P}$ it is possible to establish whether a Markov chain satisfies the conditions for ergodicity, which is to say that the iterated transition process eventually will cause the system to shake free of any arbitrary initial distribution $\boldsymbol{p}(0)$. When that is the case, it is possible to calculate the limiting (ergodic) distribution of developers among activity states.

In presenting this approach, it is important to emphasize that the limiting (ergodic) states we will exhibit are not interpreted here as long-run forecasts, because the length of the period covered by the *SF.net* archive that is under examination here is quite short, and far too brief to allow meaningful predictions for this evolving environment. On the contrary, the Markov chain apparatus is here used as a way to expose the persisting dynamic forces of the system, abstracting from the effects of transient influences that may have affected the initial distribution of the individuals. As Quah suggests with respect to his Markovian analysis of countries' growth paths: "The steady-state distributions should not be read as forecasts of what will

---

[20] The only "foundations" we take into account are those who generated projects whose founders are still part of at the end of the month in which they were launched. A part from this minimum requirement, inasmuch as the foregoing comparisons aggregate all founders and make no attempt to distinguish the successful from the unsuccessful projects that they have launched, there is no warrant here for concluding that there are no individual characteristics or qualities of founders that matter, or that would differentiate the successful leaders of new projects from the rest of the developers' population.

[21] The developer is considered active if she did post at least one item classified as "patch", "bug report" or "feature request" to the tracker system of *SF.net* of at least one project during the analyzed period. Otherwise, she or he is considered inactive. Notice that *SF.net* allows developers to post also other typologies of "artifacts", for example "support requests". We excluded this category because it did not represent the same level of contribution as bugs, patches and feature requests do. Moreover, following Christley and Madey (2005), also the small percentage of artifacts (around 3% of the overall number of artifacts) created *ad hoc* by some project administrators were excluded because their level of contribution could not be easily identified.

[22] As done for the foundation activity, an individual is here considered to be a member of a project if she or he enters that project in a certain period and is still part of it at the end of that period.

[23] If the Markov chain is ergodic, the probability of remaining in each state goes to a constant (which is the inverse of the mean recurrence time) at the limit, and the resulting limiting distribution is found as the solution of $\boldsymbol{p} = \boldsymbol{pP}$.

happen in the future - government policies might change; Important, unforeseen events might occur. Rather, these distributions should be interpreted simply as characterizations of tendencies in the post-War history that actually realized." (Quah, 1993; p. 431). In our context, the simulated evolution of the *SF.net* community is just a means of grasping the interaction of the array of forces that have been at work. In other words, the Markov chain apparatus is employed as a sort of "*centrifugal separation machinery*", allowing the observer to see the effects of the persisting internal dynamics of developers' participation in *SF.net*.

The questions that we seek to answer make it necessary to specify the time dimension of the states and the transition process by reference to fixed durations of developers' *experiences* on the platform, i.e. the spans of time elapsed after each individual's initial entry – which must be defined independently of the particular calendar dates on which she or he registered on *SF.net*. Further, for the purpose of studying transitions between any pair of "activity states," we need to compare the positions of the members of a cohort of developers among the activity states in successive "epochs" of uniform duration. Therefore, for a given transition, we divide the experience of each developer between two epochs (*A* and *B*) that are of equal absolute length: T_A, corresponding to the days observed on *SF.net* from x to y, and T_B, the days observed from $y+1$ to z, where $y - x = z - (y+1)$.

To fix the values of x, y and z that bound the two epochs for the distributions of individuals among the "activity-states", it is necessary to consider the temporal structure of the underlying phenomena to which those "summary" states refer. In particular, in order to obtain estimates of the transition probabilities that form a first-order, or current state-dependent chain, it is important that the duration of those epochs is not so brief that the probabilities of any specific transitions from any of the states will be influenced by the individual's position among the states at some more remote, anterior point in time. In Appendix 1 we report the considerations that have led to the selection of "standard" observational epochs having durations of 180 days, on the basis of time series analysis of monthly (30-day) observations of events of project-joining, and, similarly of project-founding, for the sub-set of individuals who registered on *SF.net* during the period from September 1, to October 30, 2000. Following this cohort over the ensuing 840 days provides a large enough number of 30-day periods (28) to extract the possible cycles by spectral analysis of the two series after filtering to remove the trends.

Among the procedural details of this preliminary data analysis that should be mentioned here, the first derives from our ultimate concern to examine the behaviors of developers who were active in creating or joining projects: the observational sample of individuals in the time-series studies has been restricted to the sub-sample who founded or joined at least 1 project during their first 30-day period on the *SF.net* platform. Doing so serves to remove many passive or marginally active registrants from the series examined by spectral analysis and thereby render more readily apparent the periodicities of the behaviors of joiners and founders. Moreover, this creates a common stating point (the first project launched) for all the individuals, and sets the beginning of the series at the moment of the first foundation, thus employing the whole series time span to describe successive foundation activities. In the case of the time-series of project-launchings, the data from the first two of those 30-day intervals (from day 1 to day 60) were omitted – in order to reduce the influence of the "importation" of pre-existing projects in the first months of individuals' experience of the *SF.net* platform and thereby better reveal such periodicities exist in the "indigenous" process of project-founding on the platform. Other anomalies were detected in the case project joining, so that the first and the last period of the series were omitted from the analysis (see appendix 1 for a discussion on these points).

The upshot of the preliminary data analysis reported in Appendix 1 is, first, that there are indeed periodicities in both kinds of events, and second, that these can be taken into account by defining 6-month long observational epochs. The latter are sufficiently extended to capture completion of most of the important periodic structure of events that drive transitions among the activity-states, yet not so long as to foreclose the opportunity to observe more than one typical cycle and thereby prevent examination of the temporal stationary of the estimated transition matrix.[24]

[24] It should be clear that this analysis is undertaken solely for the purpose of minimizing the amount of noise (and possible bias) by imposing arbitrary periods when calculating the frequencies of transitions between pairs of states. There is no intention here of investigating whether it is appropriate to characterize either (or both) of these micro-level behaviors as continuous time Markov renewal processes (see Pyke, 1961; Cox, 1962). For applications of Markov chain theory and renewal models to micro-demographic data, see, for example, Sheps and Menken (1973).

5. A first look at the dynamics of the process

In calculating average probabilities of transitions between activity states defined for each pair of consecutive 6-month long epochs, the data from the 222,835 registrants during their first 90 days on SF.net were not employed. The same considerations that led to discarding the first 60 days of observations on these individuals activities for the purposes of the preliminary time series analysis (Appendix 1) apply here: this is intended to filter out the effects of "imported projects" and the associated project members among the newest cohorts on the platform.[25]

5.1 Estimation of the transition probability matrix

Thus, in estimating the transition probabilities we considered two intervals: epoch *A*, based on observations that span the 30-day months from the 4th to the 9th (so that *x=90* and *y=270*): epoch *B*, based on observations pertaining to 30-day intervals 10 to 15 (i.e. *z=450*). By considering all 222,835 individuals who registered on *SF.net* during the (approximately) 14 months from September 1, 2000 to October 26, 2001 it was possible to retrieve from the database observations covering each individual's behavior throughout a 360-day time span. The distribution of the sample among the seven states at the close of period *A* is presented by Table 1, and for the purposes of this computation we refer to this as the population's "initial distribution".

Table 1. The Initial (epoch *A*) distribution of the whole sample

State	**Frequency**	Proportion
0	**192051**	0.862
1	**2202**	0.010
2	**21848**	0.098
3	**3280**	0.015
4	**1702**	0.008
5	**1405**	0.006
6	**347**	0.002

Entries for the transition probability matrix are obtained using the MLE estimator described by Amemiya (1985). The resulting matrix ***P*** describing the transition probabilities between the states attained at the end of in epoch *A* (rows) and epoch *B* (columns) is reported in Table 2 and depicted in Figure 1. The shading of the zones in the figure indicates equal probability bands, and the steepness of the gradient between a given band and the one adjacent is approximately represented by the width of the band in question. To assess the precision of each estimate we also computed the relative standard deviation theoretically, using the formula derived by Bode (1998) as reported in Bickenbach and Bode (2001):

$$\sigma_{\hat{p}_{ij}} = \sqrt{\frac{p_{ij}(1-p_{ij})}{n_i}}$$

where p_{ij} is the estimated transition probability, n_i the number of observations in state *i* in epoch *A*.

We also computed the standard deviation empirically, using the bootstrapping technique. We resampled with replacement a number of observations equal to that of the original sample 500 times, we estimated the transition probabilities for each one of the 500 new samples, and then we computed the standard deviation of their distribution across these samples. The convergence between the results of the two

[25] We extended by one more month the set of excluded periods because it is possible that delays in the formal organization and launch of projects formed substantially on the basis of "imported code" may extend beyond the initial two-month period following registration. The need to extensively restructure the architecture of the code of a project that had been previously developed by a closed process, in order to make it more suitable for further development by a decentralized, virtual community, was recognized in the well known case of Mozilla. That is only one of the possible circumstances that, along with decisions about governance and licensing, might contribute to extended delays between the individual registrations by a core of project developers, and the date of the formal launching of their project.

techniques assures the reliability of our standard deviation estimations. Table 2 reports the standard deviation (in parenthesis) as obtained through the formula described above.

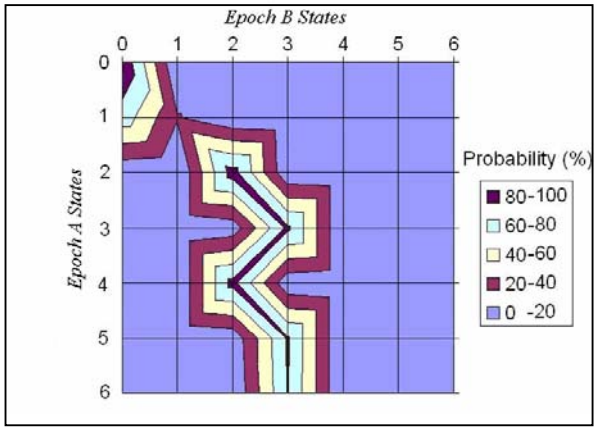
5.2 A system property is revealed: resource dissipation

The matrix estimates described above reveal the strong inertia within each of states 0, 2 and 3, indicated by the high probabilities of persistence from one 6-month period to the next. In particular, state 0 (inactivity) almost resembles an absorbing state, as there is a 0.98 probability of remaining in state 0 for two successive periods. This is shown by the dark region positioned in the upper-left corner of the Figure 1. By contrast, persistence at the higher levels of activity is much rarer: individuals reaching each of the three activity states above 3 (all of which involve launching projects) are very likely not to found any new projects in the following epoch. Instead, they tend to return to states "lower" than the one just attained: only 4.7% of those who reach state 4 (on average) will proceed to join another project in addition to the one they founded (state 3), and only 7.3% will launch at least one other project (states 4, 5 and 6) in the following 6-month epoch.

Table 2. The transition matrix P_{AB} for the whole sample of *SF.Net* registrants

		Epoch B States							
	State	**0**	1	**2**	3	**4**	5	**6**	# developers
Epoch A States	0	**0.9821 (0.0003)**	0.0072 (0.0002)	**0.0047 (0.0002)**	0.0004 (0.0000)	**0.0046 (0.0002)**	**0.0005 (0.0001)**	**0.0005 (0.0000)**	*192051*
	1	**0.7134 (0.0096)**	0.2153 (0.0088)	**0.0395 (0.0042)**	0.0023 (0.0010)	**0.0232 (0.0032)**	**0.0050 (0.0015)**	**0.0014 (0.0008)**	*2202*
	2	**0.0485 (0.0015)**	0.0009 (0.0002)	**0.8892 (0.0021)**	0.0295 (0.0011)	**0.0010 (0.0002)**	**0.0275 (0.0011)**	**0.0034 (0.0004)**	*21848*
	3	**0.0116 (0.0019)**	0.0000 (0.0000)	**0.0753 (0.0046)**	0.8424 (0.0064)	**0.0003 (0.0003)**	**0.0646 (0.0043)**	**0.0058 (0.0013)**	*3280*
	4	**0.0100 (0.0024)**	0.0000 (0.0000)	**0.8702 (0.0081)**	0.0470 (0.0051)	**0.0018 (0.0010)**	**0.0652 (0.0060)**	**0.0059 (0.0019)**	*1702*
	5	**0.0021 (0.0012)**	0.0000 (0.0000)	**0.0498 (0.0058)**	0.8206 (0.0102)	**0.0000 (0.0000)**	**0.1060 (0.0082)**	**0.0214 (0.0039)**	*1405*
	6	**0.0000 (0.0000)**	0.0000 (0.0000)	**0.0115 (0.0057)**	0.8098 (0.0211)	**0.0000 (0.0000)**	**0.1268 (0.0179)**	**0.0519 (0.0119)**	*347*

Figure 1. Representation of the Transition Matrix P_{AB}



An intuitively appealing inference is that project *launching* involves time-consuming commitments, and therefore is a current activity state that is only very rarely sustained from one 6-month interval to the next. A second consideration undoubtedly is reflected in the specific way lower activity states exert an attractive pull on recent project founders: because developers may remain officially recorded by SourceForge as members of a project they previously joined, even though they ceased contributing to it during the period when they were launching a project of their own, such individuals seem to subsequently drop back to the immediately lower activity states (i.e., from 5 to 4 or 4 to 3) just as an artificial consequence of the platform's record-keeping procedures. But the phenomenon cannot be wholly ascribed to that cause, because the archives do record the delisting of developers from project groups. Indeed, between September 2000 and January 2003 about 15,000 individuals were removed from the projects to which they had once belonged, and who were participant in 1400 projects lost their group membership status as a result of the deletion of the entire project from the *SF.net* platform. Even if it is true that these "exit" movements represent only about one sixth of the "entry" movements due to project joining, new project creation and project re-activation, they remain relevant for our interpretation of the observed pattern of transitions. Moreover, within the Markov process framework, what really matters is the relative concentration of particular transitions in the different rows of the matrix represented in Figure 1. A comparative small number of transitions actually may exert a strong influence on the dynamics of the process if they tend to cluster in the less populated rows. Given this, the observed degree of inertia in states 2 and 3, even though expected, is a non-trivial consequence of the developers' "laws of motion" on *SF.net*.

The local strength of state 0 as an "attractor", as already noted, is another marked asymmetry in the dynamics revealed by the estimated transition matrix (Table 2 and Figure 1). Not only is the mean probability of exiting state 0 quite small (0.018), but the probability of becoming inactive (state 0) after a period of activity as "non-member" (state 1) is almost forty times larger (0.713). Evidently, even those developers that enter and maintain a state of minimal activity represent a distinct minority of those who visit this FLOSS collaboration environment: for the average registrant on *SF.net* the probability of a first emergence from state 0 in each successive month is 0.003. From this it follows that the likelihoods of progressing to project membership and/or project launching (states from 3 to 6) via state 1 (i.e. starting as an unattached "lurker") must be very small.[26] At such rates, within a 6-month interval *only 1 in two thousand registrants* could be expected to make a passage such as is envisaged by "the onion model" – from inactivity to peripheral activity and thence to project membership.[27]

It is apparent that if the "onion model" has any substantive empirical relevance here, one must take it to pertain to the population of individuals who have some intention of becoming open source developers, whereas the great mass of individuals that register on *SF.net* should be regarded essentially as transient "spectators" rather than potential "players.". Therefore, viewing the boundary of the "active" system (defined for any given 6-month period) to be located *between* states 0 and 1, only those developers that attain state 1 will be considered to belong to the region corresponding to the periphery of the onion model. Initial project-joining, then, is most reasonably reckoned as a transition from that minimal activity state. One may then see directly from Table 2 that the mean probability of entering a project group (state 2) having been "in the periphery" (state 1) during a previous 6-month interval is close to 0.0395, and although this really is quite low, it still almost 8 times greater than the calculable mean probability (0.0048) of accomplishing the first passage to group membership conditional on having not been part of any project on *SF.net* in a previous 6-month interval.[28] By contrast, once a developer has joined or founded a project, the probability of not returning to any of the activity states below 2 is very much higher, indeed, almost a certainty (0.960).

---

[26] Since the 6-month probability (shown in Table 2) for remaining in initial state 0 is 0.9821, the corresponding monthly probability of persistence is calculated as the sixth-root, and the mean probability of emerging into activity is found to be 0.003, as reported in the text. This calculation assumes that the 6-month transition rates are generated by constant monthly probabilities.

[27] From Table 2 the probability of progressing directly to any of the states 2-6 from state 0 via state 1 is found for 6-month transitions as (0.007) [1-(0.713+0.216)] = 0.0005, where 0.007 is the probability of going from state 0 to state 1, 0.713 the probability of remaining in state 1, and 0.216 the probability of returning from 1 to state 0.

[28] The estimated mean probability of transiting from state 1 to 2 in a 6-month period is 0.0395 (from Table 2). The estimated mean probability of making a first passage to group membership from prior "inactive status" within a 6-month period (0.0048) is arrived at by considering the interval formed by two consecutive 6-month periods and

The preceding results highlight an important characteristic of the dynamical system we are examining: the *dissipative* nature of the FLOSS mode of innovation." The comparative ease with which FLOSS development projects can be launched combines with the mobilization of developers from the ranks of those who are at least active in this environment to couple exploration and exploitation processes in a highly dissipative system, i.e. a system that is generating outputs by "burning resources" at a high rate.[29]

To make this point clearer, we may suppose that developers approaching the platform for the first time typically do so in order to examine the code of a particular program, perhaps with the intention of downloading and customizing some portion of it to suit some specific needs. In the long run, however, those who continue and participate in community activity explain their engagement in FLOSS development by citing a completely different set of motivations, such their enjoyment of programming or fixing bugs as "hobbyists" (Shah, 2006).[30] One explanation for such behavior may be found in their interactions with the others in the community: as Rullani (2006b) reports, the more intensely developers are exposed to the social environment of a FLOSS project community, the greater is the number of new projects they may be expected to launch. Further, the small portion of developers who remain is comprised of those who are the most reactive to the stimuli generated during such interactions (Rullani, 2006a).

Considerable resources are involved in creating and maintaining the sort of interaction environment in which such things are likely to happen, and this must not me minimized. For those individuals to emerge in the project-founding activity-states, an enormously larger number – almost 65 times larger -- initially visited the platform, as the absolute figures in Table 2 reveal. When one puts aside potential resources represented by the mass of transient "sight-seers" from whom nothing is heard after they registered on the *SF.net* platform, one sees that the active population is a little less than an order of magnitude larger than the cadre of those who are launching projects.

But, inasmuch as "innovative output" is something beyond the exploratory process of launching projects, and viable outputs entail also the exploitation phase of resource mobilization, development work, and the maintenance of maturing projects' code, a more complete view of the system requires noticing a second striking feature of the process: the skewed size-distribution of the projects on *SF.net*. An aspect of this phenomenon was first noticed by Krishnamurthy (2002), who, having studied a sample of 100 "mature" SourceForge projects, reported that a surprisingly large proportion of them were far from vibrant "communities" of developers; rather, they more closely resembled "caves" inhabited by one or two rather dormant occupants. Being able here to consider the entire population of projects on *SF.net* in January 2003, the situation is found to be much more extreme that suggested by Krishnamurthy's small sample: of all projects that had at least one member, 67% had no more than one member. Thus it appears that the ecology of FLOSS represented on this particular collaborative development environment is generating many new ventures that represent "exploration," but of which only a small fraction succeed in actually mobilizing development resources. Another possibility is, of course, that the platform is simply an attractor for many pre-existing small software products that already are "mature" when re-released under an "open source" license, and, having little need to mobilize additional developers, consequently remain dormant with a single administrator/project member.

---

summing the following probabilities, again taken from Table 2, for : (i) making a direct passage from 0 to 2 in the first of these (0.0047), and then in the second period adopting whichever of the behaviors (i), or (ii) (0.0047) conditional on having remained inactive during the first period (0.9821), plus (iii) entering state 1 from 0 in the first period (0.0072), and proceeding thence to state 2 (0.0395). The mean 6-month figure in the text is found as the geometric rate that corresponds to the mean probability of making a first passage within the year (0.0097).

[29] This allusion to the metaphor of a thermodynamic system hardly is new to the organizational science literature. For example, Anderson (1999:p.222) writes: "Social entities always self-organize as long as their members contribute work; this is why informal structures emerge and persist in a way that is remarkably robust to changes in the formal organizational structure. Those with influence and/or authority *turn the heat up or down on an organization by recruiting new sources of energy* (e.g., members, suppliers, partners, and customers) …"(Emphasis added)

[30] For general discussion of the economics of FLOSS developers' motivations, see,Lerner and Tirole (2004); for empirical evidence regarding changes in individual developers'expressed reasons for their involvement, see Glott (2004), Ghosh, Glott and Krieger (2004), and David and Shapiro (2007).

The two foregoing explanations are not mutually exclusive. Yet, were the latter of them to be the dominant cause of the observed concentration at the minimum project member size, that certainly would be a serious challenge to the view that what is taking place on *SF.net* represents "exploration" through the launching of new projects. Doubts arising on that score, however, can be quickly set aside by comparing the size distributions of two groups of projects: the first is the set of projects formed within months 1-3 of their founder-members' respective dates of registration on platform, and the second, larger set is formed from the projects that had been launched during months 4-15 of the period commencing with the founder-member's arrival (registration). Within the first set, projects launched with the first 90 days of the founder's presence on *SF.net* reasonably can be regarded as "migrant" projects, whereas the second group (formed 90 days or more after the founder registered) quite unambiguously are projects that should be viewed as "indigenously" generated. It turns out that in each of these two distributions (from months 1-3 vs. months 4-15) only one-third of the projects with any members had more than a single member in January 2003, in this mapping the overall size distribution observed on *SF.net* in the same date.[31]

Thus one may say that, *SF.net* is not just a site that attracts "sightseeing" developers, most of whom contribute no resources, and that it is more than a platform that provides convenient facilities for those registrants who bring to it their ideas (and possibly co-developers) for projects conceived before they had any significant experience of interacting with others in that environment. The environment generates a population of new projects launched by cohorts of registrants well after they arrived, and the size distribution attained by those projects reproduces that of the projects that are in a sense brought to the platform by the active core among the newly arriving developers. Nevertheless, recognition the smallness of the proportion of those registered who move into even minimal levels of "activity" suggests that in order to bring the micro-level processes of project-launching and project-joining into clearer view, attention should now focus on that part of the population. The analysis of the following sections therefore leaves the mass of inactive registrants (those remaining in state 0) in the background, while continuing to work with Markov transition probability matrices estimated from data on individuals' behaviors during their 4th through 15th months on the platform.

6. Project-joining and project-founding: the main forces at work

The preceding initial examination of the transition matrix for the system (Table 2) leads to the conclusion that, on average, developers tend to decrease their involvement in the *SF.net* environment. Those who are in state 0, 2 or 3 tend to maintain their level of participation, but those in states 1, 4, 5 and 6 are strongly attracted by lower activity states. When the dynamics induced by this process is unfolded, one might suppose that there would be a marked decrease in the percentage of developers observed to be launching new projects, and the overall picture would be found to be characterized by aggregate levels of activity lower than those initially observed. But the main mechanisms at work within the system are partially obscured by non-persistent forces. The analysis in this section shows that when these transient effects are removed – by iterating the transition process until it shakes free of the influence of those conditions – then a different picture emerges.

Consider first the steps needed to discern the persistent micro-dynamic processes that are at work within the system constituted by *SF.net*. In order to proceed from the transition probabilities described above to the distribution characterizing the "steady state" of the process requires showing the stationarity of the Markov chains formed by those estimated transition rates. Observation of *SF.net* suggests the occurrence of some "environmental" changes in the platform over time, so that stability of the developers' "law of motion" is rather unlikely. Statistical analysis of the data confirms that the MLE estimates of the transition probabilities in Table 2 for the whole sample of 222,835 developers do not collectively describe a Markov

[31] For example, Krisnamurthy (2002: Table 2) reports 22 percent of his sample of mature projects having 1 developer, only one-third of the corresponding proportion found by examining the entire *SF.net* project population at the beginning of 2003. Thirty-four percent of Krishnamurthy's mature project sample fell into the 1-2 developer size range; whereas David and Shapiro (2007) find that of a total of 847 FLOSS developers who responded to broadcast web surveys in 2002-03 and could be subsequently linked to projects of known membership sizes, the 49.8 percent were in projects in the 1-2 size range. The distribution of developers by project size naturally would show less concentration at the low end of the size range than that observed in the distribution of project sizes.

process that is strictly time-stationary within the full span of observation.[32] Moreover, when stratifying developers in 90-day cohorts according to their "entry period", it is found that most of the times two different cohorts thus formed not only have different initial distributions, they are also characterized by different transition probability matrices.[33]

To partly mitigate the effect of the absence of stationarity in the process described by the transition matrix for the whole population of registrants in the *SFnetDataset* (Table 2), in what follows our analysis of the properties revealed by the iterated Markov chain will be based on re-estimating the transition probability matrix for the same consecutive 6-month epochs *A* and *B* using only the data pertaining to members of the last among the cohorts of registrants. The sample population that can be used for this purpose is thus reduced to the 79,983 individuals whose entry dates fell within the 120 day span extending from June 28th to October 26th, 2001. Table 3 presents the transition probability matrix obtained by the MLE method applied to this data, where it is denoted as $(\boldsymbol{P}_{\mathbf{AB}})^{\mathbf{last}}$.

Finding the limiting distribution implied by these transition probabilities requires that one first establish that $(\boldsymbol{P}_{\mathbf{AB}})^{\mathbf{last}}$ determines a regular Markov chain, or has an absorbing state. While one may see immediately from Table 3 that there are some zero elements in that matrix (as it was the case also for $\boldsymbol{P}_{\mathbf{AB}}$ shown in Table 2), it turns out that all the entries in the third power of the matrix are positive, so that $(\boldsymbol{P}_{\mathbf{AB}})^{\mathbf{last}}$ is indeed a regular transition matrix and its associated Markov chain is regular.

Table 3. The transition matrix $(P_{\mathrm{AB}})^{\mathrm{last}}$ for registrants from June 28th to October 26th, 2001.

		Epoch B States							
	States	**0**	1	**2**	3	**4**	5	**6**	# developers
Epoch A States	0	**0.9845 (0.0005)**	0.0063 (0.0003)	**0.0037 (0.0002)**	0.0004 (0.0001)	**0.0041 (0.0002)**	0.0004 (0.0001)	**0.0006 (0.0001)**	*70766*
	1	**0.7241 (0.0157)**	0.2143 (0.0144)	**0.0369 (0.0066)**	0.0025 (0.0017)	**0.0172 (0.0046)**	0.0025 (0.0017)	**0.0025 (0.0017)**	*812*
	2	**0.0561 (0.0029)**	0.0009 (0.0004)	**0.8882 (0.0039)**	0.0249 (0.0019)	**0.0009 (0.0004)**	0.0266 (0.0020)	**0.0023 (0.0006)**	*6458*
	3	**0.0139 (0.0038)**	0.0000 (0.0000)	**0.0897 (0.0093)**	0.8301 (0.0123)	**0.0011 (0.0011)**	0.0577 (0.0076)	**0.0075 (0.0028)**	*936*
	4	**0.0116 (0.0047)**	0.0000 (0.0000)	**0.8702 (0.0148)**	0.0407 (0.0087)	**0.0039 (0.0027)**	0.0678 (0.0111)	**0.0058 (0.0033)**	*516*

[32] To illustrate this we can compare $\boldsymbol{P}_{\mathbf{ab}}$ (i.e. the matrix expressing developers' movements from the first 180 days of their experience in *SF.net*, epoch *A*, to the second 180-day, epoch *B*) to $\boldsymbol{P}_{\mathbf{bc}}$, representing the movements of the same developers from the second (*B*) to the third 180-day epoch (*C*). If the "law of motion" is time invariant, the two matrixes should be the very similar. To see if this is the case, we use the Bickenbach and Bode's (2001: p. 12) statistic aimed at testing whether the estimated transition probability p_{ij} significantly differs from a specified value $p_{ij}^{\,0}$ (the test is a modified version of the test by Anderson and Goodman's, 1957: p. 97). We perform two different tests considering alternatively $\boldsymbol{P}_{\mathbf{ab}}$ or $\boldsymbol{P}_{\mathbf{bc}}$ as the matrix containing the non-random elements $p_{ij}^{\,0}$. As said, the test is just an illustrative assessment of the difference, because we "fix" one matrix as non-random. However, the results are useful in showing what we already expected: Considering $\boldsymbol{P}_{\mathbf{bc}}$ as the non-random matrix, the test-statistic $\chi^2(34)$ is equal to 279.99, much larger the critical value for the 1% level of significance (56.06). Similar numbers are found when $\boldsymbol{P}_{\mathbf{ab}}$ is taken to be the non-random matrix: $\chi^2(32)$=244.34 and critical value for 1%=53.49. The distance between the values of the statistics and the critical values illustrate precisely the difficulties in assuming the "law of motion" behind $\boldsymbol{P}$ as stable.

[33] To test for equality of initial distributions we took two samples (two cohorts or one cohort and the whole sample) and create a dichotomous variable distinguishing between the two. Then we used a logistic regression model to estimate the effect of being in each state on the probability of belonging to a specific initial distribution. If at least one coefficient is significant this corresponds to a situation in which being in a specific state makes it more likely to belong to one initial distribution instead of the other, and thus corresponds to the situation in which the two distributions can be considered different. To test for the equality of the transition matrixes we used the test proposed by Bickenbach and Bode (2001: p. 8). This test is an adaptation of the Anderson and Goodman's (1957: p. 99) that can be used to check for homogeneity with respect to the entry period and other exogenous traits.

5	**0.0000 (0.0000)**	0.0000 (0.0000)	**0.0670 (0.0122)**	0.7847 (0.0201)	**0.0000 (0.0000)**	0.1196 (0.0159)	**0.0287 (0.0082)**	*418*
6	**0.0000 (0.0000)**	0.0000 (0.0000)	**0.0000 (0.0000)**	0.7922 (0.0462)	**0.0000 (0.0000)**	0.1299 (0.0383)	**0.0779 (0.0305)**	*77*

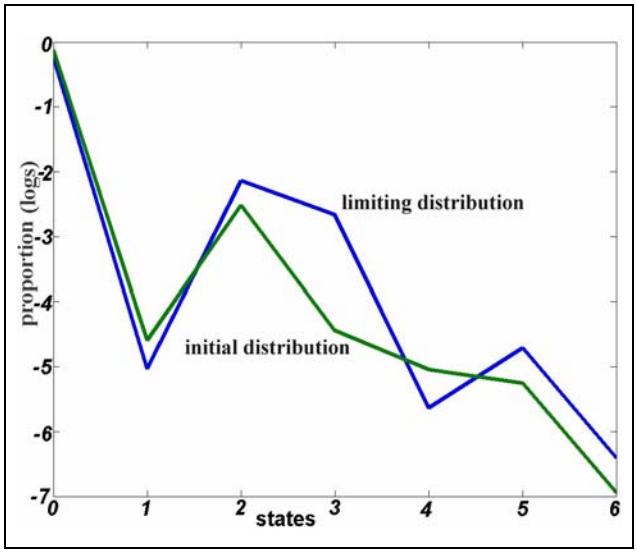
Since regular Markov chains are ergodic, one may readily solve for the unique limiting distribution of the population among all the states of the system (see Amemiya 1985, p. 421). The result from the transitions probability matrix in Table 3 is reported in Table 4, and graphed on a logarithmic scale in Figure 2, where it is compare with the corresponding values of the initial distribution for the same cohort.

Table 4 shows that in the limit, those who do not belong to any project represent about 80% of the population, while about 18% of the individuals become part of one or more projects. The remaining 1-2% fall into one state corresponding to new projects foundation. When one moves from the static view to a more dynamic picture by comparing these estimates of the limiting distribution with the initial distribution for the same sample,[34] quite different properties of the system appear. The comparison reveals the tendency for the overall proportion of lurkers to contract, and for an increasing fraction of developers to occupy states 2 and 3: those remaining in passive states become less prevalent, and the joining of existing projects becomes a more widely diffused behavior. Moreover, an enlarged proportion of developers wind up in one of the two states associated with the most active participation and new projects foundation (states 5 and 6).

Table 4. Initial and limiting distributions: developers stratified by their state(s) in month 3 on *SF.net.*

Distribution	*State 0*	*State 1*	*State 2*	*State 3*	*State 4*	*State 5*	*State 6*
Initial	0.885	0.010	0.081	0.012	0.007	0.005	0.001
Limiting	0.792	0.007	0.118	0.070	0.004	0.009	0.002
L – I	**-0.093**	**-0.004**	**0.037**	**0.058**	**-0.003**	**0.004**	**0.001**

Figure 2. Initial and limiting distributions for registrants from June 28th to October 26th, 2001.



The difference really is striking. From Table 4 one might draw the conclusion that the "attractive power" of system's inactive states, and especially of the state 0, would be in the limit all but overwhelming, and most of the individuals not ending there – those in states 2 and 3 – would slightly increase their

[34] Kremer, Onatski and Stock (2001) test some hypotheses on the ergodic distribution imposing a specific assumption on the "true" transition probabilities. They notice that "Without [this assumption], simple linear restrictions on π would be equivalent to complex nonlinear restrictions on the transition probabilities, so that one would expect very bad finite sample properties of the asymptotic test", Kremer *et al*. (2001), p. 14. Quah (1993) computes the limiting distribution to discuss the law of motion of the underlying Markov process and performs some robustness checks to be sure of the results he unfolds. This second methodology is also appropriate here where the dissipation property discussed earlier generates the need to amplify small, even apparently non significant, signals.

proportion. Moreover, Table 4 also discloses that the number of developers who continue to enter existing projects rise markedly (the proportion in state 3 increases by 5.83 times), and limiting proportions of developers launching multiple projects (states 5 and 6) are approximately twice as large as they are in the initial distribution. In other words, absent other changes, there is an underlying tendency for greater "exploratory vigor" and "exploitation activity" to emerge, especially among those who become minimally active on the platform. These findings are robust to different specifications. The estimated matrix in Table 3 may be compared to the matrix estimated from a still smaller sample of observations in a later entry cohort – namely that for the last two months in the available sequence (from August 27th to October 26th, 2001). In both cases the associated Markov chains converge to a unique distribution which exhibits the properties of the process that have just been described.

These observations, combined with the necessity of highlighting the micro-dynamics FLOSS development activity against the background of the system's dissipative nature, reveal that *the FLOSS mode of innovation is able to generate a substantial amount of both exploration and exploitation.*

7. Exploring the heterogeneity of the sub-population of "project founders"

Having examined the implications for the limiting rates of project participation and the consequent mobilization of development resources implied by the estimated transition probabilities in Figure 2, we now focus specifically on the micro-dynamics of project-founding.

7.1 The persistence of "exploratory spirits"

The previous picture may be enriched by assessing how "*persistent*" or "*durable*" is the "exploratory spirit" among those who emerge in the founder's role on *SF.net*. Consider only those developers registered from September 1, 2000, to October 26, 2001, who founded at least one project in the first 15 periods of their experience on the platform: when a developer in this sub-group is involved at least in a single project, the probability that he or she has founded that project is increasing slightly during the first three months, on average, but decreases thereafter. This fluctuation, however, has a range of only a few percentage points, so that probability can be reasonably viewed as stable in the neighborhood of 0.90. An implication following from this line of considerations is that exploration is not being undertaken at the expenses of exploitation, since founders tend to remain part of the projects they have launched – even though in the case of unsuccessful projects this simply means remaining listed as a member of a moribund project while re-directing their energies elsewhere.

7.2 The timing of the realization of "exploratory spirits"

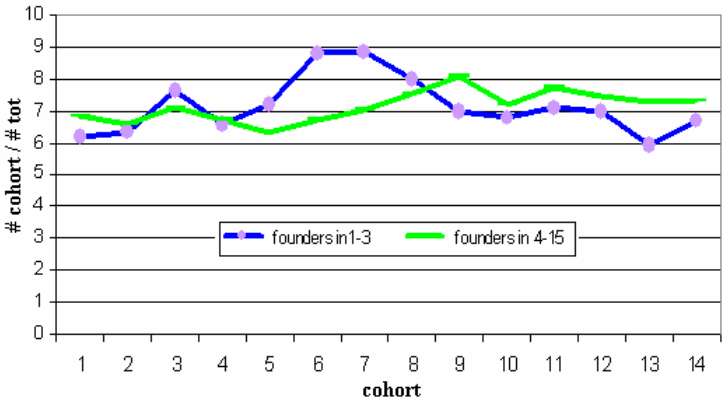
By stratifying the sample according to the developers' founding activity it is possible to further clarify the different expressions of this "exploratory spirit." In particular, we may proceed by dividing the whole population into three sub-samples according to the period within which the developers launched their first projects: in the intervals [1 – 15], [4 – 15] and [1 – 3]. The last interval comprehends those periods we excluded in the first place precisely because we are now interested in investigating if the initial "urge" to found new projects (or to import form outside *SF.net* already existing projects) has any consequences on the dynamics of the possible subsequent new projects foundations. Moreover, this group can be used as a benchmark to evaluate how much "persistent" is the activity of those who launch projects in a later phase of their experience in *SF.net* and how much important is their role as "explorers".

Notice that, as was the practice in the previous sections when comparing these subsamples, the cohorts from months 11 to 14 are the ones to be considered. This is because – as it has been shown – cohorts follow different "laws of motion" and are differently affected by changes in their environment on the platform. Aggregating all the developers would render it impossible to separate the effect of the different cohort compositions from the average "law of motion." But, given that we are now concerned with subsamples of the already limited population of founders, to constrain our analysis to cohorts from months 11 to 14 will seriously decreases the available number of observations and result in imprecise estimates, very sensitive to the possible errors in the data. To overcome the latter problem, we studied the inner structure of each sample. As Figure 3 shows, the cohort composition is basically the same for the two samples. Since we derive our conclusions from the comparison between the two groups of developers, every factor influencing

the different behavior of each cohort is affecting a very similar proportion of users in both the samples, so that its effect should not influence the results of the comparison.

The negative side of this strategy is that we lose the specificities of the last cohorts, and instead look at a sort of "weighted average behavior" across all the cohorts. Given that we do not want to produce forecasts, but observe the emerging trends in the window of analysis, this loss seems something the analysis can bear. Moreover, the cohort composition appears to be even also inside each sample, being every cohort's contribution between 6 and 9 per cent of the overall number of developers in the subsample. This assures that the incidence of each cohort in determining the observed behavior is the very close to that of the other cohorts in the sample. Thus, no cohorts are "privileged" and the outcome of the process can be thought of as representative of the whole population of *SF.net* captured by the sample.

Figure 3. cohorts composition for the two subsample of founders in [1 - 3] and [4 - 15]



Aggregating all the developers in the population (sample **A**, n=222,835) and stratifying them as described, we form: subsample **B** (founders in period 1 – 15; n=15,825); subsample **C1** (founders in period 4 – 15; n=5,514); subsample **C2** (founders in period 1 – 3; n=11,875). Notice that these groups are not mutually exclusive: developers founding projects in the first 90 days of their experience in *SF.net* and in the subsequent 6 months (about 1500 individuals) will be considered as part of **C1** as well as of **C2**. By comparing these samples, it is possible to derive some insights regarding the "timing" of project founding and its relationship with repetitive, persistent exploration.

Table 5. Initial and limiting distributions according to the period of project launching

Sample	Distrib.	*State 0*	*State 1*	*State 2*	*State 3*	*State 4*	*State 5*	*State 6*	# devel.	Iter. to eq.
A *(whole population, n=222835)*	*Initial*	0.862	0.010	0.098	0.015	0.008	0.006	0.002		
	Limiting	0.727	0.007	0.141	0.107	0.004	0.013	0.002	*222835*	*177*
	L - I	**-0.135**	**-0.003**	**0.043**	**0.092**	**-0.004**	**0.007**	**0.000**		
B *(founders in period 1 – 15, n=15825)*	*Initial*	0.074	0.004	0.584	0.119	0.108	0.089	0.022		
	Limiting	0.004	0.000	0.310	0.573	0.004	0.097	0.011	*15825*	*43*
	L - I	**-0.070**	**-0.004**	**-0.274**	**0.454**	**-0.104**	**0.008**	**-0.011**		
C1 *(founders in period 4 – 15, n=5514)*	*Initial*	0.193	0.012	0.126	0.042	0.309	0.255	0.063		
	Limiting	0.001	0.000	0.028	0.436	0.004	0.480	0.052	*5514*	*47*
	L - I	**-0.192**	**-0.012**	**-0.098**	**0.394**	**-0.305**	**0.225**	**-0.011**		
C2 *(founders in period 1 – 3, n=11875)*	*Initial*	0.009	0.000	0.756	0.153	0.000	0.068	0.013		
	Limiting	0.140	0.004	0.326	0.460	0.003	0.059	0.008	*11875*	*224*
	L - I	**0.131**	**0.004**	**-0.430**	**0.307**	**0.003**	**-0.009**	**-0.005**		

We begin by looking at the initial distributions of sub-sample **C2** in Table 5. The group of developers founding at least 1 project in the first 90 days of their experience in *SF.net* is distributed in the following 6 months in a rather different way than the developers belonging to sample **C1**. Part of this difference is due to the definition of the subsamples: **C1** is composed of developers founding projects from period 4 to period 15, and who therefore have greater opportunities to reach states 4, 5 and 6 in greater proportions. But, the table also reveals – in the high proportion of those developers moving to state 2 – that during periods from 4 to 9 a great proportion of **C2**-type developers do tend to remain part of the project they have founded (or, if they leave the one they have founded, just join another project). In other words, their "exploratory drive" seems largely confined to first projects they launched after arriving on the platform (and which may well have had it origins in an externally formed software package). On the contrary, a high proportion of **C1**-type developers founded projects in periods 4-9. In particular, state 4 -- which corresponds roughly to being a member only in the project founded by the individual in question -- shows that a consistent portion of **C1**-type developers launch their projects in periods 4-9 without having founded any project during the previous three "months". This marks the difference between the two groups of developers.

Consider now how these initial distributions evolve over time, and in particular focus on the speed at which each process converges to the equilibrium. In Table 5, and in all the subsequent tables, we report the number of iteration $\boldsymbol{t}$ needed for each row of the transition matrix $\boldsymbol{P}^{\boldsymbol{t}}$ to match the limiting distribution (allowing 0.0009 as maximum discrepancy for each element). This can be considered a rather rough measure of the convergence speed, but it is intuitively clear and easy to implement computationally. Another more sophisticated measure has been used to check for robustness, as Appendix 2 explains.

When examining the number of iterations needed for the process to reach the equilibrium it is easy to see that developers belonging to sample **C2** are much slower in reaching the limiting distribution than the **C1**-type developers.[35] The interesting thing is that when **C1** and **C2**-types are pooled together in sample **B**, the number of iterations needed to reach the limiting distribution is very close to, and even a bit smaller than, the corresponding number for the **C1** sub-group. Even if those developers who founded a project in periods 4-15 comprise half of those who founded a project in periods 1-3, they appear to be those who really are driving the process, and that would qualify **C1**-type developers for recognition as the cadre responsible for the continuity of the *SF.net* "exploratory spirits".

Further results in the same vein can be extracted from Table 5. Among **C1**-type developers, the limiting proportions states 0 and 1 shrink in relation to the initial distribution to a point that the disappearance of "lurkers" is almost complete. **C2**-type developers, however, tend to be driven in the opposite direction, with the proportion of lurkers starting at a level close to zero and becoming larger in the limiting distribution. Moreover, the greater tendency of this latter group of developers to join existing projects rather than creating new ones is confirmed by the enlarged proportion that tend to wind up in state 3 – at the expense of state 2. In addition, it may be observed that **C1**-type developers confirm their "vocation" in that the proportion settling in state 5, at the expense of state 4, reflects their tendency to not only belong to multiple projects but go on launching new ones.

8. "*SourceForge* careers": a retrospective view of project joining and project founding

The analysis to this point has sought to describe current activity states and the distributions of the *SF.net* population among them. But career histories and their cumulative achievements also are of interest: individuals' careers are relevant to their acquisition of experience and reputations, and sometimes to their futures, even in circumstances where "history does not matter" in the strong sense of dynamic processes being "path dependent."[36] In what follows, the focus is explicitly historical or, more properly speaking,

[35] It has been found that the transition matrixes pertaining to sub-samples **B** and **C1** both have a single transient state, namely state 1. This jeopardizes the ergodic properties of the associated Markov processes. Moreover the fine grain of the state definition determines few **C2**-type developers falling in state 1 and 4 in epoch *A* of the relative transition matrix, so that the relative estimates are basically "weak". Eventually, again for sample **C1**, state 0 results in a very skewed distribution at the end of epoch *A*. Such "weak" result needs to be carefully judged and imply an additional set of checks for the robustness of the results. All these analysis are described in Appendix 3.

[36] See David (2001, 2007) on the formal conceptualization of path dependence in terms of non-ergodic dynamical

retrospective, in that the analysis focuses on the *cumulative* experiences of project joining and project launching among the developers on *SF.net* during these early years of the platform's collective history. Although it would be possible to use the current activity transition matrices to generate a stochastic simulation of the distributions of individual careers that are implied by the constant activity-state dependent transition probabilities obtained from the analysis already described, we here opt to construct a simpler descriptive apparatus. This approach defines a new set of "states" in terms of "cumulative career achievement," and estimates the corresponding transition probabilities for the resulting triangular matrices.[37]

8.1 Career dynamics: project-founding

Consider first developers' distribution with respect to the *cumulated* number of founded projects in the whole time span of the transition matrix (from period 4th to 15th). Ideally, using one state for each possible number of founded projects would be the perfect representation of the process, able to account for all the steps along developers' path towards a higher level of participation. However, when the number of founded projects is taken as the measure discriminating among the states we are going to build, aggregating those developers who found more than two projects is a necessary condition to assure that developers "falling" into the states relative to the highest number of founded projects are, both in terms of absolute number and as a percentage of the whole sample, enough to yield accurate probability estimates. This procedure also helps in setting the higher level of activity in terms that are not absolute, but with respect to the distribution of the foundation activity itself. This will turn out to be useful in subsequent comparisons.

Given this constraint, two definitions for the states are possible:

State space 1:
0=founder of 0 projects in periods 4-15;
1=founder of 1 project (p. 4-15);
2=founder of 2 projects (p. 4-15);
3=founder of more than 2 projects (p. 4-15).

State space 2:
0=founder of 0 projects (p. 4-15);
1=founder of at least 1 project (p. 4-15).

State space 1 takes into account the foundation activity at a finer grain, while states space 2 is useful in isolating the passage from being a non-founder to becoming one. Accordingly, two different transition matrices have been constructed to describe developers' movements in these career state spaces. In each matrix the initial state is assigned according to the number of projects developer *i* founds from period 4 to period 9, while her or his state in the next epoch is determined by the cumulated number of projects she or he has launched since the beginning of the process, i.e. from period 4 to period 15. Notice that these two periods are precisely epochs *A* and *B* used in section 3. Because this strategy inevitably produces triangular matrixes, it is likely to result in Markov chains that have an absorbing state at whatever is designated to be the highest recognized number of founded projects. Thus, limiting distributions are not relevant in this case. Instead, the focus here is on the properties of the process, including the speed with which the overwhelming proportion of initial cohorts of developers eventually reaches the absorbing state.

A point to be noted at the outset is that however the state is defined, and whatever the initial population cohort considered, the Markov chain for processes of this kind generates results with common features in terms of the shapes of curves tracing the distributions among the states over time. In particular, state 0 and the absorbing state have an inverse monotonic behavior, whereas all the other states initially increase their proportion of developers and then slowly decrease it towards 0. But those commonalities aside, the results of the analysis are quite different in terms of the initial distributions, their subsequent values, the number of iterations at which the highest proportion of developers is reached in each of the states, and the total number of iterations needed to reach the equilibrium where all the developers are in the absorbing state. These properties of the evolving distributions are summarized in Table 6.

Comparing the number of iterations the four samples considered above require to reach the absorbing state, a first conclusion is that those who found a project in the periods between 4 and 15 "move

---

systems – a condition that, as has been shown here, does not obtain in the case of the Markov chains describing the behavior of the developer population on *SF.net*.

[37] Triangularity of the transition matrix is implied by keeping cumulative track of events, which cannot reverse project joining and foundings: all progressions through the career achievement states must be unidirectionally upwards.

faster" than all the other groups. In the case of state space 1, the number of iterations (78) after which this group (**C1**) as a whole would have founded 3 or more projects is much smaller than the number needed for the whole population (749); and it is smaller than the number of iterations required by group **C2** (136). Moreover, the representative individual in group **C1** seems to develop greater "career momentum": when **C1** and **C2** are pooled together in **B**, the iterations required to reach the absorbing state decrease from 136 to 86, thus moving much closer to 78. This finding parallels the results obtained in the previous section, namely that those who founded projects later turn out to be more likely to found new projects in the future within a shorter interval, and to "lead the ranks" of all the founders on the platform in this subsequent exploratory phase.[38]

Table 6. Evolution of the distributions (number of projects founded): two alternative state spaces

State space	Sample	Property	States 0	**1**	2	**3**	Iterations to equil.
1	**A** *(whole population, n=222835)*	*Init. Distribution*	0.984	**0.014**	0.001	**0.000**	
		Value of the max	0.984	**0.069**	0.044	**1.000**	749
		Max at iteration	0	**23-24**	32-35	**749**	
	B *(founders in period 1 – 15, n=15825)*	*Init. Distribution*	0.782	**0.196**	0.017	**0.005**	
		Value of the max	0.782	**0.458**	0.227	**1.000**	86
		Max at iteration	0	**6**	12	**86**	
	C1 *(founders in period 4 – 15, n=5514)*	*Init. Distribution*	0.374	**0.563**	0.049	**0.014**	
		Value of the max	0.374	**0.766**	0.275	**1.000**	78
		Max at iteration	0	**1**	7	**78**	
	C2 *(founders in period 1 – 3, n=11875)*	*Init. Distribution*	0.918	**0.068**	0.010	**0.004**	
		Value of the max	0.918	**0.204**	0.108	**1.000**	136
		Max at iteration	0	**9**	14	**136**	
2	**A** *(whole population, n=222835)*	*Init. Distribution*	0.984	**0.016**			
		Value of the max	0.984	**1.000**			733
		Max at iteration	0	**733**			
	B *(founders in period 1 – 15, n=15825)*	*Init. Distribution*	0.782	**0.218**			
		Value of the max	0.782	**1.000**			38
		Max at iteration	0	**38**			
	C1 *(founders in period 4 – 15, n=5514)*	*Init. Distribution*	0.374	**0.626**			
		Value of the max	0.374	**1.000**			1
		Max at iteration	0	**1**			
	C2 *(founders in period 1 – 3, n=11875)*	*Init. Distribution*	0.918	**0.082**			
		Value of the max	0.918	**1.000**			124
		Max at iteration	0	**124**			

Compare now the result relative to the two state spaces, so that the properties of the passage becoming a founder and evolving to the role of "serial founder" (i.e. repeatedly launching new projects) become clear. Within state space 2, **C2**-type developers are more likely to subsequently be in the status of non-founders (state 0); but when they progress to founding new projects, they do so *en mass*. This can be seen from the difference between the 124 iterations needed to reach the equilibrium in state space 2, where every developer has founded at least one project, and the 136 iterations needed to bring every developer to

[38] Notice that the results for **C1** under the definition of state space 2 – in which only one iteration put the group in the absorbing state – is not informative, as it would appear to be merely a consequence of way that group **C1** has been defined.

the level of having founded more than 2 projects. That difference is much smaller than the one observed for **C1**-type developers (78 *vis-à-vis* 1). Moreover, as far as state space 1 is concerned, the maximum level of state 2 is reached earlier and with higher proportions of developers in the case of subsample **C1**. This is due to the specific definition of the subsample **C1**, which requires developers to have founded at least one project in the periods from 4 to the 15. But that "bias" is not off-set by a faster approach to the equilibrium: both sub-samples **C1** and **C2** reach the maximum of state 2 in one-tenth of the total number of iterations needed to arrive at the absorbing state. **C2**-type developers seems faster in climbing the stairs of "serial founding" than developers belonging to sample **C1**.

In sum, the foregoing analysis confirms this simple characterization of the difference between the sub-samples **C1** and **C2:** that the smaller sample of "late launchers" is the core of the "exploratory engine" of the community. However, it also characterizes their activity in a non-trivial way. These developers are more likely to start new projects thereafter, but at a lower rate than those very few "early launchers" who try to launch a new project for the second time. So, this turns out to be yet another story of the (few) early-starting turtles who are able to overtake the hesitant hares.

The lesson we learn from this is that those developers whose "exploratory spirit" appears to be crucial for the expansion of the knowledge basis of the *SF.net* community behave in peculiar manner within the platform environment. At the beginning they go through an initial phase of lurking. After a certain amount of time, they start exploring "piling up" new projects. From that moment on, their launching activity proceeds regularly, albeit at a slower average pace than developers with other histories and experiences.

8.2 Career dynamics: project-joining

Consider now the evolution of the *cumulated* number of projects joined by the developers. In this case all developers belonging to 5 or more projects have to be aggregated to assure, on the one hand, a fine grain state space and, on the other hand, a high enough number of developers falling in the state with the maximum number of project participations to produce meaningful estimations. As before, this also guarantees that the maximum level of activity is set on the basis of the underlying distribution of projects joined rather than in absolute terms. This in turn allows the comparison between the results obtained for the following space and the previous ones defined with respect to individuals' foundation activities.

Given this, the two new states spaces are:

State space 3:	State space 4:
0=member of 0 projects in periods 4-15;	*0= member of 0 projects (p. 4-15);*
1=member of 1 project (p. 4-15);	*1= member of at least 1 project (p. 4-15).*
2=member of 2 projects (p. 4-15);	
3=member of 3 projects (p. 4-15);	
4=member of 4 projects (p. 4-15);	
5=member of more than 4 projects (p. 1-15).	

The joining activity as defined by the two previous spaces will be compared to the process of new projects foundation using the same sample (**A**) and the same time periods (4-15). As a second step, the same spaces will be used to describe the evolution of the joining activity over the same periods but using another sample, **M**, composed by the population of 31,460 individuals who were members of at least one project within the periods from 1 to 15. The results will be compared to those of the subsample **C1** described above. The comparison should not be influenced by the environmental changes occurred over the period of analysis because sample **M** has a cohort composition very similar to that of **C1**, and actually its representation in Figure 3 would lie for the most part in between the two lines relative to **C1** and **C2**.

The following table summarizes the dynamic properties emerging from the application of the two state spaces to samples **A** and **M**. First of all, one should notice that the process of project-joining is faster (in terms of number of iterations) in bringing the whole population **A** into its absorbing state than the process of project founding. This result must be carefully interpreted as the two absorbing states are in a sense incomparable: project founding is a subset of project joining, inasmuch as every founder, by definition, is recorded as having joined the launched projects. Thus, only a portion of those who move from state 0 to the

other states in Table 7 are also among those who make the same transition in Table 6, even if the population **A** is the same. Given this, the observed difference in the transversal rate of approach to the respective absorbing states is to be expected. What is more interesting here is that a wider difference in the mean speeds of the approach to the absorbing states might have been anticipated, while the number of iterations needed to reach the equilibrium in the case of the launching activity are only one-sixth greater than those needed in the "membership" case. In this respect, the two processes do not seem to structurally differ.

Table 7. Evolution of the distributions (number of projects joined): two alternative state spaces.

State space	Sample	Property	States						Iterations to equil.
			0	**1**	2	**3**	4	**5**	
3	**A** *(whole population, n=222835)*	*Init. distribution*	0.872	**0.106**	0.016	**0.004**	0.001	**0.001**	
		Value of the max	0.872	**0.114**	0.051	**0.031**	0.026	**1**	631
		Max at iteration	0	**12**	22-23	**27-28**	30-32	**631**	
	M *(members in period 1-15, n=31,460)*	*Init. distribution*	0.091	**0.749**	0.116	**0.030**	0.009	**0.006**	
		Value of the max	0.091	**0.749**	0.216	**0.121**	0.097	**1**	117
		Max at iteration	0	**0**	7	**12**	15	**117**	
4	**A** *(whole population, n=222835)*	*Init. distribution*	0.872	**0.128**					
		Value of the max	0.872	**1**					603
		Max at iteration	0	**603**					
	M *(members in period 1-15, n=31,460)*	*Init. distribution*	0.091	**0.909**					
		Value of the max	0.872	**1**					6
		Max at iteration	0	**6**					

Again with respect to the whole population **A**, another result along the same path is obtained by comparing the differences between the two states spaces in the case of "joining" and "launching" projects: the proportionate difference between the iteration counts needed to reach the equilibrium in the first case, (631-603)/603 = .046 is not so much greater than the corresponding difference in the second case, (749-733)/733 = .022). What is highlighted here is the importance of the transition from non-activity to participation at some level of activity: once that occurs, developers are more or less on a par in ascending the respective scales of career achievement, both in terms of project-joining and project founding. Thus the two processes do not appear to be so different in these aspects of their dynamics.

Moving now to the sample composed by individuals who joined a project in period 1 - 15, the most interesting parallelism between **M** and **C1** is represented by the proportion between the number of iterations needed to reach the equilibrium in the two state spaces. When focusing on **M** it is possible to see that, while joining a first project requires just 6 iterations, acquiring multiple memberships is a much slower process,: it takes an almost 20 times longer period. This evolution resembles the one observed for sample **C1**, as a comparison between Tables 7 and 6 clearly shows. Thus, the projects foundation undertaken by those who seem to be the most active participants in the *SF.net* environment seems exhibit the same characteristics of the process driving potential members to join projects.

The conclusion is that exploration and exploitation, conceived here as project creation and joining, seem to be equally sustained by the dynamics observed on the platform. More than that, if the joining activity is taken to be the benchmark for a minimum level of FLOSS developers' participation, the results depicted above underscore the relative importance of the launching activity, and in particular of sample **C1**. In a world where dissipation is a prevalent tendency, the exploratory spirit of the *SF.net* virtual community is far from being inconsequential even though the recurrent project founders are a numerically small core within the mass.

9. Conclusions, qualifications and opportunities for further development

This study has presented a series of empirical findings exposing the quantitative dimensions of two fundamental micro-level processes at work in the *SourceForge.net* environment: the mobilization of development resources reflected in "project joining" and the creation of novelty reflected in "project founding." That pair of dynamic processes at the micro-level are fundamental components of any system of innovation, of which the FLOSS mode of production should be considered to represent a particular type, and the attributes of these processes may be said to characterize the basic conditions affecting the regime's *sustainability* as a system of innovation. A system in which innovators launch enterprises that fail to mobilize others to work on them, or one in which existing projects and developers migrate to a locale but no new undertakings are created there, must be renewed by external resource injections. Given the relative size and prominence of the *SF.net* platform in the universe of FLOSS activities, the findings based on its performance in these dimensions are not without interest for their bearing upon the question of the sustainability of this larger phenomenon, and the long-run development of the system of community peer-based production of which FLOSS is seen by many contemporary commentators to be a paradigm.[39]

The resources mobilization process and the developers' exploratory spirit were analyzed by building a series of Markov chains from the observation of 222,835 developers' activity on *SF.net* during 2001 and 2002. This methodology has been chosen not with the aim of producing predictions, but rather as an instrument allowing us to look at the deeper dynamic patterns of motion among the various activity states that developers can occupy, abstracting from the transient influences of the initial conditions that surrounded their respective entrances on the platform. Studying the limiting distributions among the activity states that are generated by the transition probability matrices reveals that in the collaborative development environment represented by the *SF.net* platform, the system tends to raise frequency of project joining and project founding activities, including the founding of multiple projects. Moreover, examination of the quasi-ergodic behavior of the project-founding process indicates that among the small group of developers that exhibit particularly pronounced "exploratory" spirits, the average propensity to create new projects remains relatively low during the initial phase of their experience on *SF.net*, but subsequently undergoes an increase that sustains regular creation of new projects, and even multiple project-launching activity.

True, the "project founders" whose activities constitute the critical *exploration* side of the FLOSS innovation process, represent only a small portion of the developers who belong to project groups on *SF.net*. But explorers are always "few among the many". Project launching is by no means different from all the other activities typical of a self-organizing system, where *dissipation* of the mobilized resources is a necessary condition to make the "fittest" emerge. In this instance the fact of their persistence in project launching activities, taken in conjunction with the ability to mobilize developers around exiting projects and thus to *exploit* the established technologies, should go some way to allaying doubts about the *sustainability* of the FLOSS mode of collective innovation – at least on these very basic grounds.

Our analysis approaches both the new project generating and resource mobilization processes from the vantage point of the individual actors, rather than that of the projects. This is a significant limitation, reflected in the fact that this study has not dealt with the characteristics of the projects that attract members, or the pattern of circulation of developers among projects of differing sizes, purposes, or governance arrangements. From one viewpoint, "project joining" is the outcome of both individuals' propensities for involvement in FLOSS development work of different kinds, of the relative strengths of motivations and preferences for participation in collective community undertakings and independent efforts, on one side.[40] Viewed from the other side, however, it reflects the relative abilities of different projects – and the collective capacity of the ensemble of projects that embrace the generic model of "the FLOSS way of working" – to mobilize the potential pool of development resources. The picture we have been able to present here is therefore quite one-sided, inasmuch as it has implicitly ascribed participation and project joining behaviors to the individuals, conditioning these on the anterior states they occupied; and, in so doing, has not

[39] E.g., Weber (2004), and more recently Benkler (2006:chs. 2-4). On the question of the long-term success and viability of open source software as a mode of production, one should also consult the software engineering perspectives presented in the contributions to Feller et al. (2005), and particularly the concerns discussed by Brian Fitzgerald (2005) in Chapter 5 of that volume. See also the brief treatment in David (2006).

[40] The latter are the subject of an extensive empirical and speculative discussions in economics and related social sciences, concerning the nature of the motivations of participants in FLOSS development activity, creating a literature to which both authors have contributed, but into which we have quite deliberately avoiding entering on this occasion. A significant portion of it can be accessed readily from: http://opensource.mit.edu/.

considered the possible effects of time varying influences arising from the evolving distribution on the *SF.net* platform of projects having different characteristics, and consequently differential attractiveness – both vis-à-vis one another, and the external world of FLOSS projects.

Similarly, the analysis of "project founding" has been restricted here to describing a propensity, and examining its association with the other dimensions of founders' behaviors within this particular environment. There are, of course, many other aspects of experience that might shape the behavior of founders, and these have not been recognized explicitly in the present discussion, much less analyzed. The success of the individual's previously launched projects in attracting members beyond the founding group surely is one factor whose influence on subsequent project-founding behaviors deserves closer study. Are founders impelled to start new projects by the failures of their previous efforts, or is the dominant effect that of encouragement by previous success – and the accompanying reputation for being able effectively to mobilize the support of others for launching a new venture? In principle it would be feasible to expand the state space of our Markov chain to consider the dependence of the probability of founding a second project on the "success status" of the individual's first project, and so on. At a minimum, extending the analysis in that direction also could address the question of whether or not those who founded projects after having been registered for several months on *SF.net* are more likely to create projects that successfully attract more than one or two members – compared with those individuals who launch a project when they arrive on the platform, or very shortly thereafter.

Quite clearly the Markovian framework for studying FLOSS production communities is amenable to elaboration in ways that would enrich the analysis and provide greater insights into micro-level dynamics that have been dealt with here only in a partial, essentially descriptive fashion. Fortunately, the availability of the *SF.net* archives has opened the further possibility of enquiring into the stationarity of these processes, or, alternatively, the evolution of the innovation potentialities of this particular FLOSS ecology. It is to be hoped that the results of this exploratory study will encourage others to pursue and improve upon the beginning that has been made here.

References

Amemiya, T. (1985), *Advanced Econometrics*. Basil Blackwell: Oxford.

Anderson, P. (1999), 'Complexity Theory and Organization Science,' *Organization Science* 10(3).

Anderson, T. W. and L. A. Goodman (1957), 'Statistical inference about Markov chains,' *Annals of Mathematical Statistics*, 28(1), 89-110.

Bagozzi, R.P. and U.M. Dholakia (2006), 'Open Source Software User Communities: A Study of Participation in Linux User Groups,' *Management Science*, 52(7):1099-1115.

Baxter, M. and R.G. King (1999), 'Measuring business cycles: approximate band-pass filter for economic time series,' *Review of Economics and Statistics*, 81(4):575-593.

Becattini, G. (2001), *The Caterpillar and the Butterfly. An Exemplary Case of Development in the Italy of the Industrial Districts*. Le Monnier: Florence.

Benkler, Y. (2006), *The Wealth of Networks: How Social Production Transforms Markets and Freedom,* New Haven: Yale University Press.

Bickenbach, F. and E. Bode (2001), 'Markov or not Markov - This should be a question,' Working Paper, No. 1086, Kiel Institute of World Economics.

Bode, E. (1998), 'Lokale Wissensdiffusion und regionale Divergenz in Deutschland', *Kieler Studien*, 293, Tübingen: Mohr.

Braunerhjelm, P. and M. P. Feldman (eds.) (2006), *Cluster Genesis: The origins and emergence of technology-based economic development*, Oxford: Oxford University Press.

Castilla, E.J., H. Hwang., H. Granovetter and M. Granovetter (2000), 'Social networks in Silicon Valley' in Lee, C-M., Miller, C.F., Hancock, M.C. and Rowen, H.R. (eds.) *The Silicon Valley Edge: A Habitat for Innovation and Entrepreneurship*, Stanford, CA: Stanford University Press: 218-247.

Cefis, E. (2003), 'Is there persistence in innovative activities?', *International Journal of Industrial Organization*, 21: 489–515

Christley S., and G. Madey (2007), 'Global and Temporal Analysis of Social Positions at SourceForge.net,' presented at The Third International Conference on Open Source Systems, IFIP WG 2.13, Limerick, Ireland, June 2007.

Christley, S. and G. Madey (2005), 'Collection of Activity Data for SourceForge Projects,' Technical Report, No. TR-2005-15, Department of Computer Science and Engineering, University of Notre Dame, October.

Comino, S., F.M. Manenti and M.L. Parisi (2005), 'From Planning to Mature: on the Determinants of Open Source Take Off,' Discussion paper, No. 2005/17, Università degli Studi di Trento.

Cox, D. R. (1962), *Renewal Theory*, New York: Barnes and Noble.

Crowston K., and J. Howison (2005), 'The social structure of Free and Open Source software development' *First Monday* 10(2). Available from: http://firstmonday.org/issues/issue10_2/.

Cyert, R. M., H. J. Davidson and G. L. Thompson (1962), 'Estimation of the Allowance for Doubtful Accounts by Markov Chains', *Management Science*, 8(3): 287-303.

Dahlander, L., and M.W. Wallin (2006), 'A man on the inside: Unlocking communities as complementary assets', *Research Policy*, 35: 1243–1259.

Dalle, J.-M. and P.A. David (2005), 'The Allocation of Software Development Resources in 'Open Source' Production Mode,' in J. Feller et al. (eds.), *Perspectives on Open Source and Free Software.* MIT Press: Cambridge MA. Preprint available at: http://siepr.stanford.edu/papers/pdf/02-27.pdf.

David, P. A. (1974). 'Fortune, Risk and the Micro-Economics of Migration,' in P. A. David and M. W. Reder, eds., *Households and Nations in Economic Growth, Essays in Honor of Moses Abramovitz,* New York: Academic Press: 21-88.

David, P. A. (2001). 'Path Dependence, its Critics, and the Quest for 'Historical Economics', in Garrouste P. and S. Ioannides (eds), *Evolution and Path Dependence in Economic Ideas: Past and Present*, Edward Elgar Publishing, Cheltenham, England.

David, P.A. (2006), 'A Multi-dimensional View of the "Sustainability" of Free & Open Source Software Development: Sustaining Commitment, Innovation and Maintainability with Growth,' Paper presented at the OSS Watch Conference on Open Source and Sustainability, Oxford. April 10-12. Revised as SIEPR Policy Paper 06-007 , Stanford University (August) [available at: http://siepr.stanford.edu/papers/pdf/06-07.html.]

David, P. A. (2007), 'Path Dependence – A Foundational Concept for Historical Social Science," *Cliometrica*, 1(2), July: 91-114. [Preprint available at: http://siepr.stanford.edu/papers/pdf/06-08.html]

David, P. A. and J. L. Rosenbloom (1990), 'Marshallian Factor Market Externalities and the Dynamics of Industrial Localization,' *Journal of Urban Economics*, 28: 349-370.

David, P. A. and J. S. Shapiro (2007), 'Who are the developers in community-based FLOSS projects, and why does that matter?,' Paper presented to the OSSEMP 2007 Workshop of the 3rd International Conference on Open Source Systems (OSS2007), University of Limerick, 14th June. (Revised version forthcoming at: http://siepr.stanford.edu/papers/).

David, P. A., D. Foray and J.-M. Dalle (1998), 'Marshallian Externalities and the Emergence and Spatial Stability of Technological Enclaves,' *Economics of Innovation and New Technologies* (Special Issue on Economics of Localized Technical Change, ed. C. Antonelli), 6 (2&3): 147-182.

Dosi, G. (1982), 'Technological paradigms and technological trajectories: A suggested interpretation of the determinants and directions of technical change,' *Research Policy*, 11(3): 147-162.

Ezcurra, R., P. Pascual and M. Rapùn (2006), 'The Dynamics of Industrial Concentration in the Regions of the European Union', *Growth and Change*, 37(2): 200–229

Feller, J., B. Fitzgerald, S. A. Hissam and K. R. Lakhani, eds. (2005), *Perspectives on Open Source Software*. Cambridge, MA: MIT Press.

Fershtman, C. and N. Gandal (2004), 'The Determinants of Output Per Contributor in Open Source Projects: An Empirical Examination,' Discussion paper, No. 4329, CEPR, London.

Fingleton, B. (1997), 'Specification and testing of Markov chain models: an application to convergence in the European union', *Oxford Bulletin of Economics and Statistics*, 59(3): 385-403.

Fitzgerald, B. (2005), 'Has Open Source a Future?', Ch. 5 in J. Feller, *et al*., eds. (2005), *Perspectives on Open Source Software***.** Cambridge, MA: MIT Press.

Giuri, P., F. Rullani and S. Torrisi (2006b), 'Explaining Leadership in Open Source Software Projects', presented at the Pre-conference of the 13th Anniversary Organization Science Winter Conference, 7-8 February, 2007, Steamboat Springs, CO, USA.

Giuri, P., G. Rocchetti and S. Torrisi (2002), 'Open source software: from open science to new marketing models an enquiry into the economics and management of open source software,' LEM working paper series, 2002/23. Available from: http://www.lem.sssup.it/WPLem/files/2002-23.pdf.

Giuri, P., M. Ploner, F. Rullani and S. Torrisi (2006a), 'Skills, Division of Labor and Performance in Collective Inventions. Evidence from the Open Source Software', LEM working paper series, 2004/19, July. Available from: http://www.lem.sssup.it/WPLem/files/2004-19.pdf.

Glott, R. (2004),"Towards integration of research approaches to FLOSS communities," Oxford Workshop on Libre Software (OWLS), Oxford Internet Institute, June 25-26, 2004. [Available at: www.oii.ox.ac.uk/fiveowlsgohoot/postevent/owls-presentation_r.glott.pdf .]

Glott, R., R. A. Ghosh and B. Krieger (2004), "Motivations of free/libre and open source developers," International Infonomics Institute Working Paper, University of Maastricht (May).

Hamilton, J. D. (1994), *Time Series Analysis*. Princeton, NJ: Princeton University Press

Herraiz, I., G. Robles, J.J. Amor, T. Romera and J.M. González-Barahona (2006), 'The processes of joining in global distributed software projects,' *Proceedings of the 28th International Conference on Software Engineering*, Shanghai, China, 20-28 May 2006. Available from: http://libresoft.urjc.es/Contact/index_html.

Howison, J. and K. Crowston (2004), 'The perils and pitfalls of mining SourceForge,' in *Proceedings of Workshop on Mining Software Repositories*. International Conference on Software Engineering: Edinburgh, Scotland. May 25.

Howison, J., M. S. Conklin and K. Crowston (2005), 'OSSmole: A collaborative repository for FLOSS research data and analysis,' in *Proceedings of the 1st International Conference on Open Source Systems (OSS)*: Genova, Italy. July 11-15.

Hunt, F. and P. Johnson (2002), 'On the Pareto distribution of SourceForge projects,' in *Proceedings of the Open Source Software Development Workshop*: Newcastle, UK. February 25-26, pp. 122-129.

Jensen, C. and W. Scacchi (2005), 'Modeling recruitment and role migration processes in OSSD projects,' in *Proceedings of 6th International Workshop on Software Process Simulation and Modeling*: St. Louis, USA, May 14-15.

Klincewicz K. (2005),'Innovativeness of open source software projects', Working Paper, Tokyo Institute of Technology, August, Tokio, JP.

Konings, J. and F. Roodhooft (1997), 'Financial ratio cross-section dynamics: a non-paramentric approach', *Journal of Business Finance and Accounting*, 24(9) & (10): 1331-1342.

Kremer M., A. Onatski and J. Stock (2001), 'Searching for Prosperity,' NBER Working Paper, No. 8250, April, Cambridge, MA.

Krishnamurthy, S. (2002), 'Cave or Community? An Empirical Examination of 100 Mature Open Source Projects,' *First Monday* 7(6). Available from http://firstmonday.org/issues/issue7_6/.

Lécuyer, C. (2005), *Making Silicon Valley: Innovation and the Growth of High-Tech, 1930-1970*. Cambridge MA: MIT Press.

Lee, G.K. and R.E. Cole (2003), 'From a Firm-Based to a Community-Based Model of Knowledge Creation: The Case of the Linux Kernel Development', *Organization Science*, 14(6): 633–649.

Lerner, J. and J. Tirole (2004), 'The Economics of Technology Sharing: Open Source and Beyond,' NBER Working Paper, No. 10956, December, Cambridge, MA.

Lerner, J. and J. Tirole (2005), 'The Scope of Open Source Licensing,' *Journal of Law, Economics, and Organization*, 21, 20-56.

Lerner, J., P. A. Pathak and J. Tirole (2006), 'The Dynamics of Open-Source Contributers,' American Economic Review, 96(2), May: 114-118.

Madey G., V. Freeh and R. Tynan (2004), 'Modelling the free/open source software community: A quantitative investigation,' in S. Koch (ed.), *Free/Open Source Software Development*. Idea Group Publishing: Hershey, PA.

Maggioni, M. A. (2004), 'The Dynamics of Open Source Software Communities and Industrial Districts: the Role of Market and Non-Market Interactions,' *Revue d'Economie Industrielle*, 107: 127-150.

March, J. G. (1991), 'Exploration and Exploitation in Organizational Learning', *Organization Science*, 2(1): 71-87, Special Issue on Organizational Learning: Papers in Honor of (and by) James G. March.

Mateos Garcia, J. and W. E. Steinmueller (2003), 'The Open Source Way of Working: A New Paradigm for the Division of Labour in Software Development?,' Working Paper, No 1(January), SPRU, Science and Technology Policy Research, Open Source Movement Research, INK: Brighton. Available from http://www.sussex.ac.uk/spru/publications/imprint/sewps/sewp92/sewp92.pdf.

McCall, J.J. (1971), 'A Markovian model of income dynamics', *Journal of American Statistical Association*, 66(335): 439-447.

Mockus, A., R.T. Fielding and J. Herbsleb (2000), 'A case study of open source software development: the Apache server,' *Proceedings of the 22nd International Conference on Software Engineering*, Limerick, Ireland, ACM Press, 263-272.

Nelson, C. R. and C. I. Plosser (1982), 'Trends and random walks in macroeconomic time series,' *Journal of Monetary Economics*, 10: 129 - 162.

Olsson, O., and B.S. Frey (2002), "Entrepreneurship as Recombinant Growth", *Small Business Economics*,19: 69–80.

Page, S. E. (2007), *The Difference: How the Power of Diversity Creates Better Groups, Firms, Schools, and Societies***,** Princeton, N.J.: Princeton University Press.

Pyke, R. (1961), "Markov renewal processes: Definitions and preliminary properties," *Annals of Mathematical Statistics*, 32: 1231-1242.

Quah, D. (1993), 'Empirical cross-section dynamics in economic growth,' *European Economic Review*, 37: 426-434.

Rainer, A. and S. Gale (2005), 'Evaluating the Quality and the Quantity of Data on open Source Software Projects,' in *Proceedings of the 1st International Conference on Open Source Systems (OSS):* Genova, Italy. July 11-15.

Robles, G. and J. M. Gonzalez-Barahona (2006), 'Geographic Location of Developers at SourceForge,' in *MSR 2006 International Workshop on Mining Software Repositories*, ICSE 2006: Shanghai, China, May 22-23.

Rosenkopf, L. and A. Nerkar. (2001), 'Beyond Local Search: Boundary-Spanning, Exploration, and Impact in the Optical Disk Industry', *Strategic Management Journal*, 22(4): 287-306.

Rullani F. (2006a), 'The debate and the community. 'Reflexive identity' in the FLOSS community', presented at the workshop: Socio-technical Dynamics in the Free/Libre Open Source Software (FLOSS) Social World, January 19 – 20, Helsinki, Finland.

Rullani, F. (2006b), 'Dragging developers towards the core. How the Free/Libre/Open Source Software community enhances developers' contribution', presented at the CCC Thirteenth Annual Colloquium for Doctoral Student Research, May 19 - 21, 2006, Lausanne, Switzerland.

Saxenian, A. (1994), *Regional Advantage: Culture and Competition in Silicon Valley and Route 128*. Harvard University Press: Cambridge, MA.

Shah, S. (2006), 'Motivation, Governance, and the Viability of Hybrid Forms in Open Source Software Development,' *Management Science*, 52(7).

Sheps, M.C. and J. A. Menken (1973), *Mathematical Models of Conception and Birth*, Chicago: University of Chicago Press.

Shorrocks, A. F. (1978), 'The Measurement of Mobility', *Econometrica*, vol. 46 number 5, September: 1013-1024.

Shorrocks, A.F. (1976), 'Income mobility and the Markov Assumption', *The Economic Journal*, 86(343): 566-578.

von Krogh, G., S. Spaeth and K. R. Lakhani (2003), "Community joining, and specialization in open source software innovation: A case study," *Research Policy*, 32: 1227-1241.

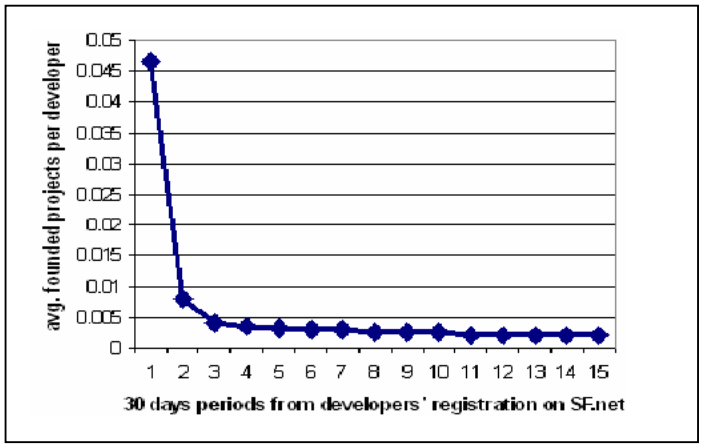
Weber, S. (2004), *The Success of Open Source*. Cambridge, MA: Harvard Business School Press.

Appendix 1: Setting the time constants of the Markov chain

Launching new projects

Figure A1-1 shows the average number of projects founded in all the successive 30-day intervals within the period of observation by the 222,835 individuals in the sample. The peculiar importance of the period immediately following entry to *SF.net* is apparent.

Figure A1-1. Average Number of founded projects per developer in each 'month'



A simple explanation of this can be given by exploiting the metaphor of the district described in the text (section 1.2). Regions are not only areas where the autochthonous resources are mobilized, but also are potential attractors of already established enterprises. Existing firms can decide to relocate in those areas simply because they see the knowledge-spillover externalities and agglomeration economies to be advantageous. Similarly, platforms where FLOSS is developed can be attractors for already existing projects. Developers and team leaders can decide to move FLOSS projects they are working on to the platform in order to benefit from the agglomeration economies and the facilities offered by the platform itself. When this is the case, we observe developers entering *SF.net* and suddenly establishing a project. The project is not new in itself, but new to the platform. Indeed existing source code may simply be made visible and re-leased under an open source license. This practice is relatively diffuse. Figure A1-1 suggests that developers are likely to enter *SF.net* bringing with them the projects they were carrying on *independently* of *SF.net*. In this case, the foundation activity is not a proper one, and should not be considered in our analysis.

Analyzing the behavior along all the 28 'months' of their experience in *SF.net* of the sub-sample of 1289 developers registered from September the 1st to October the 30th, 2000, who also founded a project in the first period does not change the results.[41] The average number of founded projects per 30-day 'month' is reduced from 1.071 to 0.049 in the first 60 days, and thereafter decreases only slightly, oscillating between 0.026 and 0.006. Thus, in order to detect the cycle, we need to focus on this last series dropping the first and second months. The Augmented Dickey-Fuller unit-root test is performed on the resulting series and suggests it is non-stationary[42]. The series is then detrended. Since we have no reason to believe that the source of non-stationarity is a stochastic trend, nor to opt for a deterministic trend, we consider both cases and detrend the series using differences and also polynomial trend removal (see, among others, Baxter, King, 1999: Nelson, Plosser, 1982). Considering the commonalities between the series resulting from

[41] To detect individuals' propensities to found new projects, in determining the cycles of project-launching we compute how many projects each user has founded every 30 days *regardless* of the situation at the end of each 'month', i.e. we do not take into account whether the founded projects are still "alive" nor if the founder is still part of them at the end of the 30 days. This way, we are able to account for the periodicity of project foundation irrespective of the actual realization and possible success of the projects previously put forward.

[42] In all the following analysis we use the Augmented Dickey-Fuller unit-root test without constant and without trend. This specification has been preferred over the other possible choices because in this context, where spectral density is employed to detect the most important cycles, as being able to detect and clear all the possible trends is crucial. In order to increase our insight into the series' behavior, however, the test also has been performed by introducing into the regression the first and the second lags, also by using different specifications (with the constant and with the constant and the trend). These results then have been compared to the main ones, in order to confirm or qualify our conclusions on the question of stationarity. The findings reported in the text reflects the outcome of this decision process.

first-differencing (**DS1**) *and* the series resulting from the removal of the polynomial trend of order 1 (**POLY1**) enables us to avoid the possible bias induced by the choice of a method that does not match the nature of the series. Both first difference and linear trend removal result in a stationary series, as shown by the Augmented Dickey-Fuller test, even if this result is less robust for the latter case. Eventually, the spectral analysis of both the detrended series is computed and the relative peaks are ranked in terms of their importance, as shown in Table A1-1.

Table A1-1. Cycles of Project Foundings

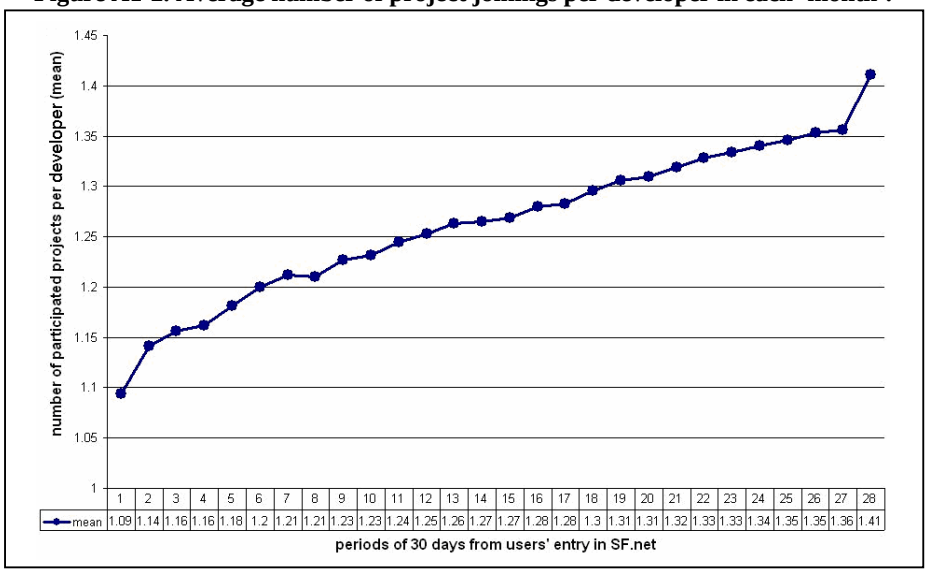
N peak	**POLY1**	Rank	Days	**DS1**	Rank	days
1	0.30	1st	628	0.51	4th	365
2	3.14	2nd	60	3.14	1st	60
3	1.70	3rd	110	1.75	2nd	107
4	1.15	4th	163	1.15	3rd	163

The spectral densities reveal that there are 4 main cycles for both (de-trended) time series, and that they are more or less congruent between the two series. The frequencies show that the structure of cycles is not very coherent, so that we have to use some degree of approximation in evaluating what time period is the best for our purpose. Notice that in most cases the longer waves emerge only when the spectral density is drawn with a low level of smoothness. To see which of these results hold also for modifications of the time window, we also compute the spectral density of **DS1** and **POLY1** for the last year (namely, from the 17th to the 28th month) and in both cases we obtain two cycles, one long about 170 days and another one spanning about 70 days. Similarly, we check the robustness of the results by computing the spectral density for the original time series, but keeping in mind our previous results on its stationarity. The results, *mutatis mutandis*, are very close to those obtained for **DS1**.

Participating in the projects

A similar sub-sample and a similar analysis can be carried out in the case of project membership. In this case, the extremes of the series appear to behave differently from the whole series, as Figure A1-2 suggests. In the last of the observed 30-day periods (the 28th 'month') the increase is due to the different composition of the sample. Those who registered in month 2 cannot be observed in their 28th period, and thus the average "jumps" to the higher value of the first cohort. In the first period, the explanation of the lower average number of founded projects can be found in the transformation from continuous to discrete time: while during other periods each individual's behavior is observed for 30 days, in her or his first period on *SF.net* the number of days of observation depends on the moment of her or his registration on *SF.net* with respect to the date used as the limit for the monthly interval. Thus, if the monthly interval spans from September the 1st to October the 1st, 2000, and the individual registers on September the 29th, her or his first period spans just 2 days. Hence, the lower average number of projects being founded. To avoid the biases induced by this problem, in the following we cut the extreme values of the series and restrict the analysis to periods 2-27.

Figure A1-2. Average number of project joinings per developer in each 'month'.



Developers registered from September the 1st to October the 30th, 2000, conditional on being a project member in the first 30-day 'month' their registration on SF.net.

Applying the same procedure undertaken before, we first study the stationarity of the series. The Augmented Dickey-Fuller test suggests that the series is non stationary. We thus compute **DS1** and **POLY1**. Both the series are stationary at the 5% level but results vary a lot between different specifications. Moreover the spectral density of **DS1** appears to be heavily influenced by a flat trend. Thus, we detrend the original series both twice-differencing it (and obtaining **DS2**) and fitting a polynomial of order 2 (whose residuals are called **POLY2**). Both series are stationary at 1% level, even if the a certain level of instability is still observed for **POLY2**. The space of the relative spectral densities and their peaks is drawn in Table A1-2.

Table A1-2. Cycles of Project-Joinings

N peak	**POLY2**	Rank	days	**DS2**	Rank	days
1	0.40	1st	471	-	-	-
2	0.98	2nd	192	1.13	3rd	167
3	2.00	3rd	94	1.96	2nd	96
4	3.00	4th	62	2.97	1st	63

It is easy to see that Table A1-2 resembles Table A1-1 when short cycles are considered: cycles close to 60 and 90/110 days are shared by both series in both tables. Also the 160-day cycle enters the picture for most of the series. For longer cycles, however, the variability is higher, but this is expected given the short length of the time window at our disposal. Similar results are obtained when considering just the last year, from the 17th to the 28th month. In this case **DS2** shows basically one wave around 70 days, while **POLY2** placed the most important frequencies from 60 to 110 days. Given the lack of clear-cut results on stationarity, we also compute the spectral density for **DS1** and **POLY1**. In **DS1** the relative importance of the longer waves increases, as expected, and the 160-day cycle stretches up to 200-250 days, according to the level of smoothness used to compute the spectral density. The other results reported in the table are by and large confirmed.

Remarks

Bearing in mind that the window to observe the registrants only spans the months from September 2000 to December 2002, i.e. 28 months, adopting a long period for the transition matrix (say, one year) would seriously limit this analysis. The longer the period we choose, however, the more the cycles we are able to account for in the transition matrix. Given this constraint, choosing time spans of 6 months seems to be the compromise best suited to the circumstances. The 180-day window for events on the basis of which the individuals' activity states can be determined should amply allow for influences of antecedent short cycles, given the fact that it embraces the 60-day cycle, the 90/110-day cycles, and even the 160-day cycle – with small biases with respect to their lengths. Moreover, this choice is consistent with the fact that the shorter cycles are precisely those which are shared by all the analyzed series. Thus, periods of 6 months seem to be adequate to generate a first order Markov process.

Appendix 2: Asymptotic half-life of convergence

The analysis presented in the text is carried out considering the number of iteration needed for each considered Markov process to reach the limiting distribution. In the literature it is often considered another measure of the speed of the process, called *asymptotic half-life of convergence* (Shorrocks, 1978). In the present framework, this measure represents the number of 6-month periods needed to reduce by half the norm of the difference between the limiting and the initial distributions. The formula by which the asymptotic half-life of convergence is computed is:

$$h = -\frac{\log(2)}{\log|\lambda_2|}$$

where $|\lambda_2|$ is computed ordering the modules of the eigen values associated to the transition matrix and considering the second largest one.

Table A2-1 compares the asymptotic half-life of convergence and the iterations needed for the Markov process of each sample to reach the limiting distribution. As it is possible to see from the table, the proportion between the two measures is almost constant and equal to one tenth. Thus, since all the results we have described in the text are based on the proportion between the numbers of iterations of different samples, they still hold even when only the asymptotic half-life of convergence is considered. Notice that the constant proportion implies that for every subsample the dynamics at which the process converges is always the same: there is a quick adjustment towards the limiting distribution at the beginning and then a "fine tuning phase" which takes a much longer time to be completed.

Table A2-1. Asymptotic half-life of convergence and number of iteration to the limiting distribution

Asymptotic half-life of convergence (*h*)	Iterations to equilibrium (*T*)	**Proportion** $\left(\frac{h}{T}\right)$	Sample	**# developers**
17.40	178	**0.098**	*Whole population*	**222,835**
14.96	148	**0.101**	*Cohorts from 11 to 14*	**79,983**
4.41	43	**0.103**	*Founders [1 – 15]*	**15,825**
22.89	224	**0.102**	*Founders [1 – 3]*	**11,875**
4.96	47	**0.106**	*Founders [4 – 15]*	**5,514**
73.47	750	**0.098**	*Whole population; cumulative, state space 1*	**222,835**
6.76	87	**0.078**	*Founders [1 – 15]; cumulative, state space 1*	**15,825**
12.36	137	**0.090**	*Founders [1 – 3]; cumulative, state space 1*	**11,875**
6.76	79	**0.086**	*Founders [4 – 15]; cumulative, state space 1*	**5,514**
10.01	117	**0.085**	*Members; cumulative, state space 1*	**31,460**
73.47	734	**0.100**	*Whole population; cumulative, state space 2*	**222,835**
3.81	39	**0.098**	*Founders [1 – 15]; cumulative, state space 2*	**15,825**
12.36	125	**0.099**	*Founders [1 – 3]; cumulative, state space 2*	**11,875**
0.47	6	**0.078**	*Members; cumulative, state space 2*	**31,460**

Appendix 3: Treatment of the anomalies encountered during transition matrix estimation

Effects of the presence of a transient state

When a transition matrix leads to the emergence of a transient state, the existence of a limiting distribution is not assured. In the present case, state 1 turns out to be transient when transition probabilities are estimated from samples **B** and **C1**.

The easiest and fastest way to check if this process is able to reach equilibrium is simply to calculate the powers of the matrixes. The 43rd and the 47th power of those matrixes, respectively, are composed of identical rows (all equal to the limiting distributions shown in table 5) and they remain constant for all successive powers. This may be seen directly from the fact that, for those powers $(\boldsymbol{P}^t \bullet \boldsymbol{P}) = \boldsymbol{P}^t$., which assures the existence of a limiting distribution independent of the initial one.

This latter statement can be confirmed by considering that for an initial distribution $\underline{\boldsymbol{i}}$, where $i_1+\ldots+ i_n=1$ (i.e. we express the distribution in proportions), and a transition matrix whose t^{th} power $\boldsymbol{P}^t$ has all identical rows, the distribution of the process at time t is $\underline{\boldsymbol{l}}=\underline{\boldsymbol{i}}\bullet \boldsymbol{P}^t$.

Thus, $l_j=i_1\bullet p^t_{1j}+i_2\bullet p^t_{2j}+\ldots+i_n\bullet p^t_{nj}$ and since $p^t_{1j}=p^t_{2j}=\ldots=p^t_{nj}=p^t_j$, then $l_j=p^t_j(i_1+i_2+\ldots+i_n)$.

Given the definition of $\underline{\boldsymbol{i}}$, this implies $l_j=p^t_j$ for every j^{th} column of $\boldsymbol{P}^t$ and entry of $\underline{\mathbf{l}}$.

In other words, the j^{th} entry of the limiting distribution is obtained by summing the same proportion (indicated by the j^{th} column of the matrix) for all elements of the initial distribution, so that in the limit the proportion of individuals in each state is totally independent of the initial distribution and depends only on the transition probabilities.

Simulations "around" weak estimates

The estimates relating to subsample **C2** show exhibit two different sets of anomalies. On the one hand in *epoch A* states 1 and 4 attract only very few developers, so that the estimates for the relative transition probabilities are unreliable. This result is due to the fact that the fine-grain definition of states is coupled with a subsample composed only by those who found a project in the periods from 4 to 15, and they are small in number. Another side effect of the limited number of developers populating these states is that the resulting estimates tend to become extremely skewed.

In order to cope with this problem, we estimated the relative transition matrix and computed the associated limiting distribution excluding both states, one at a time and together. Further we ran a series of simulations in which the distribution of state 1 and of state 4 were smoothed, thereby redistributing some of the mass centered in states (1;0) and (4;2), respectively, to the other states more evenly.

In some cases the results are qualitatively the same as those described in the text, in some others they move even further along the path we depicted. For example, state 1 seems to have a limited influence on the behavior of the whole process. When running the simulations described above with respect to this state, the resulting limiting distributions are not very sensitive to the changes in state 1's transition probabilities, even when the state is excluded *in toto* from the state space. On the contrary, state 4 exerts a much stronger influence on the dynamics of the Markov chain, so that when it is left out of the space, it limits even further the progression of **C2**-type developers towards the most active states of the distribution. This strengthens our results.

A similar problem has been found with respect to state 0 and the same sample **C2**. Users in state 0 in epoch *A* tend to cluster in the cell (0;0), creating a skewed distribution. Moreover, in order to be in this cell, developers had to found at least one project in period 1-3 and then to drop back to the non-active state for the remainder of their registered stay on *SF.net*. This situation is very peculiar, and needed to be handled with care because it could simply be the result of some of the biases affecting the data described in the beginning of the empirical analysis. We therefore have undertaken two kinds of studies to assess the seriousness of the problem.

First, we double-checked the history of a sample of developers in cell (0;0) of the transition matrix under analysis. Some of the observations were in fact generated by problems in the data. For example, a relatively small percentage of those who have multiple *user_id*'s seem to abandon the old *user_id* after their first failure of a newly founded project. But none of the problems that have been detected, once accounted for, generate relevant differences in the estimates. Moreover, further inspection of a sample of these developers has shown that there are some individuals who simply decide not to undertake any other activity after having abandoned the project they initially founded, or having seen it shut down by the *SF.net* staff after few months of activity. Thus, the results are robust to this particular problem in tracking the identity of individuals. .

Secondly, state 0 generates a highly skewed distribution -- so much so that it almost resembles an absorbing state. Even though the data generating this distribution have been double-checked, in order to be perfectly sure of the soundness of the empirical results we run some simulations smoothing the distribution of the transition probabilities associated with an initial state equal to 0. The result is that the emerging limiting distribution leads to the same conclusions described in the text. It is worth noticing, however, that differences emerged during this exercise with respect to states 5 and 0. While in the analysis described in the text the percentage of developers in state 5 decreases from 6.8% to 5.9% in equilibrium, when 20% of the developers in cell (0;0) of the transition matrix are distributed evenly in the other states, state 5 is represented by 6.9% of the population in the resulting limiting distribution, showing an increase in the number of developers founding one new project and being member of more than one existing project. Correspondingly, in equilibrium the fraction of developers placed in state 0 is increased by 0.131 in the context described in the text, whereas the increase is only by 0.015 when running the simulation.