Transactions in GIS

Transactions in GIS, 2012, 16(4): 581–597

Research Article

Mapping Collaboration in Open Source Geospatial Ecosystem

Jianhua Shao
Horizon Doctorial Training Centre
University of Nottingham

George Kuk
Business School
University of Nottingham

Suchith Anand
Centre for Geospatial Science
University of Nottingham

Jeremy G. Morley
Centre for Geospatial Science
University of Nottingham

Mike J. Jackson
Centre for Geospatial Science
University of Nottingham

Tyler Mitchell
Open Source Geospatial Foundation

[Correction added on 06 September 2012 after initial online publication on 2 August 2012: The ordering of the author names has been amended in this online version of the article. The correct order is Jianhua Shao, George Kuk, Suchith Anand, Jeremy G. Morley, Mike J. Jackson and Tyler Mitchell.]

Abstract

Over the last decade, there has been a tremendous growth and exploitation of open source geospatial software and technologies. A combination of factors is driving this momentum, including the contributions made by hundreds of developers and the leading role played by the Open Source Geospatial Foundation (OSGeo), aiming primarily to support and promote the collaborative development of open source geospatial technologies and data. This article seeks to map out the social history of collaborative activities within the OSGeo ecosystem. We used the archival logs of developers' contributions, specifically looking for boundary spanning activities where contributions crossed multiple projects. The analysis and visualization of these activities allow us to have a better understanding of the role of boundary spanning in the resourcing of each project, the incubation mechanism advocated by OSGeo, and the significance of the social interrelatedness among projects. The data consisted of the subversion (SVN) commit history made by individual developers in the programming code repository. We applied several network analytical and visualization techniques to explore the data. Our findings indicate that more than one in seven developers

Address for correspondence: Jianhua Shao, Horizon Doctoral Training Centre, School of Computer Science, Jubilee Campus, University of Nottingham, Nottingham, NG8 1BB, UK. E-mail: psxjs4@nottingham.ac.uk

doi: 10.1111/j.1467-9671.2012.01305.x

spanned multiple projects which potentially had the effects of shaping the projects' directions, and increased knowledge flow and innovation. In addition, the OSGeo's incubation mechanism provided an important encouragement for boundary spanning and increased knowledge sharing. By studying the social history of contributions, further tools can be developed in future to assist tracking of the social history, and make developers mindful of the significance of the interdependence among projects and hence continuously contribute to the health of the OSGeo ecosystem.

1 Introduction

Open source development attracts a great number of participants from various backgrounds. Participants perform different roles to push forward the development cycles, including code developers, software testers, policy makers, and normal users (Mockus et al. 2000). They contribute to and influence development within open source communities, and themselves benefit by being involved in this ecosystem (Lerner and Tirole 2002). Open source geospatial communities are a large group showing rapid growth in the last decade (Steiniger and Bocher 2009). A more recent development among some communities is to have a formalized organization to help assist projects developing under its umbrella. In the geospatial open source domain, this organization is the Open Source Geospatial Foundation (OSGeo).

OSGeo (http://www.osgeo.org/) is a non-profit, non-governmental organization whose mission is to support and promote the collaborative development of open geospatial technologies and data (Mitchell 2010). The foundation was formed in February 2006 to provide financial, organizational, and legal support to the broader free and open source geospatial community. It also serves as an independent legal entity to which community members can contribute code, funding and other resources, secure in the knowledge that their contributions will be maintained for public benefit. OSGeo provides a common forum and shared infrastructure for improvising cross-project collaboration.

OSGeo provides an umbrella organization with multiple sub-projects applying to join (Mitchell 2010). It is different from other open source communities in terms of the governance structure and the use of incubation as a decision mechanism. The decisions to sanction new projects are largely subject to an incubation mechanism, where OSGeo incubators have the responsibility of preparing and recommending newly joined projects to the OSGeo Board of Directors (Christl 2010). The incubators essentially review applications and appoint mentors for incubating projects. Yet there is no mandate that incubated projects have to work collaboratively with other existing OSGeo projects as each project can develop separately and in some instances, compete with others for developers. Although anecdotal evidence exists suggesting that interaction among projects is common, and that innovation and knowledge can be transferred and shared across OSGeo projects, it is not clear whether the interactions are largely restricted to the technical dependency among projects, for example, similar projects using the same library of codes. We know less about the social aspects of OSGeo ecosystem relative to the technical interrelatedness of projects through code reuse (Sanz 2009).

It is vital to understand that the collaboration among projects can positively affect the health of the ecosystem. Independent development can only deter wider adoption by the industry, as it can have deleterious effects. For example, in assessing the impact of developer activities on the known security vulnerabilities in the Red Hat Enterprise Linux

4 kernel, Meneely and Williams (2009) found that too many developers contributing codes independently, without subjecting their codes to rigorous peer review, can lead to submission of vulnerable software patches. Open source projects that are developed in tandem with each other are more likely to be innovative and attractive to developers (Kuk 2010). This often requires developers to be mindful with each other's contributions, and reuse quality patch of codes that others have developed (Maillart et al. 2008). This article seeks to examine the interrelatedness among projects based on the ways developers collaboratively contribute to projects. Our premise is that having developers that contribute to and crisscross multiple projects can increase the liquidity of knowledge flow and sharing, and collectively move innovation forward.

The remainder of the article is set out as follows. Section 2 outlines the data collection processes giving details about the database structure and a summary of the SVN data analysed. Data analysis is described in Section 3, with results from social network analysis, development effort overlap, boundary spanning and the impact of OSGeo incubation. Section 4 gives a summary of the work, looking into the limitations of the study and future work.

2 Data Collection

Open source software development leaves a rich trace of developers' online actions and interactions, including those who directly contribute to the code repository of a project. Often they make brief comments to underline the changes that they made to the code base or commits, identify the places that code could be improved and in some cases discuss how the code may be used in practice (Kogut and Metiu 2001). These comments serve the purpose of communicating to other developers in the project, providing references and pointers for further contributions and collaboration. The developers' commits to the project reflect not only the technical contributions but also the social and collaborative aspect of those contributions. For this article, we used three publicly available data sources to explore the technical and social aspects of contributions by OSGeo developers. They are: Subversion (SVN) code repository commits; mailing lists; and issue trackers (Gutwin et al. 2004). SVN commits are used to track individual and aggregate contributions of source code and are taken directly from a project's source code repository. Most projects in OSGeo use SVN to track and manage source code development, and it makes it easy to publicly fetch for analysis (see Table 1). The OSGeo communities also use mailing lists to communicate with each other for a broad range of purposes. Each project has its own set of mailing lists and the OSGeo communities share a general set of mailing lists for cross-project or organizational discussion. Issue trackers are another data source available for studying the interactions among community members. People report, create and assign bugs, tasks and feature requests within communities, at various scales, from individual functions that are project-specific to the shared code libraries for the whole OSGeo ecosystem. Here we focus on SVN commits as our data source, as they are easiest to access and understand. Further research will consider the other two types of data sources.

SVN commits reflect the development history of an individual project. They also reflect development interaction among developers and/or users from different projects (Perera et al. 2007, Sowe et al. 2008). SVN is a software versioning and a revision control system (Collins-Sussman et al. 2004). Developers use SVN to maintain current and historical versions of files in projects, such as source code and documentation. SVN

Table 1 The list of SVN Commits (October 1998 to April 2011). The number of developers is the number over time (including those that left now)

Project	SVN URL	Commits	Developers	Developers on multi-projects	Boundary spanning rate %
deegree	https://svn.wald.intevation.org/svn/deegree/	9624	26	1	3
geomajas	https://svn.geomajas.org/majas/	6915	16	3	18
GeoServer	http://svn.codehaus.org/geoserver/	12103	58	29	50
MapBender	https://svn.osgeo.org/mapbender/	7723	38	3	7
MapBuilder	http://svn.codehaus.org/mapbuilder/	3111	30	10	33
MapFish	http://www.mapfish.org/svn/mapfish/	3758	31	14	45
MapGuide	https://svn.osgeo.org/mapguide/	5072	59	18	30
MapServer	http://svn.osgeo.org/mapserver/	11089	53	18	33
OpenLayer	http://svn.openlayers.org/	4654	15	10	66
GRASS GIS	https://svn.osgeo.org/grass/grass/	39941	74	8	10
Quantum GIS	https://svn.osgeo.org/qgis/	14514	50	9	18
gvSIG	http://subversion.gvsig.org/gvSIG/	31441	79	4	5
FDO	http://svn.osgeo.org/fdo/	5029	39	17	43
GDAL/OGR	http://svn.osgeo.org/gdal/	21345	74	24	32
GEOS	http://svn.osgeo.org/geos/	3275	17	10	58
GeoTools	http://svn.osgeo.org/geotools/	24164	113	34	30
MetaCRS	http://svn.osgeo.org/metacrs/	1991	30	21	70
OSSIM	http://svn.osgeo.org/ossim/	17600	27	5	18
PostGIS	http://svn.osgeo.org/postgis/	6324	23	9	39
GeoNetwork	https://geonetwork.svn.sourceforge.net/	2831	17	1	5

maintains versioning for file directories, source files, and file metadata. There are many other versioning control tools, like CVS (Cederqvist and Pesch 1993) and Git (Torvalds and Hamano 2010), but SVN is, currently the most popular tool (Hammond et al. 2011). The OSGeo communities have used SVN widely; nearly all OSGeo projects use SVN or provide a SVN mirror. A commit, in the context of SVN, refers to submitting the latest changes of the source code to the repository, making these changes part of the head version of the repository and then allowing them to be synchronized with other users. Therefore, people make commits when they make any change to the source code, and the changing records reflect how that project's source code has developed. Thus it is a great source for analysing the process and for gauging the relations between the technical and the social aspects of coding in OSGeo.

For SVN commits, there are three separate folders in each project: trunk, tags and branches (Collins-Sussman et al. 2004). The trunk folder holds the main body of development, from the start of the project until the present. It generally has all the newest features. The tags folder is a point in time on the trunk or a branch that individual developers wish to preserve. The two main reasons for preservation would be that either this is a major release of the software, or this is the main stable point of the software before major revisions on the trunk were applied. The branches folder holds a copy of code derived from a certain point in the trunk that is used for applying major changes to the code while preserving the integrity of the code in the trunk. It always represents a smaller release and new features. SVN uses a standard method for locating those three sources. It all starts with a root repository URL, and then follows with the resource name. Therefore, once we know one resource address, it is easy to figure out the other resource addresses. For example, the commit path for the *PostGIS* project's trunk is http://svn.osgeo.org/postgis/trunk.

Each SVN commit message is well structured and we can retrieve it as a consistent rich data source. SVN Commits are stored internally within the repository but can be output in XML format using a command similar to the following:

```
svn log svn_path –v –xml > output_file_name
```

Each commit includes data such as revision, author, date, path and message. Table 2 shows an example from the *deegree* project. Each commit is a *logentry* message. Each *logentry* will show the unique revision ID within the project; it will also record who made the change to the code and what time it happened. SVN commits also record what files have been changed and the actions, such as adding, deleting or modifying a file. Developers usually leave a meaningful, human-readable message for each change. All SVN commits data sources use this same format to host the message, regardless of whether it is trunk, tags or branches. The only difference is where those commits come from within the resource URL, and the *msg* tag which will have a different message. As with the message demonstrated below, we can easily understand that it is a commit from a tag by looking at the *msg* element.

We designed and implemented a data collection server to aggregate statistics on the public development activities within the OSGeo community. Based on the SVN commits standard, we designed and implemented a database as shown in Figure 1. It was designed to be easy to add new commit logs and provide quick query searching. This database is also easy to adapt when we study new data sources. The design is implemented using SQLite3 (http://www.sqlite.org/) for its flexibility. The data collection server was written in the Python language (http://www.python.org/) and has been running and stable for

Table 2 An example to show how SVN Commits look

```
<?xml version="1.0"?>
  <log>
  <logentry>
    <revision="29280">
    <author>jwilden</author>
    <date>2011-01-13T09:08:15.390041Z</date>
    <paths>
      <path kind=""copyfrom-rev="29279"action="A"
        copyfrom-path="/base/branches/2.5_testing" >
        base/tags/2.5-rc1
      </path>
    </paths>
    <msg>Tagged the module to version 2.5-rc1 </msg>
  </logentry>
<log>
```

more than three months from February 2011. For this article we studied only the mature projects that had already passed the period of the OSGeo Incubation. At the time of writing this article, the database file had passed 300 MB in size with more than 300,000 SVN commits records. These records were historic information from the SVN repositories of each OSGeo project.

3 Data Analysis

Through some basic analysis of the data, we found there are about 700 developers who have been actively contributing to the source code development now and in the past, and left their names and/or user ID. It is reasonable to expect that there is a much larger number of people who use the software but do not contribute to the source code. More importantly, we find about one in seven of the active developers contribute to two or more projects (see Figure 2). About 80% of the developers working on multiple projects only work on two projects. A small number of developers work on more than five projects. We even found one developer working for seven different projects and maintaining his activities within the wider OSGeo communities (see Figure 2). To study the social history of collaboration, we are interested in analysing developers' crisscrossing activities with and between projects. We describe the analysis process and results below from four successive angles: social network analysis; development effort overlap; boundary spanning; and the impact of OSGeo incubation.

3.1 Social Network Analysis

To understand the social aspect of contributions, we used the methodology of social network analysis (Scott 2000, Lopez-Fernandez et al. 2004). Social network analysis applies graph theory to map out the social ties among contributors. A simple network is an

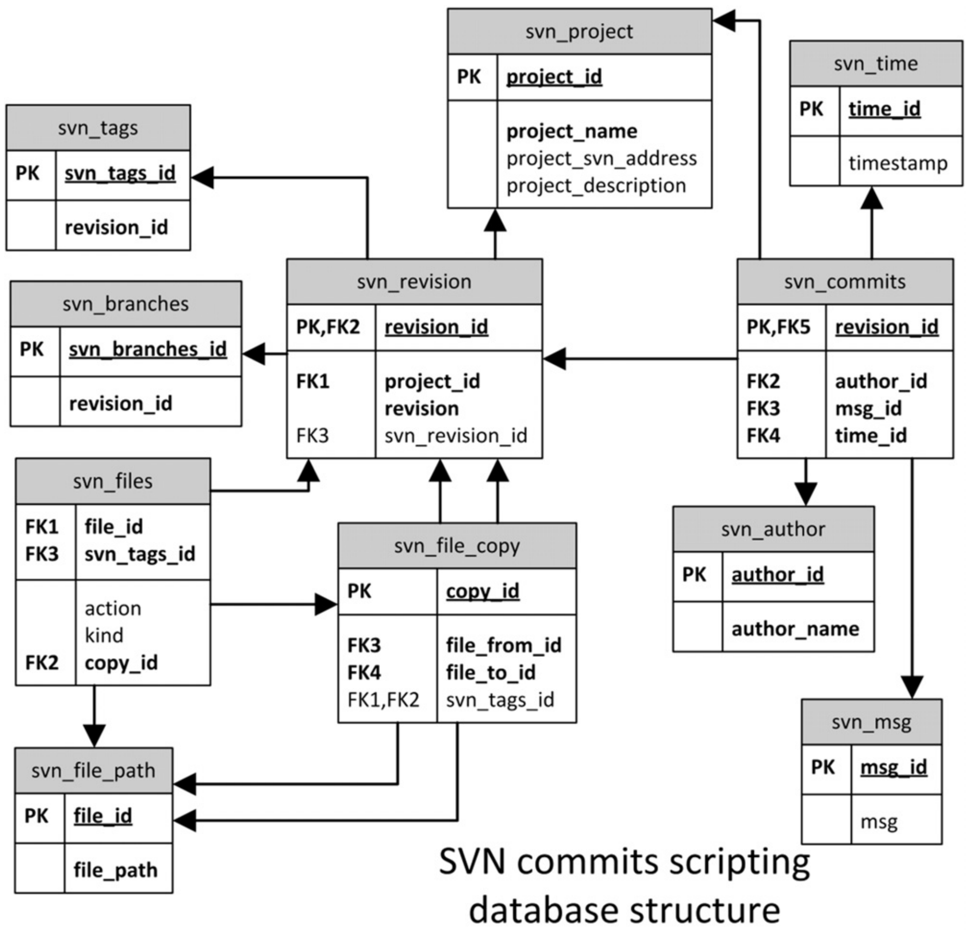

Figure 1 A schematic diagram of the SVN commits database

unigraph comprising of actors as nodes, tied or connected by one or more specific types of interdependency, such as common interest, knowledge sharing, and in this case, commits by developers to files, packages, and/or modules of a specific project. We can present the contributions to projects as a bi-graph with one node representing the developers and the other the project (consisting of either code patch, files, packages, or modules). Their ties are commits made by developers. Whether it is unigraph or bigraph, the resulting graph-based structure is often very complex and there can be many kinds of ties among the nodes. Research in a number of academic fields has shown social networks operate on many levels (Barnes 1954, Barthes and Duisit 1975), from families up to the level of nations, and play a critical role in determining the way problems are solved, organizations are run, and the degree to which individuals succeed in achieving their goals. The small world phenomenon (Watts and Strogatz 1998) is the hypothesis we are interested in for this research, in that the chain of social acquaintances connecting one arbitrary person to another arbitrary person anywhere in the world is generally short. In this theory, the network comprises a number of cohesive groups or connected components. They are connected to each other via bridges or "short-cuts".

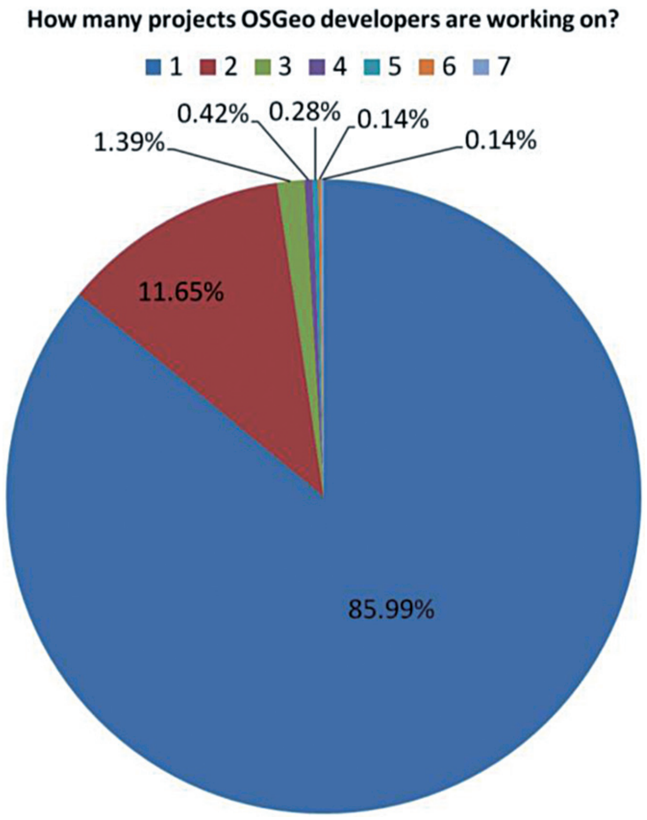

Figure 2 Distribution of around 700 developers by the number of OSGeo projects they contribute to. 85.99% of developers work on single projects and 11.65% work on two projects, 1.39% of developer work on three projects in parallel, 0.42% work on four projects, 0.28% work on five projects, 0.14% of developers are working on six projects, and another 0.14% work on seven project

Social networks may reflect on knowledge innovation and transformation (Müller-Prothmann 2006). The shape of a social network helps determine a network's usefulness to its individuals. Smaller, tighter networks can be less useful to their members than networks with lots of loose connections (weak ties) to individuals outside the main network (Granovetter 1973). More open networks, with many weak ties and social connections, are more likely to introduce new ideas and opportunities to their members than closed networks with many redundant ties. In other words, a group of friends who do things with each other already only share the same knowledge and opportunities. A group of individuals with connections to other social groups is likely to have access to a wider range of information. It is better for individual success to have connections to a variety of networks rather than many connections within a single network. Similarly, individuals can exercise influence, or act as brokers, within their social networks by bridging two networks that are not directly linked. The OSGeo community has a social network that is interesting to study. It is, firstly, working as a substantial foundation with many sub-projects, and then each project attracts many people who contribute. Mean-

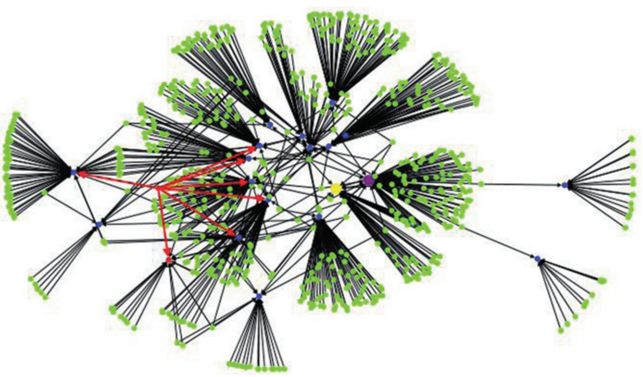

Figure 3 Diagram to show the relationship between contributor and projects. Each green dot represents a developer who contributes to the source code in one project and a blue dot marks each individual OSGeo project. Red arrows are a sample relationship showing how one developer contributes on seven different projects (see Table 2). Yellow and purple dots identify two projects, *GeoTools* and *GeoServer*. Because they have dependency relationships, we find many contributors boundary spans both projects

while, many people are communicating across different projects, even outside of the OSGeo community. The social network relationship in OSGeo should naturally show diversity, but they all aim for the same target: to encourage the use and development of OSGeo software. These reasons have attracted us to study the social collaboration within OSGeo.

3.2 Development Effort Overlap

We can view the connection between projects by looking at the relationship of developers to projects. Developers are definitely connecting to a project if they contributed to project source code development – they have to understand existing code before committing code. Therefore, individual developers may also establish connections with each other if they contribute to the same part of the source code with operations like create, copy, delete, rewrite and modify. Based on this assumption, we created Figure 3, which is a diagram based on historic SVN commit data that represents the relationship of developers to projects. Each green dot represents an individual developer and each blue dot indicates one individual project. The edge, or tie, bridging a green dot and a blue dot means the developer has contributed to the project. This figure does not include any edge weights, but this is analysed in Section 3.3 with weights calculated from how much contribution a developer has made to a project.

By looking at the diagram (Figure 3), we can see that most developers concentrate on one project, but some of them spread their efforts across several projects. Unlike other kinds of contribution, code contribution requires professional coding skills, good understanding of existing source code, and much time invested in the application area. Therefore, concentration on one project makes development work much easier, and this is common in traditional open source development. However, it is slightly different in OSGeo, as nearly all OSGeo projects have multiple developers working on other projects and those developers connect different projects together. Those developers distribute their development time into different projects, and more importantly we find that they transfer knowledge and innovation between projects. For example, the *GeoServer* project is partly built based on the *GeoTools* project. There are a number of developers working on both projects. They would import libraries from the *GeoTools* project into *GeoServer* projects, like data access, rendering and referencing. Meanwhile, those developers would also transform functional demands from the *GeoServer* project into the *GeoTools* project, like efficiency of data rendering and support for a new data format. Because OSGeo is a big umbrella over a number of open source projects, it provides an opportunity for developers to communicate and share new ideas and information, so it makes this kind of overlapping development effort more common compared with other open source communities, like the Linux Foundation (http://www.linuxfoundation.org/) and the Apache Foundation (http://www.apache.org/). The Linux Foundation is a single-source open source community where most development efforts are implemented around the Linux operating system. Even though there are several projects on going, it is still considered as a single project because those projects integrate with each other so much (Welsh 1994). The Apache Foundation is the opposite: it also holds many projects, but those projects are loosely connected (Roberts et al. 2006). OSGeo's operation lies in the middle. It has many projects, but those projects are not considered as components of a giant system; they are all excellent in their own application areas. Those projects are also not loosely connected, as they all address geospatial subjects. This unique phenomenon significantly encourages the communication between projects in the OSGeo community.

In addition, this network diagram (Figure 3) also illustrates a cohesion distance between the projects (represented as blue dots in the diagram). The cohesion distance is represented by the distance between the pairs of project dots. This distance is calculated based on the Harel-Koren Fast Multiscale algorithm (Harel and Koren 2001) to present the interaction of development between projects. Projects in the middle of the diagram, which have more open connections to other projects, have a much shorter distance than individual, closed projects away from the middle of diagram. Openness here means how much connection it has to others, which could be project software dependencies and also participants' involvement. The open projects normally also have more participants involved in development. OSGeo projects have dependencies between projects, for example the relationship between the *GeoServer* and *GeoTools* projects, and so we see much interaction between clustered projects. Yellow and purple dots in the diagram represent the *GeoTools* and *GeoServer* projects. The *GeoServer* project is heavily dependent on data access, rendering and referencing system libraries from *GeoTools*. *GeoServer* developers have a policy to contribute back to *GeoTools* whatever code has the potential to be shared with other projects. We can also view this relationship from the diagram. Their distance is very short and many developers connect both projects. This contributes greatly to boundary spanning which will be discussed more in Section 3.3.

Because of the project dependencies and similarity in objectives between projects, participants have more opportunities to contribute within OSGeo, as they can more easily become involved in different projects. In the middle of the diagram, some developers (represented by green dots) contribute to several projects, acting as hubs for project-to-project connectivity. Those projects are closer to each other and they normally have more contributors. In turn, such short project distances would also increase the possibility of people becoming involved in different projects. Because those projects share similarities in development, the developers should find it easier to apply their knowledge in a different project. The diagram also shows that developers working on different projects are more commonly clustered in the middle of diagram, close to the projects with more open connectivity. The red arrows are an example demonstrating that one developer is committing to seven projects and is near to this central cluster. It is interesting to see, in Section 3.3, what value these individuals make by connecting different projects.

3.3 Boundary Spanning

Boundary spanning describes the phenomena whereby people divide their time between different projects, linking the development between different projects across the "boundaries" of projects (Daft 2009). The linking could be information exchange, relationship building, and using "boundary objects" as a way to create shared meaning and trust across the boundary (Williams 2002). It has been studied several times as a topic in leadership (Yip et al. 2009). Here we have applied this to the context of OSGeo community. From the literature, boundary spanning is concerned with the detection of information. It has two primary roles (Adams 1976). One is to detect information and bring it into the organization. The second is to send information into the environment, presenting the organization in a favourable light. Essentially the boundary spanners perform the role of a gatekeeper, through their crisscrossing activities; they facilitate the in- and out-flow of information and knowledge including technical design, development and insights between projects.

From the analysis in the last section, we find that boundary spanning is common in the OSGeo community. Around one in seven OSGeo developers have experienced, or are experiencing, boundary spanning by contributing to multiple projects. Most of them have a primary project to work on, in that they will contribute most of their time primarily to a single project. Yet, for many reasons, they spend part of their time on several other projects. Boundary spanners serve strategic roles in OSGeo projects by gathering critical information, obtaining feedback and perception from other projects through their networks and then interpreting and translating by filtering the most relevant and significant information and knowledge back to their primary projects. For example, some boundary spanners in *GeoServer* projects would bring feedback and expectation back to their primary project, *GeoTools*, to support the development of the additional library functions necessary for *GeoServer*. Boundary spanning can lead to innovation in strategy, processes or products (Ansett 2005). We summarize the key activities of the boundary spanner in OSGeo as:

1. Creating internal and external networks for open source projects;
2. Acting as the identifier for solution development;
3. Translating the knowledge back into the project culture; and
4. Influencing and educating internal and external users of projects.

Even though boundary spanning brings many benefits, boundary spanning still has trade-offs, especially in OSGeo code contribution. Because code contribution takes professional skills and project familiarity, it is a time-consuming activity. If people work on too many projects, they may not be able to contribute as significantly on a single project or even all projects at an average level. However, when people concentrate on single projects, they more easily sharpen their skills within the project. Open source collaboration is different from what occurs in other commercial organizations, since developers may be more readily motivated by reputation. Boundary spanning connects to such a motivation, helping to enumerate a developer's contribution to innovation and

Table 3 Metrics to show how a real developer distributes his development contribution into seven different projects

Project name	SVN Commits count
MapGuide	1
MapServer	398
Quantum GIS	2
FDO	69
GDAL/OGR	2603
GEOS	26
MetaCRS	194

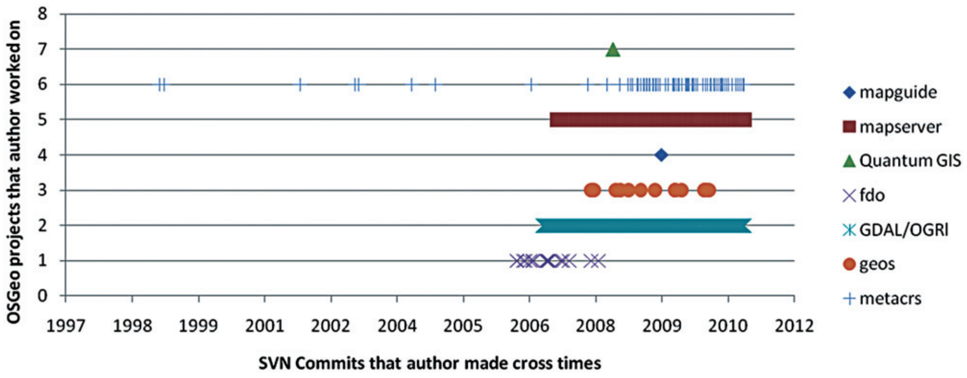

Figure 4 Boundary spanning diagram for a person who contributes to seven OSGeo projects

knowledge transformation across projects. In practice, boundary spanning would be evaluated indirectly over time. Studying OSGeo's boundary spanning can help us understand the open source development organization and motivation. Table 3 and Figure 4 demonstrate the metrics for one boundary spanning developer who has been working on seven projects. From the statistics, we can see his primary project is GDAL, but he also spends significant effort on *MapServer* and other projects. GDAL (Geospatial Data Abstraction Library) is an access and translation library for geospatial data formats. *MapServer* supports rendering maps from various data sources, using, in particular, the *GDAL* library. As with other projects, this person's boundary spans different projects to transform the knowledge and innovation among them, thereby reducing development costs and increasing the development speed and code reuse across projects.

3.4 Incubation

The incubation process also distinguishes the OSGeo community from other open source communities. The purpose of the OSGeo incubation process is to ensure projects follow the basic official OSGeo guidelines on Open Source software development (Christl 2010).

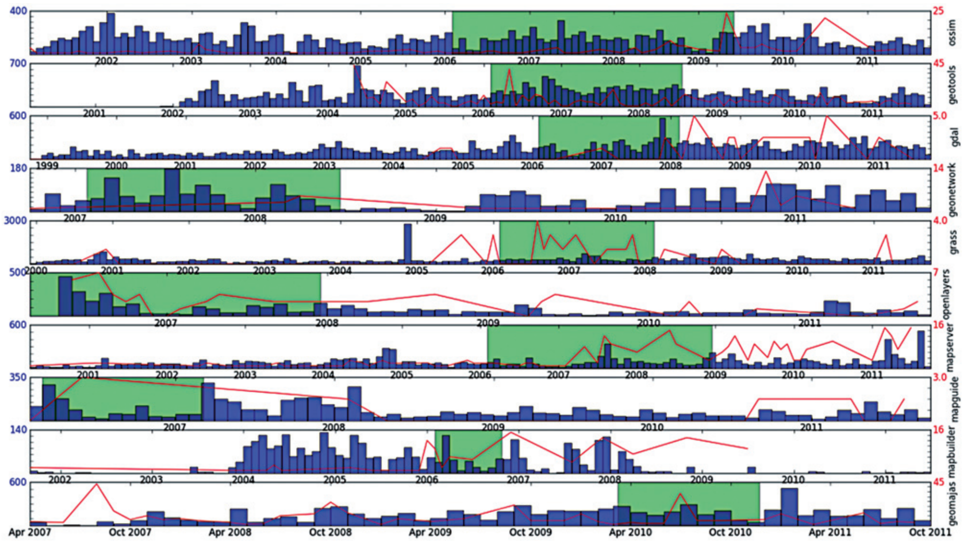

Figure 5 Diagram to show SVN trunk and tags commits distribution over time for selected mature OSGeo projects. Each row is a subplot to represent the SVN activities from one project. Each subplot includes two sub charts. The blue bar chart stands for SVN trunk commits in one project. The red line chart stands for SVN tags commits in one project

It means all OSGeo projects would have a successfully operating, open and collaborative development community to lead each project through the incubation process. Open Source software projects originate from very different backgrounds and cover all types of programming environments. Some OSGeo projects have been Open Source from the start and evolved over many years like the GRASS project that started more than 25 years ago. Other projects now published under an Open Source license were initially managed by a single commercial entity, for example MapGuide Open Source where Autodesk changed the development and governance structures so that the code was opened to the broader community. To become a successfully incubated project, the project team should manage themselves, striving for consensus and encouraging participation from all contributors, from beginning users to advanced developers. Contributors are the scarce resource. OSGeo encouraged successful incubated projects to share their contributors. Projects are also encouraged to adopt open standards and collaborate with other OSGeo projects; they are responsible for reviewing and controlling their code bases to ensure the integrity of the open source baselines.

By measuring the SVN commits of all OSGeo projects, based on timeline, we can prove that most OSGeo projects enjoy the benefits from OSGeo incubation with resource sharing. An individual project manages its development roadmap, but also shares the communication channel with other projects within the OSGeo ecosystem. By looking at Figure 5, we can see that most OSGeo projects have more active participants during the incubation (green block in Figure 5) which has a increase of SVN commits which means direct contributions to coding. By looking at the release of a product (red line in Figure 5), we can also see a larger increase in output appearing during the period immediately before, during or after incubation. When OSGeo projects graduate from incubation there are two

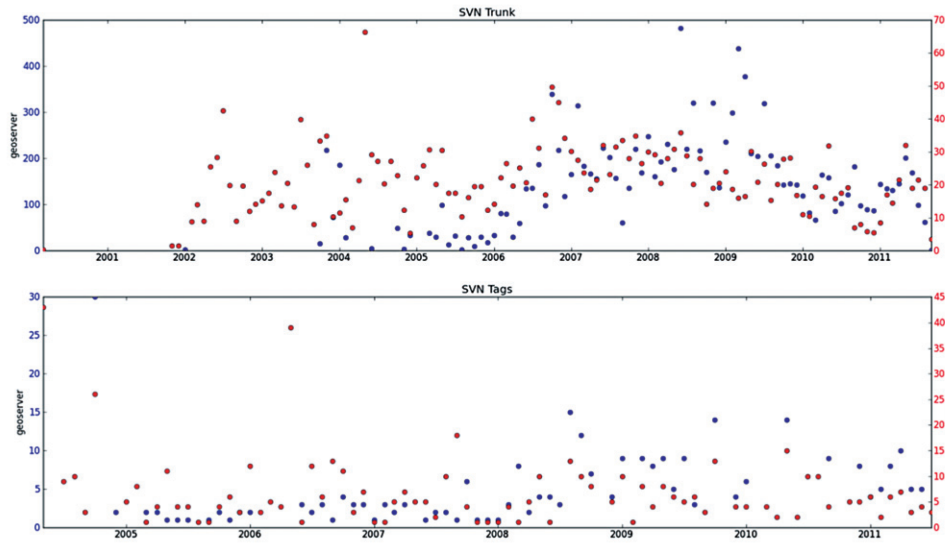

Figure 6 Distribution of SVN trunk and tags for GeoServer and GeoTools project over time. Blue dot represents the SVN commits in month for GeoServer and red dot represents GeoTools project

patterns for continued project development. One pattern is to continue with high, or even higher, code development outputs through the development of further functionality. The other pattern is of decreasing output as most bugs and functions have been developed before graduation from incubation. Some projects started a long time before incubation through OSGeo was available. With a successful set of projects that had graduated from incubation, OSGeo attracted many other projects that joined in incubation, which contributes to the continued growth of the resources available through a form of snowballing effect. We also measured the ratio of coding contribution and project output, especially for projects that have dependencies with each other, like GeoServer and GeoTools (see Figure 6). We find those projects have similar development output rates both in SVN Trunk and SVN Tags. The findings suggest that incubation is instrumental in creating a social space of mature projects, helping the incubated projects, and concentrating resources and interactions. The boundary spanning activities also provide the conduits to ensure that mature and incubated projects are developed in tandem.

4 Conclusions and Future Work

The aim of this project is to explore the social and collaborative aspect of contributions in the OSGeo community. Even though this article only introduces the project and the initial work being carried out, further research activities have been planned and more activities will be introduced as data collection and analysis improve. At present, we only collect SVN commits as a starting point to understand the social history of collaboration in resourcing the development and vitality of the OSGeo community. SVN commits records the history of project code evaluation. Yet SVN commits cannot help us measure the project delivery in a more precise way. There are three reasons for this:

1. In practice, the SVN commits do not necessarily identify real people. We can find the SVN commits' user names but this is not a match to a real name and we cannot guarantee that multiple users do not share the same account. SVN commit accounts are not open to all people who want to contribute to the codes. Therefore, we are missing some people who may want to contribute but are not recorded in the central repository.
2. Different understanding on project output standards. Small releases are treated as important outputs in some projects. Because a version number is controlled by a project's administrator, there is a wide variation between projects in the understanding of version division. We believe future study is needed to explore this further. One uncompleted analysis to be reported in a further article is temporal emergence. This is to study the patterns in how people spend time on open source development. It helps to understand the development trajectory of open source.
3. Due to limitations in the scope of the data. SVN commits only record people who directly contribute to coding, therefore we can only track the programmers in OSGeo. However, there are many people who would contribute to the development of source code indirectly, like source code testers and product users, who help to identify the places where source code could be improved. OSGeo is not only code developer driven, but also influenced by other contributors. Therefore, SVN commits is only a starting point as a data source. We plan to collect mailing list and issue tracker data to enrich our data source in the next phase. The study model introduced in this article can also be applied to studies with those two sources. The particular value of this further research is to build a model to visualize the evolution of open source communities like OSGeo.

In summary, this article shares the results of a multidisciplinary research project at the University of Nottingham to study the social history of collaboration in the various communities in the Open Source Geospatial Foundation (OSGeo with more than 20 collaborative open source projects). This article seeks to map out the social history of collaborative activities within the OSGeo ecosystem. It describes the data collection methodology used on SVN commits from each mutual project to reflect user activities in OSGeo. The project has built a server written in Python to capture a synchronous picture of the development evolution of the whole OSGeo community. We used the archival logs of developers' contributions, specifically looking for boundary spanning activities where contributions crossed multiple projects. The analysis and visualization of these activities allow us to have a better understanding of the role of boundary spanning in the resourcing of each project, the incubation mechanism advocated by OSGeo, and the significance of the social interrelatedness among projects. We applied several network analytical and visualization techniques to explore the data. Our findings indicate that more than one in seven developers spanned multiple projects which potentially had the effects of shaping the projects' directions, and increased knowledge flow and innovation. The OSGeo's incubation mechanism also provided an important encouragement for boundary spanning and increased knowledge sharing. By studying the social history of contributions, further tools can be developed in future to assist in tracking the social history, make developers mindful of the significance of the interdependence among projects, and hence continuously contribute to the health of the OSGeo ecosystem.

Most of those who boundary span work on two projects, but we also find some boundary spanners are able to manage contributions to a larger numbers of projects.

Although core developers can sharpen their professional contribution within projects, we find boundary spanners can contribute innovation transformation and knowledge-sharing among the OSGeo community. OSGeo incubation provides a critical period to concentrate attention and resources; and, through crisscrossing by a small percentage of developers, mature and incubated projects can develop in tandem and increase the inflow and outflow of knowledge. This concentration of resources including a small percentage of developers crisscrossing multiple projects is instrumental to the vitality and success of open source software development (Kuk 2006). The approaches introduced in this article could be applied to other public data sources, such as mailing lists and issue trackers.

For future work, we plan to continue to build a searchable index to reflect the social history of collaboration and allow search and identification of the key boundary spanners within the OSGeo community. In doing so, through understanding the social history of code contribution and collaboration, it will accrue a range of benefits including expediting the identification of critical resources, bug fixes and code maintenance, and the integration of products/services for adoption for the future.

Acknowledgements

The project is supported by the Horizon Doctoral Training Centre at the University of Nottingham (RCUK Grant No. EP/G037574/1), and was carried out through internship funding through the Open Source Geospatial Lab at the Centre for Geospatial Science. We also thank the Ordnance Survey for funding Suchith Anand's research through the Future Data project.

References

Adams J S 1976 The structure and dynamics of behaviour in organizational boundary roles. In Dunnette M (ed) *Handbook of Industrial and Organizational Psychology*. Chicago, IL, Rand McNally: 1175–99

Ansett S 2005 Boundary spanner: The gatekeeper of innovation in partnerships. *Accountability Forum* 6: 36–44

Barnes J A 1954 Class and committees in a Norwegian island parish. *Human Relations* 7: 39–58

Barthes R and Duisit L 1975 An introduction to the structural analysis of narrative. *New Literary History* 6: 237–72

Cederqvist P and Pesch R 1993 *Version Management with CVS*. Linkoping, Sweden, Signum Support AB

Christl A 2010 The state of OSGeo and the global SDI. In *Proceedings of IV Jornadas de SIG Libe*, Gerona, Spain

Collins-Sussman B, Fitzpatrick B W, and Pilato C M 2004 *Version Control with Subversion*. Sebastopol, CA, O'Reilly Media

Daft R L 2009 *Organization Theory and Design*. Cincinnati, OH, South-Western College Publishing

Granovetter M S 1973 The strength of weak ties. *American Journal of Sociology* 78: 1360–80

Gutwin C, Penner R, and Schneider K 2004 Group awareness in distributed software development. In *Proceedings of the ACM Conference on Computer Supported Cooperative Work (CSCW 2004)*, Chicago, Illinois: 72–81

Hammond J S, Gilpin M, and Knoll A 2011 *The State of Application Development in Enterprises and SMBs: Forrsights North America and Europe*. Cambridge, MA, Forrester Research

Harel D and Koren Y 2001 A fast multi-scale method for drawing large graphs. In *Proceedings of the Eighth International Symposium on Graph Drawing*, Vienna, Austria: 183–96

Kogut B and Metiu A 2001 Open source software development and distributed innovation. *Oxford Review of Economic Policy* 17: 248

Kuk G 2006 Strategic interaction and knowledge sharing in the KDE developer mailing list. *Management Science* 52: 1031–42

Kuk G 2010 Eyeballs, bugs, and releases in open source software. In *Proceedings of the European Conference on Information Systems*, Pretoria, South Africa

Lerner J and Tirole J 2002 Some simple economics of open source. *Journal of Industrial Economics* 50: 197–234

Lopez-Fernandez L, Robles G J, and Gonzalez-Barahona M 2004 Applying social network analysis to the information in cvs repositories. In *Proceedings of the International Workshop on Mining Software Repositories*, Edinburgh, Scotland

Maillart T, Sornette D, Spaeth S, and von Krogh G 2008 Empirical tests of Zipf's law mechanism in open source Linux distribution. *Physical Review Letters* 101: 218701

Mitchell T 2010 Annual report 2009: All reports. *OSGeo Journal* 7(1): 28

Meneely A and Williams L 2009 Secure open source collaboration: An empirical study of Linus' law. In *Proceedings of the Sixteenth ACM Conference on Computer and Communications Security*, Chicago, Illinois

Mockus A, Fielding R T, and Herbsleb J 2000 A case study of open source software development: the Apache server. In *Proceedings of the Twenty-second International Conference on Software Engineering*, Redwood City, California: 263–72

Müller-Prothmann T 2006 *Leveraging Knowledge Communication for Innovation: Framework, Methods and Applications of Social Network Analysis in Research and Development*. Frankfurt, Peter Lang Publishers

Perera D, Kay J, Yacef K, and Koprinska I 2007 Mining learners' traces from an online collaboration tool. In *Proceedings of the International Conference of Artificial Intelligence in Education*, Marina del Rey, California

Roberts J A, Hann I, and Slaughter S A 2006 Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects. *Management Science* 52: 984

Sanz S 2009 Current panorama of the FOSS4G ecosystem. *Upgrade* 10(2): 43–51

Scott J 2000 *Social Network Analysis: A Handbook* (Second Edition). London, Sage

Sowe S K, Samoladas I, Stamelos I, and Angelis L 2008 Are FLOSS developers committing to CVS/SVN as much as they are talking in mailing lists? Challenges for integrating data from multiple repositories. In *Proceeding of the Third International Workshop on Public Data about Software Development (WoPDaSD)*, Milan, Italy

Steiniger S and Bocher E 2009 An overview on current free and open source desktop GIS developments. *International Journal of Geographical Information Science* 23: 1345–70

Torvalds L and Hamano J 2010 GIT-fast Version Control Ssystem. WWW document, http://gitscm.com

Watts D and Strogatz S 1998 Small world. *Nature* 393: 440–42

Welsh M 1994 Cooking with Linux: Linux leadership. *Linux Journal* 1994(2es): 13

Williams P 2002 The competent boundary spanner. *Public Administration* 80: 103–24

Yip J, Ernst C, and Campbell M 2009 *Boundary Spanning Leadership: Mission Critical Perspectives from the Executive Suite*. Greensboro, NC, Center for Creative Leadership (available at http://www.ccl.org/leaders/pdf/research/BoundarySpanningLeadership.pdf)