The Onion has Cancer: Some Social Network Analysis Visualizations of Open Source Project Communication

Christopher Oezbek
Freie Universität Berlin
Berlin, Germany
oezbek@inf.fu-berlin.de

Lutz Prechelt
Freie Universität Berlin
Berlin, Germany
prechelt@inf.fu-berlin.de

Florian Thiel
Freie Universität Berlin
Berlin, Germany
thiel@inf.fu-berlin.de

ABSTRACT

Background: People contribute to OSS projects in wildly different degrees, from reporting a single defect once and never coming back to spending many hours each workday on the project over several years – or anything in between. It is a common conception that these degrees of participation sort the participants into a number of similar groups which are layered like the peels of an onion: The onion model. *Objective:* We check whether this model of gradually different degrees of participation is valid with respect to the participation in OSS project mailing-list traffic. *Methods:* We perform social network analysis based on replies to mailing-list messages and use visualization to check the nature of three different groups of participants. *Results:* There appears to be a discontinuity with respect to core members: The degree to which very active core members (as opposed to less active co-developers) react to e-mails of senders from the project's periphery is significantly higher than would be expected from their level of activity in general. *Limitations:* The effect might be an artifact of the assumption that each mailing-list message can be treated the same. *Conclusions:* We conclude that core member status may be qualitatively (rather than just quantitatively) different and the transition of individual mailing-list participants towards ever higher participation is qualitatively discontinuous.

Categories and Subject Descriptors

D.2.9 [**Software Engineering**]: Management; J.4 [**Computer Applications**]: Social and Behavioral Sciences

General Terms

Measurement, Human Factors

Keywords

social network analysis, open source process, communication structure

FLOSS'10 May 8 2010, Cape Town, South Africa

1. INTRODUCTION

We are concerned here with projects that have not just a free software license for their product, but also an open participation model for process conduct: Everybody who is interested in the product or project is free to participate in discussions (on an open mailing-list) and contribute to development (coordinated via publicly readable version control systems to which participants gain write access after contributing for a while) [9]. We call such projects OSS projects.

Ever since Mockus et al.'s description of the Apache httpd project [29] it is common to think of the participation structure in OSS projects in terms of what has been called the *onion model*: nested spheres of participation, where participation and influence increase from outer to more inward layers of the onion while the number of participants increases from inner (core developers) to outer layers [42, 17, 11].

The Apache study considered three such spheres (has reported a defect, has contributed a defect fix, is core developer), but other studies suggest there are more facets of participation (participating in discussions, contributing to one vs. many modules, being able to grant write-access, etc.) that create further stages [19], and it may be best to think of the participation career of a will-be core developer as a smooth, continuous process of increasing participation and integration [13]. This would indeed be more similar to a real onion, which typically has eight or more layers.

The contribution of the present paper is to suggest that the idea of a smooth transition from outer layers to the core may be misleading. Rather, the core may be special and have a qualitatively different (and not just quantitatively different) role in an OSS project even when compared to the heavy but not-quite-yet-core developers of the same project.

This observation stems from a visualization-based analysis of the social network that arises from e-mail communication. In the particular visualization style chosen, three layers of participants can easily be discerned in medium-sized projects, the middle one of which hugs the core in a manner that looks somewhat like a tumor growing out of an organ, hence the flashy title of the paper.

We will now explain social network analysis (SNA) and the role and dangers of visualization (Section 2), sketch previous studies of OSS using SNA (Section 3), describe the origin of the data we have analyzed (Section 4) and the design decisions used for the visualization (Section 5). We will then show the visualizations themselves and discuss our finding (Section 6).

2. SOCIAL NETWORK ANALYSIS (SNA)

Social Network Analysis or SNA [20] is the analysis of graphs in which a node represents an individual (e.g. a developer, but could

be a group or organization as well) and a vertex represents a relationship between two individuals (e.g. they have modified the same code module or one has sent e-mail to the other).

This idea originated in the social sciences as "sociometrics" [16] and has been applied to many types of data such as local communities of fishermen in Norway [3], the spread of AIDS through sexual contacts [21], the interlock of enterprises via debt or share holders [4], or collaboration among authors of scientific publications [30].

Computing various global or node-specific *metrics* for a network is useful for making general statements about specific networks or classes of networks. Examples of such metrics are betweenness, diameter, distance, density, betweenness centrality, degree centrality, or eigenvector centrality [31, 32]. Examples of general effects and principles found for social networks are the small world phenomenon [28, 41, 10], 0-1-2 effect [1], preferential attachment [31], or triadic closure [22]. Note that we are interested in the social phenomena *underlying* the network. However, the metrics tend to be so abstract that one can easily loose track of what they really mean, which is dangerous [36].

This is why we prefer the complementary approach to SNA here: visualization. Visualization of a social network graph [18, 40, 15] is useful for identifying key individuals, for understanding the structure of a particular social network, or for comparing presumably similar networks in order to identify peculiarities.

However, visualization has its pitfalls as well: Visualizing large networks requires automation, which implies choosing a set of rules for how to place the nodes, a layout algorithm. Layout algorithms attempt to minimize the number of edge intersections (which would make the image harder to read) and node overlaps (which would make the image very hard to read). They may also have various other functions, such as keeping certain designated subgroups of nodes together or biasing the overall structure towards a certain shape [14]. For instance, arranging all nodes in a single wide ring always allows a layout without any edge intersecting a third node but often leads to very uneven use of the layout area and to confusing images in which even strong substructures are hard to find. Other choices have different drawbacks and one must keep in mind that a striking feature of a visualization is always to some degree an artifact of the particular visualization rules chosen [39, 8].

Size, shape, or color can be used to encode further information in a node (node type and weight) or an edge (relationship type, connection strength). This increases the expressive power of the visualization, but may also create additional artifacts.

For our visualizations we use the GraphViz package [14], which provides various layout algorithms and powerful options for customizing the resulting visualization.

3. SNA IN THE OSS LITERATURE

A number of different relationship types within OSS projects have been analyzed.

Madey et al. looked at how projects are linked by individuals participating in more than one project [26, 27] and suggested there are individuals who are important boundary-spanners between many projects.

Crowston and Howison looked at how developers are linked by working on the same entry in the defect tracking system. Their study considers many projects and makes heavy use of metrics. It finds for instance that it is less likely in a large project that one developer dominates the communication regarding a defect entry [11].

Lopez-Fernandez et al. looked at how developers are linked by contributing to the same 'module' (source code directory) in three large projects (KDE, Gnome, Apache) and at how modules are linked by the same developer working on both [25, 24]. They found that both networks are loosely connected and exhibit small world characteristics. Spaeth has similar results for medium-sized projects [37].

de Souza et al. extended this by static call graph analysis and followed the evolution of the situation over a long time. They found shifts in participation from the periphery to the core of a project and vice versa, as well as changes to the ownership of files over time [12].

Bird et al. considered five large OSS projects and looked at developers working together on the same files and reply-to relationships on the mailing-list [7]. They found that (1) the communication network was modular, i.e. sub-groups could be identified, (2) that developers discussed product-centric topics in a smaller group, while other topics were discussed more broadly in the community, and (3) that people who interacted on the mailing-list also were much more likely to work together on the same files in the repository.

Ducheneaut attempted to understand the 'career' of a single, specific developer, Fred, and uses SNA (based on both mailing-list communication and work on files) as one of several qualitative and quantitative methods in this longitudinal individual-centric study. The result is a rich description of how Fred gradually moves towards the core of the Python project until eventually he is a respected core member. It characterizes joining as a learning process involving e.g. identity construction and join scripts and as a political process [13].

Keep in mind that SNA can be criticized as over-simplifying reality [5]. For instance, the reply-to relationship was often used to model an association between actors, without accounting for the content of the e-mail or the actual quotation patterns [2] and without considering the relationship strength to decay over time [22].

4. OUR DATA

We study the introduction of process innovations in Open Source projects [33] by manually extracting innovation episodes from archives of mailing-lists and analyzing these episodes qualitatively by the Grounded Theory Method [38].

In this context, we wanted to validate whether a social network visualization could be used to provide background information that helps interpreting an innovation episode. To this end, we took all messages from the mailing-list archives in 2007 of the projects we were studying, turned each participant into a node (unifying multiple e-mail addresses where needed [6]), and computed relationship strength between A and B as the number of e-mails that are a reply of B to a message from A or vice versa, according to the in-reply-to header of the e-mail.

For understanding innovation introduction, by the way, the result was negative: The only helpful information shown in the visualization was the 'weight' of each of the episode participants, which can be obtained much more easily by a simple count of the mailing-list contributions.

Our data set covers 11 of the 13 projects (from 7 different domains, selected from mailing-list archive Gmane to build a diverse set of projects) for which we analyzed innovation episodes. They include three workflow applications (Bugzilla, Flyspray, Request Tracker), two desktop environments (Rox, Xfce), two design tools (ArgoUML, a UML CASE tool; gEDA, a set of electronic design automation tools), one bootloader (Grub), one hardware emulator (Bochs), one operating system (FreeDOS), and one database management system (MonetDB). Data was cleaned as to unify multiple e-mail addresses used by the same person and Spam was re-

moved [6]. The two other projects (U-Boot, KVM) were so large that the data created scaling problems for our graph layout software. Since the resulting images would be overcrowded and hardly useful anyway, we simply leave them out.

5. VISUALIZATION RULES USED

Each node represents a person who has posted to the mailing-list at least once in 2007. Each such person is represented, even if nobody has reacted to these e-mails and the person is hence not connected to any other. Each node is represented by a circle. The area of the circle represents the number of e-mails sent by the person.

Each edge represents an undirected replies-to relationship between two persons (except when crossing the core-group boundary, as described below) and is represented as a line. The width of the line represents the number of e-mails (in any of the two directions).

The color of a circle represents the number of calendar months during which that person has sent at least one e-mail (the darker the color the more months of activity); frequent participants are not always also regular participants and vice versa.

All of these visualization design decisions are quite straightforward. Now for the crucial one: We treat core members specially, as follows.

- We consider participants to be members of the project core according to a formal criterion. A participant is in the core if s/he has sent at least one e-mail to the list in at least k calendar months, with $k = 8$. This simple criterion works acceptably well: We experimented with various other definitions, in particular one based on a community finding algorithm [35, 34], and different values of k, but the above definition produced the subjectively most convincing core groups across the various projects.
- We layout the core nodes as a separate subgraph.
- This subgraph is drawn inside a grey square to make the core group readily visible.
- Edges between core nodes are drawn as usual.
- Edges between non-core nodes are drawn as usual.
- An edge between a core node and a non-core node is *not actually drawn*. Rather, all edges between this non-core node and *any* of the core nodes are collapsed into a single edge and drawn towards the center of the core rectangle.

The latter rule greatly reduces the number of lines to be drawn, makes the image correspondingly clearer, and practically turns the core group into a single entity. This has to be firmly kept in mind when interpreting the image.

6. RESULTS

The resulting visualizations for our 11 projects can be seen in Figure 1. The script in these figures is too small to read in print, but is hardly relevant; it can be enlarged arbitrarily in the digital version.

All figures show a similar structure of (1) a tightly integrated core, (2) a loosely collected set of co-developers which are strongly oriented to the core but also share some links between each other, and (3) a periphery of project participants, most of which are either only connected to the project core or not at all (that is, their e-mails were all ignored).

In contrast to the "onion" model, which describes influence and role advancement possibilities, the communication social network is thus more appropriately named "earth, moon, and stars" with the project core being the earth around which most communication revolves, the co-developers forming a crescent-like set of "moon" developers hugging the core, and the peripheral participants are dotted as stars around the core.

A central oddity struck us when looking at these graphs: Communication between peripheral participants and co-developers and inside the co-developer group is surprisingly low, showing little signs of the strong peer-to-peer assistance commonly assumed in OSS projects [23]. Rather the core developers hold an disproportionally large share of communication with the periphery.

This visual impression is confirmed by a quantitative analysis: When a group g produces some fraction f_g of overall e-mail traffic, we would expect it to have the same fraction $f_{g,P}$ of the communication with the periphery P as well: We define $T_g = f_g / f_{g,P}$ and expect $T_g = 100\%$. We compare these fractions for the core C and co-developers D via a χ^2-test[1] and assume co-developer status for all mailing-list participants active for three months and more, who are not in the core. We find that $T_D < 100\%$ in all 11 cases. In three cases, the difference is not statistically significant: For Request Tracker and Bochs because their core groups are so small ($n = 2, n = 1$), for Xfce because the expectation is only almost met ($T_D = 95\%$). For the remaining 8 projects, the difference is statistically significant ($p < 0.01$) and T_D ranges from 31.2% (ArgoUML) to 69.2% (Bugzilla).

7. CONCLUSION AND LIMITATIONS

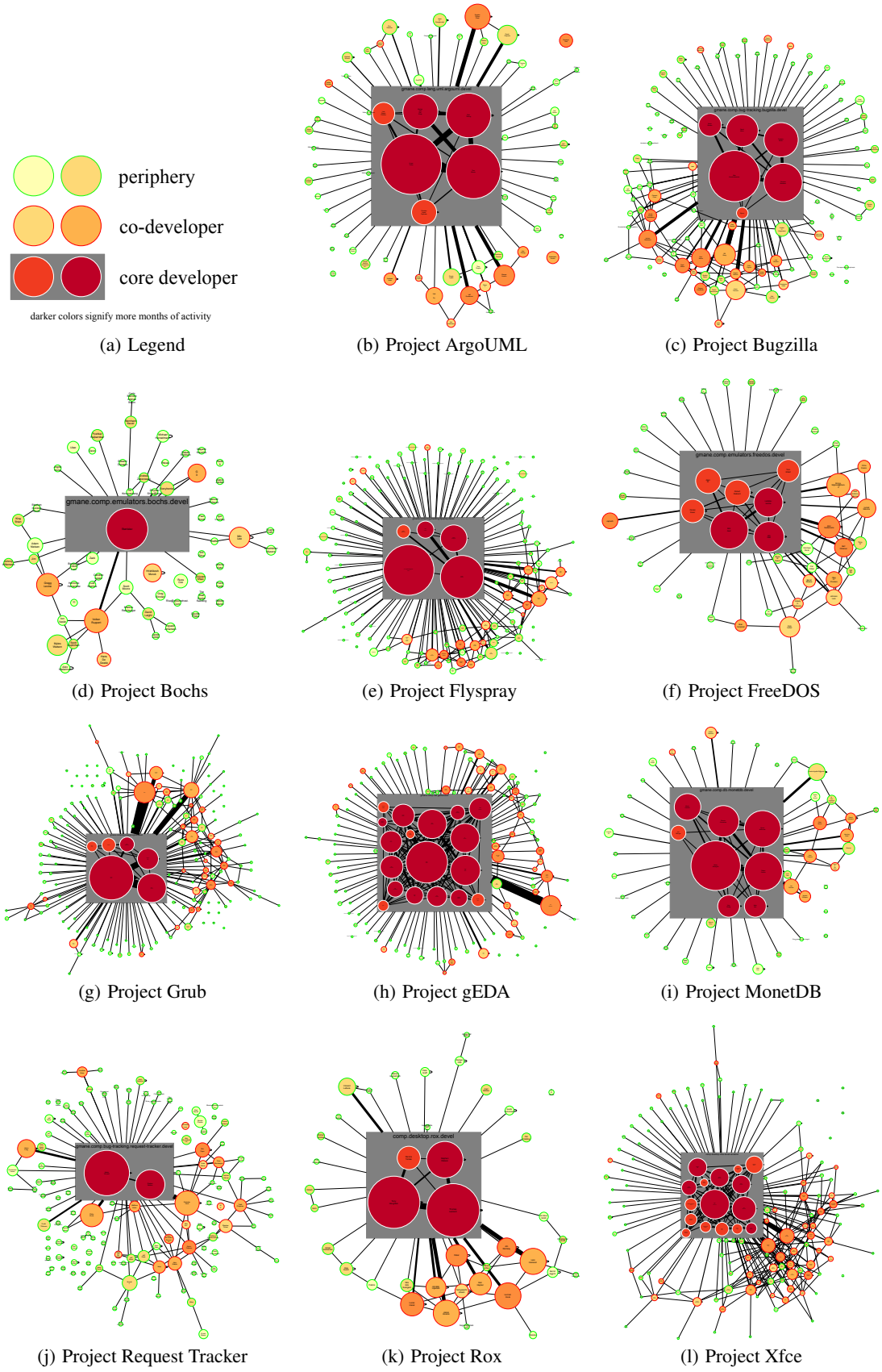
We conclude that the core group is not just a group of developers with particularly intense participation. Rather, the core appears to have a *qualitatively* different role as well, so that the many-layered onion model is misleading. Core developers appear to take part in the project in a way that is inherently more comprehensive, even *beyond* the fact that they simply do more. The group of co-developers on the other hand, while connected to some degree, seems strongly oriented to the core, causing us to describe it as "cancerous" or rather as following a "sun, moon, stars"-model of communication activity.

Although it is not clear to us what this qualitative difference really is, we do not find it surprising that it exists. What we do find surprising, however, is that the difference is so strong that it is clearly visible despite the crude metric we used for participation (e-mail counts with no weighting) and our definition of the core group that is as simple as a nursery rhyme *"Active at all // for eight months or more? // You're in the core!"*.

Nevertheless, the result needs to be validated with refined metrics.[2] Alternatively, the suitability of the metrics itself needs further investigation and support. We need to understand whether the e-mail-count metric is distorted by the existence of high-volume, off-topic traffic, whether effects such as days-per-month activity or hours-per-day activity of contributors skews who captures the (presumably often simple, one-shot) periphery traffic, and whether our months-activity rule accurately captures what conventional definitions would consider the core (such as official members, committers, high-frequency committers, high-volume committers).

[1]Note that due to the existence of threads in e-mail communication, the events counted are not all strictly independent, which distorts the test results a bit. We will therefore require $p < 0.01$ before we consider a difference significant.

[2]Barcellini et al. have for instance cautioned to the use of reply-relationships when quotations might provide a substantially better mapping of discussion flow [2].



(a) Legend (b) Project ArgoUML (c) Project Bugzilla

(d) Project Bochs (e) Project Flyspray (f) Project FreeDOS

(g) Project Grub (h) Project gEDA (i) Project MonetDB

(j) Project Request Tracker (k) Project Rox (l) Project Xfce

Figure 1: Visualizations (as defined in Section 5) of the e-mail communication structure for 11 OSS projects.

8. ACKNOWLEDGMENTS

We thank Christian Bird for providing his implementation of a community modularization algorithm as a starting point for experimentation.

9. REFERENCES

[1] L. Backstrom, D. Huttenlocher, J. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 44–54, New York, NY, USA, 2006. ACM.

[2] F. Barcellini, F. Détienne, J.-M. Burkhardt, and W. Sack. A study of online discussions in an Open-Source software community: Reconstructing thematic coherence and argumentation from quotation practices. In P. van den Besselaar, G. de Michelis, J. Preece, and C. Simone, editors, *Second Communities and Technologies Conference, Milano 2005*, pages 121–140, Milano, Italy, May 2005. Springer. http://www.springer.com/sgw/cda/frontpage/0,,1-40393-22-46608953-detailsPage

[3] J. A. Barnes. Class and committees in a Norwegian island parish. *Human Relations*, 7(1):39–58, 1954.

[4] J. Bearden, W. Atwood, P. Freitag, C. Hendricks, B. Mintz, and M. Schwartz. The nature and extent of bank centrality of corporate networks. Unpublished paper submitted to the American Sociological Association. Reprinted in Scott J. *Eds*. Social networks: critical concepts in sociology, Volume 3. Taylor & Francis, 2002., 1975.

[5] E. Berdou. *Managing the bazaar: Commercialization and peripheral participation in mature, community-led F/OS software projects*. Doctoral dissertation, London School of Economics and Political Science, Department of Media and Communications, 2007.

[6] C. Bird, A. Gourley, P. Devanbu, M. Gertz, and A. Swaminathan. Mining email social networks. In *MSR '06: Proceedings of the 2006 international workshop on Mining software repositories*, pages 137–143, New York, NY, USA, 2006. ACM.

[7] C. Bird, D. Pattison, R. D'Souza, V. Filkov, and P. Devanbu. Latent social structure in Open Source projects. In *SIGSOFT '08/FSE-16: Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 24–35, New York, NY, USA, 2008. ACM.

[8] S. Bresciani and M. J. Eppler. The risks of visualization: a classification of disadvantages associated with graphic representations of information. In P. Schulz, U. Hartung, and S. Keller, editors, *Identität und Vielfalt der Kommunikationswissenschaft*, pages 165–178. UVK Verlagsgesellschaft mbH, Konstanz, Germany, 2009.

[9] A. W. Brown and G. Booch. Reusing Open-Source Software and practices: The impact of Open-Source on commercial vendors. In *ICSR-7: Proceedings of the 7th International Conference on Software Reuse*, pages 123–136, London, UK, 2002. Springer-Verlag.

[10] J. J. Collins and C. C. Chow. It's a small world. *Nature*, 393:409–410, June 1998.

[11] K. Crowston and J. Howison. The social structure of Free and Open Source software development. *First Monday*, 10(2):n/a, 2005.

[12] C. de Souza, J. Froehlich, and P. Dourish. Seeking the source: Software source code as a social and technical artifact. In *GROUP '05: Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work*, pages 197–206, New York, NY, USA, 2005. ACM.

[13] N. Ducheneaut. Socialization in an Open Source Software community: A socio-technical analysis. *Computer Supported Cooperative Work (CSCW)*, V14(4):323–368, Aug. 2005.

[14] J. Ellson, E. Gansner, L. Koutsofios, S. North, and G. Woodhull. Graphviz: Open source graph drawing tools. In *Graph Drawing*, volume 2265/2002 of *Lecture notes in computer science*, pages 594–597, Berlin, Germany, 2002. Springer.

[15] L. C. Freeman. Visualizing social networks. *Journal of Social Structure*, 1(1), 2000.

[16] L. C. Freeman. *The Development Of Social Network Analysis— A Study In The Sociology Of Science*. Empirical Press, Vancouver, BC, Canada, 2004.

[17] C. Gacek and B. Arief. The many meanings of open source. *IEEE Software*, 21(1):34–40, January/February 2004. Good introductury paper.

[18] J. Heer and D. Boyd. Vizster: Visualizing online social networks. In *INFOVIS '05: Proceedings of the 2005 IEEE Symposium on Information Visualization*, pages 32–39, Washington, DC, USA, 2005. IEEE Computer Society.

[19] C. Jensen and W. Scacchi. Role migration and advancement processes in OSSD projects: A comparative case study. In *ICSE '07: Proceedings of the 29th International Conference on Software Engineering*, pages 364–374, Washington, DC, USA, 2007. IEEE Computer Society.

[20] J. Kleinberg. The convergence of social and technological networks. *Commun. ACM*, 51(11):66–72, 2008.

[21] A. S. Klovdahl. Social networks and the spread of infectious diseases: the AIDS example. *Social Science and Medicine*, 21(11):1203–1216, 1985.

[22] G. Kossinets and D. J. Watts. Empirical Analysis of an Evolving Social Network. *Science*, 311(5757):88–90, Jan. 2006.

[23] K. R. Lakhani and E. von Hippel. How Open Source software works: "free" user-to-user assistance. *Research Policy*, 32(6):923–943, June 2003.

[24] L. López-Fernández, G. Robles, J. Gonzalez-Barahona, and I. Herráiz. Applying social network analysis techniques to community-driven Libre Software projects. *International Journal of Information Technology and Web Engineering*, 1(3):27–48, 2006.

[25] L. López-Fernández, G. Robles, and J. M. Gonzalez-Barahona. Applying social network analysis to the information in CVS repositories. *IEE Seminar Digests*, 2004(917):101–105, 2004.

[26] G. Madey, V. Freeh, and R. Tynan. The Open Source software development phenomenon: An analysis based on social network theory. In *8th Americas Conference on Information Systems (AMCIS2002)*, pages 1806–1813, Dallas, TX, 2002.

[27] G. Madey, V. Freeh, and R. Tynan. Modeling the F/OSS community: A quantitative investigation. In S. Koch, editor, *Free/Open Source Software Development*, chapter 9, pages 203–220. Idea Group Publishing, 2005.

[28] S. Milgram. The small-world problem. *Psychology Today*,

1(1):61–67, May 1967.

[29] A. Mockus, R. T. Fielding, and J. Herbsleb. Two case studies of Open Source Software development: Apache and Mozilla. *ACM Transactions on Software Engineering and Methodology*, 11(3):309–346, 2002.

[30] M. E. J. Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences of the United States of America (PNAS)*, 98(2):404–409, Jan. 2001.

[31] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003.

[32] M. E. J. Newman. Analysis of weighted networks. *Phys. Rev. E*, 70(5):056131, Nov. 2004.

[33] C. Oezbek and L. Prechelt. On understanding how to introduce an innovation to an Open Source project. In *Proceedings of the 29th International Conference on Software Engineering Workshops (ICSEW '07)*, Washington, DC, USA, 2007. IEEE Computer Society. reprinted in UPGRADE, The European Journal for the Informatics Professional 8(6):40-44, December 2007.

[34] G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435(7043):814–818, June 2005.

[35] M. A. Porter, J.-P. Onnela, and P. J. Mucha. Communities in networks. *Notices of the AMS*, 56(9):1082–1097, Oct. 2009.

[36] J. Scott. Social Network Analysis. *Sociology*, 22(1):109–127, 1988.

[37] S. Spaeth. *Coordination in Open Source Projects – A Social Network Analysis using CVS data*. Dissertation, Universität St. Gallen, St. Gallen, Switzerland, Oct. 2005.

[38] A. L. Strauss and J. M. Corbin. *Basics of Qualitative Research: Grounded Theory Procedures and Techniques*. SAGE, 1990.

[39] M. van der Meulen, R. H. Logie, Y. Freer, C. Sykes, N. McIntosh, and J. Hunter. When a graph is poorer than 100 words: A comparison of computerised natural language generation, human generated descriptions and graphical displays in neonatal intensive care. *Applied Cognitive Psychology*, Early View:n/a, Dec. 2008.

[40] visone Software for the Analysis and Visualization of Social Networks. *Michael Baur*. Disseration, Fakultät für Informatik, Universität Karlsruhe (TH), Karlsruhe, Nov. 2008.

[41] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998.

[42] Y. Ye and K. Kishida. Toward an understanding of the motivation of Open Source Software developers. In *Proceedings of the of the 25th International Conference on Software-Engineering (Portland, Oregon)*, 2003.