OPEN UNIVERSITY UK

EXTENDED ABSTRACT

Evaluating phylogenetic methods for quantifying risks and opportunities presented by forks in open source software

Author:
Alvaro ORTIZ TRONCOSO

Supervisor:
Dr. Doug LEITH

Motivation and research aim

The nature of software makes software engineering a different endeavour than other forms of engineering. The artefacts produced during software development can be copied and distributed at almost no cost, challenging traditional business models based on the distribution of physical products and strong intellectual property protection [Kogut and Metiu, 2001]. Besides intellectual property challenges, software development also questions the idea of a development lifecycle resulting in a stable and long-lived product. Software is required to evolve, even after it has been released for production [Lehman, 1980]. Software evolution is driven by the changing expectations of its users, a dynamic hardware and software ecosystem and an innovative business environment.

Open source software development is a development paradigm that purports to increase the value of software by embracing these difficulties. Instead of applying strong intellectual property protection to create an economy of scarcity, open source licenses guarantee the free flow of ideas. Furthermore, open source development embraces constantly changing requirements by blurring the distinction between users and developers, allowing anyone to participate in development [von Hippel and von Krogh, 2003]. In this way, software developers can review each other's code to produce better quality products, project managers profit from an expanded palette of team governance possibilities, and business people can generate hitherto unworkable business models.

However, the freedom to copy source code exposes open source projects to a new kind of risk: forking. Forking happens when part of the team takes off in a new direction, resulting in an autonomous project, with its own name, infrastructure, code base and community of developers [Robles and González-Barahona, 2012]. The code structure and the governance of open source projects are tailored to prevent this risk [Kogut and Metiu, 2001]: the code base is often very modular, allowing for extensive customization, and the governance of projects is often non-hierarchical, so that all participants have a say as to where development is heading.

Nevertheless, many organisations regard forking as an opportunity to try out new bold ideas (technical or business) or to restore the balance between stakeholders [Nyman and Lindman, 2013]. Indeed, forking is not an uncommon phenomenon,

and examples of projects and companies that have resorted to forking to successfully create value from hobbling projects abound [Robles and González-Barahona, 2012].

Whether forking is a risk or an opportunity is therefore an open question. The aim of this research is to assess methods to quantify these risks and opportunities.

Summary of the methodology

Three cases of forks were selected for the availability of their complete history through freely available online repositories as well as for the variety of reasons that led to the fork. The chosen projects were: the MariaDB database engine forked from the MySQL project after disagreements about technical, community and legal issues [Widenius, 2012]; the popular Android operating system for mobile devices forked from Linux as a way to explore a new business strategy[Elgin, 2005]; The LibreOffice suite of office applications forked from OpenOffice after the original project had been abandoned by its new owners [Gamalielsson and Lundell, 2014]. Furthermore, the popularity of these projects and the size of their development teams emphasise the importance of forking for software engineering.

The present research postulates that the evolution of software can be described using terms and methods borrowed from evolutionary biology. Borrowing techniques from evolutionary biology and applying them to the study of software enables reconstructing the evolution of a population of software releases, i.e. estimating a phylogenetic tree. An example is given in the tree below (figure 1): the phylogenetic tree of the MariaDB / MySQL fork shows (1) the mysql.5.0.0 release, that pre-dates the fork, (2) the MariaDB releases and (3) the MySQL releases. This tree was obtained by applying the "Neighbour-Joining" algorithm to software metrics obtained from each release. The "Neighbour-Joining" algorithm is one of a family of algorithms used in evolutionary biology to reconstruct the relationships between organisms [Saitou and Nei, 1987]. The metrics are common metrics used in software engineering [Nagappan et al., 2008]: code churn (the amount of change to the code base), team composition (presence of team members in each release), edit frequency (edit count per team member) and code ownership (commit count per team member).

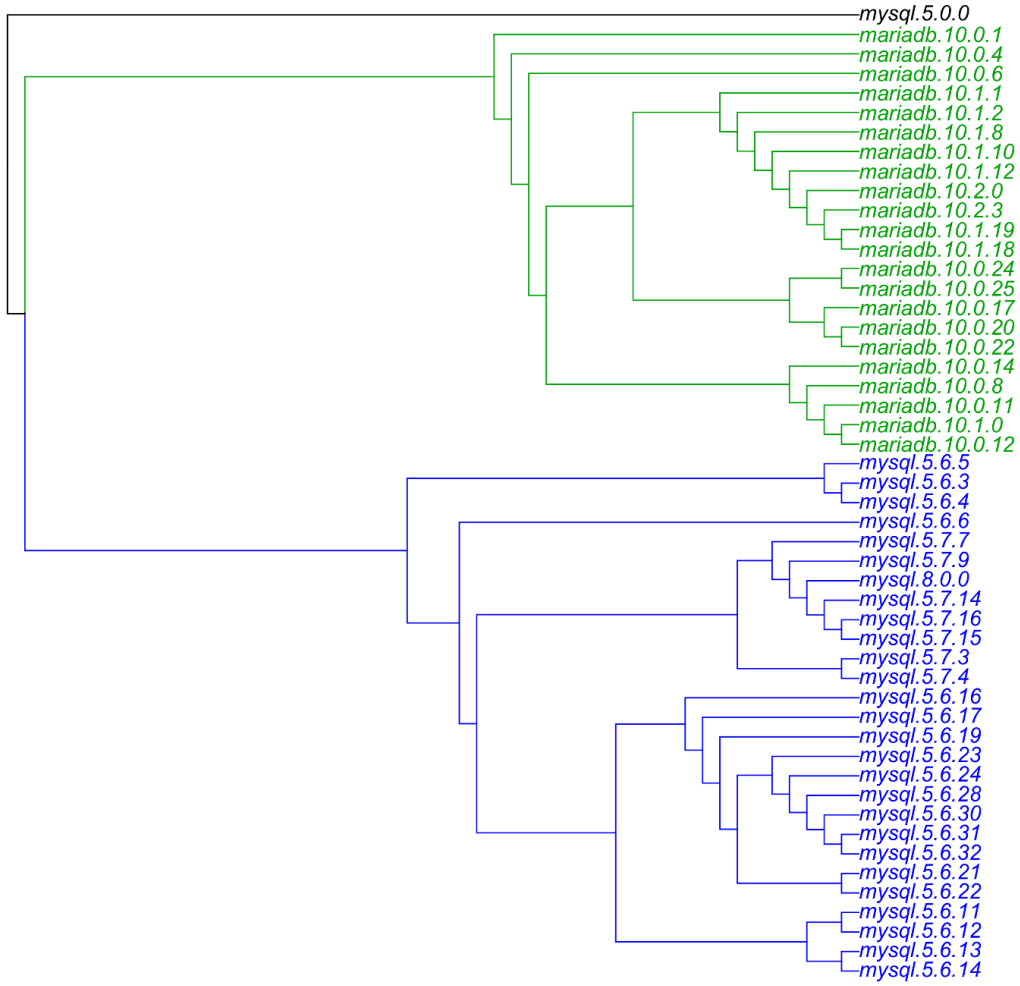

FIGURE 1: A phylogenetic tree of the MariaDB / MySQL fork.

Key results

Phylogenetic methods can, at least in the example shown above, provide a plausible reconstruction of the history of a fork. While this result is interesting as a proof-of-concept, it is not a key result, as a similar tree could have been obtained by examining the logs of the software repositories where the forked projects are kept. From a project management point of view, it is more interesting to know whether two parallel development strands are about to cross a threshold beyond which a fork will be unavoidable. As the analysis detailed in the next paragraph shows, it is possible to measure the pairwise dissimilarity between development branches within a project and therefore to detect those branches that have wandered the furthest from the main branch and could potentially fork.

Forks can have different outcomes. Forked projects may become each other's competitors, or the fork may result in the discontinuation of the original project. Alternatively, a project may fork yet the split teams may continue to exchange knowledge in a collaborative way. The second key result of the research is that the chosen measurements of intrinsic project properties are not sufficient to explain the outcome of a fork.

Summary analysis

Having constructed a phylogenetic tree, it becomes possible to measure the distance between the nodes of the tree. A statistical analysis of variance showed that, within each pair of forked projects examined, the mean pairwise distance between two releases on the same project is significantly smaller than the mean pairwise distance between releases on different projects, thus leading to the first key result.

If a fork results in competing projects, then different teams could be expected to produce different code bases. On the contrary, if the fork results in collaborating projects, different teams could be expected to produce code bases with many similarities. This idea was explored by constructing separate phylogenetic trees from measurements of (1) the code-base and of (2) the team data. A statistical analysis of variance did not find a significant correlation between the outcome of forks and the code-base- and team- measurements examined in this research, thus supporting the conclusion that other factors are responsible for the outcome of a fork.

Summary discussion

By constructing the phylogenetic tree of the forked projects and applying the analysis methods discussed above, it is possible to detect development strands that have wandered away from the main development branch, and therefore to estimate whether a development strand is likely to fork, even before the fork is effective. This is particularly interesting if forks are considered a risk.

On the other hand, the outcome of the forks used as examples for the research could not be related to the measurements obtained from the software's code base and team.

Key outcomes

The selected case studies substantiate that well-known and large open source projects have been affected by forks and that major software companies have in some cases successfully used forks as a business strategy. Hence, forking is an important phenomenon in open source software development.

Methods for constructing phylogenetic trees, ported from evolutionary biology, proved useful for estimating the relationships between development strands. In particular, evidence was gained that forks can be predicted. Additionally, the research also suggests that managerial, economic, social or other socio-technical factors have a profound influence on the outcome of a fork.

Furthermore, the vocabulary used by evolutionary biologists to describe phylogenetic trees could be applicable for describing software evolution: in particular, the concepts of relatedness, clade, parsimony and convergent evolution seem be relevant to the history of software projects.

Bibliography

B. Elgin. Google Buys Android for Its Mobile Arsenal, 2005. URL http://web.archive.org/web/20110205190729/http://www.businessweek.com/technology/content/aug2005/tc20050817{_}0949{_}tc024.htm.

J. Gamalielsson and B. Lundell. Sustainability of Open Source software communities beyond a fork: How and why has the LibreOffice project evolved? *Journal of Systems and Software*, 89(1):128–145, 2014. ISSN 01641212. doi: 10.1016/j.jss.2013.11.1077. URL http://dx.doi.org/10.1016/j.jss.2013.11.1077.

B. Kogut and A. Metiu. Open-source software development and distributed innovation. *Oxford Review of Economic Policy*, 17(2):248–264, 2001. ISSN 0266903X. doi: 10.1093/oxrep/17.2.248.

M. M. Lehman. Programs, Life Cycles, and Laws of Software Evolution. *Proceedings of the IEEE*, 1980. doi: 10.1109/PROC.1980.11805.

N. Nagappan, B. Murphy, and V. R. Basili. The Influence of Organizational Structure on Software Quality: An Empirical Case Study. 2008.

L. Nyman and J. Lindman. Code Forking, Governance, and Sustainability in Open Source Software. *Technology Information Management*, (January):7–12, 2013. ISSN 1927-0321.

G. Robles and J. M. González-Barahona. A Comprehensive Study of Software Forks: Dates, Reasons and Outcomes. 2012.

N. Saitou and M. Nei. The neighbor-joining method : a new method for reconstructing phylogenetic trees. *Molecular and Biological Evolution*, 4(4):406–425, 1987. ISSN 07374038. doi: citeulike-article-id:93683.

E. von Hippel and G. von Krogh. Open Source Software and the "Private-Collective" Innovation Model: Issues for Issues for Organization Science. *Organization Science*, 14(2):209–223, 2003. URL http://www.jstor.org/stable/4135161http://about.jstor.org/terms.

M. Widenius. Now is a good time to be part of the future of MariaDB, 2012. URL http://monty-says.blogspot.de/2012/09/now-is-good-time-to-be-part-of-future.html.