2016

First Annual State of InnerSource Commons Survey

Klaas-Jan Stol, Lero—the Irish Software Research Centre
and **Tim Yao**, Nokia

The InnerSource Commons is supported by:

PayPal Bloomberg AUTODESK. NOKIA

Contents

Executive summary

Open Source continues to play an important role in the software industry, not only because of the many high quality open source software products, but also because organizations increasingly realize that open development approaches offer many benefits. Hence, many organizations are leveraging the open source development paradigm within their boundaries. Tim O'Reilly coined the term InnerSource for this.

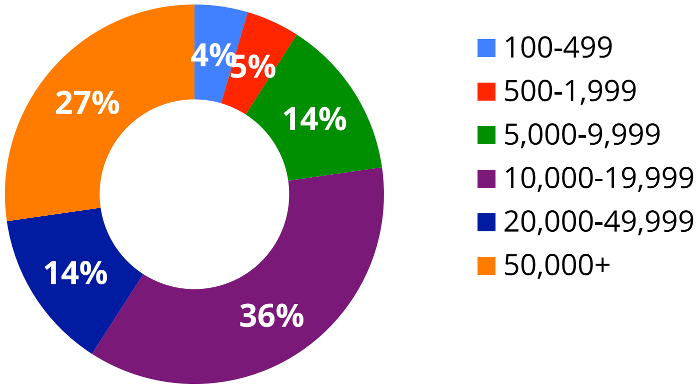
In the past few years, InnerSource is gaining much attention from companies around the globe. In 2015, the InnerSource Commons community was founded by Danese Cooper from PayPal, and the community is growing quickly, with over 120 members in the InnerSource Commons Slack channel. This community of software professionals are experimenting with open source development practices to overcome the many barriers and challenges that exist in many software organizations.

The goal of this First Annual State of InnerSource Commons survey was to establish a first baseline of how organizations adopt InnerSource. The survey addresses many aspects, including development methods, practices, quality assurance, tools, and motivation and organizational support.

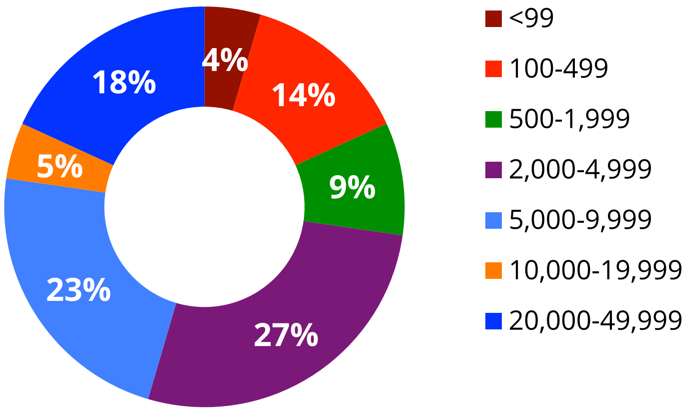
The InnerSource Commons is a vibrant and active community, and we invite you to join, learn from the community, and share your own experiences!

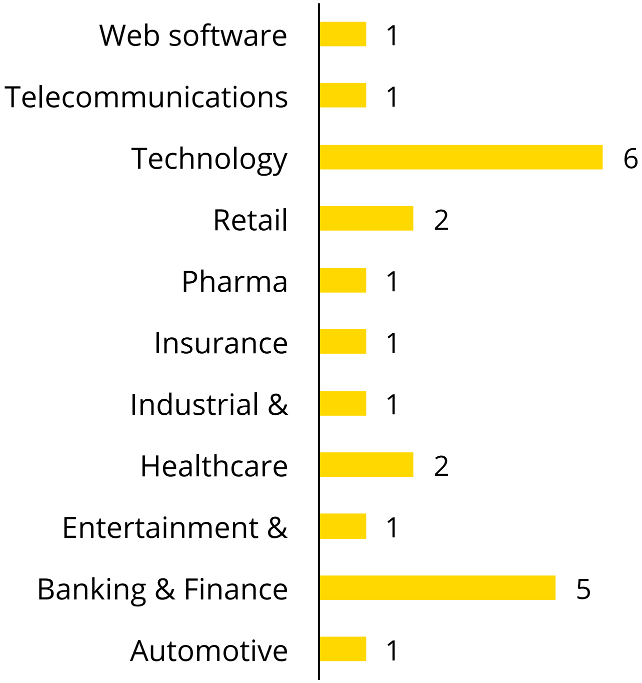
Demographics

Organization size



Number of software developers in organization

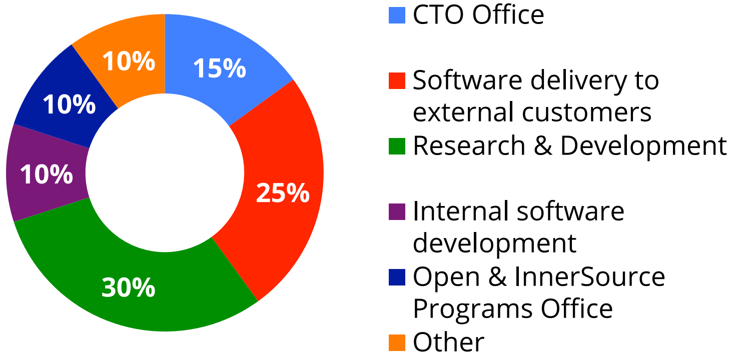




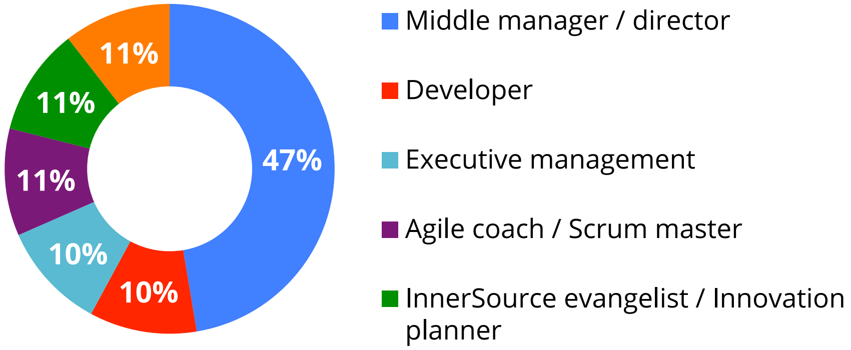
The sample represents organizations of all sizes, with medium-sized companies of up to 500 employees to global enterprises employing over 50,000 people. The respondent organizations also represent many different domains.

Demographics

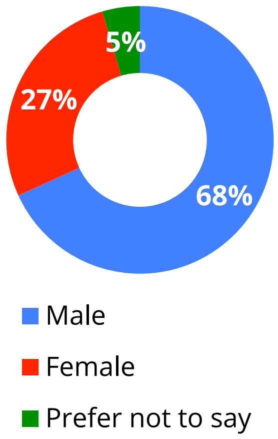
Organizational Unit



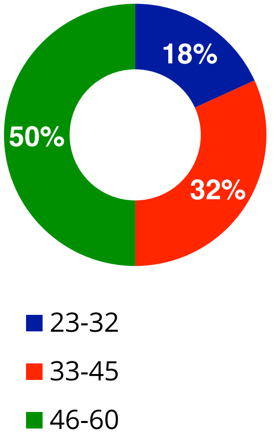
Position



Gender

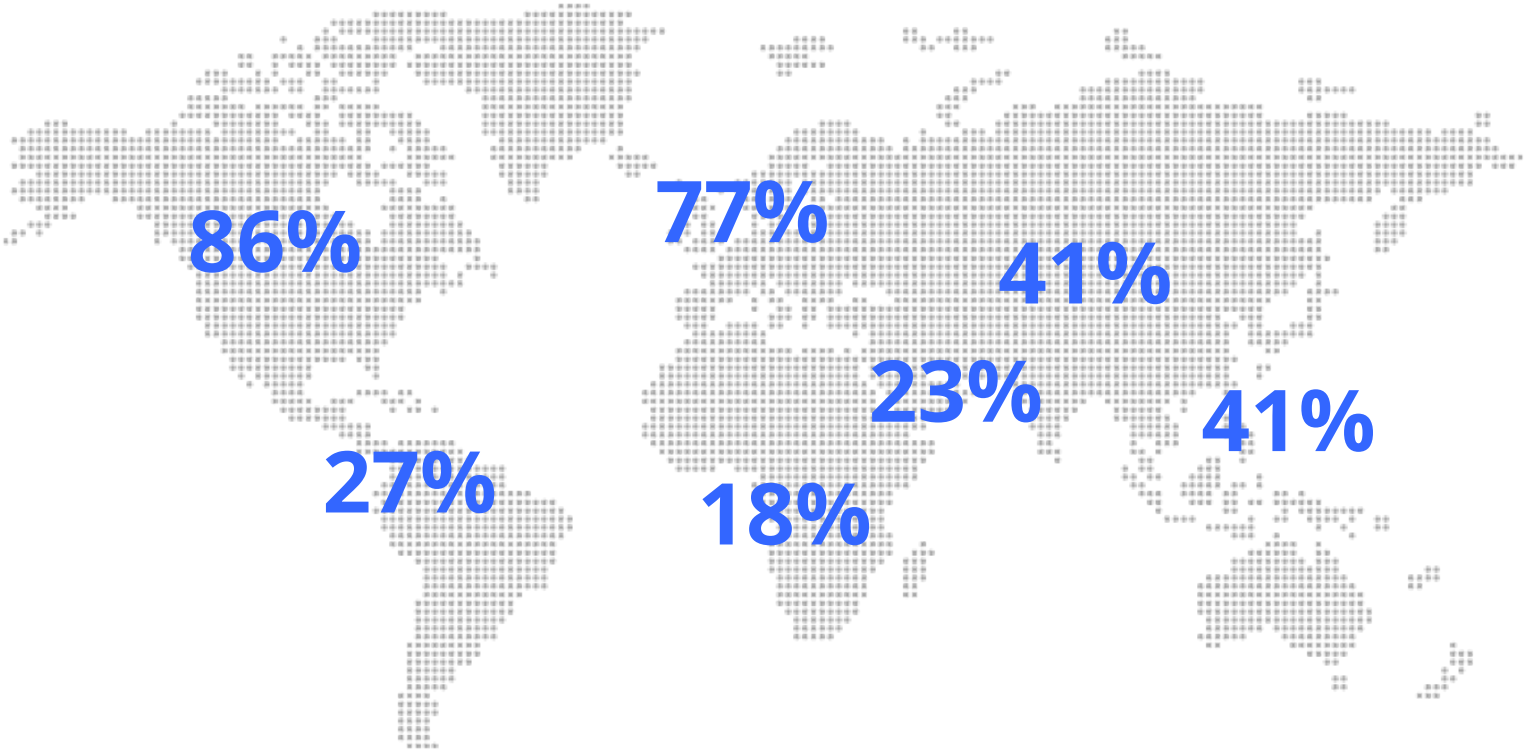


Age group



Respondents worked at a variety of organizational units, from the CTO office to business units that deliver software to external customers, as well as Research & Development units and departments that deliver software internally. It is interesting to see that some organizations have dedicated open & inner source program offices. Some of the respondents are dedicated InnerSource evangelists, and one respondent identified as an “Innovation Planner.” Most of the respondents were male, and most were in the 46-60 age bracket.

Demographics



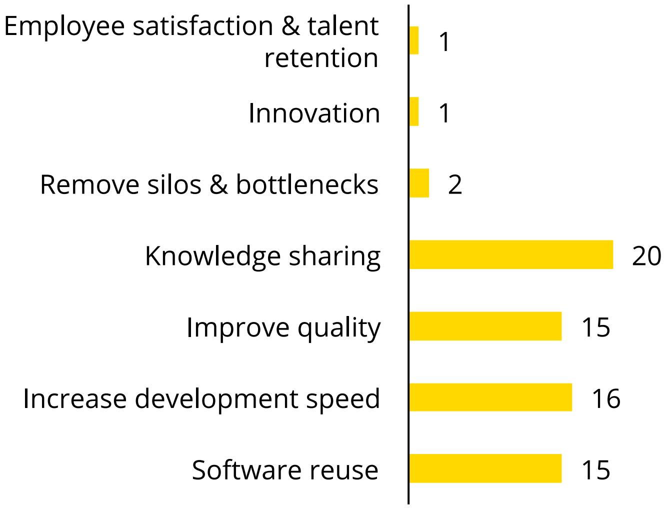
Most respondents worked at organizations which have operations across the globe. 86% of the respondents' organizations have operations in North America, and 77% in Europe. Other regions where respondent organizations have operations are Central & South America (27% of respondents), Africa (18%), Middle-East (23%), Central & South Asia (41%), and East Asia & Pacific (41%).

Why do organizations adopt InnerSource?

Organizations adopt InnerSource for a variety of reasons.[1,2] An important goal is to share knowledge across different organizational units. By involving others from different organizational units (teams, departments, etc.), developers can draw on those "internal outside experts." Joy's Law states that *"no matter who you are, most of the smartest people work for someone else."* That someone else might just be a different unit in the organization.

Software reuse and increasing the speed of development are also important drivers, which help to shorten time-to-market. Duplicated functionality is an extremely common phenomenon in large organizations, where a lack of transparency hides what is being developed in an organization. If developers don't know what software assets are available, reuse simply won't happen.

Another goal is to improve quality, and organizations hope to benefit from *Linus's Law*: *'given enough eyeballs, all bugs are shallow.'*

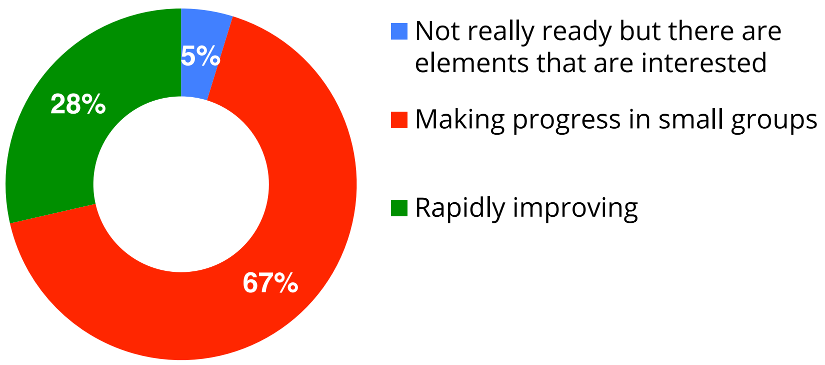


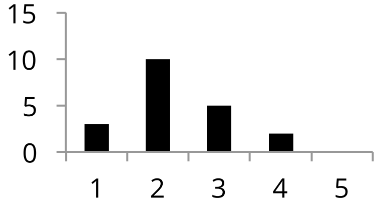
[1] K. Stol, B. Fitzgerald: Inner Source—Adopting open source development practices within organizations: A tutorial, *IEEE Software*, vol. 32, 2015

[2] M. Capraro, D. Riehle: Inner Source Definition, Benefits, and Challenges, *ACM Computing Surveys*, vol. 49, 2016

Organizational culture

Is your organization ready for InnerSource?





Failures (e.g. product release delays, downtime due to software defects, etc.) leads to scapegoating of the responsible person(s) involved

Note: We use the same convention for all Likert-scale questions in this report:

1 = Strongly disagree, 2 = Disagree, 3 = Neutral, 4 = Agree, 5 = Strong agree

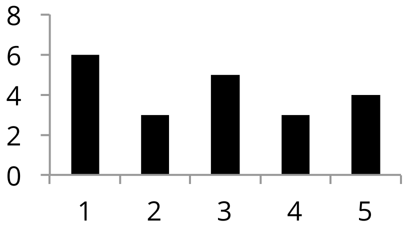
Organizational culture is perhaps the most critical factor when adopting InnerSource. Adopting the “open paradigm” represents a major shift in how the people in an organization see themselves and their responsibilities. InnerSource is all about open collaborations, and empowering developers to do the things they deem important.

InnerSource aims to increase transparency so that it becomes more clear who’s working on what, but also allows developers to contribute where they can. Some developers might be uncomfortable with this at first, as they might be embarrassed by the quality of their code. It’s important that developers work in a respectful environment where failure doesn’t lead to scapegoating, because this will reduce developers’ ability to try new things. Guy Martin highlights the importance of *psychological safety*[3] for InnerSource.[4]

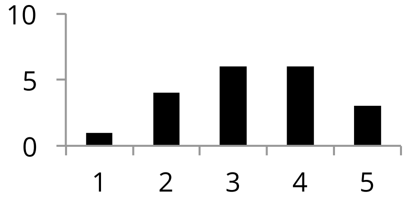
[3] A. Edmondson: Psychological Safety and Learning Behavior in Work Teams, *Administrative Science Quarterly*, vol. 44, 1999

[4] http://www.slideshare.net/GuyMartin18/inner-source-building-blocks-pull-request-culture-psychological-safety

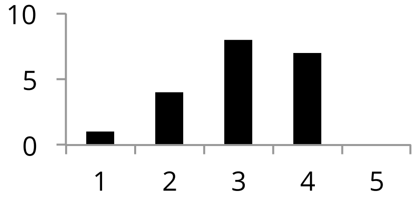
Organizational culture



All source code is universally accessible by anyone within my organization



My organization proactively encourages knowledge sharing



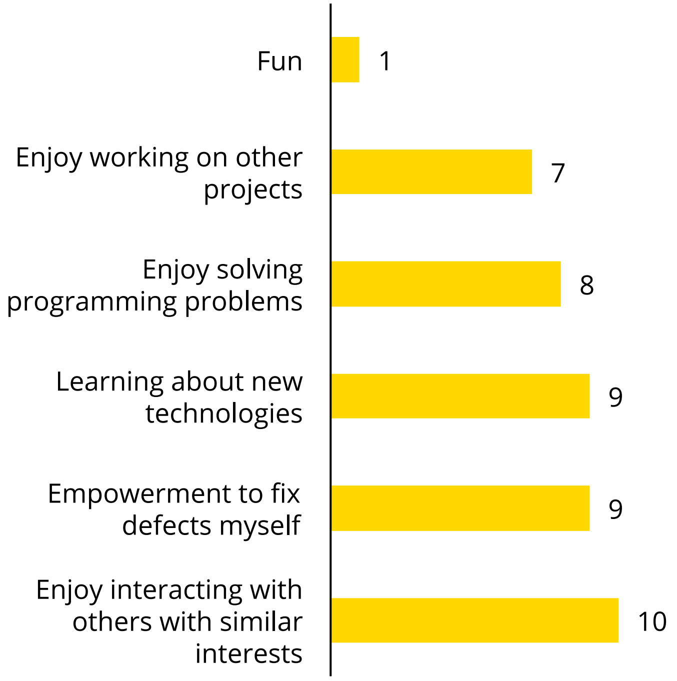
My organization is focused on performance instead of rules

Transparency implies that all development assets are available throughout the organization so as to enable developers to inspect others' code, and propose improvements where possible.

Some organizations actively encourage knowledge sharing, whereas actively maintain silos of knowledge. InnerSource aims to break down those silos, so that knowledge actively spreads. The degree to which an organization supports such initiatives is a sign of how "ready" it is for adopting InnerSource.

Furthermore, organizations that focus on rigidly complying with rules and procedures may have to reconsider their goals. InnerSource can only thrive in organizations that understand the value of end-to-end thinking, rather than seeing individual business units as profit centers which ultimately leads to local optimizations.

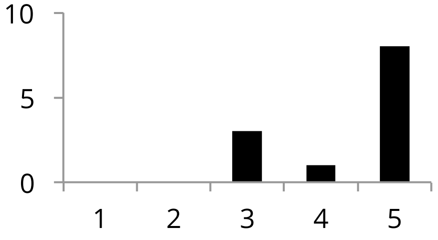
What motivates developers?



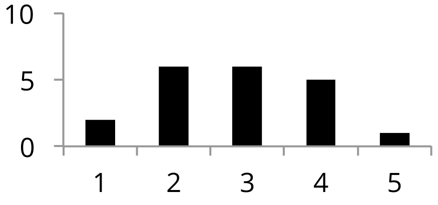
Respondents indicated a variety of reasons to participate in InnerSource programs. Many of these motivations can also be found in research on why developers work on open source projects.[5] Of the respondents who actively participated in InnerSource programs within their organizations, most indicated that interacting with other people with similar interests is a major reason to participate. For many respondents, empowerment also played an important role: rather than waiting for others to fix a bug, InnerSource enables developers to do it themselves, which can lead to a shorter time-to-market. This in turn may greatly increase job satisfaction. Enjoyment in solving programming problems was also a prevalent reason, as well as working with others. In addition, one respondent indicated that participating in InnerSource was fun in general.

[5] G. Von Krogh, S. Haefliger, S. Spaeth, M.W. Wallin: Carrots and Rainbows: Motivation and Social Practice in Open Source Software Development, *MIS Quarterly*, vol. 36, 2012

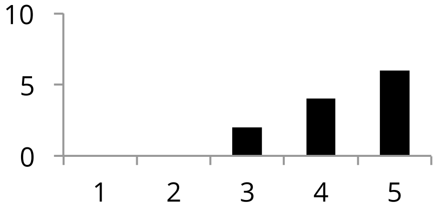
What motivates developers?



Participating in InnerSource development has improved my job satisfaction.



I spend more hours at work as a result of InnerSource program



InnerSource helps me to be more effective in my job.

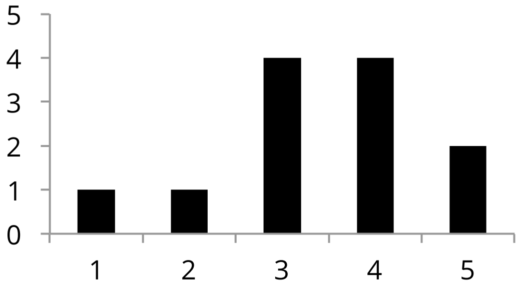
Most respondents indicated that InnerSource improved their job satisfaction. Job satisfaction is increasingly important for organizations who wish to retain their talent. Furthermore, satisfied employees are happier, and consequently may be more productive.[6]

InnerSource programs rely on motivated individuals, who go beyond their normal job description to make things happen. Some studies indicated that InnerSource contributors spend more time on the job,[7] but our survey was inconclusive on that, with widely varying results. What the survey did find was that most respondents thought that InnerSource helped them to be more effective in their job.

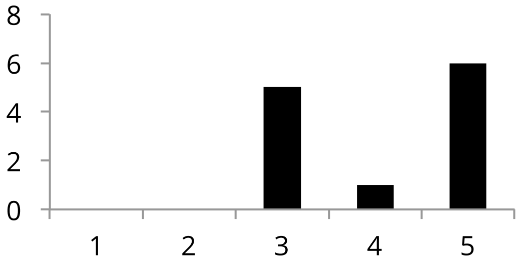
[6] D. Graziotin, X. Wang, and P. Abrahamsson: Happy software developers solve problems better: psychological measurements in empirical software engineering, *PeerJ*, 2:e289, 2014

[7] V.K. Gurbani, A. Garvert, and J.D. Herbsleb: A Case Study of a Corporate Open Source Development Model, *Proc. International Conference on Software Engineering*, 2006

Management support



New ideas for product features or process improvements are welcomed by my manager



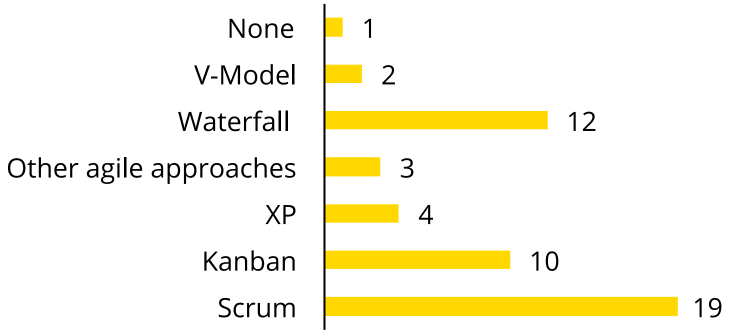
My manager supports me in contributing to InnerSource projects, even if they are not of direct use to my team or department.

Management support is a key success factor for InnerSource programs.[8] Support is needed at all levels of the organization, from executive to operational management. Executive support is required to ensure that resources are made available and also to make sure that the organization makes a long-term investment in the InnerSource program. Middle management must also support the program – it is all too common that a manager is evaluated based on the performance of his or her department, but such policies encourage local optimization of that specific department, rather than the whole organization. Most respondents indicated that their managers supported them to work on InnerSource projects.

[8] K. Stol, P. Avgeriou, M. Babar, Y. Lucas, and B. Fitzgerald: Key Factors for Adopting Inner Source, *ACM Transactions on Software Engineering and Methodology*, vol. 23, 2014

Methods and practices

Software development methods

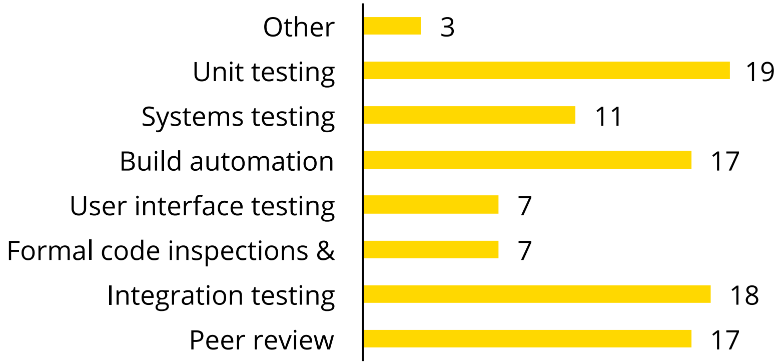


Respondents indicates a variety of development methods. Agile and lean methods are popular, with the agile method Scrum and lean Kanban practice being in widespread use. However, several respondents reported that their organizations still use plan-driven methods based on the waterfall approach and the V-model.

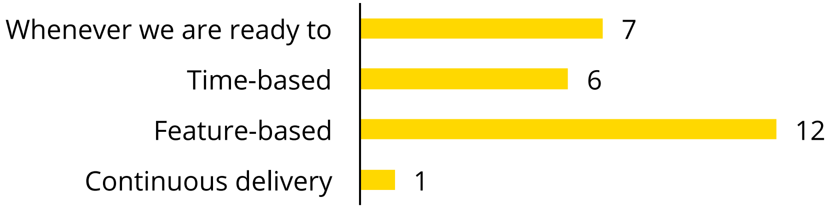
A variety of quality assurance practices is used as well. Unit testing and integration testing are prevalent, but peer review is also widespread.

A variety of release strategies is reported as well. Many open source projects have moved to a time-based release strategy as this can lead to better software quality. However, a feature-based strategy still seems to be more common among the respondent companies.

Quality assurance practices

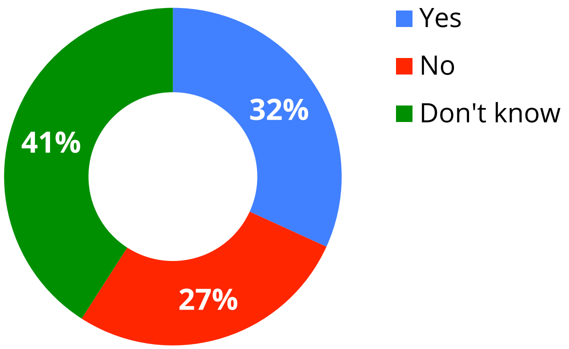


Release strategies

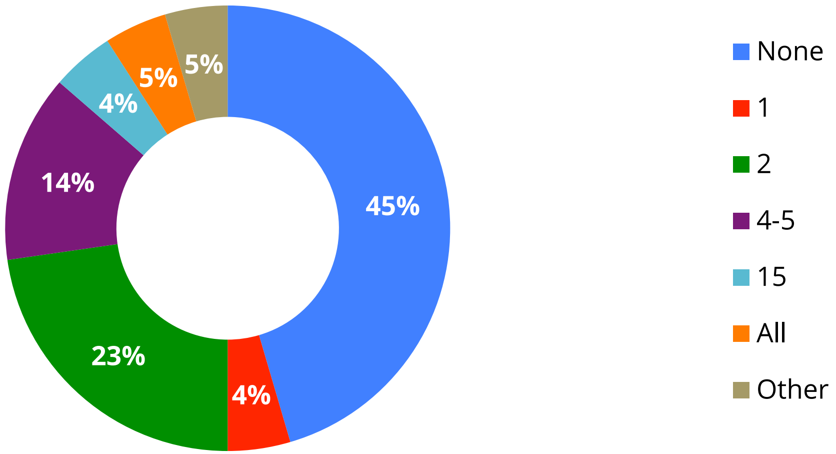


Methods and practices

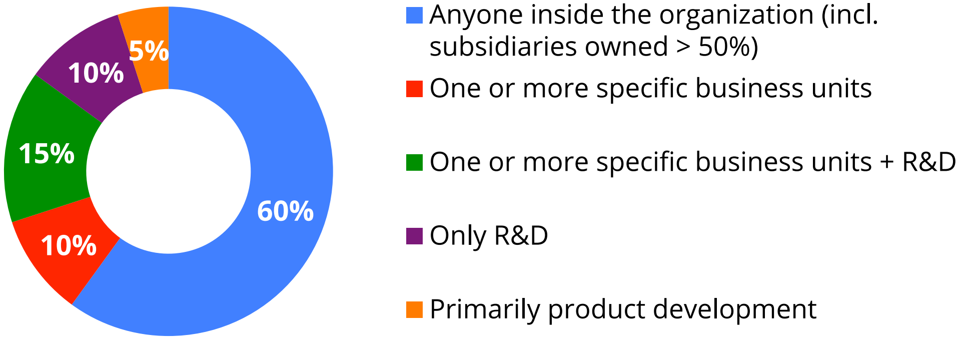
InnerSource projects subject to regulations



How many InnerSource projects do you contribute to?

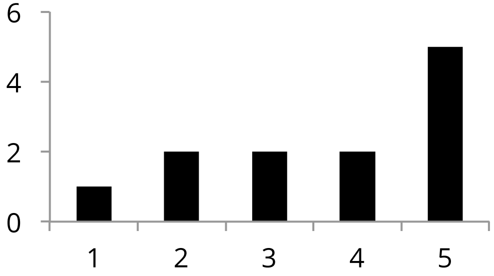


Access to source code

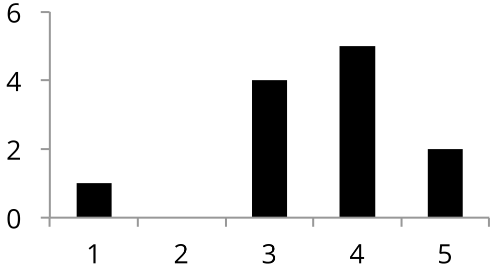


About a third of respondents indicated that some of their InnerSource projects are subject to regulations (e.g. FDA). Access to source code is generally universal, but in several cases only for specific business units or only the R&D division. Participation in InnerSource projects varied from 1 to “all” projects.

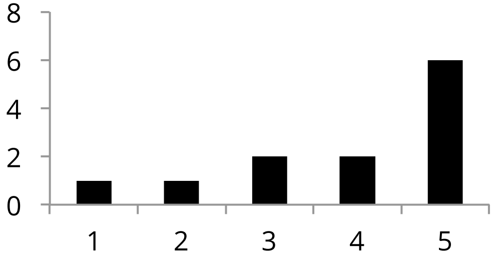
Methods and practices



The InnerSource project I work on is self-organizing - tasks are not assigned



I can completely self-direct my time, or in other words, I can always decide what I am working on



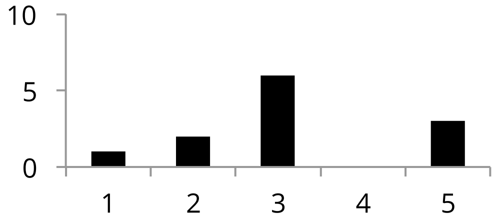
InnerSource enables me to work with colleagues with whom I wouldn't collaborate otherwise.

InnerSource is defined as the leveraging of Open Source development practices within the boundaries of an organization—or using Eric Raymond's metaphors, a *Bazaar within the Cathedral*.[9] A key characteristic of open source projects is that developers are self-organizing, which means that developers self-select those tasks that they want to do—either because they believe those tasks are important, or because they enjoy doing them.

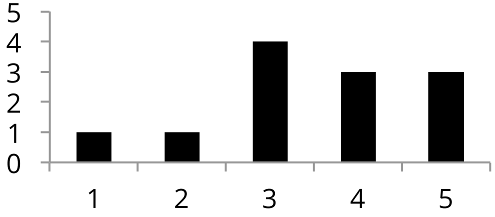
Most (though not all) respondents agreed that their InnerSource project they contribute on is self-organizing. Most developers also indicated, though to a lesser degree, that they are able to self-direct their time. Also, InnerSource offers more collaboration opportunities that otherwise mightn't happen.

[9] J. Wesselius: The Bazaar inside the Cathedral: Business Models for Internal Markets, *IEEE Software*, vol. 23, 2008

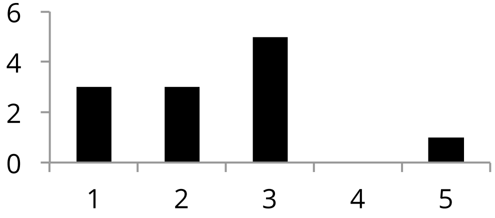
Methods and practices



The InnerSource project I work on has regular releases.



My contributions have been peer reviewed by people I have not met in real life.



When I review contributions, I feel obliged to be more gentle in my feedback if I know the contributor personally than when I don't know him/her.

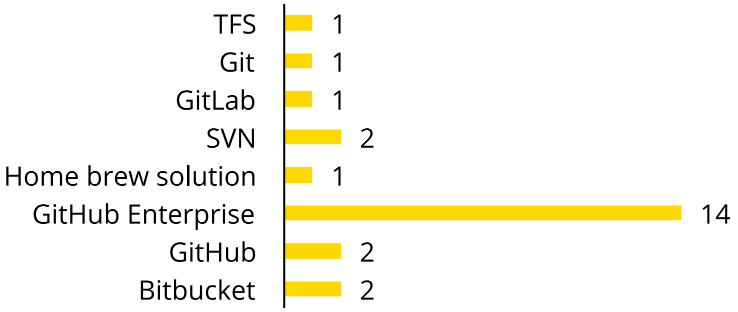
Release management plays an important role in any software project. A common approach is to release a new version whenever a given set of features are implemented—this is a feature-based release strategy. Previous research suggests that a time-based release strategy may improve quality.[10] Rather than finishing all planned features for a release, which may result in delayed releases, a time-based release results in regular new versions, no matter how small the additional features are. Most respondents didn't seem to use this strategy, however.

Another open source QA practice is peer review. Rather than having contributions reviewed by a friendly colleague, review by 'unknown' colleagues might be better as this means that the feedback is more objective. The respondents didn't feel the need to be more gentle in their feedback when they know the contribution's author, however.

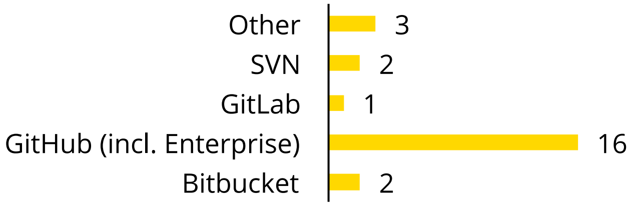
[10] M. Michlmayr, B. Fitzgerald, and K. Stol: Why and How Should Open Source Projects Adopt Time-Based Releases? *IEEE Software*, vol. 32, 2015

Tools

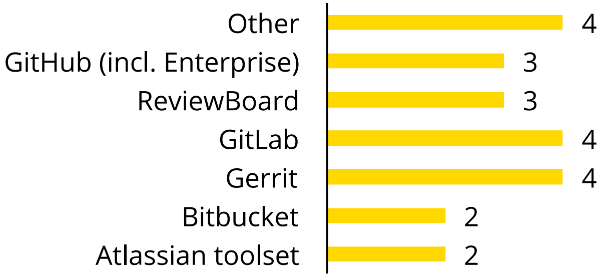
Code sharing platforms in use



Dominant code sharing platform

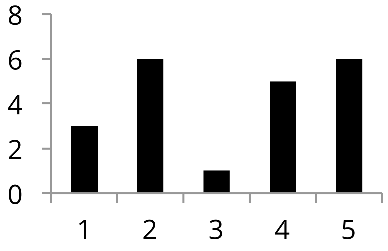


Code review tools

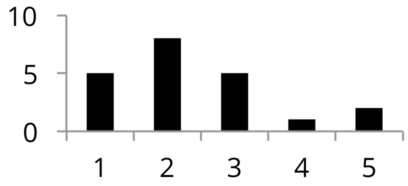


A wide variety of tools is in use, but GitHub Enterprise is by far the most widely adopted code sharing platform. Tools to support code peer review as well with a more uniform distribution across the different tools.

The use of different toolsets affects accessibility, and as such this might negatively impact cross-team collaborations. We found that most respondents didn't perceive setting up new tools or infrastructure to be problematic in their organizations.

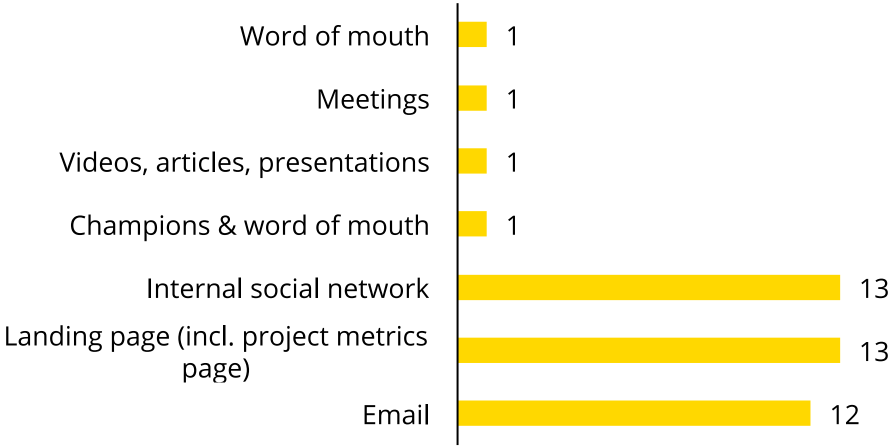


The variety in tools used in my organization inhibits collaboration and contributions from people in other teams or departments.



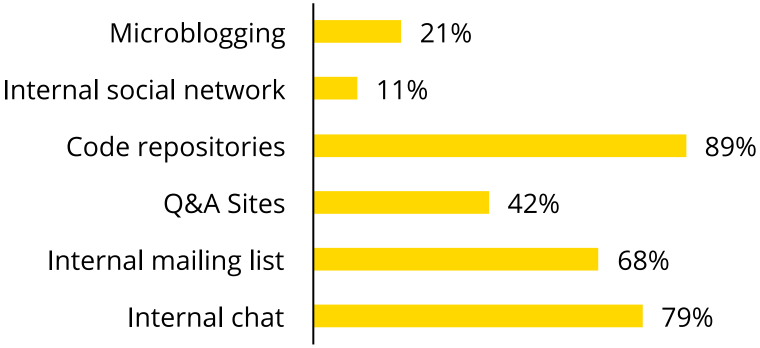
Getting new tools or infrastructure set up is straightforward in my organization or department.

Communication



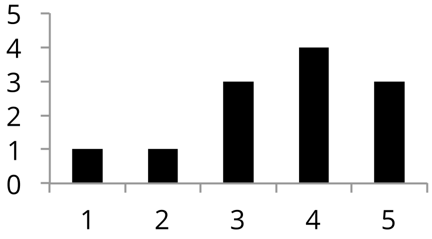
← In order to share information about InnerSource projecs within organizations, a variety of channels are used. Internal social networks and an InnerSource project landing page are very common, as is email. Word of mouth, meetings, and prepared advertising material (videos, articles, presentations) were also mentioned but were not common.

A variety of channels are used to facilitate communication in InnerSource settings. Microblogging (e.g. Twitter) was used by a fifth of respondents. Microblogging has become a channel also commonly used in open source projects.[11] Around a tenth indicated to use internal social networks. The most common channel is code repositories, followed by internal chat, mailing lists, and Q&A sites. →

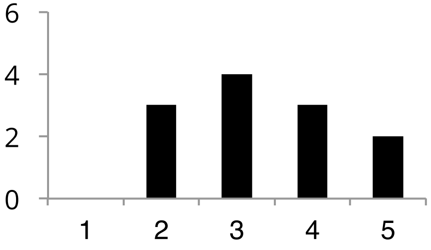


[11] X. Wang, I. Kuzmickaja, K. Stol, P. Abrahamsson, and B. Fitzgerald: Microblogging in Open Source Software Development: The Case of Drupal Using Twitter, *IEEE Software*, vol. 31, 2014

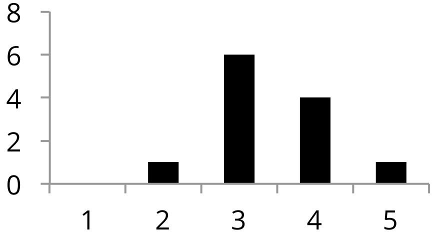
Contributions



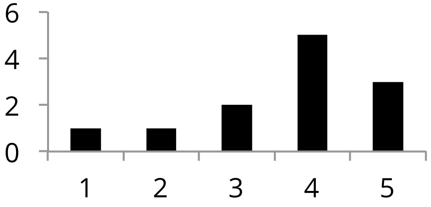
The contributions that are made vary widely in terms of variety of features and type of contribution (e.g. docs, code)



It is difficult to balance the need to release an InnerSource project as a product for the market and cultivating it as a "clean" project with good architecture and high quality



My team strictly follows a predefined development process



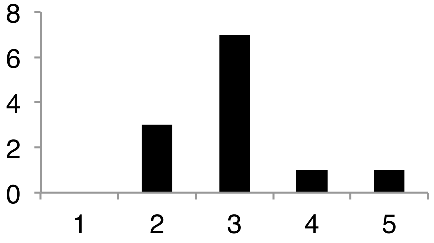
I have re-implemented or refactored parts of the InnerSource project I work on to make it better

One goal of InnerSource is to bring together different stakeholders, each bringing in different expertise and know-how. Our findings suggest that indeed most contributions vary in terms of type as well as different features. One concern in managing contributions and evolving a product is finding a sound balance between maintaining a clean architecture and developing a product that may have specific requirements from different stakeholders that might affect the product architecture. Some of the respondents indicated finding this balance can be challenging, but others disagreed.

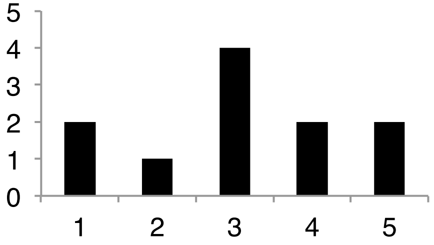
Respondents were divided over the question whether or not their teams followed a predefined process. Most respondents indicated 'neutral', though others generally agreed that a process was followed. Interestingly, a considerable group of respondents indicated to have re-implemented or refactored parts of an InnerSource project. Reimplementation (and refactoring) is a typical open source development practice,[12] whereby code is continuously improved.

[12] W. Scacchi: Free and open source development practices in the game community, *IEEE Software* vol. 21, 2004

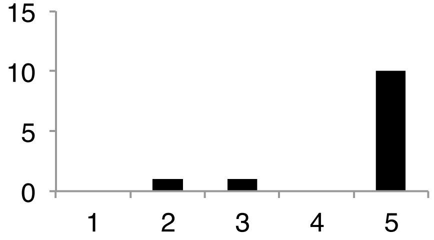
Contributions



Feature requirements are negotiated/discussed at great length with others before anybody starts working on them.



When making contributions to an InnerSource project, I try to write better code than when the contribution is not to an InnerSource project.

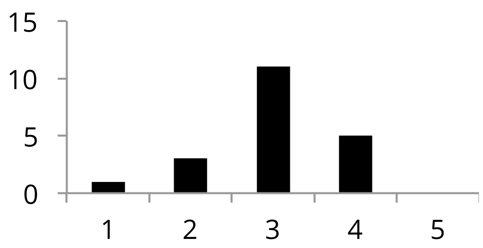


I don't mind anybody seeing the contributions I make.

Feature requirements are typical of traditional software development, and academic research in software engineering has focused extensively on *requirements engineering*, based on the assumption that requirements are the most important to get right from the outset, as changing requirements during a project is very costly. Unfortunately, getting requirements right is very difficult, which explains the popularity of agile methods such as Scrum. Agile methods "embrace change," and as such are designed to facilitate changes even late in a project. Requirements engineering is not typically done in OSS development—rather, requirements are often "asserted after the fact." As Linus Torvalds once said: "Show me the code!" Rather than negotiating over requirements, open source developers tend to value working code more.

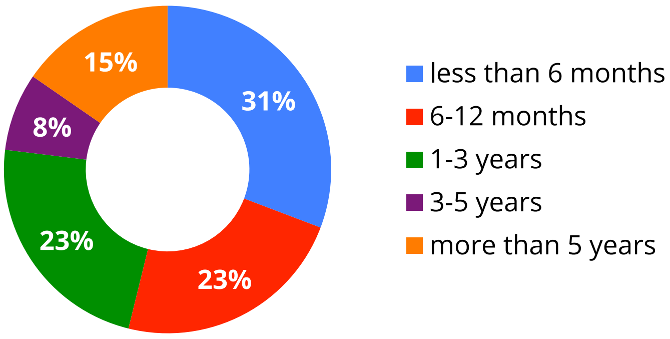
Almost all respondents indicated not to be bothered by an increased transparency and scrutiny of their contributions. This is interesting as this is a potential challenge in adopting InnerSource. While some developers make a bigger effort to write better code when contributing to an InnerSource project, others did not. In fact, most respondents were neutral on this issue.

InnerSource success



In your opinion, how successful is your organization's InnerSource program? (1=Not successful, 5=Very successful)

How long have you been contributing to Inner Source?

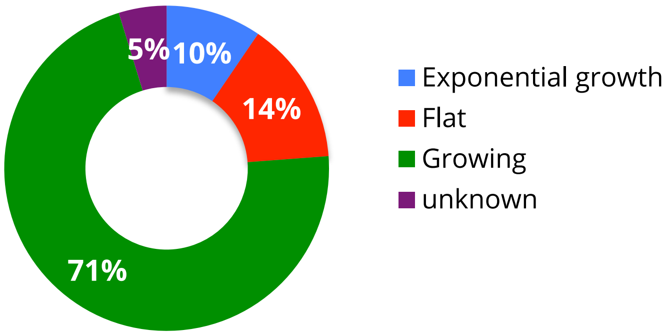


InnerSource is attracting considerable momentum with a thriving InnerSource Commons community. Most InnerSource programs have run less than one year, but there are organizations who have experimented with InnerSource for more than five years.

Most respondents judged the degree of success of their InnerSource program as "neutral," with only a few indicating no success. However, our data analysis also suggests a positive correlation between the duration of InnerSource contributions and program success.

Most respondents also indicated their InnerSource program is currently steadily growing, though some also perceived it to be stagnant. However, it is important that adopting InnerSource is not done overnight, and achieving success can take considerable time and effort.

Growth of InnerSource in your organization



Conclusion and outlook

InnerSource is an emerging trend in the software industry. The idea that open development methods offer many benefits is widely accepted in young companies and start-ups. However, for most established organizations used to hierarchy and development silos that inhibit cross-team collaborations, opening up the organization is very challenging. An industry-led community has emerged in recent years to share experiences on adopting InnerSource. This survey aims at learning more about the state of InnerSource in the software industry.

We believe InnerSource is a highly promising approach with great potential to improve software development processes, help overcome organizational barriers, improve software quality, shorten time-to-market, and improve job satisfaction, and consequently, retention of talented developers.

In the coming months, the InnerSource Commons community will continue working on sharing their expertise, experiences and knowledge, and we aim to codify this knowledge in reusable patterns. We will also explore the use of metrics to quantitatively characterize InnerSource programs.

Thank you for your interest in the State of InnerSource Survey, and we hope you will participate in future editions!

Methodology

We designed an online questionnaire targeting members of the InnerSource Commons community, which we advertised through the InnerSource Commons Slack channel. We received 22 responses in total. Twelve respondents indicated they contributed to an InnerSource project and were invited to answer an additional set of questions. As the number of respondents is limited, we cannot draw any conclusions that are statistically significant.

About the InnerSource Commons

The InnerSource Commons was founded in 2015 and is an industry-led initiative to advocate open development practices within organizations. The InnerSource Commons community interacts through an archived Slack channel, a dedicated mailing list, and organizes several events per year. Further information on the InnerSource Commons can be found on its website: www.innersourcecommons.org

Acknowledgments

We are grateful to all respondents for participating in this survey.

Authors

This survey was conducted by Dr. Klaas-Jan Stol of Lero—the Irish Software Research Centre, and Dr. Tim Yao of Nokia. Dr. Bora Caglayan of Lero contributed to the survey design. Any questions regarding this survey can be sent to: klaas-jan.stol@lero.ie

This report is supported by:

lero THE IRISH SOFTWARE RESEARCH CENTRE