Interactive Learning Environments

ISSN: 1049-4820 (Print) 1744-5191 (Online) Journal homepage: https://www.tandfonline.com/loi/nile20

Crowded house: an analysis of how the Virtual Learning Environment Moodle is built via bug tracker participants

Eamon Costello, Keith Johnston & Vincent Wade

To cite this article: Eamon Costello, Keith Johnston & Vincent Wade (2019): Crowded house: an analysis of how the Virtual Learning Environment Moodle is built via bug tracker participants, Interactive Learning Environments, DOI: 10.1080/10494820.2019.1678488

To link to this article: https://doi.org/10.1080/10494820.2019.1678488

INTERACTIVE LEARNING ENVIRONMENTS
https://doi.org/10.1080/10494820.2019.1678488

Crowded house: an analysis of how the Virtual Learning Environment Moodle is built via bug tracker participants

Eamon Costello [a], Keith Johnston [b] and Vincent Wade [b]

[a]Dublin City University, Dublin, Ireland; [b]Trinity College Dublin, The University of Dublin, Dublin, Ireland

ABSTRACT
This research investigated how the bug tracker database of the Virtual Learning Environment (VLE) Moodle is developed as an application of crowd work. The bug tracker is used by software developers, who write and maintain Moodle's code, but also by a wider public world of ordinary Moodle users who can report bugs. Despite many studies of the phenomenon of open source bug fixing and software building, much remains to be answered. Specifically, we sought to analyse the implications of this massively distributed collaborative development process for education and educational technology. The research examined the ways educators interface and contribute to the development of the VLE Moodle at the granular level of bug fixing as an example of a global crowdsourced activity. In this study, twenty community participants were interviewed, from fringe members, to key actors, including lead developers from the Open University, Moodle HQ and Moodle founder Martin Dougiamas. We uncovered rich stories of practices of community members. We found that projects are complex interplays of many actors assuming different roles and identities, and that brokers, or "kindly souls", play a key role in activities such as filing reports on behalf of others, or inducting new members.

ARTICLE HISTORY
Received 15 August 2018
Accepted 7 October 2019

KEYWORDS
Crowdsourcing; open source; Virtual Learning Environment; Moodle

Introduction

Perhaps one of the defining features of humankind is the ability to cooperate and collaboratively create artefacts and indeed knowledge. For most of history, this has involved a form of development over time – each new generation building on the knowledge and discoveries of the former – or simultaneously as a group, usually with some hierarchical forms of communication between the group members. However, a more recent phenomenon has been our increased ability to develop artefacts through the efforts of many people working simultaneously and in much more ad-hoc and less formal ways (Howe, 2006; Shirky, 2008). Indeed, people may work together who do not know each other, who are not formally designated to a task, and in ways in which the outcome is uncertain (Howe, 2006; Shirky, 2008). Howe (2006) gives what is generally regarded as the earliest accepted definition of crowdsourcing, describing it as the act of an organization "taking a function once performed by employees and outsourcing it to an undefined (and generally large) network of people in the form of an open call". Or, more succinctly, crowdsourcing can be seen as a "story of cooperation, aggregation, teamwork, consensus, and creativity" (Brabham, 2013, p. 1).

The open, decentralised, massive and accessible nature of the web, making it possible to bring the efforts of huge numbers of people simultaneously to bear, is one of the keys to crowdsourcing as a

CONTACT Eamon Costello eamon.costello@dcu.ie

phenomenon (Howe, 2008). Several analyses have been made that attempt to classify crowdsourcing and clarify or distinguish its relationship to other concepts such as open source (Brabham, 2008; Estellés-Arolas & González-Ladrón-de-Guevara, 2012; Geiger, Seedorf, Schulze, Nickerson, & Schader, 2011; Schenk, 2009). To examine the relation of open source software and crowdsourcing it is worth examining some aspects of open source itself including its genesis in *free software*. The free software movement came from those who believed in copyleft licensing which ensures that open source software cannot legally be changed into proprietary or closed source software. It operates according to the principle that those who modify free software are "compelled to leave copies behind for others to benefit" (Lakhani & Von Hippel, 2004, p. 293). The open source movement sought to break with the copyleft requirements of free software licences and to instead allow for mixing of proprietary and open source. While free software saw access to source code as a basic right, open source adherents took a more nuanced position that access to source code was desirable, but open software could co-exist with proprietary software (Lakhani & Von Hippel, 2004). This is in part because companies use open software according to a variety of business models (Bonaccorsi, Giannangeli, & Rossi, 2006; Chesbrough, 2006). Moreover, motivations to contribute to open source projects are complex and involve both extrinsic and intrinsic factors (Erikson, 1974; Orr, 2006; Waterman, Schwartz, & Conti, 2008).

As open source software projects have developed over time they have reached more areas of economic life (Mombach, Valente, Chen, Bruntink, & Pinto, 2018; Shaikh, 2016). As they increasingly interface with non-specialized end users, it becomes clear that open source is not simply a concern of software developers. Rather, such projects can be conceived of as having software developers who work and organise according to known open source models (Crowston & Howison, 2005) but also an outer layer of non-software developers, who interact according to patterns more akin to crowdsourcing models. A project may have translators, documentation writers, and bug report contributors. None of whom are software developers. They operate according to models of crowdsourcing rather than open source models (Ardichvili, Page, & Wentling, 2003).

The Virtual Learning Environment Moodle

The Virtual Learning Environment (VLE) Moodle is an example of an open source project that operates in the specialised area of education. VLEs or Learning Management Systems (LMSes) have become critical higher education infrastructure, comparable, it has been argued, to the traditional importance of libraries to universities (Costello, 2014; Williams van Rooij, 2011). Moodle was created by Martin Dougiamas (Dougiamas, 2007; Dougiamas & Taylor, 2003) and was soon rapidly adopted in higher education (Costello, 2014; Sclater, 2008). It has a bug tracking database which provides public data on the history and status of both bugs and feature requests that users such as instructional designers, system administrators, and teachers have submitted. The bug tracker can be conceived of as a boundary object (Allen, 2009) between project insiders and outsiders. Both groups together comprise a community of practice (Lave & Wenger, 1991; Wenger, 1999) of the form particular to distributed efforts such as open source and crowdsourcing. Participants may not know each other but contribute to a shared enterprise, by practices they engage in and take on roles and identities that come to shape and define such practices (Ardichvili et al., 2003). Although these models are well known, they have never been studied to the authors knowledge in the field of education. The present study hence sought to address this gap.

Although there is much research on how open source software develops including models of participation and joining (Mäenpää, Kilamo, Mikkonen, & Männistö, 2017; Von Krogh, Haefliger, Spaeth, & Wallin, 2012; Von Krogh, Spaeth, & Lakhani, 2003) there is a lack of research on specific projects in specialist domains (such as education) and also a relative dearth of qualitative narrative accounts from participants (Zimmermann et al., 2010). Specifically, this study sought to analyse how software developers and non-software developers combine to contribute to open source software through a case study of the open source Moodle bug tracker database. It also sought to address a significant gap in the scholarly understanding of this topic through interviews with project leaders which have not been conducted to date in this context.

Methods

The goal of this case study (Yin, 2009) was to investigate the practices of the different participants in the community of the Moodle bug tracker as they report and attempt to resolve issues with the software. The following guiding Research Questions (RQs) were formulated to direct the research enquiry:

RQ1: Which factors and processes contribute to the successful resolution of issues in the Moodle bug tracker?

RQ2: What are the role and identities of participants in this community including of non-software developers and other non-technical participants?

RQ3: How do educators such as teachers come to gain access to inner levels of this community?

Following from a social constructivist and interpretivist perspective a case study methodology was adopted as one that is both synonymous with both qualitative research and mixed methods (Creswell, 2012; Yin, 2009). In the first phase of the research, 100 participant profiles were analysed from the Moodle bug tracker database. In addition, 20,830 tracker issues were retrieved through queries to the Moodle bug tracker database. This was carried out by making queries to the JIRA bug tracking database which has a public interface that allows powerful queries of its contents via its own query language known as JQL (Jira Query Language) (Radigan, 2015). A full explication of the structure and findings of this phase of the research is beyond the scope of this paper however, they provided information on how long it took issues in the database to be resolved, how many people vote for issues to be fixed or resolved and who could be assigned to issues. These concepts informed the development of an interview schedule and also helped identify key members of the community worth interviewing. Participants were also used to identify other participants via "snowball sampling" (Guba & Lincoln, 1985, p. 233). Strategies to help counter potential bias included selecting members who had left the community and people with only a minimal involvement in the community, both of whom might be likely to speak more freely.

The semi-structured interview protocol (Seidman, 2006) followed from the research questions and focused on: factors interviewees perceived to be important in issue resolution or non-resolution; narrative accounts of resolutions; describing the roles of insiders i.e. those capable of being assigned to issues (*assignees*) and outside, or more casual, contributors – *non-assignees*.

Institutional Ethical approval was attained to conduct a series of interviews. A variety of different interviewees were sought and twenty participants were interviewed for on average 53 min in face to face (2) and skype (18) meetings. Twelve of the interviews were bug tracker issue assignees, and eight were non-assignees. A majority were experienced software developers (15) and the remainder (5) had teaching or higher education backgrounds. Four of the 16 software developers were also former teachers. Four interviewees were female and sixteen male. Interviewees were based in the UK (7), Ireland (2), Australia (2), New Zealand (2), Belgium (2), Czech Republic, Italy, Austria, Germany and the USA. English was the second language of five of the interviewees. All interviews were conducted through English however with the first author and quotes are reproduced as close to the original conversation as possible. Three participants had left the Moodle community i.e. were no longer involved, whereas the rest were still active members at the time of interview. This composition was selected as these two types could potentially have quite different perspectives. Six interviewees were working in universities, four in Moodle HQ, four in Moodle Partners, four in schools and two in miscellaneous others. This provided a heterogeneous sample which proved important for achieving variation and aiding the search for disconfirming evidence in the analysis phase (Kuzel, 1992). Key actors in the core Moodle community were happy to waive their right to anonymity and be identified including Michael du Raadt the head of software development, Tim Hunt a lead developer from the OU UK, Helen Foster the Moodle community manager, and Moodle founder Martin Dougiamas.

The interviews were transcribed into full orthographic transcripts and imported in the qualitative analysis software Nvivo. A dual cycle method of coding was employed (Saldaña, 2009). In the first cycle of coding, used to generate what in Nvivo are termed "free nodes", there were three related methods employed in tandem: *descriptive* coding, *in vivo* coding and *process* coding (Saldaña, 2009). *Descriptive* coding was used to create a basic vocabulary of the data. *In vivo* coding was used to identify terms that were common to participants in describing their world. Thirdly, *process* coding was used as many of the interview questions centred on processes, on resolution of problems, and also on the dynamics of entering and leaving the community. So for example RQI which centred on issue resolution led to several process-type codes e.g. "Decid*ing*", "Wait*ing*". These codes were then synthesised and aggregated into higher level themes during a second cycle of coding (Corbin & Strauss, 2008).

Results

Accounts of issue resolution factors

RQ1 sought to analyse the factors and processes involved in the resolution of issues in the Moodle tracker. Over 30 factors that interviewees believed to be important to issue resolution were initially identified. These views formed a belief matrix of the community that contained both commonalities and contrasts. An example of this is illustrated in Figure 1 below:

These factors were then divided into three code groups in the second coding phase. Each group involved either: the submitter of the issue, the assignee assigned to fix the issue or the eventual submitter of any code to fix the issue. The potential paths to successful resolution were mapped as per Figure 2 below which shows an example for the issue submitter:

We will next outline selected factors which participants professed to be important and illustrate via relevant quotes. All participants claimed that the quality of the information given in an issue was important to its resolution. Although these steps were sometimes described as informational – "numbered points, you-know, do this then do that, then do that" (Interview M10) the ability of the bug reporter to catch the reader's attention and tell a persuasive story of the bug's impact was also important. Another strong theme that emerged was responsiveness of the issue submitter as illustrated by this assignee account:

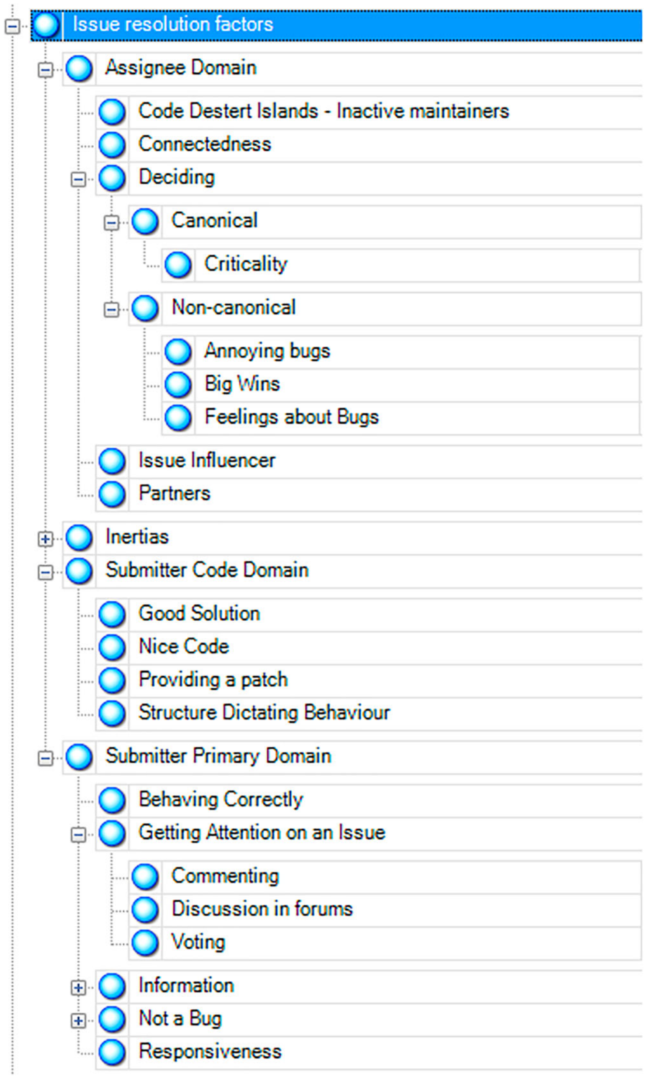
> It helps to be responsive also. If a reviewer looks at it and says there's a problem here, if the person is very reactive and goes 'Okay I have fixed those now', you-know, 'What do you think of it now?' That really helps. I think it's a big thing to be involved and engaged obviously. It's human nature. (Interview M20).

Moreover, an assignee's tenacity in their responsiveness could even result in a closed issue being re-opened or an issue classed as invalid (i.e. not a bug) being reclassed as valid.

Validity is also a critical juncture in the issue lifecycle (see Figure 2), as if an issue is not deemed valid by the assignee it will almost never be fixed. Invalid issues may be misreported or may be duplicates of other issues:

> I mean it's quite funny because way back when we were still testing out Moodle two point zero, before it came out, I found an issue in the File-Picker and we sorted it and then, about two or three months later, Martin [Dougiamas] reported the same issue **aw right!** Because he hadn't realised **ha, ha** that I'd already reported it and it was a duplicate and I thought: well, even *he* can do it, you-know? (Interview M16).

A common factor that many participants believed important to issue resolution was the behaviour of the submitter. Some interviewees professed strong feelings that specific normative behaviours of patience and politeness should be expressed by submitters. Submitters exhibiting such behaviours were felt to be more committed to the project or "on the same ship" (Interview M2) and participants professed that "most people are really agreeable and understanding" (Interview M19). By contrast behaviour considered as rude can be a negative factor in issue progression. It may influence which issues an assignee chose to work on and this view was not uncommon:



- Issue resolution factors
  - Assignee Domain
    - Code Destert Islands - Inactive maintainers
    - Connectedness
    - Deciding
      - Canonical
        - Criticality
      - Non-canonical
        - Annoying bugs
        - Big Wins
        - Feelings about Bugs
    - Issue Influencer
    - Partners
  - Inertias
  - Submitter Code Domain
    - Good Solution
    - Nice Code
    - Providing a patch
    - Structure Dictating Behaviour
  - Submitter Primary Domain
    - Behaving Correctly
    - Getting Attention on an Issue
      - Commenting
      - Discussion in forums
      - Voting
    - Information
    - Not a Bug
    - Responsiveness

Figure 1. Snapshot from Nvivo of issue resolution success factors coding.

> So if people are getting upset and fired up I'm more inclined to go and help someone who's being polite about it and look at their book rather than the person who's getting upset. Getting upset's not going to help the process at all; it's going to make it worse for them (Assignee view).

Newcomers must make efforts to understand and demonstrate the "ethos of the community" and may require an induction to it: "there's a fair amount of education and almost training that needs to be done to teach them how to participate in an open source project" (Interview M3).

Lastly, new feature development was seen to rank lower in importance than bug requests as determined from querying the database. A Chi Square test was conducted on all the issues in the database which confirmed that new features were less likely to be implemented than bug fixes. This success is significant χ^2 (1, $N = 14{,}119$) = 1278.9, $p = 4.42^{-280}$.

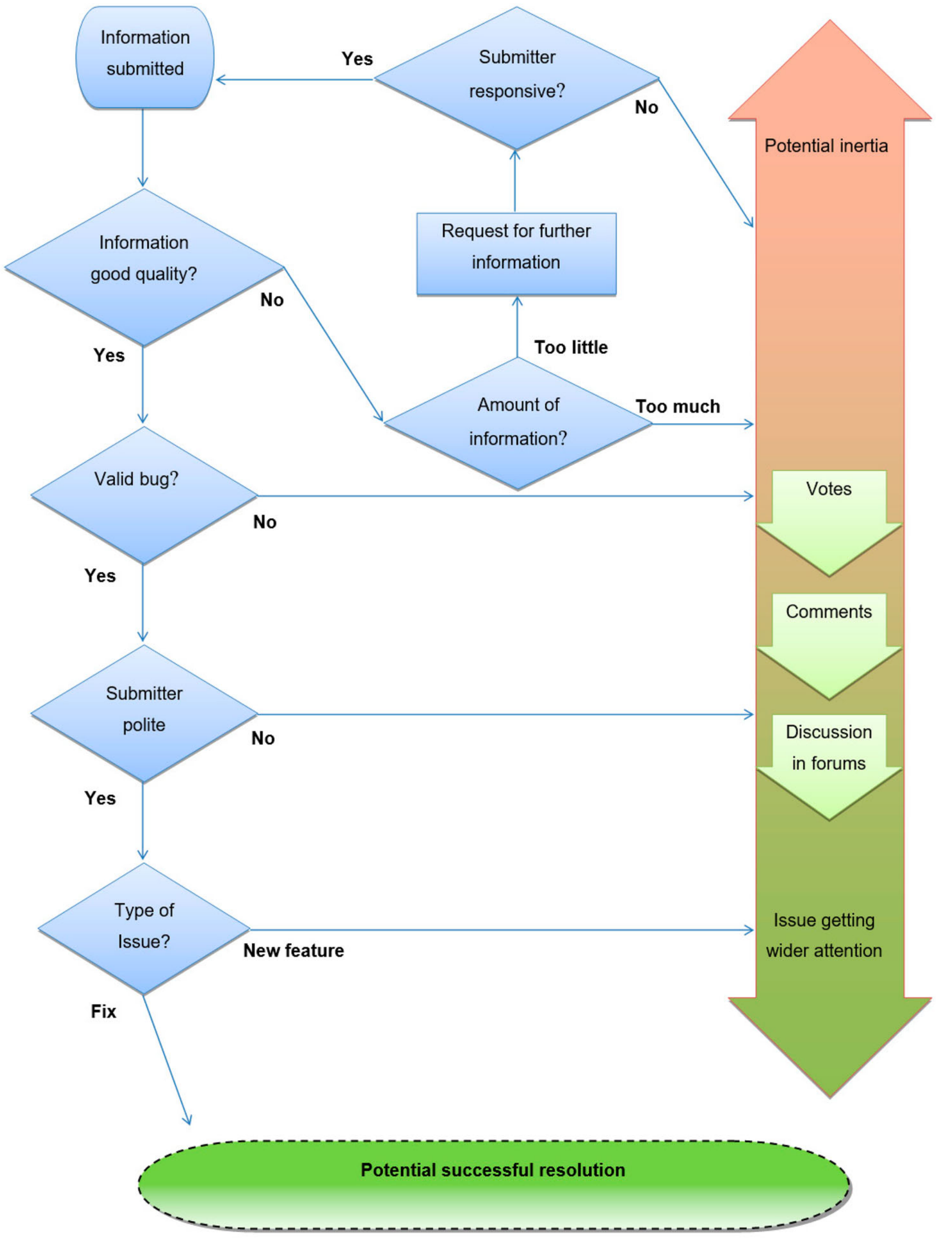


Figure 2. Potential paths for issue submitter.

Roles and identities

RQ2 Sought to determine the roles and identities of participants in the community including the roles played by non-software developers and other non-technical participants. Just over half of participants self-identified as "developers". One participant when asked which skills allowed them to contribute, conveyed this succinctly: "well being able to write good code goes without saying" (Interview M1). The ability of developers to write good code was critical to the identity of these participants. Hence they can "enjoy the complexities of some of the challenges" of developing Moodle

(Interview M4). They may display a "passion" for what they do, one that may develop from "fiddling around with Moodle" (Interview M10) into "a bit of an obsession" (Interview M1).

A common developer theme was a need to "scratch an itch" (Interview M1) a compulsion to try and fix or solve a problem. Founder Martin Dougiamas described Moodle's origins in this vein: "I just had this need to build this thing to prove to myself that I could do it" (Interview with Martin Dougiamas, 2013). Bugs could be "annoying" or "interesting", implying developers may have an intrinsic motivational impetus to work on them. This building and tinkering, which open source code allows for, was important to these interviewees' individual identities. Learning similarly is a key part of their makeup:

> The reason for my involvement in open source is usually linked to the pleasure of understanding things. Usually I am not in need to know for working reasons. It's hungry for knowledge – that is my food! (Interview M12).

Non-core developers demonstrated excitation about interaction with core developers:

> And it's also cool that you, we are sometimes in contact with Petr Skoda and David Mudrak and I think it's … well for me it's kind of cool that wow these are the core developers! (Interview M9).

Several participants had originally been lecturers/teachers but had transitioned into software development. They considered their teaching background to be important to their role within the project as it provided them with an insight into the ultimate end users (students/teachers) as they "know" the people who "actually use Moodle […] from an educational point of view, rather than just [a] purely technical point of view" (Interview M8).

The role of brokers emerged as an important one. Although these people are not numerous they act as mediators between people in different layers of the project such as for instance the Moodle Development Manager who triages issues:

> I'll pipe it to him [named developer], and that doesn't mean that they'll necessarily start working on it straight away, but it means that they'll come up in their list so they'll probably go and have a look at it and if they've been working in the area recently they can have a sort of second opinion on it (Interview with Michael du Raadt, 2012).

Helen Foster, Moodle Community Manager may act in a similar role as a link between submitters and assignees and use the discussion forums as a source of information:

> When people post in the forums and they mention a tracker issue and I go and look at it and if I think it hasn't got enough attention then I will go and contact an HQ developer and say 'hey is there any chance you could have a look at fixing this?' (Interview with Helen Foster, 2012).

Brokers are deemed to be connected individuals. An active community member may be able to progress an issue that they are not formally assigned to by virtue of these connections:

> If it's really necessary I know a lot of the developers personally okay, and I know who works on what, and if it's really urgent then I can make a little use of that. (Assignee view).

Teaching can be an aspect of the identity of a broker who "straddles both camps":

> Being a teacher gives me real key scenarios about how things are put into practice and so I've sort of seen myself almost as a translator at times, in the community between the user and the developer (Interview M3).

Participants could identify themselves as mediators between technological and commercial/business domains and not just educational ones such as claiming to be: "50% technology, 50% training and 50% business" making them "a bridge between the use case and the systems" (Interview M7).

A common and critical act of brokerage involved filing a bug report on someone else's behalf:

> Sometimes, people actually just don't sign up for tracker accounts […] then hopefully some *kindly soul* in the forums will spot it and write it up as a bug report (Interview M10 emphasis added).

Component leads are developers who are in charge of a section of the code. They cited these "kindly souls" as fulfilling an important function for them. These could be key members of the community such as the Community Manager Helen Foster – an example of someone who specialises in general brokerage. For instance, she encourages or even reports issues herself on behalf of others (Interview with Helen Foster, 2012) for any module. A more common occurrence is when a broker will file issues only on behalf of users for the particular module or code piece that they are interested in.

Trajectories

RQ3 sought to determine how educators such as teachers can come to gain access to inner levels of the community. A particular trajectory of joining and entering a deeper layer of the community can happen via brokerage. A broker may file an issue on behalf of an outsider member or bring it to a developer's attention. Several interviewees described how they felt they owed part of their involvement in the project to key individuals such as receptive lead developers. The establishment of rapport was important here in order to establish trust. If a participant can demonstrate their ability they may get additional rights or status on foot of this:

> I submitted quite a lot of patches within a short space of time and at that point I got given the pull request privileges on the Jira instance. (Interview M6).

This process however is "informal" (Interviews with Martin Dougiamas, Helen Foster, Michael du Raadt). The newcomer must peripherally participate in discussion forums, the bug tracker, or developer chat meetings to determine who the key brokers are as this is not always apparent and as the process is not formalised.

Martin Dougiamas himself is a key broker and describes his role in the earliest days of project as being concentrated on inducting new members:

> [In the early days] I was really about encouraging, […] trying to encourage everyone to be open, and you-know making relationships with people. […]A lot of the weirdness of [the Moodle code], some of the imperfections if you like, have gone in that way. But it was hard to say no when it was smaller and someone had to spend three months, you know, busting their gut to get this code done. Maybe it wasn't quite how I would have done it, but I wanted just to be open and accepting. (Interview with Martin Dougiamas).

One early participant in the project describes his direct influence upon him: "The guy, author, was really friendly and there was a cool spirit in their small community […] and I fell in love with Moodle" (Interview M2). Another participant recalls that the "lead developer was there constantly on the forums and if you had a question he was there so it was very nice to work with" (Interview M17).

Dougiamas describes the level of effort required to sustain such interaction:

> And then [Moodle] rapidly became used, it just became all my life and just eighteen hours a day while doing my PhD I was basically just waking up in the morning, going to bed at midnight and just basically just powering through it for a couple of years (Interview with Martin Dougiamas).

Discussion

The preceding sections examined the identities and associated roles of the Moodle community and analysed how educators participate in such a community to effect change. The community was examined through the lens of identity: both those of individuals in their roles as developers or teachers, specialists or brokers and also of their community memberships and loyalties. Roles and identities are not fixed and can be changed as participants move along a trajectory according to their involement.

The resolution of issues in a bug tracker may turn out to be complex stories and processes (Hooimeijer & Weimer, 2007; Aranda & Venolia, 2009). Some aspects of factors suggested here from narrative accounts are not widely reported in the existing literature such as submitter responsiveness. This is not a prominent theme in the literature though is highlighted by Breu, Premraj, Sillito, and Zimmermann (2010).

Another factor identified was the politeness of the issue submitter. It is hence important that accounts of bug fixing consider this factor. However, we should also balance this against the contrary view of one participant who described the importance of "really having to fight with the [assignee]" to get an issue resolved i.e. politeness was the advised strategy of assignees for non-assignees, but only non-assignees themselves made a case for a forceful approach.

Perhaps the complexity of this aspect of bug resolution is why it does not appear prominently in the literature i.e. it is difficult to answer systematically, relative to other questions. Bug-fixing is, however, a complex social process (Sack et al., 2006) and we should not be entirely surprised that it is subject to strong non-predicable influences.

Our findings contribute examples from a prominent open source project in the area of Education of which little was reported previously. They re-enforce the point that "the histories of even simple bugs are strongly dependent on social, organizational, and technical knowledge that cannot be solely extracted through automation of electronic repositories, and that such automation provides incomplete and often erroneous accounts of coordination" (Aranda & Venolia, 2009, p. 1).

An important brokerage act that was identified in this research was that of filing bug reports on behalf of a third party. No studies on this phenomenon were found in the literature of open source bug fixing. There is no way to tell from a bug tracker itself whether someone has filed an issue for someone else or not. It is hence difficult to quantify how pervasive this form of proxy issue reporting is. Essentially, we need to be careful about ascribing ownership or provenance of issues too tightly. They are part of the social fabric of the community.

It may be that filing issues on behalf of others is common practice in bug trackers, but it is also likely that there is a particular type of this brokerage that is linked to projects like Moodle. We heard how interviewee M12 filed issues on behalf of others after reading the discussion forums dedicated to the particular area of Moodle they were interested in. In the case of one participant however, we found that she filed bug reports on behalf of others for *any* area of Moodle i.e. she is *generally* helping users and the project more so than a specific area of Moodle over which she feels some ownership or expertise.

Brokers are important in providing "a buffer between developers and peripheral users" (Crowston & Howison, 2005). This may become more significant as a project matures, becomes larger and more complex. Hence we see in Moodle highly specialised roles founded on activities of mediation such as those of Helen Foster, Community Manager of the Moodle discussion forums or Michael Du Raadt as Development Manager who described a key aspect of his role as being to triage issues in the tracker and "pipe" them to potential, suitable fixers.

Code contributions are important for joining the inner core community but advancing on this conception this study attempted to consider the complexity of the tasks and roles involved in the "hybrid weaving accomplished by the actors of this distributed, collective design process" (Sack et al., 2006, p. 229). There is more than simply the gift-giving of code going on. Being a helpful community member was a strong emergent theme relating to a participant's establishment in the community. Establishing this identity may not require any coding and many other tasks were found to fall into this category, such as writing documentation and filing bug reports on behalf of others. Although these non-programming tasks are mentioned in other open source projects their importance will be greater as projects mature and become more complex (Barham, 2012) such as Moodle has. Harnessing the power of the crowd for such activities may prove critical to open source projects in the future.

Conclusion

This study contributes to the scholarly understanding of how crowds may work to build a shared enterprise in the field of educational technology. It examined the interplay between specialists and non-specialists in a large open source project and in so doing addressed a gap in the literature, as there are few if any qualitative studies on open source educational projects. Significant findings included the importance of submitter behaviour in issue resolution, the identification that bug fixes are complex interplays of actors who assume different roles and identities, and finally the key role that brokers play in such projects by filing reports on behalf of others as “kindly souls” or in inducting new members into such a community.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Dr Eamon Costello is Head of Open Education in the National Institute for Digital Learning in Dublin City University. He has research interests in digital, online and open learning; STEM education and Open Science. He has extensive expertise in developing and delivering online programmes.

Dr Keith Johnston lectures in the area of ICTs in education in the School of Education, Trinity College Dublin, The University of Dublin. He teaches courses in ICTs (Information and Communications Technologies) in Education as part of the Professional Master in Education and the Master in Education programme. His main research interests are the development and implementation of ICT policy in primary and post-primary education, and in the use of ICTs to support teaching and learning at these levels.

Professor Vincent Wade holds the Chair of Computer Science in the School of Computer Science and Statistics, Trinity College Dublin, the University of Dublin and the personal chair of Artificial Intelligence. Vincent is CEO and Director of the ADAPT Research Centre for Digital Content Platforms and Application Research. He has secured in excess of 50 million euro in grant funding and authored over three hundred scientific papers in peer reviewed research journals and international conferences.

ORCID

Eamon Costello http://orcid.org/0000-0002-2775-6006
Keith Johnston http://orcid.org/0000-0001-9382-6335
Vincent Wade http://orcid.org/0000-0001-6133-5160

References

Allen, W. (2009). Boundary objects in hybrid commercial/open-source software development firms.

Aranda, J., & Venolia, G. (2009). The secret life of bugs: Going past the errors and omissions in software repositories. *Proceedings of the 31st International Conference on Software Engineering*. Vancouver, Canada, IEEE Computer Society.

Ardichvili, A., Page, V., & Wentling, T. (2003). Motivation and barriers to participation in virtual knowledge-sharing communities of practice. *Journal of Knowledge Management*, *7*(1), 64–77.

Barham, Adina. (2012). The impact of formal QA practices on FLOSS communities – the case of mozilla. *Open Source Systems: Long-Term Sustainability*, *378*, 262–267. https://doi.org/10.1007/978-3-642-33442-9

Bonaccorsi, A., Giannangeli, S., & Rossi, C. (2006). Entry strategies under competing standards: Hybrid business models in the open source software industry. *Management Science*, *52*(7), 1085–1098.

Brabham, D. C. (2008). Crowdsourcing as a model for problem solving: An introduction and cases. *Convergence: The International Journal of Research into New Media Technologies*, *14*(1), 75–90. doi:10.1177/1354856507084420

Brabham, D. C. (2013). *Crowdsourcing*. Cambridge, MA: The MIT Press.

Breu, S., Premraj, R., Sillito, J., & Zimmermann, T. (2010). Information needs in bug reports: improving cooperation between developers and users. *Presented at the Proceedings of the 2010 ACM conference on computer supported cooperative work* (pp. 301–310), ACM.

Chesbrough, H. (2006). *Open business models: How to thrive in the new innovation landscape*. Brighton: Harvard Business Press.

Corbin, J., & Strauss, A. (2008). *Basics of qualitative research: Techniques and procedures for developing grounded theory*. Thousand Oaks, CA: Sage.

Costello, E. (2014). Opening up to open source: Looking at how Moodle was adopted in higher education. *Open Learning: The Journal of Open and Distance Learning*, *28*, 2. doi:10.1080/02680513.2013.856289

Creswell, J. W. (2012). *Qualitative inquiry and research design: Choosing among five approaches*. London: Sage Publications.

Crowston, K., & Howison, J. (2005). The social structure of free and open source software development. *First Monday*, *10* (2–7). https://firstmonday.org/ojs/index.php/fm/article/view/1207/1127

Dougiamas, M. (2007). OSS Watch – Moodle: A case study in sustainability. Retrieved from http://www.oss-watch.ac.uk/resources/cs-moodle.xml

Dougiamas, M., & Taylor, P. (2003). Moodle: Using learning communities to create an open source course management system. In D. Lassner & C. McNaught (Eds.) (pp. 171–178). *Presented at the Proceedings of the EDMEDIA 2003 Conference*, AACE.

Erikson, E. H. (1974). *Dimensions of a new identity: The 1973 Jefferson lectures in the Humanities*. London: WW Norton.

Estellés-Arolas, E., & González-Ladrón-de-Guevara, F. (2012). Towards an integrated crowdsourcing definition. *Journal of Information Science*, *38*(2), 189–200. doi:10.1177/0165551512437638

Geiger, D., Seedorf, S., Schulze, T., Nickerson, R. C., & Schader, M. (2011). *Managing the crowd: Towards a taxonomy of crowdsourcing processes*, 12.

Guba, E. G., & Lincoln, Y. S. (1985). *Naturalistic inquiry*. Newbury Park, CA: Sage Publications.

Hooimeijer, P., & Weimer, W. (2007). *Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering* (pp. 34–43). ASE 07.

Howe, J. (2006). The rise of crowdsourcing. *Wired Magazine*, *14*(6), 1–4.

Howe, J. (2008). *Crowdsourcing: How the power of the crowd is driving the future of business*. Manhattan: Random House.

Kuzel, A. J. (1992). Sampling in qualitative inquiry. In B. Crabtree & W. Miller (Eds.), *Doing qualitative research*. Newbury Park, CA: Sage Publications Inc.

Lakhani, K. R., & Von Hippel, E. (2004). How open source software works: “free” user-to-user assistance. In Cornelius Herstatt & Jan G. Sander (Eds.), *Produktentwicklung mit virtuellen Communities* (pp. 303–339). Wiesbaden: Springer.

Lave, J., & Wenger, E. (1991). *Situated learning: Legitimate peripheral participation*. New York, NY: Cambridge University Press.

Mäenpää, H., Kilamo, T., Mikkonen, T., & Männistö, T. (2017). Designing for participation: Three models for developer involvement in Hybrid OSS projects. In *IFIP International Conference on Open Source Systems* (pp. 23–33). Springer.

Mombach, T., Valente, M. T., Chen, C., Bruntink, M., & Pinto, G. (2018). *Open source development around the world: A comparative study*. ArXiv Preprint ArXiv:1805.01342.

Orr, J. E. (2006). Ten years of talking about machines. *Organization Studies*, *27*(12), 1805–1820.

Radigan, D. (2015) Search JIRA like a boss with JQL. Retrieved from https://confluence.atlassian.com/jiracore/blog/2015/07/search-jira-like-a-boss-with-jql

Sack, W., Détienne, F., & Ducheneaut, N. (2006). A methodological framework for socio-cognitive analyses of collaborative design of open source software. *Computer Supported Cooperative Work (CSCW)*, *15*(2-3), 229–250. https://doi.org/10.1007/s10606-006-9020-5

Saldaña, J. (2009). *The coding manual for qualitative researchers*. London: Sage Publications.

Schenk, E. (2009). *Crowdsourcing: What can be outsourced to the crowd, and why*? 29.

Sclater, N. (2008). Largescale open source ELearning systems at Open University UK. *Educase*, *1*(12). Retrieved from http://www.educause.edu/ir/library/pdf/ecar_so/erb/ERB0812.pdf

Seidman, I. (2006). *Interviewing as qualitative research: A guide for researchers in education and the social sciences*. New York, NY: Teachers College Press.

Shaikh, M. (2016). Negotiating open source software adoption in the UK public sector. *Government Information Quarterly*, *33*(1), 115–132.

Shirky, C. (2008). *Here comes everybody: The power of organizing without organizations*. London: Penguin.

Von Krogh, G., Haefliger, S., Spaeth, S., & Wallin, M. W. (2012). Carrots and rainbows: Motivation and social practice in open source software development. *MIS Quarterly*, *36*(2), 649–676.

Von Krogh, G., Spaeth, S., & Lakhani, K. R. (2003). Community, joining, and specialization in open source software innovation: A case study. *Research Policy*, *32*(7), 1217–1241.

Waterman, A. S., Schwartz, S. J., & Conti, R. (2008). The implications of two conceptions of happiness (hedonic enjoyment and eudaimonia) for the understanding of intrinsic motivation. *Journal of Happiness Studies*, *9*(1), 41–79.

Wenger, E. (1999). *Communities of practice: Learning, meaning, and identity*. New York, NY: Cambridge University Press.

Williams van Rooij, S. (2011). Higher education sub-cultures and open source adoption. *Computers & Education*, *57*(1), 1171–1183.

Yin, R. K. (2009). *Case study research: Design and methods* (4th ed.). London: Sage Publications.

Zimmermann, T., Premraj, R., Bettenburg, N., Just, S., Schroter, A., & Weiss, C. (2010). What makes a good bug report? *IEEE Transactions On Software Engineering*, *36*(5), 618–643.