Participatory Practices in Open Source Educational Software: The Case of the Moodle Bug Tracker Community

A thesis submitted for the award of Doctor of Education

October 2013

Eamon Costello

School of Education

University of Dublin, Trinity College

Declaration

I, Eamon Costello

declare that this thesis has not been submitted as an exercise for a degree at this or any other university; is entirely my own work, where appropriate the unpublished and/or published work of others is duly acknowledged in the text; and agree that the Library may lend or copy the thesis upon request. This permission covers only single copies made for study purposes, subject to normal conditions of acknowledgement.

Signed Date

________________________ __________________

Acknowledgements and Dedication

I would like to thank my supervisors Mr. Keith Johnson and Professor Vincent Wade for their dedication and patience, their willingness to share their expertise and experience and most of all for encouraging and motivating me by their example. The support of my wife Deirdre must be acknowledged as she could explain to the children on many occasions that even though Dad was not in College he was nonetheless working. I would also like to thank and acknowledge the support of my colleagues in Dublin City University, in particular Lorraine Delaney, Seamus Fox, Mark Glynn, Richard O Kennedy and Jean Hughes without whom I would not have been afforded this opportunity. This study would not of course have been possible without the time and generosity of spirit of the participants themselves and I hope that I have been fair and just in my appraisals.

In a time of deep economic turmoil in Ireland and across the world I would like to dedicate this thesis to all those looking for work, in particular those educating and up-skilling themselves in the hopes of a better future, and to the educators entrusted in helping them.

Abstract

The Moodle bug tracker is a boundary object that faces software developers who write and maintain Moodle's code whilst simultaneously exposing an interface to a much wider public world of ordinary Moodle users. Bugs can be fixed and new features requested by recording them in this boundary object which then tracks their progress. Such tracking has proven a powerful lure for researchers and despite much study of the phenomenon of open source bug fixing and software building, much remains to be answered. Specifically this research sought to analyse the implications of this massively distributed collaborative development process for education and educational technology (which to give it due importance, is referred to here as educational infrastructure). It examined the ways educators - who are defined inclusively as all those involved in supporting the educational enterprise - interface and contribute to the development of the Virtual Learning Environment (VLE) Moodle at this granular level of bug fixing.

Two things happen in a successful bug resolution: it is reported and then it is fixed. Only one population is skilled and empowered to engage in the latter but, in theory, anyone can be a reporter. Here data was collected and analysed about these two types of participant. Firstly archival and statistical analysis of thousands of issues contained in the tracker database itself was undertaken. These canonical accounts of bug-fixing contributed to the design and conduct of interviews of both core participants of this community and more casual or peripheral members. A broad spectrum of community participants were interviewed from fringe and casual members to some of the key actors including Tim Hunt of the Open University, Moodle HQ members Helen Foster and Michael du Raadt and Moodle founder Martin Dougiamas. Ethnographically inspired methods were utilised in the interview analysis to uncover rich stories of actual practice that were absent from the accounts of the database itself. This lead to several contributions to research being made by this thesis: a depiction of the dynamics and characteristics of an open source software community of practice dedicated to the enterprise of education; an enumeration of three complexes of factors leading to bug tracker issue resolution elicited from participants themselves; an account of particular unknown or under-reported non-canonical issue resolution factors; a model of the role of bug tracker issue mediators including the novel brokerage act of proxy issue submission and improvement; and a theory of how bug trackers are resistant to predictive models of issue resolution. These findings have implications for educational institutions reliant on VLEs and for developers of open source educational software.

Table of Contents

List of Appendices

List of Figures

List of Tables

List of Abbreviations

CIC	Collaborative Innovation with Customers
CVS	Content Versioning System
DCU	Dublin City University
Edu	Education
GPL	General Public License
ICT	Information Communications Technology
IT	Information Technology
KM	Knowledge Management
Moodle	Modular Object OrientateD Learning Environment
Moodle HQ	Moodle Head Quarters (Moodle Pty Ltd)
MS Word	Microsoft Word
NA	Not Applicable
NFQ	National Framework for Qualifications
OER	Open Educational Resources
OS	Operating System
OSI	Open Software Initiative
VLE	Virtual Learning Environment
OU	Open University (United Kingdom)
PHP	PHP Hypertext Pre-processor
SAT	Scholastic Aptitude Test
SCORM	Shareable Content Object Reference Model
STEM	Science, Technology, Engineering or Mathematics

YUI – Yahoo! User Interface (Library)

1 Introduction

1.1 Background

In January 2009 a lecturing colleague in Dublin City University (DCU) contacted me and explained a problem she was experiencing with Moodle. For some reason the tests she had set her accounting students, which they were to take using Moodle’s Quiz feature, were not working. Incorrect student answers to the multiple choice questions were not being negatively marked as she wanted but instead were being rounded up to zero. This, I discovered, was due to no fault on her part in setting up the test but rather a problem with the Moodle Quiz itself. This part of Moodle was broken. We had discovered a bug in the software.

Others had also encountered this problem and were discussing the bug online. They did so in the Moode.org community discussion forums, the first or outermost online layer of the vast online Moodle community. In these forums educators who were experiencing this bug voiced their frustration but also shared tips on how to overcome the problem. One workaround involved applying a small piece of code, or *patch* to Moodle. As its name implies, a patch does not solve the problem in the fullest sense. Moodle remained broken but using this code one could mend or patch up one’s own local copy of the software. Negative marking in the Quiz could be made to function as it should. Applying this patch to DCU’s own copy of Moodle was not an option for us however, as the University operated a strict policy of not allowing modifications, or local customisations, to its Moodle installation. Tinkering with a critical piece of the institutional infrastructure, even to fix a problem, was not an option.

What was required was for the original copy of Moodle, upon which all local copies are based, to be fixed. Beneath the Moodle discussion forums there existed a place where such a request could be made. This inner layer is known as the Moodle bug tracker, a database where users can report bugs or issues they have encountered with Moodle. Here, more so than in the discussion forums, a Moodle user affected by a bug, can interface directly with the builders of Moodle, with those who are tasked or assigned with the software’s development and maintenance. In this way I could cast my vote in favour of the fixing of the issue of the negative marking in the Moodle Quiz.

Others concerned with this bug had also found their way to the tracker. There they too had voted for it to be fixed, given details of the problem, and proffered the patch that claimed to fix it. An important person who contributed to this discussion in the tracker was Tim Hunt of the UK's Open University (OU). He was the maintainer of the Quiz component in Moodle, the developer, guardian and gatekeeper of the particular part of Moodle that allows teachers to create, among other things, online tests consisting of multiple choice questions. Casting a critical eye over the patch Tim was quick to point out that the proposed solution would not work in all cases and that a better way would need to be found. So, for the time being, Moodle remained broken.

I was intrigued to see how Moodle laid aspects of its development open through the tracker and also by the complexity that it seemed was hindering the fixing of what appeared a fairly serious flaw in this part of the software. The possibility of modifying the code of Moodle, of being party to its development was clearly there, at least in theory. Indeed, it was a prospect that had been a factor in DCU becoming one of the earliest adopters of Moodle in 2003 when an evaluation committee cited its open source nature as something they hoped would allow the University "unrestricted technical access to [develop the software], whether within our own local resources, via collaboration with other users, or by contracting out to commercial support companies" (McMullin & Munro, 2004, p. 1). There may have been good reasons why, by 2009, DCU was operating a policy of not modifying its local version of Moodle but, notwithstanding these, it was clearly against the spirit of what the University had initially hoped for. Moreover, even if an institution could not adapt and customise software to its own ends, the issue of the broken Quiz appeared to diminish the prospect that a wider global community of distributed users could affect change. Despite the community reporting the issue to the Moodle developers, giving multiple votes for its resolution and the proposal of a fix, the issue had nonetheless languished for years unfixed.

Of course this was just one issue and from my narrow personal perspective, as it has been outlined so far. Broader questions pressed to be answered such as: whether this issue was characteristic of issues *as a whole* or what factors might determine whether issues *in general* might be fixed. Clearly there was a social process at work as much as an engineering one in how issues were resolved. The Moodle bug tracker appeared a frontier between two worlds of teacher and software developer. Discovering the workings of this realm, through the prism of fixing or resolving issues, became the research imperative that motivated the study at hand.

Although, it was inspired by personal circumstances as described, the research problem is also an important one with wider significance. The work of the developers of Moodle has a global impact. A development in Moodle, even a small one, such as the bug in the Moodle

Quiz, can potentially affect over 70 million users worldwide (Moodle.org, 2013c). The imperative became to uncover some of the workings of a social entity, the Moodle bug tracker, lying in open view yet relatively unknown to educational research, at a frontier between educational and technological worlds. What these communities are, how they interact at this boundary, and how they work over time to attempt to build shared knowledge is the focus of this study.

Moodle is an important artefact that has emerged as both a product and a driver of the move to online education and the development and deployment of ICTs to this end. Divergent perspectives have been taken on the rise of educational technology in third level education. For some, such as Noble (1989), this move has represented a commodification of education, the implementation of an industrial model at the expense of the individual teacher. In the context of the VLE this debate was aptly captured by Dron (2006) in the title of his paper on the topic –*Any color you like as long as its blackboard* – which posited cottage industry versus industrial models of education arguing that the VLE can, if left unchecked, allow the technological form to dictate pedagogical patterns in subtle ways in a type of "educational Fordism". Noble's (1989) alarm stems in part from a perception that the quality of education was being diminished as it moved online. Christensen et al (2008) however, precisely warned that incumbents in a market, treating higher education as such, would disparage new entrants and more efficient forms (such as online learning) but that these forms would evolve and adapt before becoming eventual dominant new forces in the overall market. In this way they framed online education as a radically disruptive innovation.

That these new educational ICTs could disrupt education for the better (ultimately by lowering or removing cost barriers) was taken up in several quarters, one prominent example being the Open Educational Resources (OER) movement. Drawing philosophically on the open education movement, and the traditions of distance and open education, the OER movement advocated for the reuse and repurposing of digital educational resources with the ultimate aim of broadening access to educational opportunities (D'Antoni, S., 2009). Repurposing and reusing content also had analogues with the open source software movement which we will later examine.

Moodle began in Curtain University in Australia as an offshoot of an (as yet unfinished) PhD thesis by Martin Dougiamas (Dougiamas & Taylor, 2002, 2003; Dougiamas, 2007). Dougiamas experienced frustration with the inability to adapt WebCT (because of the restrictive nature of the software license) to his own ends of exploring interactive and socially constructivist forms of online learning. Instead, he started writing his own alternative learning environment that would come to be known as Moodle and instead of retaining strict ownership of the code he decided to make it freely available. Moodle represented for many a welcome confluence of the ideals of open education and open source. For example Moodle claimed to be based on a “social constructivist” pedagogy which it is argued elsewhere positively affected Moodle’s adoption (Costello, 2014) and was directly cited by some adopters (McMullin & Munro, 2004). There are several other factors that proved important as Moodle rose from relatively obscurity in 2001 (Dougiamas & Taylor, 2003) to holding a dominant share of the open source VLE market by 2013 (Costello, E., 2014). Moodle adopters also cited cost (due to its open source licensing model), fear of vendor lock-in to proprietary licensed competitors such as Blackboard and the promise of adapting the software to their own needs. A detailed analysis of 28 higher educational institutions self-professed rationale for their adoption of Moodle is made by the author in an article dedicated to the topic (Costello, 2014). Interesting network effects proved to be at play as the effective market and choice of VLEs began to shrink between 2003 and 2011 (Bennett, 2011; Browne et al., 2006; Hawkins & Rudy, 2008). The Open University of the UK’s adoption of Moodle and its commitment to its development proved a massive boon to its credibility and paved the way for more institutions to follow suit (Sclater, 2008).

Moodle is currently the dominant open source VLE in higher education. Moreover the VLE itself has become increasingly important to the actual identity of a university - such that, it has been argued, having a VLE may be as important as having a library (Costello, 2014, Williams van Rooij, 2011). This study it is hoped will provide some more understanding of something that has become critically important to the enterprise of higher education.

1.2 An Overview of the Research

1.2.1 Research Approach

This study is qualitative in nature. Although it draws on numerical data and proceeds in an analytical fashion, it nonetheless does not claim to build objective truths about independent phenomena. Rather, a social constructionist position is taken where knowledge comes from the collective because “humans create and change the world around them through action and interaction” (Corbin & Strauss, 2008, p. 6). Both the construction of Moodle and the conduct of this research are conceived of as intrinsically social activities, their outputs the product of confluences of social influence. As such a researcher is interpreting events; they become a co-constructionist as they deconstruct and then reassemble the phenomena of study to come to some final thesis. Such interpretations "cannot be separated from their own background, history context and understanding" (Creswell, 2012, p. 39). Hence the background section above which outlined (in the first person) the path of the researcher up to the outset of the study and in so doing introduced some of the influences that colour its perspective.

Communities of practice provided an important theoretical touchstone to the design of this study. The theory, as espoused by various proponents, perhaps most notably Wenger (1999), holds powerful conceptions for analysing social groups, in particular for groups not traditionally defined as organisations in legal or monetary senses as is the case with the Moodle community. Rather, actors within these communities are defined by the *practices they engage in* (such as bug tracker issue resolution) and have identities and roles within the group that come to define and shape those practices.

1.2.2 Research Questions

Following from the constructivist/interpretative research perspective, and from the nature of the phenomenon of study which was deemed unique and underexplored, a study of an exploratory and elucidative nature was designed (Yin, 2009). The study is defined by the following question:

What are the key practices of participation in open source educational software development, taking as a case study the Moodle bug tracker community?

Four sub-questions were designed to address the main research problem:

1. *What are the characteristics of participants in the Moodle bug tracker and the issues they engage in?*
2. *What factors and related processes are important in the resolution of issues in Moodle?*
3. *What are the key identities and roles of the participants?*
4. *How do educators come to participate in the inner community of Moodle development?*

Question one sought to examine the overall or aggregate appearance of tracker issues and their participants. The second question sought to analyse, from the perspective of participants, the community practice of issue resolution; to analyse what participants believed to be important success factors as they engaged in this activity of attempting to resolve issues. Question three addressed the issue of identity and its interplay with defined roles within the community. Question four sought to build on the previous three questions and tie them back together to address the overarching research aim: that of how educators (in a broad sense of the term that extends beyond simply those in the classroom) come to participate and how this affects Moodle development. That is, this question was concerned with momentums – of how people enter the community or change role within in it in the pursuit of changing Moodle itself.

1.2.3 Research Design and Methods

A case study may distinguish itself from a more purely ethnographic study by its situatedness, by its bounding of the object to be studied as a specific event or activity rather than a whole culture (Stake, 1978). Here the activity of mutual engagement in Moodle bug tracker issues, in the joint enterprise of developing Moodle and the resultant shared repertoire of communal resources that emerged, is the specific social practice under study (Wenger, 1998). A temporal bound to the case is defined by confining it to Moodle bug tracker issues that were started between January 2007 (Moodle version 1.7.1) and February 2011 (Moodle 2.0.2). Community members were included in the study if they participated at least once in the tracker during that time. The single-case study method of bounding an object of study helped demarcate the research phenomenon. The case study was also suitable as this study sought: to focus on contemporary events over which the researcher had limited control; and was an area under explored and reported to date, leading to the requirement of an elucidative and exploratory approach to the phenomenon (Yin, 2009).

A case study is also characterised by its use of multiple data types or sources (Yin, 2009). The Moodle bug tracker is itself a vast data store and hence was the inevitable starting point for collecting evidence for this study. Descriptive statistics were generated using software by analysing reports from the Moodle bug tracker. The researcher also spent time reading individual issues and three issues were selected and presented to demonstrate the unique stories that exist within the multitude. A picture was also built of the participants of the tracker community: of those who had the ability, as a result of long-standing membership, to be assigned to fix issues; and those members by contrast who reported issues and lobbied for their resolution.

A lack of participant voice was identified in the review of bug tracker literature and hence interviews – which for Yin (2009) are often the most important data source in a case study – were planned where “key respondents would be asked about the facts of a matter as well as about their opinion of events” (Yin, 2009, p. 90). The bug tracker participant analysis informed the selection of interviewees. Interviews were subsequently conducted according to a semi-structured interview schedule.

1.2.4 The Research Process

A case study research approach was undertaken with data comprising the bug tracker itself and interviews with participants. The research process proceeded iteratively as activities of literature review, data sampling and research design were conducted to various degrees in parallel, subscribing to the maxim that “analysis begins with the collection of the very first pieces of data” (Corbin & Strauss, 2008, p. 47) or, in Merriam’s (1985, p. 207) conception of the tight connection between data collection and analysis in case study research, that “checking, verifying, testing, probing, and confirming collected data as you go” is key. In April 2010 four prospective (and eventual) interviewees were identified following informal conversations with them at a Moodle conference in London and a tentative outline of the interview schedule was formed. In February 2011 systematic collection of data from the bug tracker started, and was compiled and analysed using spreadsheets and the statistical package R. The interview design was informed by this data analysis from the tracker in addition to the pilot stage of discussions with participants at the 2010 Moodle UK conference. Between January and April 2012 nineteen of the interviews were conducted with the final twentieth interview concluded in early February 2013. Transcription and analysis of the interviews and tracker data was deemed largely completed by this stage (though some level of analysis continued during all stages of the thesis write-up).

1.3 Outline of the Thesis

1.3.1 Chapter Two: Review of the Literature

This chapter is divided into two sections. The first section explores the communities of practice theory as a social model of human interaction and organisation. Key themes are drawn out such as how communities interact with each other; how participants enter or are apprenticed to a group; how a group negotiates meaning and potentially produces new knowledge. Two important aspects – *identities* of community members, as defined by their on-going practice, and *boundary objects* which interface between communities – become relevant to the second section of the chapter which deals with situated studies of communities of open source software such as Moodle. This second section of the literature review looks at the boundary object of the software bug tracker. It examines how researchers have conceptualised it and attempted to answer research questions such as why bugs or issues in trackers may or may not be resolved, the joining trajectories of people into such communities and how participants seek to affect change within the community.

1.3.2 Chapter Three: Methodology

The adoption of a single-case study was a consequence of the formulation of the research problem: one that looks at a particular group (the bug tracker community) engaged in a particular avowed practice (bug fixing). The case study is an appropriate form when the researcher is examining contemporary events over which he/she has little control. The types of questions used to tackle the research problem were informed by the perspective of the researcher and have philosophical underpinnings in constructivism. They sought to elicit the view of the participants. The questions focused on bug tracker communities, bug tracker issue resolution, participant identities and processes of joining these communities. These question topics were formulated based on relevant, related studies from the literature review. This development of the research questions and the adoption of the case study approach is outlined in Chapter Three.

The data from this study comprised the bug tracker database itself and participant interviews. Most of the interviews were conducted via Skype, directly recorded and then transcribed. They were then coded using established tools and methods from qualitative research. The issues that contributed to the adoption and design of the particular chosen methods of interviewing and coding are also discussed here as research design decisions are made clear to the reader. Similarly, an examination is given of ethical issues that pertained to the study, and how these were addressed. Finally, as a fitting conclusion to the chapter, the research's overall reliability is probed and then argued for.

1.3.3 Chapter Four: Findings

The research problem is decomposed into four sub-questions and the findings of this study are presented under the headings of these questions. The first of these questions examines the characteristics of the Moodle community in general terms i.e. in the aggregate using descriptive statistics. The first sub-section of the Findings chapter, which addresses this first research question, also maps to the first data source – the bug tracker database – and hence largely matches the chronology of the research process (notwithstanding inherent degrees of iterative parallelism). The subsequent three sections are based on the next three research questions that were primarily addressed through the interview analysis. Research question two examined the *factors* and *processes* of *issue resolution*. Its findings are presented according to three themes that emerged in the analysis; these focus on the perspectives of the bug tracker *issue submitter*, the bug tracker *assignee* (potential fixer) and the *submitter who has written a fix* to an issue. The next section addresses the *roles* and *identities* of the participants; how teachers, software developers and brokers who mediate between the two, are motivated to act within the community. Lastly the concepts uncovered during the previous sections are gathered and tied back to the main thesis statement, examining how participants come to be involved in the community and whether such involvement can affect change to Moodle's code.

1.3.4 Chapter Five: Discussion

Chapter Five follows the same outline as the Findings chapter and is divided into sections that broadly map to each of the four questions. These are not necessarily of equal weighting as some findings were found to be more significant than others. Their significance was determined by casting them in the light of the wider research literature and by drawing upon the concepts from the literature review. Firstly the aspects of the Moodle tracker that give it and its inhabitants a unique character and culture are argued for. It is also found to have many characteristics of comparable communities such as the relative influence of assignees and the importance of certain tasks such as commenting on an issue to get it fixed. The interesting factors that contribute to bug tracker issue resolution are discussed including ones that are not widely reported or represented in the literature. The approach taken uses theoretical constructs from the first section of the literature review, such as the interplay of canonical and non-canonical accounts of how work is carried out, to uncover aspects of the community practice. Unique identities (such as those of teaching) and roles (such as mediators) that are characteristic of this community are posited and examination is made of their relationship to roles and identities (such as those of software developers) that are by contrast well established in the literature.

1.3.5 Chapter Six: Conclusion

The final chapter provides a summary of the main findings and discusses possible further implications. There are two main types of implication that flow from what is presented here. The first relates to future research. This study, it is argued, has helped answer questions about the nature of the boundary object that is the Moodle bug tracker, sitting as it does between the worlds of educators and software developers. However, it has also raised interesting possibilities for future research. Part of the aim of this research was to help tell the story of this particular community, explore and describe its character through a case study in the hope of informing a research agenda for open educational software development. There is necessarily more work now to be done, to test whether, for example, the role of brokers found here is characteristic of similar communities and whether it is related to a community's maturity or is instead unique to particular ones. Brokerage as a concept is commended to open source tracker communities. They may recognise it instinctively as has been shown here but the potential role of mediators may be undervalued in other comparable communities. Lastly suggestions are made here for educational institutions. It is argued that they should engage in helping develop entities such as Moodle for the sake of education generally as it heads into an uncertain technological future.

2 Review of the Literature

2.1 Introduction

This literature review is structured in two main parts. The first deals with communities of practice from a theoretical perceptive. This leads to a second part that deals with specific studies that are closely related to the domain of the study at hand i.e. open source software communities. In this way the scope of discussion is progressively narrowed so that the context for the research question of this study, as discussed in the succeeding methodology chapter, becomes clear.

The first section, on communities of practice, starts by looking at theoretical underpinnings. This mirrors the research process itself (at least as construed in a post-facto form) where an existing theory becomes the basis or foundation for a new enquiry. In this study a core group of people involved with developing Moodle will be considered theoretically as a community of practice. In doing this we will ask who these people are, what is it that they do and how they engage in their work. To do this we must first explore the communities of practice field and then drill down into particular researchers and particular concepts in this area that will be of use to us. Although aspects of their methodological toolsets will be explored, our review of communities of practice in the first section of this literature review will be from a broadly theoretical perspective, as it contributes strongly to the theoretical basis of this study.

To bring the research back down to situated studies, a review of open source communities will follow. This second part introduces the concept of open source and its place in research. It then analyses specific research efforts that have been undertaken on open source communities, considering their concerns, their methodological outlook, relevant findings but also any lacunae, any angles which have not been fully explored or areas that are lying in wait of the researcher (such as the intersection of education and open source).

2.2 Communities of Practice

2.2.1 Introduction to Communities of Practice

The aim of this study is to examine the community that builds Moodle and particular work practices that its members engage in during their efforts. Specifically, we will look at how they engage in solving problems that arise in Moodle's software. This is a collective activity and to explore more theoretically how this is underpinned we must take a look fundamentally at communities themselves. The question we will now ask is how researchers have conceptualised community activity in education and knowledge work literatures and what ultimately this might tell us about what to expect of the Moodle community. In doing so we will draw out key themes of the Wengerian community of practice model such as how communities interact with each other; how participants enter or are apprenticed to a group; and, once a group is formed, how members negotiate meaning and potentially spur innovation.

2.2.2 Apprenticeship

Communities of practice are based on notions of disequilibrium within groups. Essentially power relations are at play and some members must weigh less than others. Resnick (1987) in her American Educational Research Association Presidential address, invoked the concept of *apprenticeship*. She called for "bridging apprenticeships" that would greater connect theoretical learning and actual practice. At around this time, anthropologists returning from the field were finding that failures in education could be attributed to the misapplication of didactic modes of teaching in situations where learning in the apprenticeship mode was more appropriate or more culturally customary (Jordan, 1989). Thus the notion of apprenticeship, as something applicable to almost all forms of learning, began to take hold in educational research, particularly after Lave and Wenger (1991). For many this was a chance to reframe learning situations and argue against a more didactic model. As such, it has a heritage in the educational tradition of such thinkers as Dewey, Vygotsky and Freire. Similar to Rogers' rechristening of *teachers* as *facilitators* (Rogers et al., 1965), a language of *apprenticeship* and *practice* was instituted to revive and renew constructivist principles. Before long commentators could claim that Lave and Wenger had become "a very influential corrective to previous educational practice" (Cox, 2005, p. 3).

Lave and Wenger's (1991) original communities of practice model was based on their critique of five individual studies of the apprenticeships of midwives, tailors, quartermasters, butchers and recovering alcoholics. In their meta-analysis of these apprenticeship studies, the importance of social interaction, social identity and negotiated admission to a cultural heritage emerged as strong themes. Failure to integrate could occur if interactions were not rich enough or systemic structures caused sharing to be obstructed. For instance in the case of the butchers, apprentices were physically segregated from masters for crucial tasks, which led to a greater likelihood of their failure to fully apprentice. On the other hand, in optimal scenarios, relatively complex mental abilities, such as mathematics, could be acquired under apprenticeship that were found to be on a par with those obtained through formal learning, as in the case of the tailors. It was this later study where Lave (1977, p. 177) found evidence that "apprenticeship training does teach general problem solving skills". The tailor and midwife studies occurred in the developing world. The butcher and quartermaster studies by contrast were of groups that were embedded within larger organisations in modern developed economies. The recovering alcoholics were not partaking in an apprenticeship in the truest sense, however the study was admitted for demonstrating many of the same characteristics as the other studies and Lave (1988) went on to study a diversity of groupings (such as dieters). Another influential ethnographic study that was to contribute to the community of practice literature, but not cited by Wenger and Lave (1991) at this point, was that of Orr's PhD work which examined a community of photocopier repair technicians, and was later published as a book (Orr, 1996). Finally, Wenger himself did most of his most important early fieldwork on a group of claims processers in an insurance company. Considering these latter studies, it is clear that the notion of apprenticeship is a loose one as it relates to communities of practice. As Cox (2005) has it, only a naive reading of the early communities of practice literature would see it as a call to the return of apprenticeship styles of learning in the modern world. However, it still serves as an important pointer in this literature to a valuable theme; that of more incarnate representations of social activity.

One of the key concepts of Lave and Wenger, appropriated from apprenticeship, is that of *Legitimate Peripheral Participation.* This was their demonstration that people could be successfully inducted to a group by simply hanging around at its fringes. Just sitting and observing the norms and language of the group was a necessary first step to participation, as something of the culture of the group must be recognised and learned before a newcomer can input and begin to become a member. In acquiring the norms, viewpoints and behaviours of the group, they become enculturated (Brown et al., 1989). Practices could not be disentangled or abstracted from the cultures in which they were enacted. Assimilation and adherence to cultural norms was a necessary prerequisite to learning (or being allowed to learn) the more explicit and formal practices of the group. There is also an element of making prospective members “wait their turn”. As one member of an open source software group puts it, “If you aren’t willing to do a little research, observe how the project functions, and figure out how to make your mark on it, do you really belong on the team?” (Krishnamurthy, 2005, p. 30).

Wenger has given perhaps the most complete subsequent theoretical model of the community of practice. His focus diverged somewhat from Lave and he concentrated more on applications relevant to industrial workplaces than to formal educational settings and in the main this is the strand of research most relevant to us here. But before looking at his theory it is worth also considering the contribution of Brown, Collins and Duguid (1991) because of its relevance and impact on educational research in general and also of its strong focus on the Orr study of photocopier technicians which is, because it focused on technicians involved in fixing things, pertinent to the research at hand.

2.2.3 Story and Practice Canons

Brown and Collins (1991) refined the community of practice model by building on the idea of *situated cognition,* that called attention back to the site of learning and of activity (Brown et al., 1989). This was arrived at by observing the common disconnect between classroom activities and their applied corollaries in the real world. They argued that school often acculturates students more to itself than to its teaching disciplines such that “what students do tends to be ersatz activity” (Brown et al., 1989, p. 38). They found an over-reliance on the explication of instruction and of activity in the classroom to the detriment of actual learning. Later, they widened their focus from formal education to organisational theory in general to also look at (ostensibly) non-educational workplaces. The disconnect between the *de jure* and the *de facto* activities is key, and led them to the claim that:

> …reliance on espoused practice (which we refer to as canonical practice) can blind an organization's core to the actual, and usually valuable practices of its members (including noncanonical practices, such as “work arounds”). It is the actual practices, however, that determine the success or failure of organizations (Brown & Duguid, 1991, p. 41).

Because of the problem caused by official work specifications, used for training and formal learning, being out of kilter with actual effective practices, a *learning-in-working* conception was proposed by Brown and Collins (1991) in an attempt to tangle doing and knowing and keep the idea up front that learning should always be situated and embodied rather than abstracted. From this standpoint learning could be seen as “the bridge between working and innovating” (Brown & Duguid, 1991).

"Actual practice inevitably involves tricky interpolations between abstract accounts and situated demands" (Brown & Duguid, 1991, p. 42) and the limited abstract accounts can be seen, to use terminology from Geertz (1973), as *thinly* described. So for example thinly described actions are given in Orr's (1990) corporate training manuals, training courses and job descriptions of the photocopy repair technicians. However, they also had unsanctioned modus operandi based on their communal experience and in-the-moment action. We can conceive thin and thick in a qualitative rather than quantitative sense: the thin formal descriptions may come in a very thick manual indeed, and likewise an effective real-world stratagem may be a simple one.

The exchange of work related anecdotes is also highlighted as an important component of problem solving in a community of practice. Orr had described in detail how the technicians exchanged "war stories" when solving a particularly intractable problem (Orr, 1990). Storytelling can never be captured in canonical accounts, though it is a vital sensemaking activity (Weick, 1995). Its role in problem identification and resolution to the photocopier repair technicians is attested by Orr: "The use of story-telling both to preserve knowledge and to consider it in subsequent diagnoses coincides with the narrative character of diagnosis" (Orr, 1990, p. 188).

There is some reflexivity going on here, as many if not most of these primary studies were carried out by researchers who privileged narrativity in their own methodology (especially Orr and Wenger for example). There is a primacy of narrative both in the subject of study and its explication: "The intent of Talking about Machines is descriptive: to present the work of the technicians and to use it to suggest what may be learned from studies of work practice in contrast to more abstracted ways of writing about work" (Orr, 2006, p. 1816).

For Brown et al. (Brown et al., 1989), the coalface, where the customers, technicians and technical experts of Orr's study interacted to solve problems, could be an exciting place characterised by *innovation*. They concluded that "through their constant adapting to changing membership and changing circumstances, evolving communities-of-practice are significant sites of innovating" (Brown et al., 1989, p. 41). Here they added not only learning, but also knowledge creation, termed innovation, to the functions of communities of practice. Innovation is a key strand in the community of practice research literature, as we will examine later.

2.2.4 Wenger's Communities of Practice Model

For Wenger communities of practice abound and we are inevitably members of several whether through work, school, home or hobbies. However, he is clear in distinguishing that communities of practice are not simply based upon geographical location, nor even on shared interest. Rather the group must be *actively engaged* in a work practice and must further exhibit a social manifestation that is defined by three particular characteristics:

> **What it is about**—its *joint enterprise* as understood and continually renegotiated by its members
>
> • **How it functions**—the relationships of *mutual engagement* that bind members together into a social entity
>
> • **What capability it has produced**—the *shared repertoire* of communal resources (routines, sensibilities, artefacts, vocabulary, styles, etc.) that members have developed over time (Wenger, 1998, p. 2).

Communities of practice are groupings within wider organisations and a constellation of such communities may exist according to the exclusive and overlapping activities in which people may be engaged. Wenger attempts to distinguish a community of practice from formal teams, project groups or networks, because of its emphasis on doing, its definition by knowledge and its identity through practice respectively. It also has a life that does not map conveniently to officially sanctioned organisational structures. Its life cycle moves from genesis and coalescence through a phase of activity and then dispersion before finally staying alive in individual and collective memories. The "informal fabric" of communities within an organisation add value, do real work and even allow an organisation to function (Wenger, 1999). Nonetheless, they may not have formal organisational approval and as such may exist upon a continuum of such communities from the completely unrecognised, which consequently have limited impact, to official and heralded entities with transformative power.

The status of a community of practice, and the degree of its legitimacy within its parent organisation, led Wenger to look at how they might be fostered and developed. To this end, organisations should be open to recognising more aspects of organisational life and in so doing *legitimise participation* in effective communities of practice. They should also work to calibrate their strategy with practices and above all be "attuned to real practices" (Wenger, 1998, p. 2). This became the dominant theme of subsequent work by Wenger. Speaking to an audience of managers based in mostly large corporate organisations, he further elaborated how communities of practice might be recognised and grown in the service of the wider organisation (Wenger et al., 2002). Although this was raising interesting research ideas for others (Kimble et al., 2001; Hildreth & Kimble, 2002; Von Krogh et al., 2003; West &

Lakhani, 2008) there were also concerns that, in the worst possible scenario, it might hide other forms of normative and networked control by organisations (Cox, 2005). Detractors might be consoled that although the promise of implementing communities of practice within organisations is great, their realisation could well be trickier and we are warned by Wenger (1998) that this may represent something of an art form as opposed to a science:

> Communities of practice do not usually require heavy institutional infrastructures, but their members do need time and space to collaborate. They do not require much management, but they can use leadership. They self-organize, but they flourish when their learning fits with their organizational environment. The art is to help such communities find resources and connections without overwhelming them with organizational meddling. This need for balance reflects the following paradox: No community can fully design the learning of another; but conversely no community can fully design its own learning (Wenger, 1998, p. 2).

As also illustrated in the above example Wenger makes extensive use of dualities, paradoxes and contradictions in his writing. Sometimes, to the frustration even of his proponents:

> …this book can be frustrating to read and understand. It is at times an excruciatingly difficult read, because of the different way that Wenger looks at and defines his underlying concepts. However, the book is worth every penny (Gillespie, 2000, p. 96).

Although this style may give rise to the criticism of making his theory less applicable, rigorous, or consistent, it is also one of its key strengths as it is arguable that it has allowed the communities of practice paradigm to be amenable to *reframing.* A *frame* is a term used to represent a social phenomenon and, for Benford and Snow (2000), the *elasticity* of a frame may be a factor in its success or usefulness; communities of practice has proved to have this property i.e. it has been reused and remixed in a diversity of contexts by subsequent researchers.

A second important point should be made about ostensible contradictions in Wenger's communities of practice model. This is that the Eastern idea of *duality*, which allows conflicting ideas to coexist without resolution, can allow new conceptions to be admitted (Hildreth & Kimble, 2002). Or, where there is resolution it "does not mean consensus [but] rather, representations, or inscriptions, contain at every stage the traces of multiple viewpoints, translations and incomplete battles" (Star & Griesemer, 1989, p. 413). It is often in the areas of tension that creativity comes about and novelty may arise. Wenger is given to looking in these areas and identifies four important dualities at the core of the communities of practice concept (Wenger, 1999) and two of these dualities are particularly important to this thesis.

2.2.5 Four Dualities

2.2.5.1 Designed/Emergent

One duality, as we have seen, involves community construction: are they *designed* or *emergent*? These are two important sources of organisational structure. Design may be a stricture upon emergence, but much that is useful only emerges during interaction that has been pre-planned i.e. designed. This idea has been pursued by researchers interested in the design of online learning environments (Barab et al., 2004).

2.2.5.2 Identification/Negotiability

Design is a proposal of identity and this creation of identity or *identification*, for Wenger (1999), is an expression of power because it attempts to nail down meaning which is counter to negotiation. Identification is totemic. Members may feel they most strongly belong to communities where their sense of identity is highest, where they have negotiated the most meaning. Over-identification can be destructive, leaving participants unable to negotiate either amongst themselves (civil war) or with external communities (cult-like behaviour). For learning to happen some identity must be given up as a novelty is negotiated. The dual processes of negotiability and identification thus form a basis for looking at learning, identity, and power in social terms.

2.2.5.3 Local/Global

A third dialectic is that a community member's identity always involves interplay between *local* and *global* events. People "come together, not only to engage in pursuing some enterprise, but also to figure out how [their] engagement fits in the broader scheme of things" (Wenger, 1999, p. 162). In the knowledge management literature this has some analogues in the relation of *buzz* (face to face and co-located activities) to *pipelines*, which are more formal external connections that are spatially distributed, such as alliances with outside groups including possibly rivals (Bathelt et al., 2004).

Interaction amongst different communities is a very important issue, as these occurrences are rarely straightforward and they have been the subject of much theorising. One of the ways in which two communities interact is via *boundary objects*. Boundary objects act as abstractions that exist in two worlds, can be adaptable to viewpoints of each, but are robust enough to maintain identity across both:

> In conducting collective work, people coming together from different social worlds frequently have the experience of addressing an object that has a different meaning for each of them. Each social world has partial jurisdiction over the resources represented by that object, and mismatches caused by the overlap become problems for negotiation (Star & Griesemer, 1989, p. 412).

For Wenger a person who operates in the intersecting boundaries of communities is a special class of actor known as a *broker*. Brokers hold a special role as they must straddle two worlds and mitigate between them:

> The job of brokering is complex. It involves processes of translation, co-ordination and alignment between perspectives. It requires enough legitimacy to influence the development of a practice, mobilise attention and address conflicting interests. It also requires the ability to link practices by facilitating transactions between them, and to cause learning by introducing into a practice elements of another (Wenger, 1998, p. 109).

Boundary objects can be artefacts, discourses or processes. What they have in common is that they are both formed by, and the subject of, *interactions* where two communities abut. For some organisations the activities that happen at their boundaries are so important that they have defined boundary practices and take steps to actively manage their peripheries. Perhaps most intriguingly of all, Wenger postulates the organisation itself as a boundary object:

> An organisational structure, for instance, is often considered as an overarching umbrella that incorporates multiple parts by specifying their relationships. But, in fact, it is more usefully designed as a boundary object intended to enable multiple practices to negotiate their relationships and connect their perspectives (Wenger, 2000, p. 235).

As we will see in Chapter Three, this idea helped to conceptualise the phenomenon of this study as set of interfaces, and hypothesise in general terms that one of the core Moodle community's defining aspects would be its local/global duality and its interaction with external enabling worlds (such as that of Higher Education).

2.2.5.4 Participation/Reification

Another important duality Wenger identifies, that is part of a long research tradition and relevant to this study, is that of *participation*/*reification*. Fundamentally, this duality relates to *knowledge*. With *Legitimate Peripheral Participation* (Lave & Wenger, 1991), knowledge for individuals involves competence and experience, while learning occurs during their interplay. Competence arises and exists in a specific social space; experience can happen either in or outside of that space; and it is the intrusion and extrusion of experience into and from the group that is at the core of this theory. Or, as Jordan puts it, knowledge is "the ability to participate meaningfully” whilst learning is the “process of becoming a member of the working community of practice” (Jordan, 1996, p. 18).

As Wenger began to move away from legitimate peripheral participation his research focused more on knowledge as a commodity, as something that organisations could seek to maximise. However, to illustrate that this commoditisation is not trivial he posed knowledge as a duality. It contained a tension between its abstract (reified) and is living (participatory) forms (Wenger, 1999). This idea has a large place in the literature. Polanyi introduced the term *tacit knowledge* as “a knowledge which we cannot tell” (Polanyi, 1967, p. 5). Hunches, guess work, unconscious habit and social idioms may all be forms of tacit knowledge that it is not possible to observe and measure but which may nonetheless be a vital contributory factor to overall knowledge. Brown and Duguid call this *know-how* and *know-what*:

> The organizational knowledge that constitutes 'core-competency' is more than 'know-what' explicit knowledge which may be shared by several. A core competency requires the more elusive 'know-how' – the particular ability to put know-what into practice (Brown & Duguid, 1998, p .91).

Wenger describes *reification* as how we a use a term as a projection of what we mean:

> It is an abstraction. It does not do the work by itself. But after a while, as I use it to think with, it starts talking to me as though it were alive. Whereas in participation we recognise ourselves in each other, in reification we project ourselves onto the world, and not having to recognise ourselves in those projections, we attribute to our meanings an independent existence. This contrast between mutuality and projection is an important difference between participation and reification (Wenger, 1999, p. 58).

Wenger sees participation and reification as complementary processes. If reification has hardened something into an inappropriate form, participation can make it malleable again e.g. a judge may interpret laws. Likewise, reification may compensate for the limitations of participation, due to its informal, subjective and irreproducible nature e.g. notes are made of a meeting. This leads to the idea of a trade-off inherent in a knowledge artefact between its reified and participatory aspects. For Hildreth and Kimble (2002) this is the degree of *hardness* or *softness* that knowledge has. If knowledge is mostly soft it has a high proportion of participation in its make-up. If it is predominantly hard it is made up of mostly reified knowledge.

2.2.5.5 Knowledge and Innovation

The participation/reification duality is examined in the Knowledge Management (KM) literature by Wenger and others and tacit knowledge becomes something of a unicorn, always elusive but vigorously hunted. However, the capture and codification of this knowledge – which is futile (Hildreth & Kimble, 2002) – is not the real goal. What is most sought after from students in classrooms or workers in multinational companies, in the literature, is innovation. Innovation is used as shorthand for the creation of new knowledge. In educational research, this may represent the pinnacle for students. In knowledge management literature, this may represent a product or process that gives competitive advantage to an organisation. Innovation has an extra significance in communities of practice where open source is used because the openness creates a huge increase in the amount of potential available information to participants: "At its root, open innovation is based on a landscape of abundant knowledge, which must be used readily if it is to provide value for the company that created it" (Chesbrough, 2003, p. 37).

For Von Krogh (1998) innovation will involve some process whereby an individual's tacit knowledge is made explicit to the group that he/she is part of. Personal knowledge, for example, because it has yet to be expressed may require new language and hence "recognition of new business opportunities might require an innovative vocabulary such as 'neutraceuticals' 'infotainment' 'edutainment' or 'cybershopping'" (Von Krogh, 1998, p. 135). This may be relevant to a development community such as Moodle because they are well educated and skilled but more importantly because they are in a wider peer community where new technologies, methodologies and beliefs can appear as rapidly as old ones may fall into obsolescence. These could be new pedagogies as much as new programming languages.

Although innovation is a useful concept it can be something of an unquestioned ideal in the literature, and we need to be careful of its exposition as an almost completely benign phenomenon. Perhaps it is that no more than one villain can be cast in a particular story. Thus, in *situated cognition*, it is abstract and institutionalised forms of teaching that are to blame. In knowledge management it is the ignorance of available or potential information that is the problem. However there remains the basic issue of unwanted or invalid innovation: creative plagiarist strategies in the classroom, bugs and viruses in computer software, outsourcing strategies in companies that lead to job losses in the economy. That is not to say that malign or deviant forms of innovation are not dealt with in the literature but just that they tend to appear in different contexts. For instance there is much research literature looking at how to increase participation in open source communities, whilst elsewhere there is a lot of research into software bugs in open source software projects, yet increasing participation could also increase bugs.

2.2.6 Problems with the Communities of Practice Model

Communities of practice, as originally formulated and later developed, were based upon anthropological studies of people at work or learning as we have seen. They are essentially unique vignettes, as much the product of the observer as of the group of study. Yet they have given rise to a great deal of theorising on the part of the researchers involved and their subsequent followers in this area. The communities of practice theory claims to model organisational behaviour in an abstract way, and be widely applicable. We may ask if this generalisability is sound. Lave and Wenger's original model was based on four apprenticeship studies all broadly ethnographic in nature, but not necessarily reproducible nor rigorously comparable to each other. Wenger refined his thinking based on his own ethnographic study of insurance claims processors, which alone became the basis of an influential book. The counterargument may be that the communities of practice model has been applied so many times by subsequent researchers that surely it is applicable. Although there is an obvious circularity to this, the effect of Wengerian communities of practice on research cannot be denied and it would be a grand delusion indeed if the theory was badly flawed but that everyone who applies it *thought* that it was useful. Indeed, in the qualitative tradition where outcomes are fuzzy and theories are generally not highly predictive in nature, the use of a theory may be the mark of its validity. Of course there may remain a problem that situated modes of being and knowing should by definition remain resistant to abstraction, to codification, to their explication by theory. An epistemology that "begins with activity and perception, which are first and foremost embedded in the world" (Brown et al., 1989, p. 41) ironically itself becomes the basis of a canonical theoretical narrative of the research literature.

There are alternative ways to model communities (Ibert, 2004). Chiu, Hsu and Wang (2006) build on work from social psychology and cognitive theory to attempt to model the factors that lead to knowledge production in an online community that can be measured statistically. Their study provides a useful synthesis of significant work relating to organisational behaviour outside of the research mentioned already here. They follow a broadly positivist paradigm and draw upon the results of research where questionnaires were combined with the results of qualitative tests, which were tightly defined or also combined with behavioural data which was again narrowly specified. For instance, they looked at the quality and quantity of contributions of online community members and compared these with the contributors' responses to a survey asking them to evaluate their level of expectation of reward for the community, expectation of reward for themselves, trust in the community, and expectation of reciprocity. Interestingly, one of their most significant claims is that expectation of reward *for the community* matters more than expectation of *personal reward* in deciding the quality and quantity of their contribution. A limiting aspect of their work is that its inputs and outputs are often narrow. So, for example, they assume that personal and community reward are orthogonal. However, this does not take into account that participants could potentially obtain personal reward by actively not participating in the community. They might be, for instance, arbiters of local knowledge in their primary community which would be surrendered if that knowledge was contributed back to the wider community. Wenger's "identities of non-participation" are relevant here.

A wider problem of this more positivistic side of the research is that many of its underlying concepts become unstable under scrutiny. Identity, motivation and innovation are complex issues whose valuation is very often in the eye of the beholder. Although self-reported data is compared with behavioural data, this is not designed so as to validate the former (which may be suspect) but rather to measure innovation. And, although they attempt to quantify innovation with some success, they do not treat it as a duality. Their findings may have most applicability in heavily institutionalised settings where successful outcomes can be rigorously defined and evaluated.

Communities of practice may be seen as ways of envisaging knowledge and learning with a strong, if not dominant, social dimension. This is in contrast to a cognitive perspective which may be concerned with how individuals attempt to acquire knowledge that is pre-existing, objective and predetermined. However, the communal and social aspect of communities of practice may become problematic if they are further magnified. For instance, organisations may become anthropomorphised, and attempts made to analyse the extent of their knowledge and learning. This is a problem because whether groups can learn and know, and what that means is contentious. Popper and Lipshitz (2000) argue that the focus should be on learning *in* organisations where the organisation is the arena for learning scenarios that happen between individuals and groups of individuals. Orr (2006) also cautions against the problem of considering learning in the aggregate, pointing out that organisations are not homogenous entities, and while a shop floor community of practice may develop effective processes, upper levels of management may not have knowledge of (nor interest in) these developments.

There have been fears that communities of practice could be used as a pretext for increased managerialism (Cox, 2005). This of course would be a dark irony as most adherents of the model urge greater flexibility for employees and for a loosening of hierarchical structures to allow communities of practice to move around with less friction. A common theme in the knowledge management communities of practice literature is the warning that communities of practice should not be actively managed, and should be nurtured rather than controlled (Brown & Duguid, 1991; Hayes & Walsham, 2000; Stamps, 2000). This is however a subtle point. It appears that the communities of practice model could work against itself, or become a predictor of its own unwanted results.

Contribution and/or communication through formal channels may often be subverted, if the group members feel that the mechanism is overly controlling (Ardichvili et al., 2003), or may otherwise impinge upon them in some way they feel to be unsatisfactory. Hayes (2000) gives a good example of this in an online community context, where users found ways around using the company-prescribed Lotus Notes electronic communication system as they felt it to be a managerial control device. So the danger exists that jaded employees, teachers or students may begin hatching their identities of non-participation as soon as they hear the dreaded managerial cliché “communities of practice.”

Despite these potential problems of its misapplication, the communities of practice theory remains an enduring account of human behaviour. In addition, as it builds on theories such as sensemaking and boundary objects, it has strong roots and linkages to the wider literature (Star & Griesemer, 1989; Weick, 1995). Several concepts we have examined here will now be important as we turn our attention to specific studies of open source communities:

- The treatment of knowledge/innovation generation has influenced the thinking of many researchers studying open source development communities and how participation may lead to new knowledge production.
- The examination of how communities bump into each other, and of the practices that occur at these collision sites, is also a powerful theoretical lens. Specifically the concept of boundary objects can be used to examine open source bug trackers.

- The dualities of local/global, which pertains to trajectories into communities and gives rise to community brokers, and that of identity/negotiation which can map to ideology and motivation of project members are also concepts to which we will now turn.

2.3 Communities of Open Source Software

2.3.1 Introduction

We have looked at a particular treatment of groups of individuals engaged in mutual endeavours as communities of their practice. This is a useful way to conceive of groups of people who develop software. It is also useful to posit such groups against other outside and interlocking groups. For instance: how does a teacher affect the development of Moodle? To do this a teacher may interact with a particular key community of Moodle – that of its software developers. They may even join it. Moreover this is a two way street – a Moodle development community must receive feedback from teachers. They need to know where Moodle is failing, where it is working well or how it could be enhanced and improved. Because much of open source software development is carried out in the open air of the web, it is possible to answer these *kinds* of questions, about inter-community relationships and interactions, via research. In this section the kinds of questions that researchers of open source software have asked will be reviewed and also their reasons for asking them. Some key themes will be explored such as the form and governance of open source projects i.e. who controls or leads them. These structural elements may be related to a community's size so project evolution is also touched upon. Next the motivations of individual participants will be examined. Individuals may be motivated to join a community by legitimately participating at the periphery of this community. They do this through the interface of its boundary object. Hence to conclude the literature review, we will take a concept from communities of practice – that of boundary objects – and use it to examine open source bug tracker research.

2.3.2 What is Open Source?

Open source generally refers to the source code of a piece of software being made publicly available. The source code is the blueprint and the building blocks of a software programme. Open source also generally means that in addition to the ability to access and read this source code, anyone can modify the programme by editing, rewriting (or fixing) the source code. The mechanism that makes this possible is the license, or legal terms, under which software is released. An important aspect of open source software development that follows from its legal basis is that it is distributed. It is not owned, in the traditional sense, by a central firm or a person. Rather, it is "community based, evolutionary knowledge creation" by people who are "dispersed across organisational and geographical boundaries, and collaborate via the Internet to produce a knowledge-intensive innovative product of high quality" (Cole & Lee, 2003, p. 663). For its most enthusiastic adherents it may thus represent a manifestation of Popper's (1968) open society, with its facilitation of an "open critical discussion that moves us closer to the truth" (Cole & Lee, 2003, p. 663).

The term "open source" itself was coined during a meeting of its adherents in 1998 in Palo Alto California. Its concepts were already well known and in extensive and growing use at this stage, but this new term, according to its proponents, "distinguished it from the philosophically and politically focused label 'free software.'" (OSI, 2012). *The free software movement* was a pre-existing and more radical group who believed in copyleft licensing, which ensured that open source software could not be turned into proprietary software (i.e. closed source). Thus anyone who modified free software would be themselves "compelled to leave copies behind for others to benefit" (Lakhani & Von Hippel, 2003, p. 293).

However, the strict copyleft viral nature of free software licences were not required by the open source movement which allowed for more mixing of proprietary and open source codebases in what was believed to be a more pragmatic approach. To promote a more inclusive agenda, the Open Software Initiative (OSI) was founded by Eric Raymond and Bruce Perens and it went on to create a definition of open source, which they used to certify whether given software was open source or not (Raymond, 1999). Whereas the free software movement proposed that access to source code was a human right, the open source movement took the less extreme position that source code of software, while highly desirable, will probably co-exist in a world alongside proprietary software. The open source movement believe that organisations that release source code, under any type of licensing, are inherently preferential to closed and proprietary codebases (Lakhani & Von Hippel, 2003).

Moodle, the software that forms the basis of this study, was released in 2001 under the General Public License (GPL) agreement. The GPL is a copy-left license, of the type advocated by the Free Software Movement (GNU.org, 2007). Anyone who modifies Moodle and then redistributes that modification must release the source code of their creation also and it must be released under the GPL license.

From this brief overview of the origins and basic forms of open source licences it should be apparent that there are strong political and ideological aspects at play. Ideology has been seen as a key marker of an individual's identity within what is known in identity research as the *social-structural domain* in which the individual operates (Schwartz, 2001). Researchers investigating the implication of ideological identification with open source culture and ideals have found it to be strongly linked with participating in particular open source projects i.e.

believing in open source tenets (“it's almost a moral duty to share information” (Raymond, 2003)) may correlate with joining and also staying in a project community (Stewart & Gosain, 2006). Stewart and Gosain (2006) found specifically that trust, born of shared ideological values, improves collaboration. However although identification with open source was seen to bind the community and attract members, it was also found that this could detract from project output, suggesting that consensus building could also be a hindrance to certain tasks. As Wenger noted, identification is a powerful binder, but it can also constrain negotiation of new meanings through either internal struggles (civil wars) or more aptly in this context, through cultish or “totalising membership” (Wenger, 1999, p. 207) which leads to a narrowing of what is negotiable.

2.3.3 Open Source Software Research

Open source research is a vast and burgeoning field. Google Scholar listed 6,250 results in December 2012 for the search: *“open source software” “social science”*. The open source terrain has been well prospected by researchers such as anthropologists, organisational behaviourists, economists and computer scientists. The literature may even emerge from open source participants themselves (Raymond, 1998; Raymond, 1999; Torvalds & Diamond, 2001; Raymond, 2004). It may also concentrate on its most famous actors and projects (Lakhani et al., 2002; Cole & Lee, 2003; Hertel et al., 2003; Herraiz et al., 2006; Krafft, , 2010); characteristics of such projects, such as how they are governed and organised (Shah, 2006; Dougiamas, 2007; Markus, 2007; O'Mahoney & Ferraro, 2007; O’Mahony, 2007); or the reasons that people are motivated to participate in them (McLure Wasko & Faraj, 2000; Lakhani & Wolf, 2003; Krishnamurthy, 2006; Oreg & Nov, 2008). Outside of these more situated studies are related research streams that examine wider effects or applications of open source, for instance its economic effects (Lerner & Tirole, 2003) or its impact on educational policy formation (Brown & Adler, 2008; Costello, 2012). Aksulu and Wade (2010) give a synthesis of the state of the art of open source research from an analysis of 618 peer reviewed articles and generate a taxonomy from their conclusions. Interestingly, they plot an evolution of the open source literature over time showing how licensing, developer motivation, open innovation and open source governance were knowledge islands that only began to emerge as the field matured. Some aspects of the research undertaken for this thesis were possible precisely because we have reached a particular maturation point i.e. that open source has been loosed from the confines of projects built by a computing community for a computing community, such as web server or an operating system, and may now pertain to artefacts that are produced by a community of programmers but intended for much more non-technical (though nonetheless specialised) field such as education.

2.3.4 Research Attractiveness of Open Source

Scholars have given various reasons as to why open source has been so thoroughly studied. Two prominent reasons cited for the research appeal of open source phenomena are: data availability; and novel characteristics of open source communities compared to other forms of human organisation (Von Krogh & Spaeth, 2007). These two motivations map onto the work at hand as they provide the artefacts of study and the theoretical grounds respectively for this research.

2.3.4.1 Access to Data

The second of these factors is straight forward: data. In this study, an initial problem was the superabundance of data. As we will see in Chapter Three ascribing the parameters of the study was an important and complex part of the research design, simply because there was so much prospective data. As open source communities conduct so much of their business in public they generate vast digital data trails. For Moodle for example there are many hundreds of thousands of discussion forum postings in Moodle dot org. Below them lies a smaller, more core, but still vast bug tracker. The bug tracker is a more focused data source because it contains less participants and a greater concentration of core Moodle community members. Taken as a boundary object, it becomes a classical artefact of community of practice study. The purpose of this study was not to focus on bug trackers in general. Rather it was to elucidate one *specific* interface in one *specific* educational technology setting, although an analysis of selected research in this area will provide important insights.

2.3.4.2 Novel features of open source projects

The other important motivational force, in addition to the lure of un-researched data that this study shares with much of the literature, is that of the novelty of the open source approach. Sometimes this novelty is directly related to its visibility. Although researchers have noted similarities between open source development and scientific peer review process (Bezroukov, 1999; Bergquist & Ljungberg, 2008), even conceiving it as an "extension of the scientific method" (DiBona & Ockman, 1999), it is more often attractive because of its apparent strangeness. Why would people engage in activity for which they may not be paid? How would such endeavours compare with their remunerated counterparts? How does governance and ownership emerge when the product is free and cannot be owned in strict legal terms?

Before looking at some of the main questions that have been asked and are relevant to this study it is worth pausing to consider the relationship between the novelty of open source and access to data. It is plausible to suppose that access to data might be relatively more important to the growth of the open source research field than other *self-professed* motivations of researchers. We cannot easily come up with good controlled research designs which would compare closed source with open source projects because we do not have access to the workings of the closed projects. In our case, for instance, we did not have such access to the work processes, nor indeed the members of the developer community of WebCT/Blackboard which is a commercial competitor to Moodle. Therefore it should be borne in mind that supposed novelties of the open source approach are not often provable. Open source is often contrasted with the characteristics of alternative software development licensing models, but the data available on those alternative approaches is more limited than for open source, so there are problems with this research approach. Thus, where novelty is used here we are following the convention of the literature and aware that this is often an *assumed* novelty. We should also note the reflexive nature of this type of research (Bergquist & Ljungberg, 2008) and of course we can entertain an hypothesis with intuitive appeal that open source may be a superior model precisely because it allows itself to be so easily researched and hence presumably improved.

2.3.5 Form and Governance

A primary novelty relating to open source projects is their inception. This in turn influences their form and so their governance. A project may start out as a hobby or passion of one developer, lacking a team, a roadmap, or any market research to justify its existence and so it is not surprising that there is a vast graveyard of open source efforts that never got off the ground. Krishnamurthy (2002), sampling 100 mature open source projects, found that most were developed by individuals rather than communities with the mean participants being four and the mode one. He noted that project growth and participant growth go hand in hand.

Once a project has more than one member it must be governed. There must be the "means of achieving the direction, control, and coordination of wholly or partially autonomous individuals and organizations on behalf of an [open source software] development project to which they jointly contribute" (Lynn et al., 2002, p. 6).

Markus (2007) identifies two basic ideas of governance in the literature as monolithic or multidimensional. The former are relatively simple conceptions of open source governance that focus on ‘‘bazaar governance’’ in contrast to traditional hierarchical governance. In our approach to the literature this might be an example of over emphasising novelty i.e. the monolithic conception may be relatively naïve because it assumes that the open source is more different to traditional forms of organisation than it actually is, that "its form is operationally viable, economically efficient, or technically superior in comparison to other forms" (Markus, 2007, p. 152). Markus contrasts this monolithic view of governance that is relatively flat and communal, with conceptions from the open source research literature that are more concerned with the different types and variations of governance mechanisms. Markus links the monolithic governance conception with a reductionist approach that gives the role of the licensing of the software primacy i.e. because of the open source license it naturally flows that a particular type of governance of the project will emerge. Governance evolution however can be influenced strongly by factors other than simply the licensing of the software. A project with no guaranteed long term future, or from which there is little to be financially gained, may make users more co-operative and hence more likely to volunteer for leadership roles or accept others in them. This has been shown in various online community contexts for example in communities of informal learning in education (Greenhow & Robelia, 2009; Scott et al., 2009) or in cooperative behaviours in online gaming communities (Williams et al., 2007; Mysirlaki & Paraskeva, 2012).

Something of the split noticed by Markus (2007) in the literature of open source community governance may even be a chronological one. The early open source landscape was a relatively more uniform and basic one. It was not until the release of the Netscape Browser in 1994 that truly commercial open source projects began to appear. These were open source projects that were not started by unpaid enthusiasts but fully paid commercial employees. Thus hybrid forms of open source software emerged where paid employees and volunteers worked together and this would have an impact on how those projects were governed (O'Mahony, 2007). Simultaneously, open source projects that started out as volunteer efforts began to be commercialised and/or attract commercial participation which necessitated evolvable governance. Governance in small emerging projects could be informal and unstructured (Von Krogh et al., 2003) but fundamental questions of control and leadership emerged when projects began to scale.

2.3.5.1 Control and Leadership

Raymond (1998) used the term "benevolent dictator" to describe the classical model of the authority wielded by an open source project founder. The founder is a dictator not because they rule by force, but because they wield great power, and largely dictate proceedings at a high level. However, they must remain benevolent or project participants could simply leave the project, take a copy of their code with them and start an alternative effort. This is known as a *fork* of the software. Leaving aside that this is a relatively crude depiction of a complex phenomenon we can say that Moodle in many respects operates along the benevolent dictatorship model. Martin Dougiamas has the ultimate say in what Moodle will or will not do (most famously in his reputed refusal of offers of large amounts of money to sell the project). This constitutes a relatively simple model of governance and is found in many famous projects such as the programming language Perl, and the Linux Kernel computer operating system (Gardler & Hanganu, 2012).

As projects evolve more people are attracted to them and sometimes other governance models emerge. These forms may have democratic elements such as those with an elected board, as in the case of the Apache web server and the Debian Linux distribution. Or, projects may be set up in this way from the outset, such as for the Sakai VLE (Farmer & Dolphin, 2005). For Gardler and Hanganu (2012) this governance is meritocratic and examples of successful forms of such governance operate "with an almost completely 'flat' structure, which means that anyone willing to contribute can engage with their projects at any level [whilst at] the other end of the 'control' spectrum is the benevolent dictator governance model, which is led by one individual" (Gardler & Hanganu, 2012, p. 1).

Regardless of how much control an individual – or a central core of developers – may have, they do not work alone but are part of a much bigger collective. The "onion model" describes how the layers of a project radiate from this core (Crowston & Howison, 2005; Crowston & Howison, 2006). The layers of the onion correspond to increasing degrees of involvement and influence in a project when viewed from the outside in:

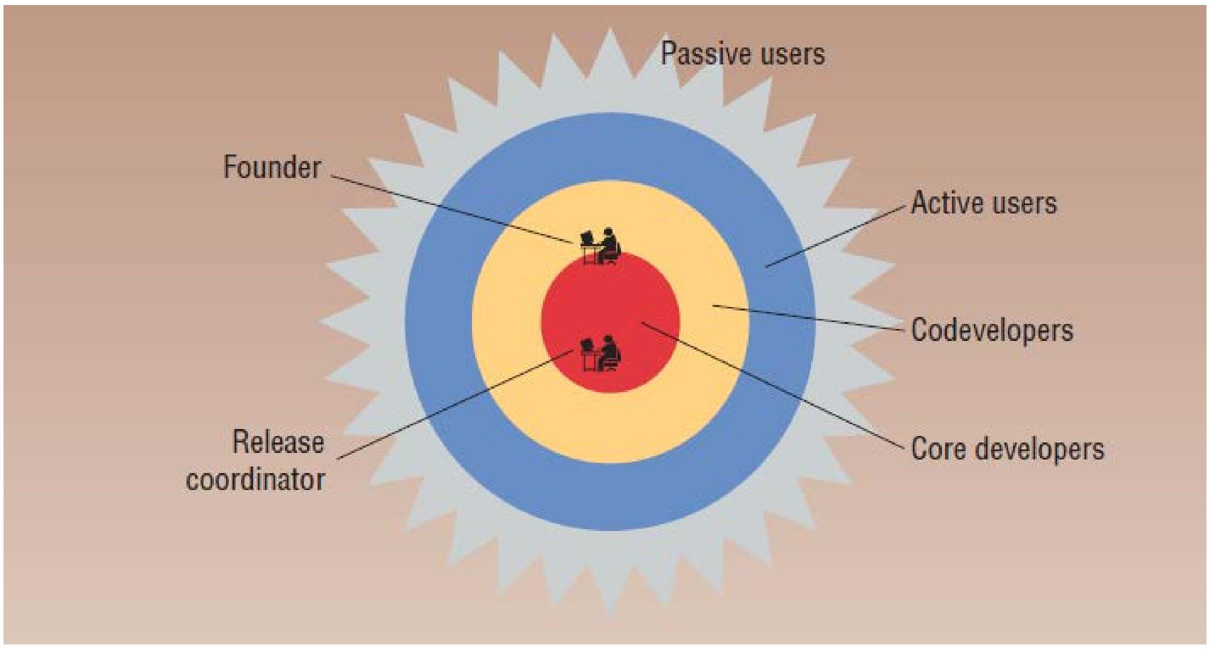


Figure 1 Onion Model of Open Source Projects (Crowston & Howison, 2006)

Crowston's model may suffer from some over-simplification in certain cases and has been developed further, such as by using Social Network Analysis (SNA) which is a growing area of research into online communities (Oezbek et al., 2010; Barham, 2012). However it still provides the theoretical touchstone for much research in this field.

2.3.6 Evolution and Project Size

Widening the concept of governance to include not just the most ostensible power relations and political and organisational structure, we can look at the processes and work practices involved i.e. the culture of the community. However, reading this literature poses problems. Much research is concentrated on the largest and most mature open source projects. These may be "marquee" communities or simply those with the largest and most available datasets. Smaller and less mature efforts may exhibit very different characteristics such as very loose formal organisational processes (Von Krogh et al., 2003; Allen, 2009) and may also lack staple practices of large projects such as software testing (Crowston & Scozzi, 2004; Barham, 2012). This was also the case with Moodle (Alier et al., 2010; Krishnamurthy & O'Connor, 2013), although there is a dearth of scholarly research on Moodle in this area.

Just as individual projects may grow and change, the field itself including its tools and methodologies is also rapidly evolving. Such may be the gulf in practices that studies carried out in even the recent past may be difficult to compare with contemporary projects. Herraiz et al (2006) note that the GNOME project was transitioning between the CVS and Subversion code tracking systems at the time of their study. A similar change occurred in Moodle a few years later when CVS was abandoned for the Git system. These tools can have a huge influence on practice so even comparing a project with itself over time throws up problems. This is an issue tackled in Chapter Three in the formulation of the temporal boundaries of the case study and we also examine a case of tool use and transition in Chapter Four.

2.3.7 Motivation and Identity

Just as projects evolve so too does their membership and the process of joining. Significant changes also occur in the ways participants move through the layers of the community. Closely related to these trajectories of community membership are motivations or why people choose to get involved with open source projects.

This area has been much researched for the relative novelty of projects that people may work on for free as opposed to their paid work. Indeed there are almost shades of pathology in certain behaviours associated with the open source project volunteer. In the Boston Hacker survey, respondents reported how much sleep they lost whilst staying up at night working as open source project volunteers before turning up for work the next morning (Lakhani et al., 2002). The concept of motivation itself has been researched extensively and Ryan gives a good conceptual encapsulation:

> [...] even brief reflection suggests that motivation is hardly a unitary phenomenon. People have not only different amounts, but also different kinds of motivation. That is, they vary not only in level of motivation (i.e., how much motivation), but also in the orientation of that motivation (i.e., what type of motivation). Orientation of motivation concerns the underlying attitudes and goals that give rise to action—that is, it concerns the why of actions (Ryan & Deci, 2000, p. 54).

This orientation of motivation is divided in the open source literature into *intrinsic* and *extrinsic* forms (Lakhani et al., 2002; Oreg & Nov, 2008). Extrinsic forms may be pay or, in purely voluntary projects where no pay is involved, the prospect of putting ones open source participation on a CV. Intrinsic motivations may be a love of coding and a passion for creating software. Motivations are complex and changeable, but a consensus of the literature in this area is that intrinsic motivation leads to better eventual outcomes and that successful open source projects contain members who exhibit strong intrinsic motivations associated with concepts such as fun, flow, learning and community (Krishnamurthy, 2006). Orr remarked on the intrinsic motivation of the members of the photocopy machine repair community of practice, attempting to single it out as a primal instinct and a primary cause:

> “[The] Technicians seem to like fixing machines. Most of the ones I knew were inveterate tinkerers, and many had been since childhood. They did not take the job to follow

> documentation but to fix machines. Nor, I think, is it primarily about creating their identity. Their interest is in the machines, not the identity; the identity follows from, and is subordinate to, their ability to fix the machines" (Orr, 2006, p. 1815).

This example also highlights the case that disentangling and classifying motivations can be more difficult than it sometimes appears in some of the open source literature because the technicians are operating broadly under a dual motivation: They are paid to fix machines (extrinsic) and they love fixing machines (intrinsic). (See Waterman et. al. (2008) for a study that attempts to quantify the distinction between intrinsic and extrinsic motivations in general.)

Although Orr foregrounds, if not elevates, the workers personal calling (intrinsic motivation) as representative of their identity, the concept of identity is never monolithic. Rather, in many models, identity is at least threefold consisting of ego, intra-psychological and social (Schwartz, 2001). In open source communities an individual's status may be affirmed by their peers whereby they build a reputation for contributing code. A project called Advogato, with perhaps the most formalised and complex mechanism for this creation of social status, was well studied by Stewart (2005), but the most expressive explanation of the phenomenon is given by Raymond:

> You become a hacker when other hackers call you a hacker. A "hacker" in this light is somebody who has shown (by contributing gifts) that he or she has technical ability and understands how the reputation game works. This judgment is mostly one of awareness and acculturation, and can only be delivered by those already well inside the culture (Raymond, 1999, p. 31).

Hence we must acknowledge that social and individual identities are mutually constitutive such that it may be difficult (and it has been argued sometimes not useful) to tell "where the sphere of the individual ends and the sphere of the collective begins" (Wenger, 1999, p. 146) and hence motivation itself, arising from identity, also has this aspect.

2.3.8 Social and Individual Identity

As identity is such an important concept both in the community of practice model but also in the open source literature, albeit mostly concentrated around the aspect of motivation, it is worth exploring this concept further. Indeed pertinent to widen our focus and consider both individual and group identity at this point precisely because of its sometimes narrow treatment in the open source software literature, and instead to frame it more broadly and draw on some of the wider literature in this area.

The concept of identity is deeply indebted to the work of Erikson (1950, 1974) whose model of identity drew upon, and attempted to integrate, ideas from both sociology and psychology (Schwartz, 2001) in essentially examining the question asked by the young adult of: Who am I? Moving away from Freud’s conception of identity as essentially composed of parental introjects, Erikson (1950) conceived the consolidation of identity as marking the end of childhood, as ego coherence versus confusion (over who one is and what one believes in). He conceptualises identity as ego synthesis versus confusion and as something that gives an individual predicable behaviour:

> An identity-synthesized person’s choices and actions are consistent with one another, such that one can predict, with some degree of certainty, what that person is going to decide or do in the context of any particular situation or life choice. (Schwartz, 2001, p. 10)

Identity was hence conceived of as the set of goals, values and beliefs that one shows to the world (Erikson, 1950). The relationship of the individual to the world was expressed in one’s identity relative to a group:

> […] social identity was identified as a sense of inner solidarity with a group’s ideals, the consolidation of elements that have been integrated into one’s sense of self from groups to which one belongs (Schwartz, 2001, p. 11)

Identities have been linked to the domains in which they operate such as personal, personal–social, and social–structural. Models orientated towards social identity often draw on the structural aspects of society and culture and can hence be termed socio-structural (Schwartz, 2001). In this area Identity Capital by Côté (1996) posits identity sociologically. Building on the economic concepts of Human Capital by Becker (2009) and Cultural Capital, a theory of social-class reproduction by Bourdieu and Passeron (1977), Côté formulates Identity Capital:

> Most generally, the term “identity capital” denotes what individuals “invest” in “who they are”. These investments potentially reap future dividends in the “identity markets” of late modern communities. To be a player in these markets, one must first establish a stable sense of self which is bolstered by the following: social and technical skills in a variety of areas; effective behavioural repertoires; psychosocial development to more advanced levels; and associations in key social and occupational networks. At the very least […] key resources for bargaining and exchanging with others in the late-modern communities are apt to involve skills in negotiating life-passages with others, such as securing validation in communities of strangers, and attaining membership in the circles and groups to which one aspires. The most successful investors in the identity markets presumably have portfolios comprising two types of assets, one more sociological and the other more psychological. (Côté, 1966, p.425).

Individuals investing in social and technical skills and establishing behavioural repertoires has obvious ostensible potential for application to areas such as open source software communities of practice. Social capital theory has been proposed as a useful model for studying distributed online communities (Daniel et. al., 2003) and it maps well to the notion of open source gift giving communities of Raymond (1999) and indeed to community of practice legitimate peripheral participation (see Table 1 Key Theoretical Concepts) however it is not without its problems. For example empirical studies of identity in groups and organisations have found that participants identify with a group simply be being in it, regardless of their activity or role within in (on which social capital theory is focused). Hog and Turner (1985, p. 51) conducted experiments which lead them to conclude that "interpersonal attraction (positive or negative) is related to group formation only in so far as it enhances intergroup distinctiveness". That is you do not necessarily need to like someone to feel like you belong to a group but more simply believe that they are similar to you. This has important implications. For instance, in the study at hand, the distinctiveness (or otherwise) of the communities under examination could be important. There is tension in the identity literature between types of innate group, almost tribal behaviours, that are disconnected from any interpersonal bindings within the group on the one-hand and on the other between individuals who may seek out (or simply stumble-upon) like-minded individuals and form communities with them. It is this second form that Wengerian communities of practice are largely predicated upon (for a detailed discussion of this issue see Ashford and Mael (1989)).

Erickon's theory of identity was elaborated upon by Marcia (1966) who framed ego identity formation as a trade-off between exploration and commitment. An individual must explore and seek out social positionings before committing to one. Commitment is "the attainment of a clear sense of self-definition or ego identity within one or more domains" (Yoder, 2000, p. 96). Too much exploration can be negative as can too little which would result in a premature commitment. The notion of commitment can be usefully thought of here as having strong analogues with the notion of identification in Wenger's work: although identification is important, even vital, it can also prove limiting as it may limit negotiation. Indeed negotiation is the group equivalent of individual identity exploration. Although researchers of identity development acknowledge that identity is a socially embedded process (Marcia, 1966; Waterman, 1988), Yoder (2000) points out that identity research often focuses, more narrowly, on the psychological perspectives of internal exploration and development of the individual. Wenger's (1998, 1999) greater focus on the group dynamic is hence useful in this regard.

One final important aspect of identity theory that is relevant is that of the core existential Eriksonian question of finding ultimate individual expression. Following from Erickson's (1974, p. 107) contention that the ego was safest "when grounded in activities" Waterman (1995) drew upon Aristotle's notion of an inner daimon that finds expression in some key activity which is of personal significance:

> From a eudaimonist perspective, the daimon represents the core of one's self. It remains unconscious and untapped until it is discovered during the course of engaging in activities that resonate with it (Schwartz, 2001, p. 10).

Waterman (1995) uses the term eudemonia to denote living in accordance with the most truthful expression of one's inner being and as representing also a state of happiness that can be contrasted with hedonic living. The search for such activities, and living in accordance with them while resisting hedonistic activities, has influenced empirical work that attempts to disentangle intrinsic and extrinsic motivations (Waterman et. al. 2008) and influenced ideas of individual identity because it follows that people may seek out groups to which they feel they may belong based on the activities that personally resonante with them. These two strands, of individual and group identity formation theory, of the wider identity research should help us contextualise the issues of identity at hand in open source communities of practice.

2.3.9 Joining and Legitimate Peripheral Participation

To now consider some examples of how motivation can operate we can start by simply saying that that once a person is motivated they are urged to some action. In our case the action will be to join a community. According to the onion model of joining, users start as silent observers or lurkers on the mailing list or the discussion forums. They progress by interacting, tentatively at first, such as in the mailing lists and then perhaps the bug tracker. Next they progress to contributing code and finally to being granted some permissions or ownership of part of the project (Crowston & Howison, 2005). This maps well to the community of practice concept of legitimate peripheral participation (Lave & Wenger, 1991).

Raymond's (1999) casting of open source as primitive "gift giving" communities can also be formulated as a community of practice phenomenon in this context. These gifts, which are lines of code, are the product of intrinsically motivated participants engaged in a form of play. The onion can thus be seen as a place where “the ‘apprentice’ is exposed to a certain environment, participates in sets of activities, handles (plays with) certain kinds of artefacts and is entrained into the sphere of specialist work the same way a child is into the home environment” as the community of practice model puts it (Jordan, 1989, p. 927).

As under the apprenticeship concept in communities of practice participants may join a project according to a “script”, such as identified by Von Krogh et al. (2003) in the Freenet project. This is an almost ritualistic process whereby newcomers go through a number of steps before being granted access to commit code to the project. As the Freenet project was an embryonic one, there were fewer layers of the onion, so after a number of emails to the mailing list a newcomer could gain access. The onion model is essentially an elaboration of this script where more practices must be learned, tools engaged with and boundary objects navigated e.g. unlike Freenet, more mature projects will have a bug tracking layer that exists over the code repository layer and beneath the email list/discussion forum layer. Indeed it has recently been pointed out that some of the most mature open source projects are gaining additional layers due to their increasing professionalism, with associated specialisation of roles within them (Barham, 2012).

Research into the GNOME project (Herraiz et al., 2006) used the onion model (Crowston & Howison, 2005; Crowston & Howison, 2006) as the baseline for an examination of joining trajectories. They excluded lurking activities which are not detectable and started tracking a participant’s entry into a project from when they first joined the mailing list and introduced themselves to the project. Following this they hypothesised that users would use the bug tracker before finally being granted authority to contribute code directly via the code repository. They found that while one group of users did fit this model, progressing through each layer of the onion to the core, others engaged in all layers simultaneously and made much more rapid progress and one joiner was a complete outlier whose trajectory did not fit either of the two groups (and which they discarded from their findings). The faster integrators were correlated with hired developers, whereas those with a slower entry to the core were volunteers. Hired developers could be working for commercial companies or be university staff.

The joining script for a project may not always be apparent and sometimes this is deliberate on the part of the existing members:

Help Wanted. We always need Heavy Lifters in code. If you’re excited about web browser technology, why not get involved in the premier Open Source browser project? We’re especially looking for people with skills in Mac OS X programming and Windows developers. Get started today by finding and fixing something. *Instructions are not provided here since figuring out how to do all of this can be considered part of the “entry requirements” ;-)* (Krishnamurthy, 2005 emphasis added)

The above quotation, from an advertisement for new members of the Firefox web browser project, indicates that entry may be purposefully difficult, that rituals of peripheral participation may be carefully draped over boundary objects.

Another important concept in joining trajectories is the role of mediators. This concept is informed by the brokers of Wenger’s (1999) community of practice theory. These are important people that operate between boundaries helping people into deeper layers. The most important of these people may be the community founder whose charisma or “benevolence” is a key success factor to a project (Raymond, 1998). Or at the outer layer they may be “active members – those with multiple interactions – [who] form a buffer between developers and peripheral users” (Crowston & Howison, 2006).

2.3.10 Open Source Boundary Objects

All successful trajectories into an open source community must negotiate boundary objects (see Table 1 Key Theoretical Concepts below) regardless of the route they take. Allen (2009) outlines a research agenda based on boundary objects in open source projects. His boundary objects are the source code of the Snort project and a seminal paper by the founder that described the project. We can find an analogy between this and Moodle, with Dougiamas’s social constructivism which is often cited by insiders (Cole & Foster, 2007). However, as a large, mature and sophisticated project, there are a range of boundary objects we could look at in Moodle, such as the community discussion forums, Moodle conferences, and of course the Moodle bug tracker. In many ways the Moodle bug tracker provides a particularly useful boundary object to study: it sits in a more enumerable social layer below the forums (though it is still vast); it contains all of the key members of the core community; and it is constantly being bombarded with bug reports. Hence an appraisal of the relevant literature on bug trackers was undertaken.

2.3.10.1 Bug Trackers: Canonical Accounts of Bug Fixing

Debugging is one of the least understood activities in software development and is practiced with the least amount of discipline; it is often approached with much hope and little planning (Ghezzi et al., 2002, p. 331).

The efficient fixing of bugs is one of the key claims of open source. As Raymond (1999) puts it, “given enough eyeballs all bugs are shallow” i.e. letting more people see something means that problems will be solved more quickly. Bug trackers are particularly well studied by researchers. Often the aims of these studies are to find success factors for bug fixing or resolution. A study of bug resolution in the open source Eclipse project found that the most influential factor affecting length of bug lifetime was commenting activity i.e. the more people that commented on an issue in the tracker the more likely it was to be fixed (Panjer, 2007). Giger et al. (2010) using statistical analysis, also found comments to be the top predictor of bug fixes in a study of three open source software projects – Eclipse, Mozilla, and Gnome. Similarly a study of the Firefox bug tracker found the amount of user comments to be significant (Hooimeijer & Weimer, 2007) and also that the amount of attachments to bug reports was important (i.e. screenshots). Interestingly, they found that the submission of patches (proposed written code fixes) had no effect.

A study of the JBoss project (Weiss et al., 2007) found that the actual amount of programming time it would take to fix a bug could be predicted by analysing the title and text of incoming bug reports and comparing these to how long developers reported that it took them to fix similar previous bugs. This study did not consider how long bugs stayed open, but more narrowly how long developers reported working on them, once it had been decided that they would be fixed. The authors also conceded that the (self-reported) work estimation data of the developers is not available in most bug tracking databases.

Bug resolution studies vary widely in the factors that they choose to analyse as being relevant to successful bug completion. So, for instance, a study might choose to consider the relevance of the starting date of an issue (which proved to be significant) (Giger et al., 2010) or more discerningly the amount of issues that were submitted at the same time, i.e. the issue submission load (Hooimeijer & Weimer, 2007). Or, studies might look at the textual descriptions of the bugs (Hooimeijer & Weimer, 2007; Weiss et al., 2007). It will come as little surprise to the student approaching his essay deadline (or the researcher as the call for abstracts submission date looms) that bug fixes have been found to be closely correlated with release dates of software i.e. that some people only get around to doing things when they really have to (Francalanci & Merlo, 2008).

Most of these studies are characterised by approaches that are primarily computational and quantitative and generally lack narrative accounts of bug fixing. A notable exception to this is

Hooimeijer and Weimer (2007). Their work has a range of interesting factors, which they consider in their bug resolution model e.g. submitter reputation, issue submission load, textual description of bug, bug severity changes etc. It is certainly arguable that we should not be surprised at this and that good data about what bugs are can best be gleaned from their narratives i.e. approaches considering narrative data may yield richer explicative models of how bugs come to be resolved. This is an important area where the community of practice model can inform our treatment of this literature.

Another aspect that many of the previous studies of bug trackers may suffer from is their reliance on canonical data sources i.e. on representations of the way things are officially supposed to work in a project. These sources are concrete and measurable but not always trustworthy and, moreover, they often do not account for useful practices that might spring up as improvisations around the canonical work practice. That a bug is fixed represents the officially sanctioned best outcome. However in trying to fix the issue other factors may come into play or it may be discovered that there are “workarounds” or alternative narratives that lead to a different but equally happy ending as shown by Orr (1990) in his community of practice study of the photo copy repair technicians. In most cases the fixing of bugs will represent the best outcome but there may be others, not addressed well in the literature, that for many users pass a “threshold of acceptability” (Schwartz et al., 2002). Hence studies of bug trackers that rely solely on enumeration of canonically approved outcomes (see Table 1 Key Theoretical Concepts below) will only tell part of the story.

2.3.11 Open Source Communities Summary

Open source projects emerge and die regularly. Some may grow and evolve as participants are spurred to join. To join these communities new entrants must negotiate particular boundary objects. These objects are filters that allow the wider world percolate into a core community in a systematic way. One such boundary object is a bug tracker and although it has been well studied and provides some insights into how these communities resolve issues and negotiate meanings with outer communities, there is still a lack of narrative accounts of these resolutions. Where narrative accounts have been developed they have provided richer and more robust models of boundary objects and their associated communities.

2.4 Chapter Conclusion

To provide a reference for the reader, and to help summarize and reinforce the link between certain aspects of the community of practice and the open source software literatures, a table is given below recapping some of the key concepts which we will draw on later in the thesis:

Table 1 Key Theoretical Concepts

Boundary Object	An item over which two communities have partial jurisdiction. Can be artefacts, discourses or processes Serves to include as well as exclude as it separates but also joins two groups A bug tracker is a boundary object allowing the wider community to contribute bug reports to a core community who fix them
Canonical Practice	The way things are supposed to work Characterised by explicit instruction. Well understood. Written down. The fields of a bug tracker database are designed to capture canonical accounts.
Non-canonical Practice	The way things actually work. Implicit or not written down. May be characterised by insider knowledge. May be characterised by narrative such as Orr's (1998) photocopy repair "war stories". Not captured in the bug tracking literature that deals only with statistical reports of bug fixed derived from bug tracker databases.

	May be derived from participant accounts
Identity	Identity is at least threefold: personal, personal–social, and social–structural (or socio-
Negotiation	If meaning is socially constructed then new knowledge must be negotiated. The joint enterprise of a community is continually renegotiated by its members (Wenger, 1998). Two communities have partial jurisdiction over the resources represented by a boundary object, and mismatches caused by the overlap become problems for negotiation (Star & Griesemer, 1989). Negotiation is influenced by identity and over-identification can crowd out space for negotiation of new meaning.
Joining	Refers to how new members join an open source community of practice. May happen via a boundary object e.g. bug tracker or mailing list. May follow a “script”. Characterised by apprenticeship and participating peripherally at first (legitimate peripheral participation). Can follow a short or long trajectory. May occur as a new member navigates the layers of the onion Model (Crowston &

	Howison, 2006).
Broker	A person who operates in the intersecting boundaries of communities. Brokerage involves processes of translation, co-ordination and alignment between perspectives. Brokerage requires legitimacy. Play important in open source projects to induct new members.
Legitimate Peripheral Participation	Based on the idea of apprenticeship. Newcomers negotiate access to the share cultural heritage of the group. They are *enculturated* to the group. In open source software literature this is related the concept of project joining (see Joining). Observe before acting: such as "lurking" on mailing lists before posting Newcomers must prove themselves by uncovering the non-canonical accounts of practice via sustained participation e.g. in an open source project "figuring out how to do things" may be considered "part of the entry requirements" (Krishnamurthy, 2005).

Communities we have seen can be best seen in the practices they engage in. This engagement is where true learning happens, learning to be part of a community. The shared enterprise of the community that is engaged in is defined both by canonical accounts of practice, what is written down and official, and non-canonical or unwritten accounts. These are not opposites but a duality, they correspond to hard and soft types of knowledge that act on each other in practice. The community is not ascribed by a boundary but rather starts and ends at boundary objects that help it to mediate with outside words.

The rules that govern communities of knowledge workers are complex, although they engage in a mutual practice they do so in a landscape of processes and artefacts that requires diligent and lengthy negotiation. Negotiating this terrain often involves vectors of joining, learning leaving and each happens via at least one definable boundary object. In some cases, such as bug trackers, the boundary object may appear well defined. However each has a unique identity situated within a community which will have its own culture and governance. The resolution of issues with bug trackers (as a way of defining the negotiation of meaning) has been studied but questions remain and new ones emerge as the field of open source rapidly grows and evolves. In particular: the question arises as to what newer open source projects that interface with a wider non-technical world, look like. The world of education is now heavily involved with and committed to open source software such as in our case Moodle. Further study of how its community operates, negotiates meaning and how ultimately the classrooms of the future are being shaped is needed and will be the focus of this study.

3 Methodology

3.1 Introduction

This study sets out to explore the development of the open source educational software Moodle during 2007 and 2011 by people who became directly involved in this activity by changing, or seeking to affect change, in the code of that software. The data for this study comprises biographical data of a sample of participants involved in the Moodle bug tracker, the bug tracker issues themselves and twenty interviews with participants. In this chapter the research problem is broken down into four sub-questions that look at: the characteristics of the community participants as a whole and the issues (i.e. the work) they engage in; the factors and processes important to the shared purpose of their work (issue resolution); the identity and roles of community members; and finally how identity relates to participation and community joining. These questions are related to the perspective of the researcher and the research approach adopted, which builds from a social constructivist perspective. The study is undertaken using the case study approach for: its use of mixed methods; its mechanisms for defining the research problem, or case boundary; and its establishment in the qualitative tradition of educational research.

Once the case study approach was adopted, the case itself was defined i.e. the research problem was formulated and three data sources were identified to address it: bug tracker member profiles, bug tracker issues and member interviews. The tools designed and selected to collect data from these sources are outlined in this chapter with particular focus on the interview analysis via the chosen coding methods and use of Nvivo software. How these tools could be ethically used with respect to the study participants was an important part of the research design outlined below. In addition to ethical integrity, the integrity of the study itself is examined and its reliability discussed.

3.2 The Research Problem

This study is defined by the following question:

What are the key practices of participation in open source educational software development, taking as a case study the Moodle bug tracker community?

This is examined by asking four research sub-questions:

1. *What are the characteristics of participants in the Moodle bug tracker and the issues they engage in?*
2. *What factors and related processes are important in the resolution of issues in Moodle?*
3. *What are the key identities and roles of the participants?*
4. *How do educators come to participate in the inner community of Moodle development?*

Two data sources are used to address the questions: question one is addressed through examination of data from the Moodle bug tracker itself and questions two, three and four from the interviews. The first question is an exploratory one: *What are the characteristics of participants in the Moodle bug tracker and the issues they engage in?* It draws on data from the bug tracker database to examine the characteristics of the participants, such as where they are from and their background. It also looks at how this may affect their involvement. This question also attempts to outline, in narrative form, what issues are. Sample issues are analysed and their salient characteristics outlined for the reader. This first question prepares the ground for the subsequent interviews and was used to formulate and then answer the subsequent questions.

Changing code to develop a software artefact is a social practice of a community as we have seen in Chapter Two. Research has shown how participants may enter such communities via established (but not necessarily explicit) joining trajectories (Von Krogh et al., 2003). A bug tracker may be a boundary object (Allen, 2009) that allows meditation between communities and may play a role in facilitating joining trajectories (Crowston & Howison, 2006). A notion crucial to communities of practice and perhaps under-explored in the literature of open source software as it relates to trajectories, is that of identities (Wenger, 1999). Individual identities become defined according to Wenger (1999) by the practices of a community and how they negotiate meaning as a group through these practices. The practice chosen to focus on in this study is that of “bug fixing” or “issue resolution”. This is the stated or canonical purpose of the Moodle bug tracker. Through this activity of fixing bugs/resolving issues the Moodle community is defined. Hence an examination of this practice is undertaken here to try to answer the second of our four questions: *What factors and related processes are important in the resolution of issues in Moodle?*

This type of question has an established research lineage as we have seen in Chapter Two, though it is mostly formulated in a narrower way, such as to find some specific determining factor or factors. Finding factors important to bug resolution in specific domains (such as with this example in software that attempts to model the domain of education) and trying to enumerate them without ranking them is important in itself (Hooimeijer & Weimer, 2007) and it should also provide the basis for other research. This question is examined here from the perspective of the beliefs of the participants, for such beliefs are the building blocks of their community. This question's second part looks at how such factors relate to each other i.e. by conceiving the enumerated factors as dynamic *processes* (or stories).
Similar to the type of the second question, of what factors are involved in issue resolution, the question of the identity of the participants is exploratory. What distinguishes the Moodle bug tracker community from others is that it requires a meeting of two communities: software development and education. Communities are defined by the practices of participants, by their roles, and the thus followed the third research question: *What are the key identities and roles of the participants?*

As the answers to these three questions build, the question can also be answered as to how individuals join and participate in these communities e.g. how a teacher might affect changes to Moodle via the tracker, or how a university might do the same. This leads us to the final research sub-question which encompasses the issues of identities, practices and joining trajectories: *How do educators come to participate in the inner community of Moodle development?* If the Crowston (2006) model of open source communities examined in Chapter Two were to hold in the case of Moodle we may also expect inner cores within the community.

This last question also tells us about the relevance and significance of this study. VLEs are changing the face of the higher educational landscape (Daniel, 2012). They exert (for better or worse) great influence, making the imperative to know about them and the details of their on-going genesis clear. Moreover, although many are opaque or unknowable, Moodle, as an open source project, provides a window into its development.

3.3 Perspective of the Researcher

The initial motivation for this research started when the researcher came across a particular bug in Moodle, following a conversation with a colleague who was using Moodle in their teaching practice as described in Chapter One. The research question thus emerged out of the researcher's direct experience, in their role of educational technologist and lecturer. The researcher is programme chair of a BSc. in Information Technology through distance learning in DCU's Distance Education Centre, Oscail and has served in various academic and eLearning roles on the programme since 2004. For the students and teachers of this programme, who have limited physically co-located contact, Moodle is effectively the campus. Given that the researcher has been using Moodle for many years in the service of distance education, the research subject is one that holds personal significance. This, together with a former life as a software developer in the eLearning industry, marks the researcher as a research 'insider' if pre-study domain knowledge is used as a barometer.

All is relative however and communities such as Moodle may have successive inner cores where the notion of 'insider' becomes ever concentrated. So, for instance, six of the interview participants had met the researcher face to face at least once before their interviews (though four of these meetings happened at one conference), whereas the majority of the interviewees were "cold-called" i.e. had no previous relationship or meeting with the researcher. The researcher also sat in on semi-open meetings of the Moodle developers, talked to various participants informally and interacted with them on social media. From that perspective, little attempt was made to distance the researcher from the participants of the study, and instead I regarded, and hopefully presented, myself as an interested peer.

3.4 Philosophical Underpinnings: Pragmatism and Social Constructivism

This research is informed by a social constructivist philosophical perceptive. In the field of education, philosophy is often incarnate, that is, lived and acted rather than merely theoretical. The research 'insider' often has the corollary role of affecting change in their domain, such as for example in action research or Marxist approaches (Loxley & Seery, 2008). For these researchers, research is a tool to activate a social or personal process. Philosophically this may be known as pragmatism, and at one end of the scale, such as the position of Rorty, may relegate any research that does not affect change to the status of mere 'wordplay' (Reason, 2003). Hence pragmatism is less concerned with truth than change; or rather, its truth is change. For this reason it has been seen as a form that is not beholden to one system of philosophy or reality. It follows that its adherents may, in the service of pursuing the research question, be "free to choose the methods, techniques and procedures that best meet their purposes" (Creswell, 2012, p. 23) – or at least that agonizing over whether certain approaches have epistemic validity may descend unnecessarily into metaphysics at which point the researcher would do better to "change the subject" (Rorty, 1982, p. 2).

Although there are many aspects of pragmatism that are germane to the approach adopted here this study does not have a strong or overt political aspect, that is, it does not directly seek to affect change. The philosophical position here taken however does accept that change *may happen* and more importantly that its possibility cannot be precluded. Any social artefact, such as a doctoral thesis, is both situated in and constituted from, the confluence of the actions and outpourings of a human network and inevitably itself becomes some small part of the sea. If we arrive at the position that "all narratives are, in a fundamental sense, co-constructed" (Salmon & Riessman, 2008, p. 80) then the researcher is some form of an agent for change, regardless of whether they have made such an agenda explicit. This human 'co-construction' is the basis of social constructivism. If one philosophy that inspires or explains the logic of this study is to be declared (as pragmatists might advise against) it is social constructivism. Creswell's outline of this philosophical worldview, as it is embodied in a research outlook, sums up well how the study at hand was informed and subsequently conducted:

> The goal of the research is to rely as much as possible on the participants' views of the situation being studied. The questions become broad and general so that the participants can construct the meaning of a situation, typically forged in discussions or interactions with other persons […] Constructivist researchers often address the processes of interaction among individuals. They also focus on the specific contexts in which people live and work, in order to understand the historical and cultural settings of the participants (Creswell, 2009, p. 8).

As meaning is social and arises from within community interaction, the researcher is inevitably generating meaning too. So although, as Creswell has it, the researcher relies as much as possible on the participant's views, the inquirer is nonetheless engaged in an inductive generative process when it comes to the production of the research findings. Reality is social and the researcher is not just studying social constructions but also engaged in them, such that "the process of bringing these realities into being is as one with the process of interpreting and reinterpreting them" (Crotty, 1998, p. 55). Social constructivism is hence linked to the theoretical perspective of interpretivism (Crotty, 1998; Creswell, 2009). Communities of practice (Wenger, 1999) lie within the broad church of social constructivist theories and also inform the theoretical perspective here. Concepts from the communities of practice model are used to direct the research and later to structure and construe data into findings at the coding phase as we will see.

3.5 The Case Study Research Approach

Yin (2009) describes a case study as something that arises from a particular type of research question. If the research question focuses on contemporary events; does not require control over those events; and asks either *how,* or *why,* something is; then a case study is appropriate. The adoption of a case study methodology may follow from social constructivism and interpretivism for several reasons, in particular for its being synonymous with both qualitative research but also mixed methods approaches (Yin, 1981; Yin, 2009). The case study also becomes a valid and useful approach here because of the nature of the phenomenon of study. The phenomenon is in many respects singular (single-case); and the phenomenon is unexplored, or an example of what Yin (2009) terms the “revelatory case”.

The adoption of ‘single-case’ study is also related to the perspective taken on generalisation. There are many problems with generalising in research, such as for example proving that the sample we are generalising from is representative of the population, leading Lincoln and Guba to conclude that “if there is a 'true' generalisation, it is that there can be no generalisation” (Lincoln & Guba, 1979, p. 39). In social contexts this problem is important because situations are complex, multivariate and mostly irreproducible. However, some events are not worth generalising about anyway. There are unique phenomena which cannot be replicated. For an action researcher these might comprise experiences of a particular teacher and his or her particular class, “unique, diverse and manifold” (Carr & Kemmis, 1986, p. 25). However, in our case, it can mean something large, dominant and unavoidable. When the OU conducted an evaluation of VLEs in 2006 it chose to adopt Moodle in part because Moodle had one of the largest and fastest growing user bases (Sclater, 2008). Moodle's success became a critical factor of its success and it has become the only viable open source VLE (with the possible exception of Sakai) (Costello, 2014). It is unique, so perhaps hard (or pointless) to generalise about, yet all the more vital to study and know. In this case a research methodology is needed for which, “such generality [as] it contrives to achieve grows out of the delicacy of its distinctions, not the sweep of its abstractions” (Geertz, 1973, p. 75), one that can show primarily the uniqueness of a situation over its generalness.

For Yin (2009), this development of Moodle would represent a good candidate for a single-case study. It fulfils the criterion of being an *extreme* or unique case. For unique cases, *descriptive* and *exploratory* approaches can be appropriate forms of a case study. Yin (2009) breaks down the types of questions such studies may ask. For exploratory approaches there are questions of “what”, “how” or “who”. In our case: *what* factors are important in the resolution of issues in Moodle?; *what* is the process of decision making in Moodle, as open source educational software?; and *who* are the actors involved (i.e. what are their identities and roles)? Explanatory research by contrast asks more *causal* type questions, including those beginning with “how” or “why”: and in our case we ask just one of these: *how* do educators come to participate in the inner community of Moodle development in issue resolution?

The second type of single-case study that applies here is the *revelatory case* where “the researcher has access to a situation previously inaccessible to scientific observation” (Yin, 2009, p. 49). Yin cites the example of *Street Corner Society,* a study of an Italian-American urban subculture, which has proved to be a prototypical social study whose findings have been replicated in many subsequent studies (Whyte, 1993 cited in Yin, 2009). This is not to say that this doctoral study will prove seminal, nor that it is any more unique or special than any topic must feel to a passionate researcher. However, it is the contention here that the development of educational software – which will itself have a growing influence on the future of education – has not been extensively researched to date. Moreover the fact that Moodle’s software is open source means that much of its development occurs in the open and is accessible to research. It is, perhaps, hiding in plain view.

3.5.1 Case Study and other Approaches

For Hammersley (1992) the definition of the case study, its boundary and what it can include, comes down to sampling. Experimental and survey approaches sample their data in a different way. In experiments, cases are created; in surveys, they are naturally occurring and large in number. Hammersley describes the selection of one research strategy over another as a trade-off. Experiment has the drawback of its artificiality – the researcher may construct an experiment that does not represent the world though it may yield interesting (and rigorous) results. By contrast the case study is anchored in “naturally occurring” data (though Hammersley does not mention that the selection of that data does seem to have parallels with the construction of the experiment). The main trade-off between experiment and case study is between control and reactivity for Hammersley. A researcher may be confident that they have eliminated sources of bias in an experiment and controlled for extraneous factors. However they must be certain before they run the experiment which factors are in fact extraneous, as unlike a case study they may be unable to react during the research to something they did not foresee at the design phase.

In comparison to the survey, the case study allows "greater detail and likely accuracy of information about particular cases at the cost of being less able to make generalisations to a larger population of cases" (Hammersley, 1992, p. 186). Hammersley grounds case studies in their commonalities with surveys and experiments, showing that their different methods exist on common continua, in this case the amount of detail they consider. The argument could also be made that a survey or case study can actually consider equal amounts of data but where they differ is in how information *is derived from that data*. Details in a survey may be abstracted into numbers whereas case study details may be summed into narrative.

A case study may allow a phenomenon to be conveyed (and not as such proved or formally defined). A crucial concept discussed by commentators on the case study is that of story-telling. Simons' (1980) exhortation "towards a science of the singular" may be seen as an alignment of the case study with narrative. A narrative must have its own internal coherency such as a beginning, middle and end (though not necessarily a wider consistency or generalisability). In this way the case study is communicable. It is a story. Its writing style may be "informal perhaps narrative, possibly with verbatim quotation, illustration and even allusion and metaphor [and while] themes and hypotheses may be important they remain subordinate to the case" (Stake, 1978, p. 3).

A case study may use a range of tools not only to gather data (interviews being often the primary data collection mechanism as here) but also to analyse it. Use is made here, described later, of coding to analyse the data. In this case coding does not proceed from a tabula rasa, as it does in certain schools of, but rather themes from the literature are more explicitly acknowledged as seeds to the coding process as we will see.

3.5.2 Case Study and Context or Boundary

Case study research is sometimes characterised as leading to "context-specific" knowledge (Flyvbjerg, 2006). That is, the context cannot be easily or meaningfully divorced from the object of study. In this sense the definition of the context itself becomes the boundary of the case. Although there is scope for emergent design where the precise boundary may be deferred until well into data collection or analysis, which can be termed a constructivist approach (Wells et al., 1995), this also contains the risk of falling into undirected (and unending) research. Instead the advice of Stake (1999) that the case boundary cannot be stretched arbitrarily, is followed here and the bounds of this case study were fixed following an exploratory phase and then remained in all important respects immutable.

The temporal boundary of this case goes from January 2007, close to the release of Moodle version 1.7.1 to the release of Moodle 2.0.2 in February 2011 i.e. three years of Moodle development. The population are those people who interacted in the bug tracker during that time. Within this population, distinction is made between those who can be assigned to issues (assignees) and those who cannot (non-assignees). The former are a much smaller population numbering just over 100 versus the 3,838 in the wider non-assignee community. Assignees are generally longstanding and active members of the Moodle bug tracker i.e. 'insiders' or core community members. Non-assignees are generally one-time or sporadic users of the bug tracker i.e. in crude terms 'outsiders' or wider community members. These are of course generalisations and one of the data collection phases detailed in Chapter Four involved profiling these two groups to build a better description and definition of them.

3.6 Data Sources and Collection Techniques

3.6.1 Tracker Participant Profile Data

The first research question – *What are the characteristics of participants in the Moodle bug tracker and the issues they engage in?* – was addressed with reference to data from the Moodle bug tracker. This tracker contains publicly available data and can be accessed from http://tracker.moodle.org. The first part of this question involved determining the basic available characteristics of a sample of the tracker participants i.e. where they were from, what they worked at, what their educational background was and their gender. (This also allowed for the subsequent research question three, on participant identities, to be framed and addressed in the interviews.)

Answering the first question involved looking at a range of community members in the bug tracker. Data was collected for 101 members of the population containing both core/assignees (45) and wider/non-assignees (56) on their:

- gender
- country of residence
- primary place of occupation
- educational background

Other salient data, that reflected some aspect of a person's involvement with Moodle, were recorded as memos e.g. the fact that a participant had written a book, or a dissertation on Moodle or their involvement with a particular part of Moodle. The data was collected and stored in a spreadsheet between December 2010 and January 2012.

Selection of individuals to add to this spreadsheet was driven by reading the tracker issues and selecting the participants who were involved in them. A tracker issue may relate to other tracker issues and these related issues were read by the researcher and their participants added i.e. proceeding in a dendritic pattern similar to how a literature review may be conducted where a paper is read, followed by the papers that reference that paper and so on. Initial starting issues to seed this process were ones which had attracted a lot of attention, including an issue about importing calendars to Moodle and an issue about negative marking in quizzes (both described in Chapter Four). Other issues were randomly chosen so as to include orphaned (ones not related to any other) issues and issues that did not have a large amount of participants. An example of one of these types of issues, relating to the Moodle Workshop, is detailed in Chapter Four. This data was analysed to generate descriptive statistics using Excel and the statistical software R.

3.6.2 Tracker Issue Data Collection and Analysis

In addition to the profiling of the 101 tracker participants, an analysis of the tracker issues themselves was also undertaken. This data was analysed to help answer the second part of research question one i.e. what Moodle tracker issues are, and to provide a grounding from which the remaining research questions could be developed. (It should be noted that all issues that fall under the bounds of the case study were analysed here whereas the trackeer participant profile data uses sample as mentioned above in the previous section).

Questions that measured the level of activity of different members of the community and the impact and type of their activity were hence asked of the data. The question of what issue resolution success factors are was also in part addressed by this data source. In particular, two factors – type of issue and type of person submitting the issue – could be tested for correlation with successful issue resolution. This data also provides a useful base for describing the case for the reader and for defining terminology that comes up in the interviews.

There were 20,830 tracker issues created during the bounds of this case. Data on these issues was retrieved by making queries to the Moodle bug tracker database, which is housed in a special bug tracking program known as Jira. This database is publically accessible on the web and any user can generate reports from it. The report generated included the following data for each issue:

- The name of the person who submitted the issue

- The name of the person assigned to (fix) the issue
- A description of the issue
- The type of issue (bug/fix or feature request)
- The date the issue was submitted
- The date the issue was resolved (if it had been)
- The number of votes cast for the issue's resolution
- The names of the individuals involved in the issue

To make sense of the large amount of data in this report, descriptive statistics were generated using Spreadsheet software and the statistical package R including:

- The number of issues submitted per user
- The top 20 issue submitters
- The number of votes cast in favour of the resolution of each issue
- The number of individuals involved in each issue
- The types of issue submitted

Inferential statistics were generated to examine whether core and peripheral members of the tracker community demonstrated significant differences in their influence on resolving issues and submitting valid issues. Chi square tests were run to test hypotheses of core/insider influence. The chi square test was deemed appropriate here as we could measure frequency data – number of valid issues submitted, number of issues fixed – and compare it across the two populations of the peripheral tracker user (non-assignee) and core tracker user (assignee). Although chi square does not does not give much information about the strength of the relationship, it is appropriate for use where the distribution of the population is non-normal (which was the case here as will be shown in Chapter Four) and where sample sizes are greater than 50 (Stevens, 2009).

These tests were run using the R statistical software package. R (R Development Core Team, 2013) is a free software environment for statistical computing and graphics available from http://www.r-project.org/. It is widely used in scientific analysis as the leading open source software alternative to statistical packages such as SPSS, SAS or Minitab. Given that this study focused upon open source software attempts were made where possible and feasible for the research itself to utilise open source software. Although R is not a typical GUI driven software, and requires some basic programming skills to properly use, it does present a robust, powerful and valid open source statistics platform (Ihaka & Gentleman, 1996).

In addition to describing issues in the aggregate, three particular issues are described, in the Findings chapter, in narrative form supported with screen-shots. These three issues serve to illustrate important issue characteristics that cannot be determined from the database reports, including how participants discuss issues in the Tracker comments and thus attempt to negotiate their resolution. These three issues are:

- An issue relating to an error in Moodle’s Workshop module.
- An issue relating to an error in negative marking in multiple choice questions in Moodle’s Quiz function.
- An issue relating to a request for importing a user’s calendar into Moodle’s Calendar function.

3.6.3 Interviewing

Interviews provided a crucial source of data, addressing the research questions two, three and four. The Moodle tracker data provided the main source for addressing the first of the research questions but also directly contributed to the interview data collection, both in helping frame the interview questions (for instance interviewees were asked why some issues stay unresolved for a long time – a fact uncovered from analysing the tracker) and also in developing a shortlist of prospective interviewees that were representative of the community.

Interviewing (along with participation observation) has proved a “staple” of education research (Brenner, 2006). Fundamentally, a research interview is a directed conversation (Lofland & Lofland, 1995). One way that research interviews are commonly classified is by structure and whether they are structured, semi-structured or unstructured. Here structure refers to the degree of license that the interviewer has to deviate from the interview schedule (Robson, 2002). Sticking to the script, by asking the same questions verbatim and in the same order of each participant, constitutes a high degree of structure. Highly structured interviews have also been characterised as having mostly limited response categories and less open ended question types (Fontana & Frey, 1994). Much structure, although it may lead to good comparability of the interviews, suffers from diminishing the role of the researcher and their capacity to innovate or improvise in what is a live (and possibly once-off) encounter. As such it is a more brittle form, and aside from issues of intimacy and mode of expression, not much different to a survey. At the far end of the continuum of interview structure there may be no schedule at all and simply a focused discussion on a topic. In between lies the semi-structured interview where a schedule exists, but where the research has some license to roam and where the researcher can "build on and explore their participant's responses" (Seidman, 2006, p. 15).

3.6.3.1 Design of the Interview Protocol

The form of interview devised was semi-structured. An interview schedule of fourteen questions was distributed to interviewees in advance. This aided in attaining agreement to participate from many interviewees. Some respondents treated the schedule as their homework and came to the interview very prepared. Although there were both upsides and downsides to these prepared answers, it did help put these participants at their ease. Fundamentally, the schedule helped establish trust (Seidman, 2006) and this allowed changes to happen during the interview, not as surprising deviations but as natural new tributaries arising from the flow of the conversation.

The interview schedule, given in Appendix A was informed by the first data collection phase of the analysis of the bug tracker which yielded a preliminary picture of the tracker and its participants. It was also informed from preparatory conversations with four of the eventual interview participants. This interview pilot phase was conducted during the 2010 UK Moodle Moot in London. The first question on the schedule asked participants how they came to be involved with Moodle. This followed the interview design principle of starting with a non-threatening, open ended question that puts interviewees at their ease (Brenner, 2006; Seidman, 2006; Knapik, 2008). This interview question was designed to address the research question of member identity. Prompts and follow-on questions explored their professional and educational background. A person's entry into the Moodle community was also hypothesised to be related to the process of issue resolution from entry trajectories reported in the literature as we have seen in Chapter Two.

Interview questions to address the second research question, of what factors and processes the participants believed to be important to issue resolution and the process of such resolutions, included:

- Factors the interviewee thought to be important in issue resolution
- Reasons interviewees believed that issue resolution might not happen or be impeded
- The interviewee description of the issue lifecycle

In accordance with the constructivist research approach participants were prompted for their own view (“what *you think* are the reasons…” “in *your experience*”). Elucidation of their beliefs was important in the analysis to build a picture of the participant world.

A question asking the interviewee to describe the role of the group (assignee or non-assignee) which they were *not a member of*, sought to address the issue of constructed identities of the community as per research question three. This issue was also probed by asking participants about their experience of other participants.

A final important question used to close the interview, asked participants what Moodle meant to them. This question was intended to allow the participant to orientate themselves in or relative to the community. This was to generate data to address the research questions three and four as to community identity and the process of individual involvement and participation. A full list of all the questions, prompts, and the themes they were designed to address, is given in *Appendix A The Interview Schedule*.

3.6.3.2 Sampling and the Number of Interviewees

During the research design the question of how many people to interview arose. Mason’s (2010) analysis of over 500 PhD theses, found the mean number of interviewees in qualitative studies to be 31. In his findings there were a wide variance of possible interview numbers and a large number that were multiples of 10 which suggested more pre-meditation than is often claimed in qualitative research sampling approaches. One systematic attempt to retrospectively find a “saturation point”, beyond which very little new data emerged from analysis, reported little new data after six interviews (of 60 in total) and none after 12 (Guest et al., 2006). Both numbers are lower than Mason’s PhD “norm”. Two points not addressed by Mason’s study are worth noting. The first is interview length – thirty hour long interviews constitute a very different volume of data to say thirty ten minute interviews – and the other is the unknown population sizes from which the samples were drawn. This muddies the issue, as is there is a lack of a discussion of confidence intervals or the uncertainty associated with a sampling method. This issue may ultimately boil down to the type of the research question. Hence, for the purposes of this study, where discovery and elucidation take precedence over generalisabily, a variety but not necessarily a particular number of views were required.

3.6.3.3 Snowball Sampling and Saturation

Case study sampling may have “an iterative or ‘rolling quality’, working in progressive waves’ as the study progresses” (Miles & Huberman, 1994, p. 29). Several of the interviewees recommended others to be interviewed. The concept of using participants to identify other participants is based on the assumption that the participants know more about the object under study than the researcher and will honestly lead them to a fuller picture. This “snowball sampling” can more quickly lead to a “saturation” point where enough data has been collected:

> In this form of sampling one identifies, in whatever way on can, a few members of the phenomenological group one wishes to study. These members are used to identify others and they in turn others. Unless the group is very large, one soon comes to a point at which the effort to net additional members cannot be justified in terms of the additional outlay of energy and resources; this point may be thought of as a point of redundancy (Guba & Lincoln, 1985, p. 233).

Using participants to identify other participants had the advantage of bringing in more wary respondents as participants allowed their names to be used as a reference. This was important in helping counter potential responder bias problems. Two other strategies to help counter bias were selecting members who had left the community and ones who had only had a minimal involvement in the community, both of whom might be likely to speak more freely.

3.6.3.4 Details of the Interviews and the Interviewees

Twenty interviews were conducted in all. A variety of different interviewees were sought. Twelve of the interviews were conducted with assignees, and eight with non-assignees. Four of the twenty were female and sixteen male. English was the second language of five of the interviewees. Three of the participants had left the Moodle community i.e. were no longer involved whereas the rest were still active members at the time of the interviews. 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 important for achieving variation and in aiding in the search for disconfirming evidence in the analysis (Kuzel, 1992). Key actors in the core Moodle bug tracker community who upon request were happy to be identified as participants included Micheal du Raadt, Tim Hunt, Helen Foster and Moodle founder Mrtin Dougiamas.

A summary of the interview formats are given in Table 21 below (Chapter Four profiles the tracker community, which these interviewees are part of, in greater detail):

Table 2 Interview Data Collection Summary

Code	Length	Transcript	Participant	Date	Format
M1	58 minutes	8,492 words	Assignee	22/02/2012	Skype audio call
M2	3 hours 15 minutes*	2,466 words*	Assignee	21/03/2012	Synchronous text Chat
M3	52 minutes	7,923 words	Assignee	28/02/2012	Skype audio call
M4	53 minutes	7,929 words	Assignee	30/01/2012	Skype audio call
M5	~50 minutes**	1,641 words**	Former Assignee	06/02/2012	Skype audio call
M6	38 minutes	4,391 words	Former non-Assignee	09/02/2012	Skype to mobile number audio call.
M7	28 minutes	4,570 words	Assignee	03/02/2012	Face to face recorded via iPad
M8	41 minutes	5,636 words	non-Assignee	30/01/2012	Skype video call
M9	34 minutes	4,257 words	non-Assignee	17/03/2012	Skype video call
M10	40 minutes	5,784 words	non-Assignee	23/01/012	Skype audio call
M11	26 minutes	3,463 words	non-Assignee	23/02/2012	Skype video call
M12	1 hour 20 minutes	7,967 words	non-Assignee	01/02/2012	Skype audio call
M13	38 minutes	3,686 words	Former non-Assignee	07/02/2012	Skype video call
M14	32 minutes	4,533 words	non-Assignee	09/03/2012	Skype audio call
M15	36 minutes	4,734 words	non-Assignee	07/03/2012	Skype to landline audio call
M16	37 minutes	4,078 words	Assignee	12/03/2012 –	Skype audio call

				14/03/2012 ***	
M17	42 minutes	3,898 words	non-Assignee	23/03/2012	Skype video call
M18	1 hour 14 minutes	9,651 words	Assignee	28/03/2012	Skype video call
M19	1 hour 8 minutes	7,993 words	Assignee	08/03/2012	Skype video call
M20	27 minutes	4,066 words	Assignee	19/02/2013	Face to face recorded via iPad
*Via synchronous text chat not voice audio. ** Audio recording failed *** Internet connectivity issue led interview to be carried out in separate parts over two dates					

Interviews were an average length of 53 minutes leading to an average transcription length of 5,358 words. The longest interview was 3 hours fifteen minutes and the shortest was 27 minutes. Interviews were mostly conducted via Skype if remote, or via an iPad if conducted face to face, and the audio recorded. In one instance the recording software malfunctioned. In this case a lot of notes had been taken during the interview and, as it was discovered directly after the interview that the recording had failed, more notes were then taken from memory. Nonetheless the notes were much less than a full transcription would have been and there was less data in this instance. One participant requested to be interviewed via text chat rather than audio conversation which resulted in a very different interview; one which though it took a long time to conduct, resulted in a relatively parsimonious transcript (and where the transcript constituted the actual interview). There was a mix of participants using the video function of Skype and those making audio only calls. There is research to suggest that video interviews may correlate with lesser levels of disclosure by interviewees (Brunet & Schmidt, 2007) but the choice was left up to the participants as to their preferred mode of audio with video, or audio only.

3.6.3.5 Interview Transcription and Presentation

Each interview was transcribed verbatim including all the interviewer utterances and including any speech disfluencies (ums, ahms, stuttering, repetitions etc.) in order to arrive at an "orthographic transcript" which Braun and Clarke (2006) determine to be a requirement of textual analysis. Transcripts included any of the interviewer's interjections and these were distinguished from the interviewee by bolding their text. The fidelity of the interviews was thus preserved for coding based on the interview as an interaction between researcher and participant, rather than attempting to whitewash the researcher's input and effect, but also to provide maximum context which allowed the researcher to "relive" the interview through the text (Seidman, 2006). Samples of excerpts from the interview transcripts are given in Appendix E. A second copy of the transcripts was then created, using MS Word to search and delete all incidents of bolded text. This gave an *in vivo* version of the transcripts and this was used, for example, to search for particular words spoken *only* by the interviewees using the text search function in Nvivo. It could also be used to determine if the researcher had seeded a conversation with a particular word, or relatedly, if a word or phrase was a true emic synonym such as "component lead" and "component maintainer" proved to be.

Individual interview participants are anonymised and are identified by a code when cited in the following chapters e.g. M1, M2 etc. as outlined in Table 21 Interview Data Collection Summary above. To preserve this anonymity certain participant quotations are sometimes identified as simply either an assignee view or a non-assignee view. Several interview quotes are attributed directly to specific interviewees in cases where the interviewee had waived their right to anonymity and where, in the context of the findings' presentation, identifying them was deemed important.

3.7 Interview Analysis

The interviews were analysed using coding in the Nvivo software package and were then organised in a largely narrative form as presented in Chapter Four with much use of direct interview excerpts to foreground the participant voice in the story of the findings. In addition to the narrative form, visual representations are also presented as the result of plotting issue resolution process graphically via flow charts.

3.7.1 Nvivo Qualitative Data Analysis Software

Nvivo is a software package widely used in qualitative data analysis. It may aid the researcher to manage data and ideas, query data and also to create models or other reports from the data (Bazeley, 2007). Care must be taken that the software does not somehow start to direct or dictate the research. However such fears as Gilbert (2002) points out are really larger ones about the research process itself. After all, any method, whether it uses software or not, still represents a technology the advantages of which may be to bring rigour and transparency to a process (Bazeley, 2007). Gilbert's (2002) advice for using software like Nvivo, such as – writing memos, maintaining a focus on the research questions and coding systematically for specific research-related themes – are thus good general practices for carrying out analysis. Software then, like Nvivo, for its functions of memoing and search (the use of which is described later), and its ability to arrange and display data and questions together was hence chosen to aid in the analysis. Its ability to get the researcher closer to his/her work processes, referred to by Norman (2002) as the "affordances" that a tool provides, were deemed valuable after trialling the software and undergoing training in it.

3.7.2 Coding

As a method, coding is a process of abstraction, and resultant codes "empower and speed up analysis" (Miles & Huberman, 1994, p. 65). Coding acts on data and also acts to transform data as it progresses (when the codes may be themselves coded). Coding starts by "fracturing" the data, organising it into categories that relate to each other and aiding comparison of the data (Corbin & Strauss, 2008). Unlike quantitative coding, qualitative coding is not simply a process of enumeration but rather one of meaning making as the fractured data is then further coded to ultimately aid in the development of theoretical concepts.

In many approaches there are (at least) two phases or cycles of coding. Broadly, a first pass generates some initial summation of the data. The resultant codes then effectively become part of the data and the following cycle acts upon this enhanced data corpus in a deeper analysis and synthesis of it. At each stage various types of coding strategies may be employed. Saldaña (2009), for example, outlines 25 discernible forms. However, usage of one coding method may not necessarily preclude the use of another and there may be overlaps between these methods.

3.7.3 First Cycle (Open) Coding

In the first cycle of coding, used to generate what in Nvivo are termed 'free nodes', there were three related basic methods employed in tandem: *descriptive* coding, *in vivo* coding and *process* coding (Saldaña, 2009). *Descriptive coding* (sometimes known as 'topic coding') can be used to create a basic vocabulary of the data. This is a base level of code that any coding scheme usually employs. The researcher uses their own terminology for the code titles. *In vivo* codes, by contrast, are defined as "concepts using the actual words of the research participants rather than being named by the analyst" (Corbin & Strauss, 2008, p. 65). They are attempts to uncover the participant diction that mark them as members of a group. *In vivo* coding was used here to identify terms that were common to participants in describing their world, or terms used by a participant that were particularly illustrative of a phenomenon. As such it was employed as an emic approach that arose as a consequence of the use of the community of practice model and from operating from a constructivist research perspective. Although *in vivo* codes necessitate giving authentic voice to the participants, and exploring their constructed worldviews, they also involve proactive interpretation:

> Interpretations are sought for understanding the actions for individual or collective actors being studied. Yet those who use grounded theory share with many other qualitative researchers a distinctive position. They accept responsibility for their interpretative roles. They do not believe it sufficient merely to report or give voice to the viewpoints of the people, groups or organizations studied. Researchers assume the further responsibility of interpreting what is observed heard or read (Strauss & Corbin, 1994, p. 275).

Finally, elements of *process coding* were also used. Process codes describe activity (Saldaña, 2009). Here we follow Charmaz and Glasser's method of identifying process codes as those created with greunds (the form of verbs ending in "ing") (Charmaz, 2006; Glasser, 2008). This coding was chosen to allow for momentum to be elucidated. Many of the interview questions centred on processes, of resolution of problems, and also of the dynamics of entering and leaving the community. So for example one of the specific research sub-questions – what is the *process* of issue resolution – derived, unsurprisingly, several process-type codes e.g. "Decid*ing*", "Wait*ing*". This method of coding was also useful for research question three that sought to examine who the community are, via the constructed identities of its participants, arriving at codes such as "Identify*ing* as teacher" and "Identify*ing* as Developer".

When moving between the three coding types employed, the initial coding was guided by the rule that it should "stick closely to the data" (Charmaz, 2006, p. 47) and a description of each code was made as it was formulated. Initially the three coding tacks were kept distinct by creating three categories and assigning codes to them. However this was revised after it became clear that a code could belong to more than one category. For example "necessity" was an in vivo word that described involvement with particular aspects of Moodle: "necessity controls my involvement…". "Necessity Controll*ing* Involvement" became a useful process code that retained some of the emic language but was more applicable as an etic generalism. Thus the initial coding scheme was iterated by reference to the three principles of in vivo coding, process coding and descriptive coding.

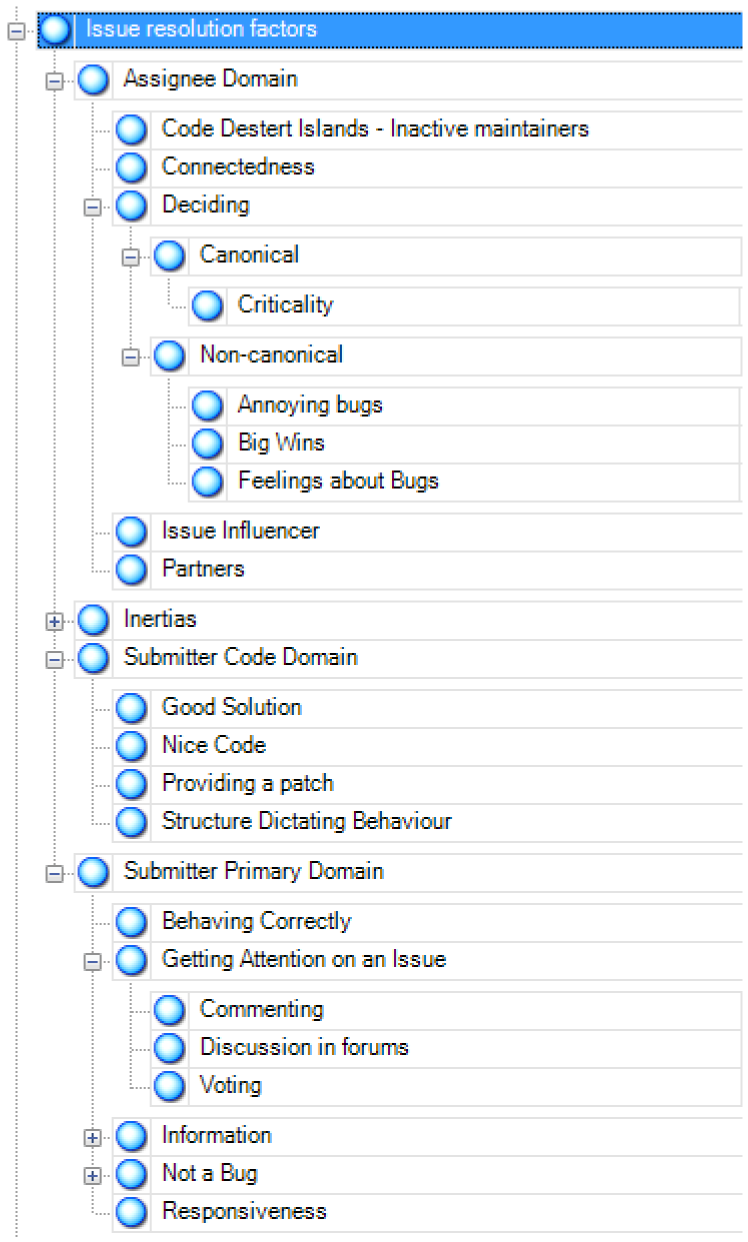
The codes themselves were largely derived from the analysis of the data, although the interview questions inevitably provided hooks and the literature review, memos and interview notes were periodically cogged during this period so that they were loaded up in the researcher's mind. Hence, although no codes were generated a priori, i.e. an inductive approach was followed, their creation was well primed. As new codes were developed they were checked for coherency with the overall pool of existing codes to ward against the evolution of what Miles and Huberman graphically term, a potential “ratbag” of codes (Miles & Huberman, 1994, p. 62). For example the code “Identifying as Teacher” was found to have a twin “Identifying as Developer” and a related code on this emerging concept of identification was “Becoming”.

Nvivo’s text search function helped facilitate the constant comparative method (Miles & Huberman, 1994; Corbin & Strauss, 2008). For instance, *Deciding* emerged as a level one code (Nvivo free node) representing participants’ views around the decision making process in issue resolution. This code was prominent in three interview questions that dealt with: the issue lifecycle; the role of the core community; and the factors leading to successful issue closures. However, it also appeared sporadically elsewhere in other question answers as it was such a key concept. Nvivo was hence used to search for the word “decide” and also to automatically detect related stemmed terms such as “deciding”, “decided”, and also synonyms such as “resolve”, “resolved” etc. Techniques such as this proved a powerful augmentation to the process of the researcher manually combing the text and aided in the eventual bringing about of the saturation point beyond which the data refused to yield any more codes (Corbin & Strauss, 2008).

3.7.4 Second Level Coding

Themes, causes, relationships between people, and emerging theoretical constructs are the four main types of meta-code used by Miles and Huberman (Miles & Huberman, 1994) at the secondary coding stage. These map to the four research questions which are based on enumeration of the issue resolution success factors, the process of issue resolution, the identities and roles of the community members, and how identities are defined by joining and participating. Corbin and Strauss’ (2008) finding that second level coding (in their case termed axial coding) should not wait until the first phase is complete was borne out, as several higher level conceptual codes were thrown up during the first or open coding stage. The codes were re-factored at the end to clearly distinguish lower level, more primary codes (Nvivo free nodes) and group them under secondary ones (Nvivo tree nodes) i.e. the secondary coding phase involved both the generation of new codes but also the reordering of existing ones to create a conceptual hierarchy. (In Nvivo this process is termed "coding on".)

The research question relating to the factors involved in successful issue resolution yielded many codes as in Figure 2 below:



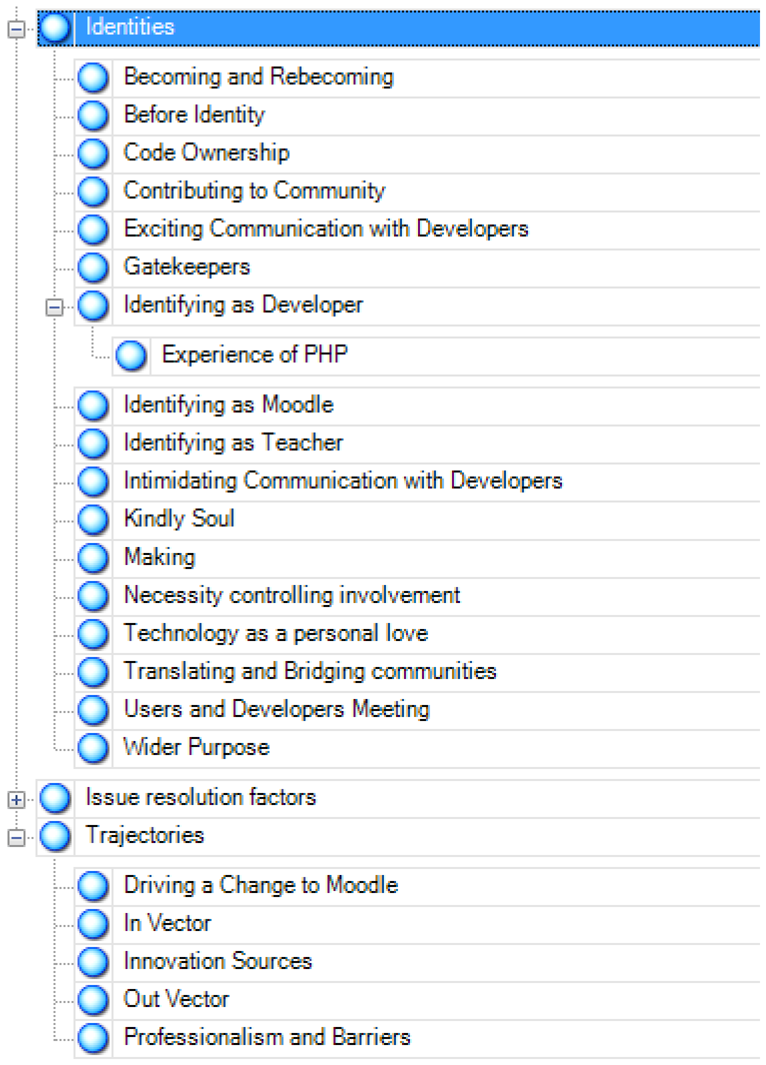
- Issue resolution factors
  - Assignee Domain
    - Code Destert Islands - Inactive maintainers
    - Connectedness
    - Deciding
      - Canonical
        - Criticality
      - Non-canonical
        - Annoying bugs
        - Big Wins
        - Feelings about Bugs
    - Issue Influencer
    - Partners
  - 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 2 Snapshot from Nvivo of Issue Resolution Success Factors Coding

These codes were arranged into graphical displays both to group them thematically and also to show their relationship to each other and hence answer the question relating to the *process* of issue resolution. The development of display charts, described in Chapter Four, was important to the analysis for forcing “a more inferential level of analysis that pulls together the data into a single summarising form” (Miles & Huberman, 1994, p. 160).

In eliciting data on what factors were deemed important to issue resolution, two related questions were put to interviewees asking them what factors might lead to issues being resolved; and why issues might not be resolved. This method – asking essentially the same question in two different ways – proved particularly effective as it helped to draw further information from participants. It also served to provide counterfactuals: one person might believe a particular factor to be important to the successful resolution of an issue whilst another might cite that same factor as a cause of the protraction of issues. (A full list of all the factors identified in the initial coding phase is included as Appendix D.

Two other important high level coded themes related to *identities* and *trajectories* as represented in the coding below in Figure 3:



- Identities
  - Becoming and Rebecoming
  - Before Identity
  - Code Ownership
  - Contributing to Community
  - Exciting Communication with Developers
  - Gatekeepers
  - Identifying as Developer
    - Experience of PHP
  - Identifying as Moodle
  - Identifying as Teacher
  - Intimidating Communication with Developers
  - Kindly Soul
  - Making
  - Necessity controlling involvement
  - Technology as a personal love
  - Translating and Bridging communities
  - Users and Developers Meeting
  - Wider Purpose
- Issue resolution factors
- Trajectories
  - Driving a Change to Moodle
  - In Vector
  - Innovation Sources
  - Out Vector
  - Professionalism and Barriers

Figure 3 Identities and Trajectories

The themes of identities and trajectories mapped to research questions three and four that addressed respectively: *What are the key identities and roles of the participants?* and *How do educators come to participate in the inner community of Moodle development?* So, for example, the concept of *brokers*, who from the literature operate between boundaries to help people into deeper layers of a community (Wenger, 1999), was a theoretically informed first cycle code called *Translating and Bridging Communities* which was then grouped under the theme of *identities* (along with other codes for people who identified as *teachers* or *developers*). Thus the coding converged back to themes from the research questions. More extensive examples of the coding as it developed are given in the two figures in Appendix C.

Although the coding was thought to be completed by the writing of the Findings chapter, in effect it continued. The act of writing became an extension of the analysis and the codes were accordingly reopened and revised during the writing stage when data that looked fine in a coding tree did not necessarily translate into a written narrative:

> It seems, in fact, that you do not truly begin to think until you attempt to lay out your ideas and information […] you are never truly inside a topic – or on top of it – until you face the hard task of explaining it to someone else (Lofland & Lofland, 1995, p. 142).

3.8 Reliability, Credibility and Ethical Considerations

3.8.1 Ethical Considerations

Intruding into the lives of others as a researcher, even if they are non-vulnerable adults and have consented to be interviewed, nonetheless presents ethical issues. Although Gomm (2004) found that the vast *majority* of social research is relatively benign – doing for the most part neither harm nor good to the participants in a significant way – ethical considerations must nonetheless be an essential part of the research design and carefully formed before data collection can begin. A primary guide was the Research Ethics Policy of the School of Education in Trinity (Trinity School of Education, 2009) which sets out the following principles to be followed by researchers:

- a commitment to the well-being, protection and safety of participants
- a duty to respect the rights and wishes of participants
- an evaluation of the relative benefits of any research to groups and individuals
- a responsibility to conduct rigorous, academic research
- a commitment to disseminate the results of research in an honest and truthful manner to all who may be affected by the research or those who should be informed about the research

Although this study was deemed to be a low risk i.e. involving non-vulnerable adults, it nonetheless addressed the concepts of informed consent, anonymity, data storage access, and data retention and destruction. A copy of the email template sent to prospective interviewees containing the ethics/privacy statement is given in Appendix B. All interviews were conducted with the offer of anonymity. Specifically, prospective respondents were informed that their answers would be “anonymised and aggregated with other data” i.e. as the community is relatively small, and some people could publicly disclose that they had been interviewed, the prospect of being identified as a participant was not completely under the researcher’s control. However, the important point was that no views were directly attributed to an identifiable person unless they had consented to this. The convention used in citing the interview quotations was to attribute a random code assigned to each interviewee, e.g. *(Interview M13)* or, where their role might make them more identifiable, simply *(Assignee* *view)* or *(non-Assignee view)*. In the main all respondents were very open and there was very little overtly controversial that arose, apart occasionally from respondents wishing to go off the record to discuss some aspect, generally of a business nature. These sections were not transcribed from the audio. Interviewees were emailed a copy of the transcript of their interview as soon as it was available providing an opportunity for them to raise any concerns, to add anything, or to withdraw.

Assurances to participants were made as to the security and anonymity of all stored data in accordance with best ethical practices and with Irish law on data protection (Government of Ireland, 1998; Government of Ireland, 2003). All of the data from the interviews was stored on a laptop which remained either in the researcher's home or workplace. A password protected and encrypted cloud backup service was used to keep a backup of this data.

All other data used in the study can be considered to be in the public domain i.e. the Moodle bug tracker database itself. However, performing research on information that is already publically available may still pose ethical issues. With increasing use of social media this represents a rapidly evolving field of ethical research, or, as the authors of the Association of Internet Researchers (AoIR) report into ethical conduct put it: "Social, academic, or regulatory delineations of public and private as a clearly recognizable binary no longer holds in everyday practice" (Ess & Jones, 2012, p. 7). The AoIR recommendations that researchers check the policies of websites for particular usage restrictions were followed here. The Moodle discussion forums have for instance detailed guidelines for their use but have nothing relevant to say about using or reproducing data. In using data from the public domain, the principles of identifying vulnerable groups and risks to participants were also applied and found not to be major concerns, though recognition was also made that those actors still have a right to "contextual integrity" achieved through cognisance and respect of social rules and norms, with both local and general values, ends, and purposes (Nissenbaum, 2010).

3.8.2 Reliability and Credibility

A case study should tell a unique story. However it is not a work of art, nor a work of journalism. Although its findings, as with most studies (Gobo, 2008), may be irreproducible in practice, its *workings* must still nonetheless be described and justified. Fundamentally, it should demonstrate that whatever claims are made of it have been stood "in the fierce light of free and fearless questioning" (Clifford, 1879, p. 184). In effect this is an extra, and essential, ethical obligation which the researcher must uphold.

Validation has been referred to by Lincoln and Gubba (1985) as a process of verifying the “credibility” and “trustworthiness” of findings. Creswell (2012) outlines why contributors such as Lincoln and Gubba amongst others have sought to affect a new language for the description of ways in which a qualitative study should defend the value of its findings as waters may be “muddied” by relying on ideas from positivist paradigms. One of the most common methods of ensuring the value of findings in qualitative research is triangulation. Although we need to be wary of taking triangulation literally – such as in using ideas from geometry to arrive at a magic number of three as the required minimum of sources in qualitative research – using more than one data source is nonetheless important. We have noted above that the data sources used here served to augment each other – specifically that data derived from the Moodle bug tracker was used to help develop the interview questions and frame the resultant data. There was also a process at work, as we shall see in Chapter Four, where certain interviewee claims are countered with alternative explanations from the bug tracker data. This can be a Popperian search for disconfirming evidence, or as in case study portrayal, the bringing of more than one evidence source to bear in the service of making the object of study more context-dependant rather than less so (Flyvbjerg, 2006).

A related form of triangulation is multiplicity of evaluators, particularly the participants themselves. To this end the participants were invited to give any comment on their transcripts (which for the most part they did not), but also the interview with Martin Dougiamas (and other conversations with him) were used to “test” some of the researcher’s conclusions of the analysis to that point. For example Dougiamas had a different take, or at least could explain more clearly, the process of Moodle insiders’ writing books about Moodle (books emerged as an artefact correlating to a person’s trajectory into an inner Moodle community in the analysis). His views on the evolution of the Moodle codebase, and the framing of the use of particular development tools as a social problem, helped confirm and enrich conceptualisations that the researcher had formed at this stage. This process is referred to by Yin (2009) as “theory triangulation” where multiple perspectives are brought to bear. It should be noted however that not all commentators are in agreement that this is a good strategy, as it may unnecessarily introduce bias of participants who “may or may not understand the [researcher’s] theory, or even like the theory if they do understand it” (Glaser, 2008, p. 25).

Another of the tenets commonly cited for the establishment of the credibility of a qualitative study is “prolonged engagement” with the data (Creswell, 2012). The well-developed and well-specified tools of coding (and the associated tool framework of the Nvivo software) proved invaluable in this regard, for establishing an eventual “critical way of seeing” brought about through “numerous cycles through a little bit of data, massive amounts of thinking about that data and slippery things like intuition and serendipity” (Agar cited in Corbin & Strauss, 2008). This study draws on the coding process for “exhausting” the data (Strauss & Corbin, 1994). Coding is a communion with the data and an analogy here may be Jung’s description of the obsessive nature of medieval alchemists: “The real nature of matter was unknown to the alchemist: he knew it only in hints. In seeking to explore it he projected the unconscious into the darkness of matter in order to illuminate it” (Jung & Hull, 1968, p. 144).

For Corbin and Strauss a key aspect is that there “should be sufficient detail for the reader to vicariously feel they are in the field” and hence be able to make judgements for themselves about a study (Corbin & Strauss, 2008, p. 300). There are two ways to interpret this advice. On a literal level we should foreground the participant voice in the findings, ensuring that their world is portrayed in their words. This strategy is followed in the Findings chapter that follows. Of course the researcher is still selecting the quotes, and so to this end the second goal, of allowing the reader to make their own judgements, is attained by including as much detail as possible about the researcher’s decision making throughout the process. That is, the rigour of the research process cannot be an addendum but rather should be ever present (Maxwell, 2004). Examples here include weighing the pros and cons of the semi-structured interview, how the interview schedule was piloted and evolved, how the participants were selected from wider populations, how those populations were conceptualised and what sampling considerations went into this design. These design conversations should be shared with the reader as the study progresses not just to show that particular academic processes have been followed, but also more fundamentally as a reminder that there is a researcher present. Someone is there, complicating matters. This is the difference between the artist or journalist and the researcher: the latter must always make their presence clear – even at the price of rendering the ensuing final artefact less exciting than one that may appear to have begotten itself.

A final point should be made at this juncture. It is worth asking what the study would have been like without the interview with Moodle founder Martin Dougiamas. With so many references to him from others, would the study have been “valid” without his chance to put his side of the story? Although his interview has been cited as a capstone and triangulating source, it is the researcher’s considered opinion that the eventual outcome would have been different with his exclusion but not necessarily any the lesser. It would simply have been a different story. In this respect the researcher is in agreement with Corbin’s view on "credibility" that it "indicates that findings are trustworthy and believable in that they reflect participants', researchers' and readers' experiences with a phenomenon but at the same time the explanation is one of only many possible 'plausible' interpretations from data" (Corbin & Strauss, 2008, p. 302). A key concept in establishing the legitimacy of a case study is thus fidelity and it follows that the onus falls on the researcher for the development of their truth of the phenomenon. As Jung put it:

> Thus it is that I have now undertaken [...] to tell my personal myth. I can only make direct statements, only "tell stories." Whether or not the stories are "true" is not the problem. The only question is whether what I tell is my fable, my truth (Jung & Hull, 1968, p. 13).

3.9 Conclusion

This chapter has outlined the researcher's perspective and the relationship of this study to the philosophies of pragmatism and social constructivism. From this a single-case study approach was adopted, one where mixed methods could be used to address the key questions of how Moodle bug tracker issues are resolved and how particular people come together to affect this process. A profile of the people involved and a profile of the bug tracker issues for a three year period provided the grist for 20 interviews and also to orientate the views of the interviewees within their world. The interview transcriptions were analysed through the process of coding which played a key role in the immersion of the researcher in the data and in developing constructs such that the findings, detailed in the next chapter, could be developed.

4 Findings

4.1 Introduction

This chapter deals with this study's findings and the path to those findings, the analysis. The evidence strands for addressing the research questions were twofold: the first data source was the Moodle bug tracker itself from which a sample of its members were drawn and also a sample of issues lying within the bounds of the case study; the second data source was the views of participants from the interview stage that followed (and which itself was informed by the bug tracker analysis). The format introduces findings firstly through key descriptive statistics of the community members; secondly through looking at the issues from the tracker themselves, which introduces key concepts; and thirdly through the researcher's account of the interviews as derived from the coding.

The order of presentation of the data sources was designed to orientate the reader; so that by the time the examination of the interviews occurs a clear picture will already have been painted of the people are who are involved and also what the bug tracker issues are composed of. The four research questions are the four main sub-headings of this chapter. The first question explores the characteristics of the tracker community participants and the bug tracker issues (the units of the work they are mutually engaged in). This exploration provides the foundation for posing and addressing the subsequent research questions. The findings from the next three questions, addressed by the interviews, are then presented: the factors believed important to Moodle bug tracker issue resolution and the processes of these resolutions; the perceived roles and identities of the participants; and how members come to participate in the community.

In order to help illustrate the characteristics of the participants of the Moodle bug tracker community as a whole, use was made of descriptive statistics and charts. The presentation style becomes more narrative as the findings from the interviews are presented and the participant voice is allowed to come to the fore through selected quotations. These quotations have been derived in turn from the coding.

The second research question, which is addressed by the interviews, asks which factors participants believed to be important to issue resolution. This analysis breaks apart the key avowed practice of the community – bug tracker issue creation and resolution – and examines its constituents in detail. The factors of issue resolution are grouped by first viewing them according to the perspective of a submitter (i.e. an 'outsider'), next from the perspective or domain of the assignee (someone with influence or an 'insider') and finally from the perspective of participants who have written code to solve an issue. The processes that link these factors and these three domains are outlined and illustrated graphically via display charts.

The next set of findings relate to research question three, regarding the identities and roles of the participants and these are presented according to subthemes uncovered by the coding including those of *developers*, *teachers* and *identifying (or not) with Moodle* itself. Finally, the ways participation was initiated and developed is examined in addressing research question four, which looks at how educators come to participate in the Moodle community. This question links many of the concepts elucidated in the preceding questions and explores the momentum of people into and within the community in the pursuance ultimately of the development of new knowledge through changing the Moodle code.

4.2 What are the Characteristics of Participants in the Moodle Bug Tracker and the Issues they Engage in?

4.2.1 Introduction to the Tracker

As we saw in Chapter Three, the case bounds of this study are people who have participated in the Moodle bug tracker between 2007 and early 2011. As the tracker is designed as a data gathering and reporting tool it is possible to generate reports by pulling datasets directly from it. In this way a snapshot of all interaction in the tracker was generated from the tracker database for the period. Analysis of the first research question – *What are the characteristics of participants in the Moodle bug tracker and the issues they engage in* – provides a landscape for the reader in which to situate the remainder of the research questions. It also provided a basis for the researcher from which the interview stage could be developed.

For the purposes of the study, the participants in the tracker were classified as either belonging to a core/inside Moodle community or a wider/outside Moodle community. One readily available way to identify these communities within the tracker is by examining whether a given user is a potential assignee to an issue. In the dataset, and in the findings that follow, *an assignee is defined as someone who has been assigned to at least one issue in the tracker* i.e. they have been designated as the person who should decide if the issue can be fixed or not and if so, how it will be done. For example if a teacher had a problem with the way negative marking of multiple choice questions works in Moodle they could file an issue in the tracker. This would be assigned to a particular *assignee* for review which might ultimately lead to that assignee resolving the issue for the teacher. A submitter can designate an assignee when they submit their issue. Alternatively an assignee, or someone from Moodle HQ, can assign someone as the assignee.

The process of how and who assigns issues in the Moodle bug tracker has changed over time. In terms of this research however, what is important is that an assignee is designated to have some power (knowledge) to solve the issue. Assignees are thus influential people within Moodle. A person can only become an assignee after being appointed by one of the core Moodle community members, typically by spending a lot of time on the other side of the fence submitting issues and, most importantly, related code fixes. Within the assignees there may be those with more perceived influence or importance than others. Although it is arguable that assignees now have less power to directly change Moodle than they did at the time this study began, they nonetheless represent a good proxy for Moodle community insiders, or more precisely, for persons with some influence in Moodle tracker issue resolution.

4.2.2 Tracker Participant Profile Sample

In order to explore the profile of the members involved in this community, an appraisal was first undertaken of their overall demographics. As discussed in Chapter Three, this helped inform the selection of interviewees, as it was an important step in conceptualising, in general terms, the community makeup. It served to both contextualise and focus the remainder of the study. In particular determining community member background was important in later exploring the constructed community identities of interviewees and how their involvement with the community came about.

Between 2007 and February 2011 there were approximately 106 assignees in the tracker. This figure may be considered as closer to 100 as accounts such as, *test user*, *nobody* and *Moodle.com*, do not correspond to defined people (whereas almost all of the other accounts in the tracker are in a person’s name). The ~100 assignees are dwarfed by the 3,838 non-assignee participants in the tracker during this period. To establish a picture of the participants in the tracker that are covered by the bounds of this study, a sampling of 101 of them was undertaken. This sampling involved profiling their country of origin, their primary place of occupation, their educational background and their gender. Other salient contextual information uncovered during this data collection phase was also recorded and the results were collated in a spreadsheet.

4.2.2.1 Country of Residence

As there are 101 people represented in this sample all the charts and tables that follow can be read either as absolute numbers or percentages of the sample. All charts and statistics were generated with Excel (descriptive) or the open source statistics package R (inferential). The sample focused mostly on the assignees, profiling 45 of them, as opposed to 56 of the non-assignees. Although the latter are a much larger population, the former are the more influential and the more engaged in the community and were thus important candidates to target for the later interview phase. Consequently the following findings are statistically significant when talking about the assignee population, though the non-assignee sample of 56 from a population of 3,826 gives a large confidence interval of 14% at the 95% level of significance. Beyond this note of caution the following data does give an insight into aspects of the basic makeup of this community. It hence addresses the research question of members' characteristic makeup and lays the foundation for exploring this area further in the interviews where the theme of participant identity is examined.

Determining the country of residence for the profile sample was relatively straightforward and reliable for almost all the sample members. This is because participants also have a profile on the Moodle.org discussion forums and this profile includes country as a mandatory field. Figure 4 is a map showing the locations of the 101 participants sampled for this study and Figure 5 shows the most commonly represented countries. The map was generated by importing a Google spreadsheet of the collected country of origin data into the Google Fusion Tables tool and plotting them by geo-location.

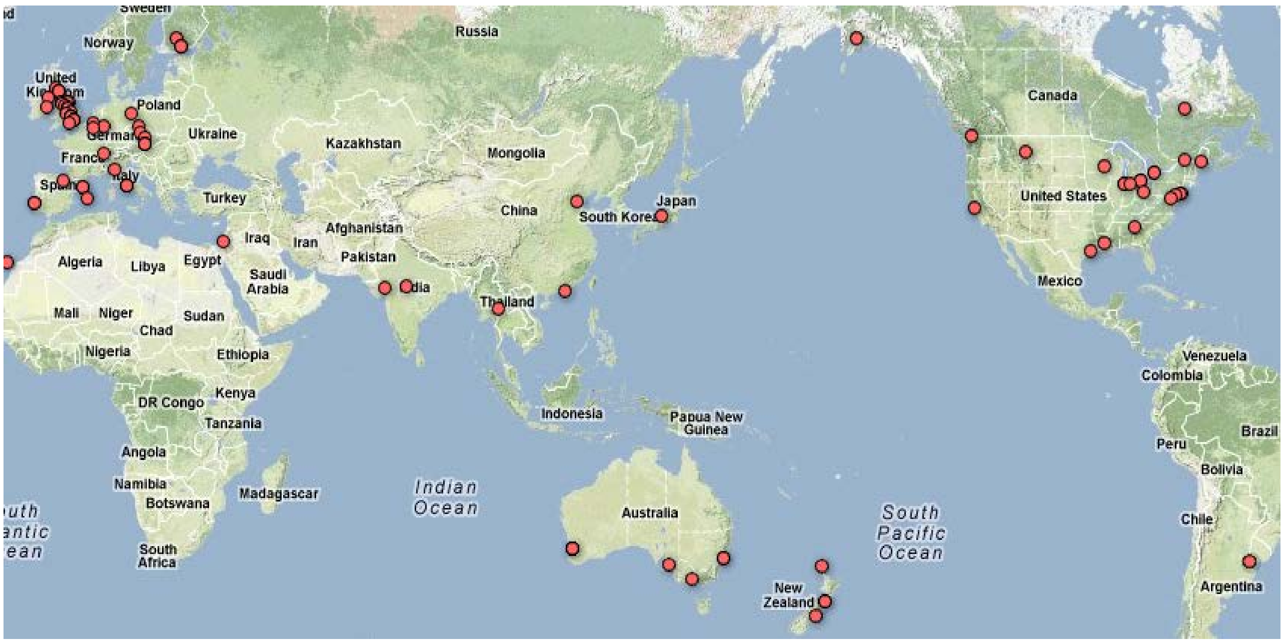


Figure 4 Map of Country of Residence of 101 Tracker Participants

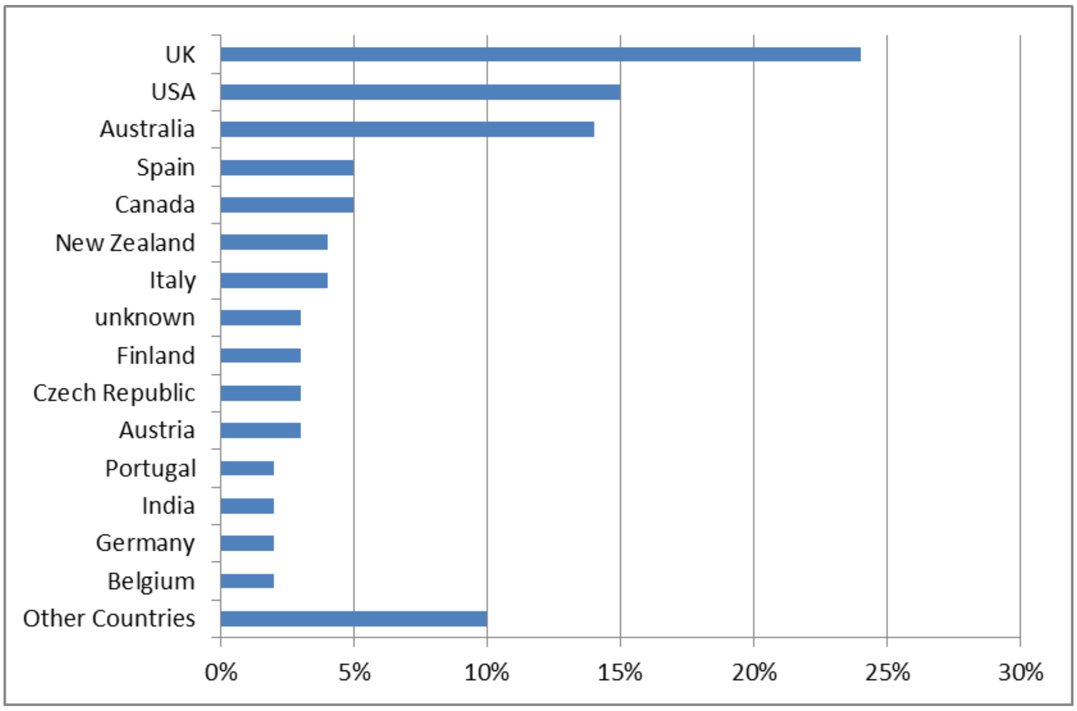


Figure 5 Countries of Residence Ranked of 101 Tracker Members

The above graph in Figure 5 shows that tracker members are concentrated in Europe, North America, Australia and New Zealand. This reflects some aspects of Moodle's overall worldwide distribution. It should also be noted that the sample was not completely random and, as previously outlined in Chapter Three, was arrived at by reading particular bug tracker issues and the participants in them. It is worth noting that English is the official language of the countries of 62% of the sample i.e. there may be a significant portion of tracker participants for whom English is a second language.

4.2.2.2 Tracker Participants Place of Work

Determination of where the participants in the tracker work was important for understanding something of the other communities they may be involved in. Many participants had an entry in their profile on the Moodle.org discussion forums where they stated their place of work or provided their work email address. For profiles where this information was not available, a Google web search for the participant often established their place of work such as university/company staff webpage or LinkedIn profile. This was a straightforward (if lengthy) process, but for 32% of participants it was not possible to find any information about their employment. The missing data was almost exclusively for low level users of the tracker. For example, a person might file only one issue and leave very few digital footprints. For core Moodle developers and those heavily involved in Moodle there was generally an abundance of biographical information. The chart below shows this data with the unknowns removed.

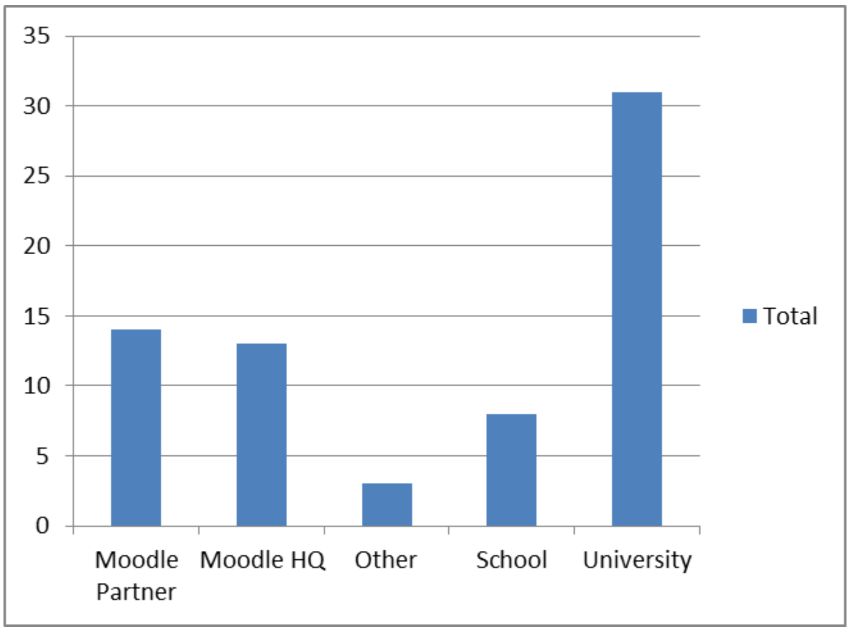


Figure 6 Known Place of Work of Tracker Member Sample

Most of the participants profiled worked in universities (31%) followed by Moodle Partners (14%) and Moodle HQ (13%) with a small amount of the participants in schools (8%). There is no directly comparable metric for the wider Moodle community but we do have information on distribution of Moodle by institutional type, again from the upper temporal bound of our case study – the beginning of 2011 (Dougiamas, 2011). This data reports respectively universities, schools and companies as the biggest users of Moodle. Schools are more prominent in their reported use of Moodle relative to the tracker profile sample. However, this only tells us about the estimated *number of schools* using Moodle and nothing of the *level of that usage* e.g. a distance learning university may effectively be using Moodle as their campus – thousands of students and lecturers using it intensively, whereas perhaps only *one* teacher in *a* particular school may be making some basic use of Moodle with one small class.

It was possible to refine the primary place of work to a specific role. These roles were then coded to the eight most common occupational categories:

- **Developer** – software programmer (40% belonged in this category)
- **Manager** (9%)

- **Lecturer** (9%)
- **Teacher** (9%)
- **Consultant** – generic IT role (7%)
- **System Administrator** – person who looks after a Moodle server (6%)
- **Educational Technologist** – University ICT in Education specialist, trainer etc. (4%)
- **Researcher** – PhD student or post-Doc (2%)
- **Miscellaneous Others** (2%)

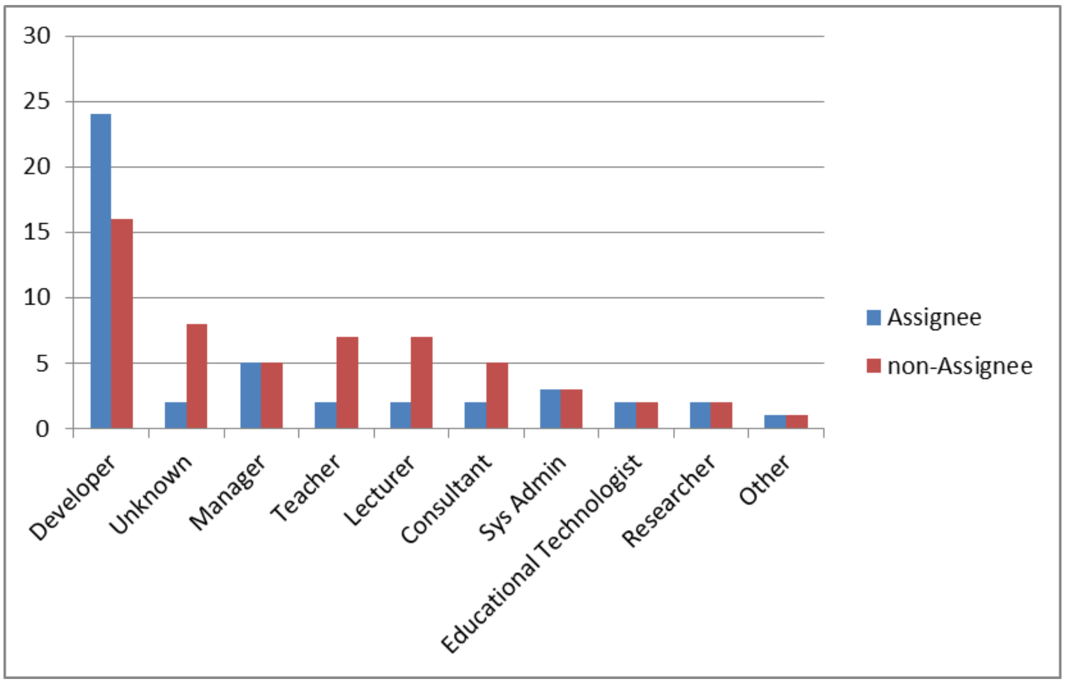


Figure 7 Occupations of Tracker Member Sample

The occupations of the participants are graphed in Figure 7 above according to their status as assignee or non-assignee in the tracker. It can be seen that developers are represented more in the assignees (24% of overall) than the non-assignees (16%). A converse disparity exists between non-assignee teachers (7%) and non-assignee lecturers (7%) versus assignee teachers and lecturers (both 2%). This would suggest that developers are more likely to be assignees whereas teachers and lecturers are not. Although caution should be exercised in making too many extrapolations from this data we can note simple absences and presences. It is for example noteworthy that all types of occupations are represented as assignees; teachers and lecturers, for example, although underrepresented, do nonetheless appear as assignees in the tracker.

4.2.2.3 Educational Background

It was possible to ascertain the highest educational attainment for many of the participants in a variety of ways, such as from LinkedIn profiles, company or particularly university profiles, from the interviews themselves, or by inference e.g. a teacher must have a primary degree. Estimating equivalence of various international educational qualifications would constitute a complex exercise in itself, so the Irish National Framework for Qualifications (NFQ) was taken as a reference point and participants' highest qualifications were coded according to levels eight, nine and ten on the NFQ corresponding to undergraduate degree, postgraduate masters degree or PhD (NFQ, 2013).

Table 3 Highest Educational Attainment Level of Tracker Member Sample

	Degree	Masters	None	PhD	Unknown	Total
Assignee	11	5	1	6	22	45
non-Assignee	12	8	1	11	24	56
Grand Total	23	13	2	17	46	101

There is much missing data here (46% unknown) and this may bias the sample (e.g. a person with no third level qualification may be less likely to publicly display this fact). However it is also to be expected that a community based around education would be highly educated. Figure 8 below shows Table 32 in bar plot form.

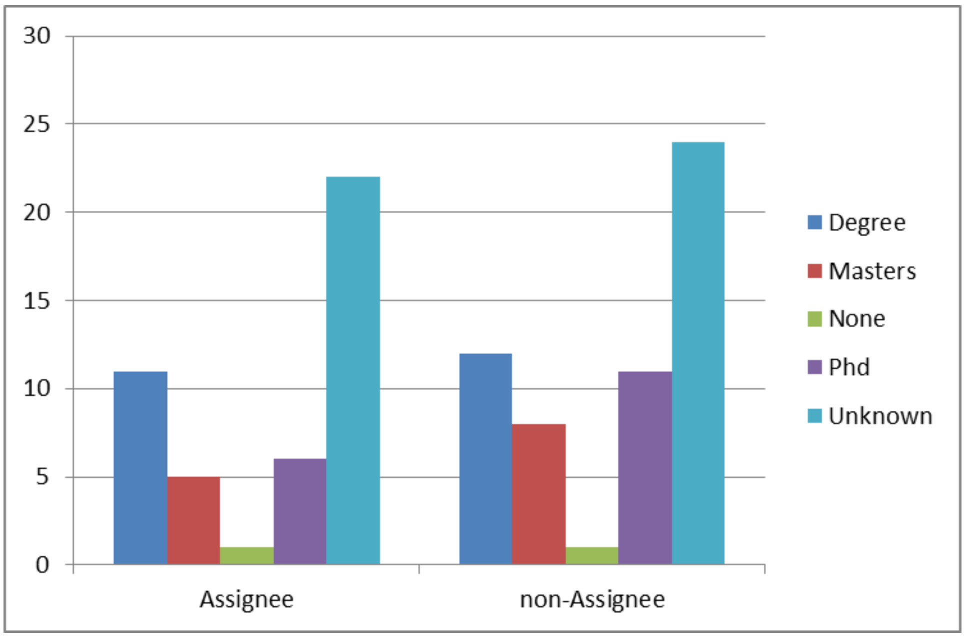


Figure 8 Known Educational Attainment Level of Tracker Member Sample

Another noteworthy aspect gleaned during the collection of this data was that at least three participants had written masters theses on some aspect of using Moodle itself in research or education and one was pursuing a PhD that involved Moodle. (Moodle itself was of course an offshoot from the unfinished PhD of Martin Dougiamas).

4.2.2.4 Education by Type

Computer science, IT and software development dominated the qualifications of many of those involved. It was common for a developer to have a BSc. in an information science for example, though they could also have a qualification in mathematics, physics or engineering. To summarise this information, educational attainments were coded according to whether they were in a Science, Technology, Engineering or Mathematics (STEM) discipline. Qualifications in education or pedagogy were also coded, and combinations of these two became another category:

Table 4 Educational Background Type of Tracker Member Sample

	Other+Edu	Edu	Other	NA	STEM+Edu	STEM	Unknown	Total
Assignee				1	4	16	24	45
non-Assignee	1	1	4	1	3	17	29	56
Grand Total	1	1	4	2	7	33	53	101

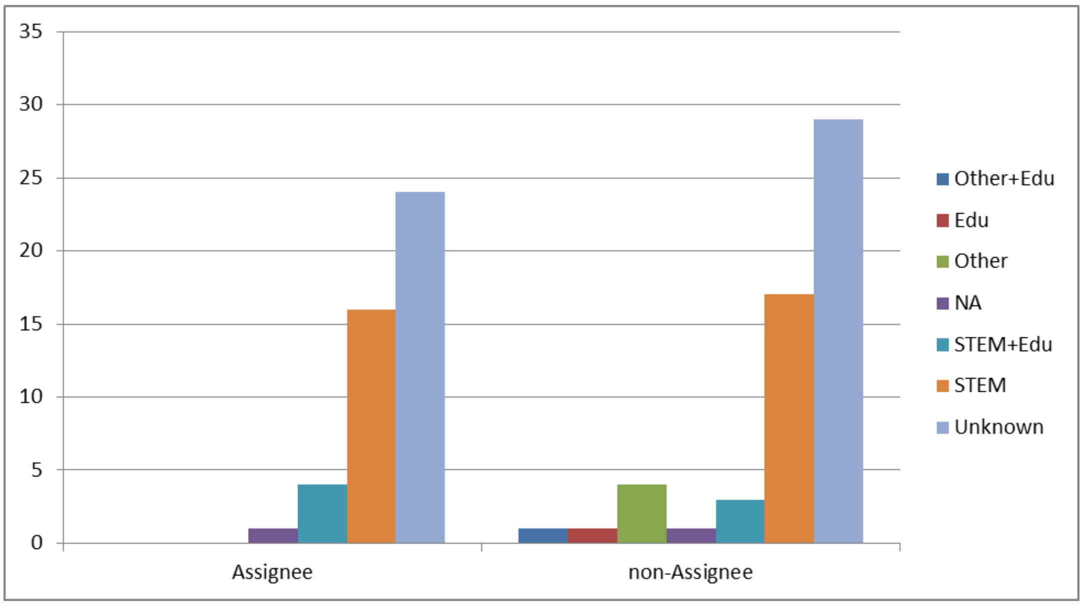


Figure 9 Educational Background Type of Tracker Member Sample

Data is only available for 47% of the sample so generalisations to the wider (particularly non-assignee) tracker population would be not be advisable here. It is worth noting that nine per cent of all the people about whom educational attainment is known have some form of qualification in education itself and this figure could be larger. Also perhaps noteworthy in terms of the assignee population, amongst whom we can make more confident inferences, is that all bar one person have some STEM educational background. These averages do not give the rich picture that individual stories do, such as the fulltime Moodle programmer with a PhD in pedagogy, another with a PhD in Mathematics, the college dropout entrepreneur, or the chemistry professor spending spare time writing fixes for the Moodle quiz.

4.2.2.5 Gender

Gender was determined by participant name. Although there is a small chance of human error, no names were judged ambiguous by the researcher. 13% of the 101 participants were women and 81% were men, with similar distributions represented in each of the categories of assignee and non-assignee.

4.2.2.6 Summary

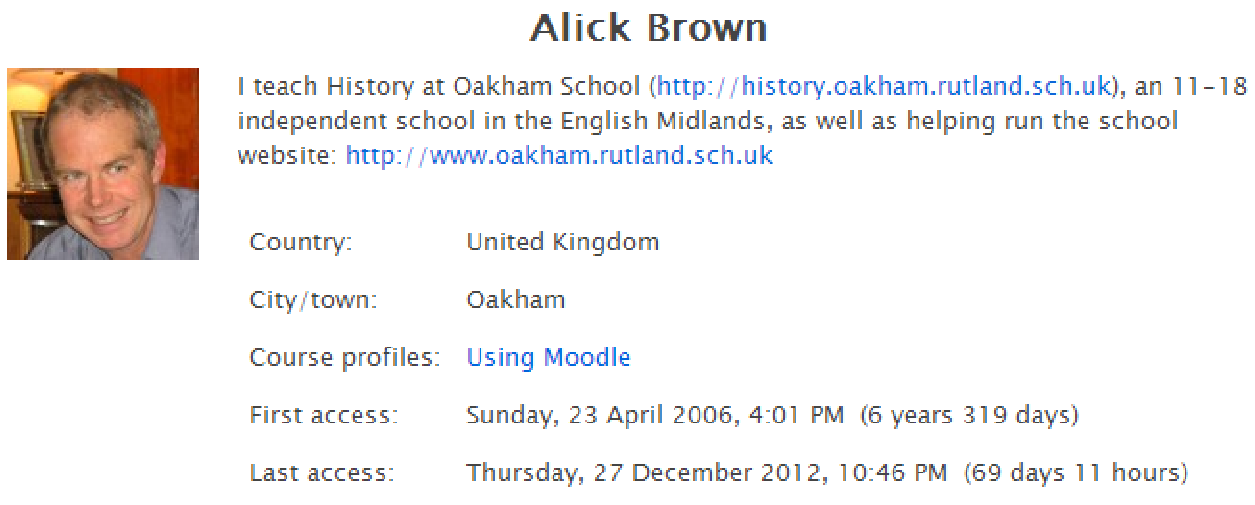
Overall we have a picture of a group of people in the tracker who are well educated. These people are drawn from cities distributed around the developed world, particularly Western Europe, North America and Australia. They work in schools, universities, IT companies and in particular companies dedicated to Moodle itself, including of course Moodle HQ. Although STEM subjects are highly represented in the educational backgrounds of these people, other fields are also represented and education and pedagogy qualifications are not uncommon. These people are mostly men though women are also present. This data was important in ensuring that these characteristics were all represented in the sample selected for interview.

4.2.3 Moodle Tracker Issue Anatomy: Three Examples

This study focuses on the Moodle bug tracker as a boundary object that simultaneously joins, and separates, communities. Inside the tracker boundary lies the Moodle community of developers and assignees. Educational worlds orbit outside of this core. The two groups interface directly through *individual* bug tracker issues. These issues are the holes in the sieve that filter communication to the core. It is to individual issues that we next turn and this section will examine three specific sample issues, presenting them in a narrative form and showing how they are represented in the bug tracker software interface. Screenshots of these sample issues are presented and one is annotated to highlight the lifecycle of an issue from opening to closure. The three issues were selected as representative examples of the variety of the characteristics that an issue may have. Issues may involve a range of actors such as teachers, university lecturers, Moodle Partners, influential assignees; be of varying lengths and complexities; involve different levels of interaction; and have different time-spans from issue opening to issue resolution. This section will also serve to introduce special terms used in the tracker in a context where examples can be given.

4.2.3.1 Alick Brown and an Error in the Workshop Module

The first issue we will look at is an example of a relatively straight forward and typical issue. Issue MDL-7932 was reported by a history teacher from Oakham secondary school in England on December 16th 2006. The discussion forum profile of this reporter – Alick Brown – is reproduced in Figure 10 below:



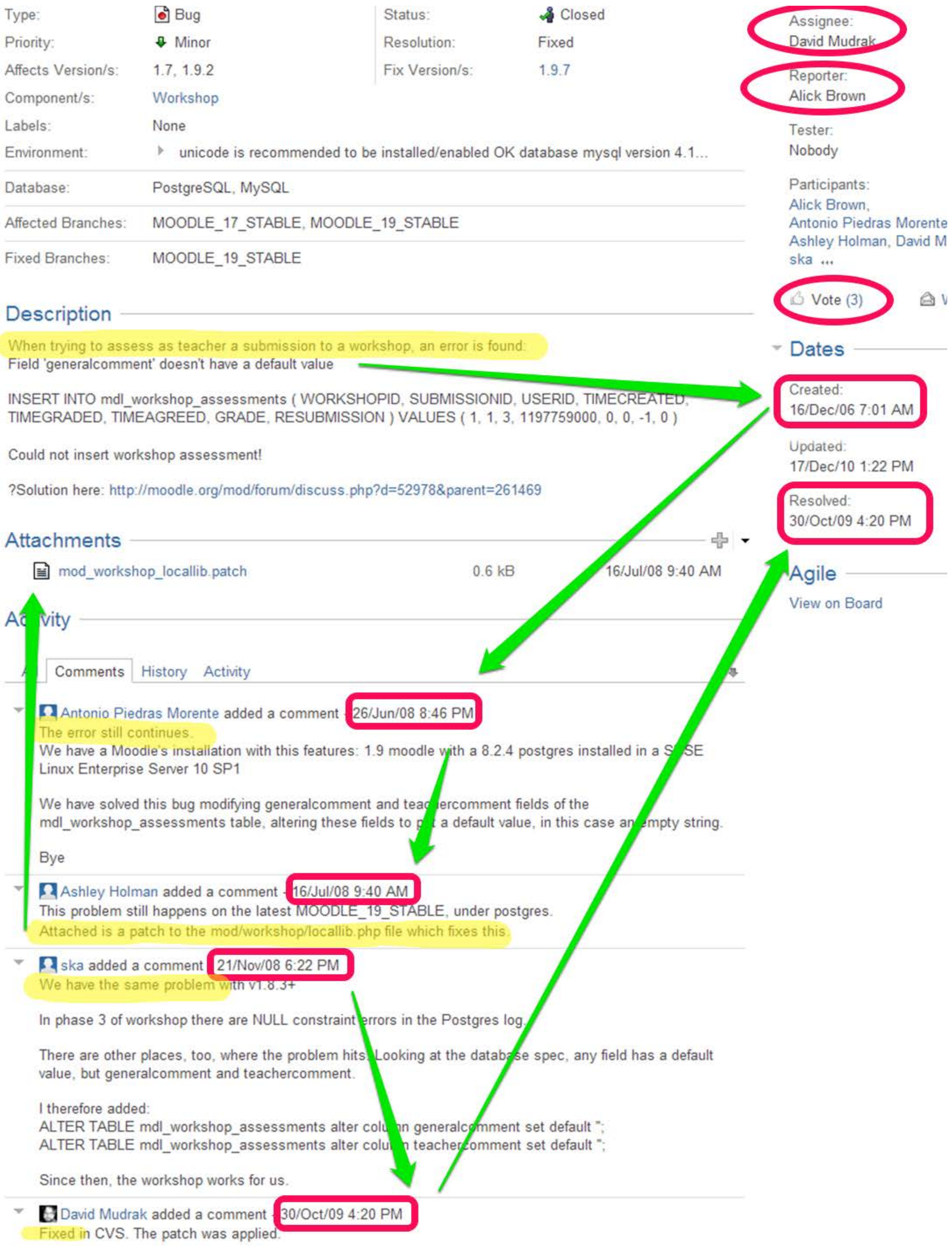
Alick Brown

I teach History at Oakham School (http://history.oakham.rutland.sch.uk), an 11-18 independent school in the English Midlands, as well as helping run the school website: http://www.oakham.rutland.sch.uk

Country:	United Kingdom
City/town:	Oakham
Course profiles:	Using Moodle
First access:	Sunday, 23 April 2006, 4:01 PM (6 years 319 days)
Last access:	Thursday, 27 December 2012, 10:46 PM (69 days 11 hours)

Figure 10 Forum Profile of a Tracker Member

Alick explains in his submission that he has tried to use the Workshop feature of Moodle but encountered an error. He points to other people debating this issue in the forums, including someone that has proposed a fix to the issue there. No more activity happens in this tracker issue until June 2008 and then, in the space of three months, three other users add that they too are experiencing the issue and that they are using a fix in their local Moodle installation. One participant in the issue attaches a *patch*, a piece of code proposed to fix the problem. A year later in October 2009, the assignee, and Moodle HQ member David Mudrak (who works remotely from the Czech Republic), fixes and closes the issue. Figure 11 below shows a screen-shot of the actual issue in the tracker. The key dates and events of the issue are highlighted in annotations by the researcher which acts as an overlay to explain the lifecycle of the issue to the reader. Although there is a significant degree of detail in this figure it can be most easily understood by looking at the red circled elements, following the green arrows and reading the text highlighted in yellow.



Type: Bug
Priority: Minor
Affects Version/s: 1.7, 1.9.2
Component/s: Workshop
Labels: None
Environment: unicode is recommended to be installed/enabled OK database mysql version 4.1...
Status: Closed
Resolution: Fixed
Fix Version/s: 1.9.7
Database: PostgreSQL, MySQL
Affected Branches: MOODLE_17_STABLE, MOODLE_19_STABLE
Fixed Branches: MOODLE_19_STABLE

Assignee: David Mudrak
Reporter: Alick Brown
Tester: Nobody
Participants: Alick Brown, Antonio Piedras Morente, Ashley Holman, David M ska ...
Vote (3)

Dates
Created: 16/Dec/06 7:01 AM
Updated: 17/Dec/10 1:22 PM
Resolved: 30/Oct/09 4:20 PM

Agile
View on Board

Description

When trying to assess as teacher a submission to a workshop, an error is found:
Field 'generalcomment' doesn't have a default value

INSERT INTO mdl_workshop_assessments (WORKSHOPID, SUBMISSIONID, USERID, TIMECREATED, TIMEGRADED, TIMEAGREED, GRADE, RESUBMISSION) VALUES (1, 1, 3, 1197759000, 0, 0, -1, 0)

Could not insert workshop assessment!

?Solution here: http://moodle.org/mod/forum/discuss.php?d=52978&parent=261469

Attachments

mod_workshop_locallib.patch 0.6 kB 16/Jul/08 9:40 AM

Activity

Comments History Activity

Antonio Piedras Morente added a comment - 26/Jun/08 8:46 PM
The error still continues.
We have a Moodle's installation with this features: 1.9 moodle with a 8.2.4 postgres installed in a SUSE Linux Enterprise Server 10 SP1

We have solved this bug modifying generalcomment and teachercomment fields of the mdl_workshop_assessments table, altering these fields to put a default value, in this case an empty string.

Bye

Ashley Holman added a comment - 16/Jul/08 9:40 AM
This problem still happens on the latest MOODLE_19_STABLE, under postgres.
Attached is a patch to the mod/workshop/locallib.php file which fixes this.

ska added a comment - 21/Nov/08 6:22 PM
We have the same problem with v1.8.3+

In phase 3 of workshop there are NULL constraint errors in the Postgres log.

There are other places, too, where the problem hits. Looking at the database spec, any field has a default value, but generalcomment and teachercomment.

I therefore added:
ALTER TABLE mdl_workshop_assessments alter column generalcomment set default '';
ALTER TABLE mdl_workshop_assessments alter column teachercomment set default '';

Since then, the workshop works for us.

David Mudrak added a comment - 30/Oct/09 4:20 PM
Fixed in CVS. The patch was applied.

Figure 11 Annotated Screenshot of Tracker Issue MDL-7932

There are five participants in this issue. We define a *participant as someone who submits an issue, comments on an issue, or is assigned to an issue*. Not shown in the graphic are the identities of three *voters* on the issue and four *watchers* of the issue. A *watcher will get an email if there is any activity on the issue they are watching*. A user becomes a watcher of an issue simply by clicking a *watch* link on the issue when they are logged in. In a similar vein, a *voter is someone who has voted on an issue by clicking vote*. In all there are eleven unique people involved in this issue (as one person who voted is also a watcher). A useful point to note is that the voters here are a mutually exclusive set of people from the participants, and share only one person in common with the watchers. What this means is that the issue submitter, the commenters and three of the four people watching the issue – a group which clearly wants this issue to be resolved – have not actually voted. Voting is a mechanism intended to act as a flag to increase the priority of issues. However from this case we can see that it is not being used by people who clearly want the issue to be resolved and hence we must be cautious about using it as a precise measure of how many people want an issue to be resolved.

This is not necessarily a typical issue, although it helps illustrate several characteristics of the Moodle tracker issue lifecycle. Indeed, the twenty thousand issues under consideration here vary widely and we later analyse them statistically.

4.2.3.2 MDL-1647 Allowing Negative Marking for Questions

An example of an issue with a relatively high number of votes is issue MDL-1647 – *Allowing Negative Marking for Questions*. This issue was opened on the 15th of July 2004 and closed on the 10th of June 2011. "That only took 6.5 years" quipped Tim Hunt of the OU, the developer who fixed the bug and hence closed this issue (Hunt, 2011). This issue had six proposed patches, 28 votes, 20 watchers, several related discussion forum threads and seven duplicates. As its name implies, *a duplicate is an issue submitted for which there already exists a prior valid issue submission* i.e. the duplicate submitter has not found the original issue and assumes that no such issue yet exists. Such issues are closed when identified and merged with the original issue.

The crux of issue MDL-1647 was a bug in the Moodle quiz which meant that negatively marked multiple choice assessment questions were not possible in Moodle. This is a requirement of many standardised tests (such as the SAT for college admissions in the United States). The fact that the negative marking did not work was not immediately apparent to the end user, as this option appeared to be available but was in fact broken (quizzes silently rounded up questions that should have attained negative marks to zero). As outlined in Chapter One this is the issue that ultimately inspired the pursuit of this research.

Tim Hunt is Moodle's Quiz component maintainer (also known as component lead), the developer who has responsibility for this particular component i.e. the Quiz of Moodle. Moodle Development manager Michael du Raadt described this role in his interview:

> Anyone who is a component lead is sort of seen as the – not the owner – but pretty much as close as you can be to an owner for an area of Moodle. In other words they are basically saying what can come and go in that part of the codebase (Interview with Michael du Raadt, 2012).

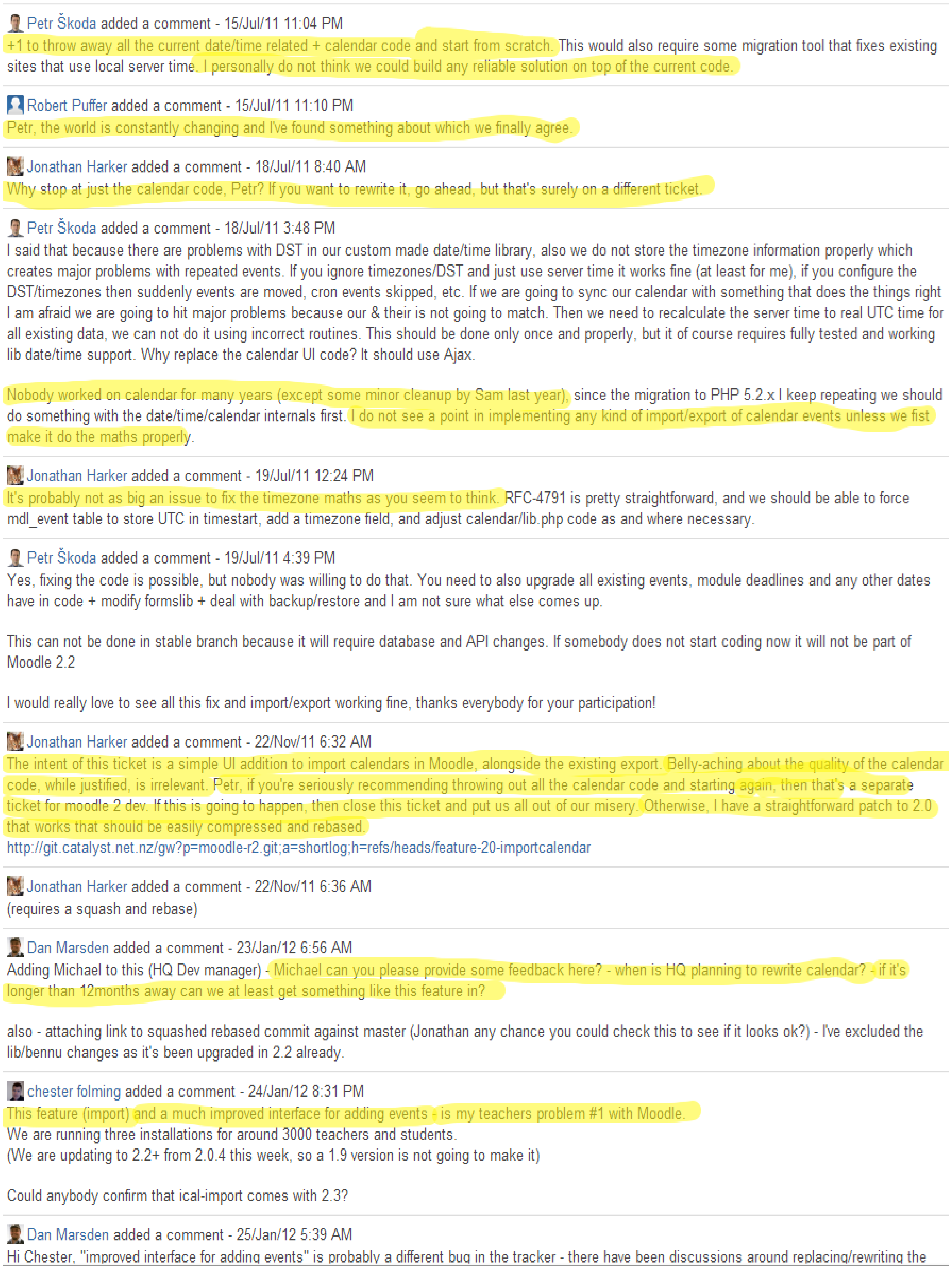
Shortly after this issue was opened in July 2004 Tim added a comment to it in the tracker explaining that it could not be fixed in a straightforward way because it could break other functionality. In January 2009, Tim then wrote that a fix to the issue would be part of a complete rewrite of the code for the quiz module (component). After the issue was finally closed in June of 2011, new participants joined the issue to say that the new functionality introduced by the fix was having some strange effects on old quizzes. The fact that an issue can technically close but still be active is something to note here.

An interesting aspect of this issue is that it may represent something of a grey area between a bug and a feature i.e. between something broken that needs to be fixed and something new that needs to be developed. Issues are typically divided into bugs (errors which require fixing) or new feature requests, but as some participants in this issue demonstrate, this can be a contested concept e.g. one of the duplicate issues is a request for this feature to be developed (implying that it does not already exist in a broken form).

4.2.3.3 Import a Calendar into Moodle: A Popular Feature Request

One of the issues with the highest number of votes in Moodle is issue MDL-16660 which had 203 votes upon its closure in November 2012. This issue, started in September 2008, comprises a request for the functionality to allow a calendar from a different application be imported into and then displayed within Moodle. For example a user might create a calendar (such as a class timetable) in an application like Google Calendar but wish to display the results from it within Moodle. This issue attracted 179 comments, had 120 watchers and 49 participants. Various patches/fixes were proposed; one was worked on by a particular developer before being taken up by a second developer about a year and half later. The issue was related to the Calendar functionality of the Moodle code which did not have an obvious designated maintainer. Some contentious discussion occurred between a Moodle HQ developer and a proposer of a fix. The issue also straddled the introduction of a new development process for Moodle in December 2010 which created new requirements for accepting any code into Moodle; in particular it required that all code must undergo a rigorous peer review process by a member of HQ before being accepted. The institution of this new process may have contributed to the protraction of this issue. As in the negative marking issue, this issue also has a follow-up request after its closure – with a user requesting that it be possible to add a location, and not just description and time, to a calendar event.

This issue is particularly noteworthy for its exceptionally large number of comments – 179. Although it is not possible to systematically count the number of comments per issue in the tracker, there would be few if any issues with so many comments and none discovered by the researcher. An annotated screenshot in Figure 12 below shows some of the comments on this issue including the aforementioned, sometimes contentious, discussion between a person working on a fix, Jonathan Harker, and Moodle HQ developer Petr Škoda. The discussion turns on many ostensibly technical details of how best to implement the desired feature such as how any implementation might affect other parts of Moodle. Relevant sections are highlighted to allow the reader to follow the debate without needing to read the more technical passages:


Petr Škoda added a comment - 15/Jul/11 11:04 PM
+1 to throw away all the current date/time related + calendar code and start from scratch. This would also require some migration tool that fixes existing sites that use local server time. I personally do not think we could build any reliable solution on top of the current code.

Robert Puffer added a comment - 15/Jul/11 11:10 PM
Petr, the world is constantly changing and I've found something about which we finally agree.

Jonathan Harker added a comment - 18/Jul/11 8:40 AM
Why stop at just the calendar code, Petr? If you want to rewrite it, go ahead, but that's surely on a different ticket.

Petr Škoda added a comment - 18/Jul/11 3:48 PM
I said that because there are problems with DST in our custom made date/time library, also we do not store the timezone information properly which creates major problems with repeated events. If you ignore timezones/DST and just use server time it works fine (at least for me), if you configure the DST/timezones then suddenly events are moved, cron events skipped, etc. If we are going to sync our calendar with something that does the things right I am afraid we are going to hit major problems because our & their is not going to match. Then we need to recalculate the server time to real UTC time for all existing data, we can not do it using incorrect routines. This should be done only once and properly, but it of course requires fully tested and working lib date/time support. Why replace the calendar UI code? It should use Ajax.

Nobody worked on calendar for many years (except some minor cleanup by Sam last year), since the migration to PHP 5.2.x I keep repeating we should do something with the date/time/calendar internals first. I do not see a point in implementing any kind of import/export of calendar events unless we fist make it do the maths properly.

Jonathan Harker added a comment - 19/Jul/11 12:24 PM
It's probably not as big an issue to fix the timezone maths as you seem to think. RFC-4791 is pretty straightforward, and we should be able to force mdl_event table to store UTC in timestart, add a timezone field, and adjust calendar/lib.php code as and where necessary.

Petr Škoda added a comment - 19/Jul/11 4:39 PM
Yes, fixing the code is possible, but nobody was willing to do that. You need to also upgrade all existing events, module deadlines and any other dates have in code + modify formslib + deal with backup/restore and I am not sure what else comes up.

This can not be done in stable branch because it will require database and API changes. If somebody does not start coding now it will not be part of Moodle 2.2

I would really love to see all this fix and import/export working fine, thanks everybody for your participation!

Jonathan Harker added a comment - 22/Nov/11 6:32 AM
The intent of this ticket is a simple UI addition to import calendars in Moodle, alongside the existing export. Belly-aching about the quality of the calendar code, while justified, is irrelevant. Petr, if you're seriously recommending throwing out all the calendar code and starting again, then that's a separate ticket for moodle 2 dev. If this is going to happen, then close this ticket and put us all out of our misery. Otherwise, I have a straightforward patch to 2.0 that works that should be easily compressed and rebased.
http://git.catalyst.net.nz/gw?p=moodle-r2.git;a=shortlog;h=refs/heads/feature-20-importcalendar

Jonathan Harker added a comment - 22/Nov/11 6:36 AM
(requires a squash and rebase)

Dan Marsden added a comment - 23/Jan/12 6:56 AM
Adding Michael to this (HQ Dev manager) - Michael can you please provide some feedback here? - when is HQ planning to rewrite calendar? - if it's longer than 12months away can we at least get something like this feature in?

also - attaching link to squashed rebased commit against master (Jonathan any chance you could check this to see if it looks ok?) - I've excluded the lib/bennu changes as it's been upgraded in 2.2 already.

chester folming added a comment - 24/Jan/12 8:31 PM
This feature (import) and a much improved interface for adding events - is my teachers problem #1 with Moodle.
We are running three installations for around 3000 teachers and students.
(We are updating to 2.2+ from 2.0.4 this week, so a 1.9 version is not going to make it)

Could anybody confirm that ical-import comes with 2.3?

Dan Marsden added a comment - 25/Jan/12 5:39 AM
Hi Chester, "improved interface for adding events" is probably a different bug in the tracker - there have been discussions around replacing/rewriting the

Figure 12 Discussion of Calendar Import Issue

As the sample of comments shows there is an obvious friction between the Moodle HQ developer, who does not like the proposed solution to the problem and advises a complete overhaul of Moodle's date/time functionality, and a developer who believes their proposed solution to be a valid one. This highlights the negotiation that often occurs in issues.

Something else noteworthy about this issue is that after it was opened Tim Hunt closed the issue mistakenly misclassifying the issue as having been previously resolved. The issue submitter then adds a comment explaining that Tim has confused Calendar import for Calendar export and asks that the issue be reopened which Tim does. Although this is an extreme example, because the error was clear to both parties and quickly resolved, it does show that an issue's status as *resolved* is negotiable.

4.2.4 Tracker Issues in the Aggregate

Unlike the three issues reviewed here, many tracker issues are resolved quickly. Many have far less votes, comments and participants. Indeed an issue may involve only one person – where a developer reports and then fixes and closes an issue they have found. This section provides an overall summary of tracker issues during the case study period.

There were 20,830 issues created in the tracker between the first of January 2007 and the ninth of February 2011. These issues were created by 2,840 unique people. (On the basis that user accounts map to unique people, though in a small number of cases this may not be so and where obvious these were removed from the data.) Of these 2,840 reporters, 106 are assignees and the remaining 2,734 are non-assignees. There were also 1,086 active participants in the tracker during this period i.e. people who did not report an issue but commented on an issue reported by somebody else. The number of voters is not determinable, though from any sample of issues with votes, such as those issues already discussed, it can be seen that issues may contain several voters who are neither reporters nor commenters. Thus we can infer that there may be a sizeable population of silent voters, who do not comment or initiate issues but create an account in the tracker and then simply click the vote link to express their interest in an issue and its resolution. Similarly to voters, there may be a number of silent or "lurking" issue watchers. As this shows, different types of members display different levels of interaction and engagement. Defined in terms of practice then there are clearly distinct communities operating here through the shared artefact of the tracker and tracker issues.

4.2.4.1 Issue Reporting

Of the 2,840 issue reporters, many only ever report a single issue. At the other end of the scale a handful of assignees submit hundreds of issues each. This is illustrated in the graph in Figure 13 below which plots issues reported per individual:

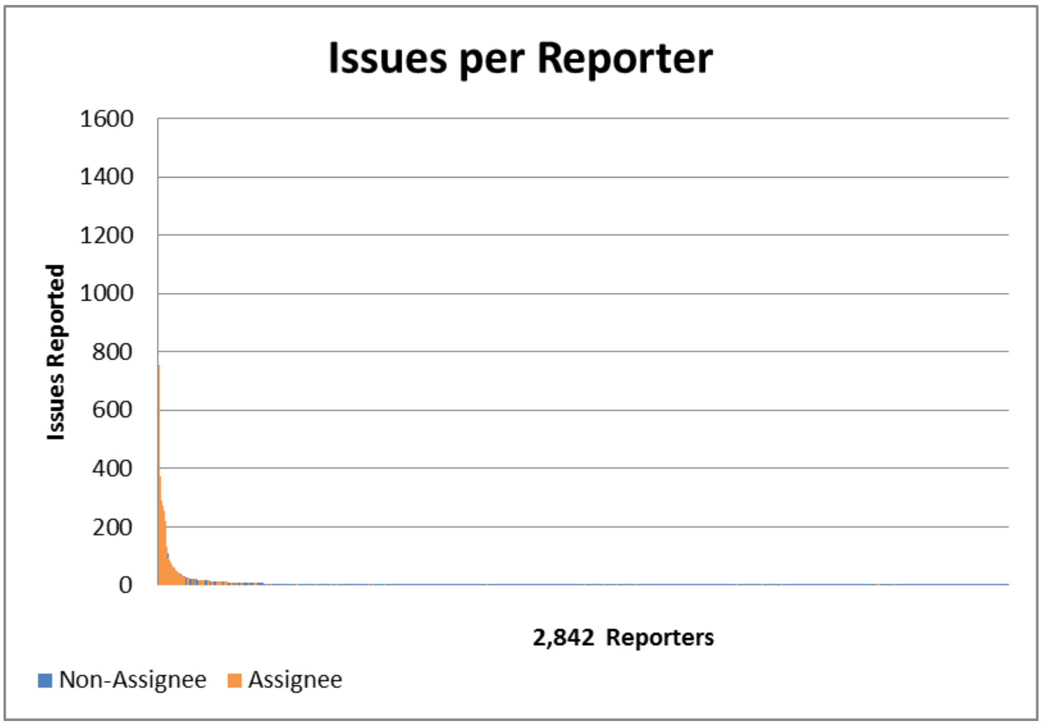


Figure 13 Issues Reported per Person

Overall, a very small number of people create a large number of total issues and conversely a vast number of issues are created by a large number of people who only ever create one issue. As can be seen from Figure 13 above, the distribution of this population is positively skewed (the skew is 19 with a high kurtosis of 496). The submission distribution is of a similar form when taking assignees only as reporters where skew is 20 and kurtosis is 528.

Even within the assignees there are a number who remain relatively dormant, submitting only one or two issues. below gives a tabulated summary of the issues reported by both assignees and non-assignees:

Table 5 Summary of Issues Reported

Issue Reports	**Assignee**	**Non-Assignee**	**All**
Total	10665	10165	20830
Mean	3.9	95.9	7.3
Median	1	20	1
Mode	1	1	1
Std Dev	16.8	190.7	43.8
Max	555	1457	1457
Min	1	1	1
Skew	20	4.4	18
Kurtosis	528	26	496
Reporters	2734	106	2840

From the high deviation and skew of the assignee population we can deduce that there is a smaller sub-population who are more active than the assignees as a whole – a sub-group of super-assignees. Indeed, the shape of many of these reports shows that in many respects Moodle consists of groups whose activity becomes increasingly concentrated. The modal value of issues reported per person is one and the maximum value is 1,457. As there are such differences between the numbers of submissions per person, it is also instructive to look at a table of top submitters ranked by their number of individual submissions as detailed in Table 6 below (as with all tracker data this is in the public domain and available from http://tracker.moodle.org):

Table 6 Top 20 Issue Submitters

	Number of Issues Submitted	
Person	**Non-Assignee**	**Assignee**
Petr Škoda (skodak)		1457
Eloy Lafuente (stronk7)		753
Tim Hunt		634
Martin Dougiamas		588
Jerome Mouneyrac		372
Howard Miller		363
Helen Foster		334
Daniele Cordella		310
Dan Poltawski		291
Ralf Hilgenstock	283	
Matt Gibson	282	
Penny Leach		279
Joseph Rézeau		276
David Mudrak		274
Ray Lawrence	273	
Andrea Bicciolo		270
Nicolas Connault		258
Sam Hemelryk		253
N Hansen	242	
Sam Marshall		227

The assignees make up most of this list with four non-assignees represented. One assignee Petr Škoda accounts for 1,457 or 7% of all issues while the first six people combined are responsible for 20% of all issue submissions. The twenty people in this table together account for a sizeable proportion (39%) of all issues reported during the period. At the other end of the scale, 19% of all issues submitted are accounted for by people who submit only five issues or less and 8% of issues (1,655 in total), are reported by people who ever only report a single issue.

4.2.4.2 Votes for Issues

It is possible to determine the number of votes per issue. Similarly to the number of issues reported per person, the distribution of this data is skewed: some issues get a large number of votes, while most issues get no votes at all. In fact only 17% (3,589) of issues have any votes at all. This is illustrated in the graph in Figure 14 below:

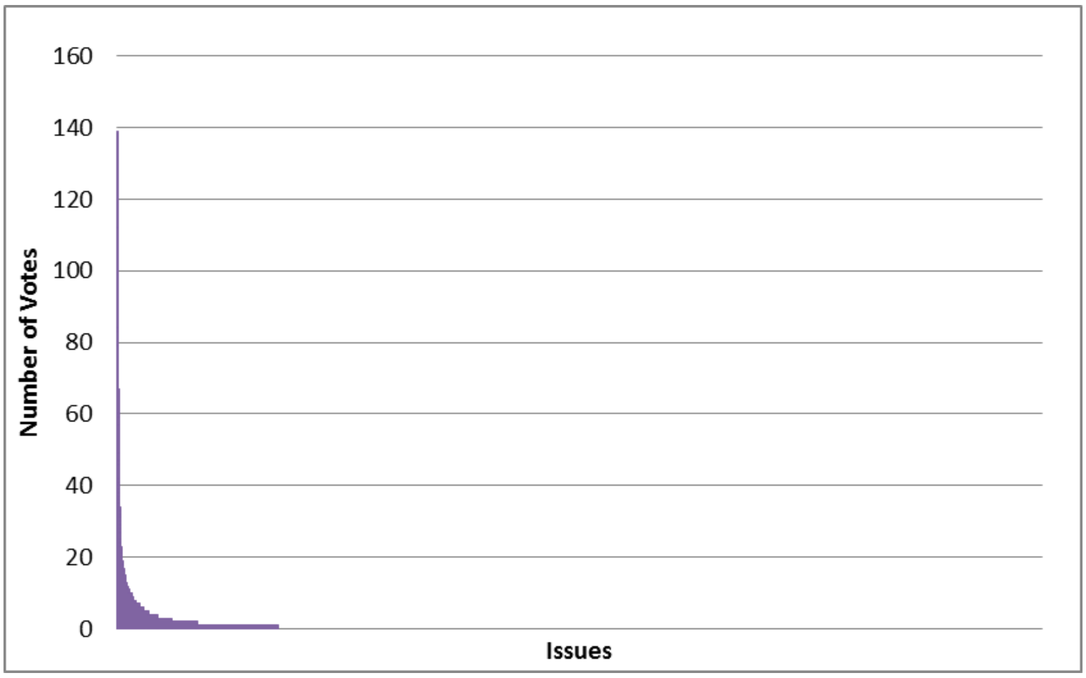


Figure 14 Votes per Issue

Table 7 Summary of Votes per Issue

Started By	Total	Mean	Min	Max	Std Dev	Median	Mode	Skew	Kurt
Assignee	4,094	0.4	0	79	2.3	0	0	14.4	309
non-Assignee	9,095	0.9	0	139	3.8	0	0	14.4	338
Total	13,189	0.6	0	139	3.2	0	0	15.6	410

A summary of the voting data is presented in Table 7 above. The data shows that non-assignees (0.9 votes) voted slightly more per issue on average than assignees (0.4) but in both cases the overriding feature is that most issues have zero votes. In fact 88% (8,907) of the issues that were reported by non-assignees had no votes. This demonstrates that most issues attract little attention, while also showing that people who submit issues generally do not vote for them – even non-assignees i.e. even people we might assume do want those issues to be fixed as evidenced by their taking the time to report them.

4.2.4.3 Participants in Issues

It is also possible to determine the number of participants per issue. This is presented in Table 87 below:

Table 8 Participants per Issue

Started by	Total	Mean	Min	Max	Std Dev	Median	Mode	Skew	Kurt
Assignee	25,413	2.5	1	22	1.5	2	2	2.4	13
non-Assignee	32,367	3.0	1	34	1.6	3	2	5.3	57
Total	57,780	2.8	1	34	1.6	2	2	4	41

Some aspects of note about participants include that there are most often only two participants per issue (the mode) i.e. the reporter and the assignee, and a reporter can assign an issue to themselves, potentially leading to an issue with only one participant (the minimum). During the case study period were 2,323 issues with only one participant i.e. 11% of all issues. Participant numbers vary between issues – though not as widely as the numbers of votes or submissions. This point shows through clearly in the lower (relative to the previous graphs) skew and kurtosis values and is illustrated by graphing the numbers of participants per issue. As can be seen from Figure 15 below, this graph, while still positively skewed, represents a smaller scale than those of the graphs of Figure 13 and Figure 14 e.g. the maximum value in this distribution is 34. That is, overall, participants per issue are low and more uniformly distributed, relative to the other cases.

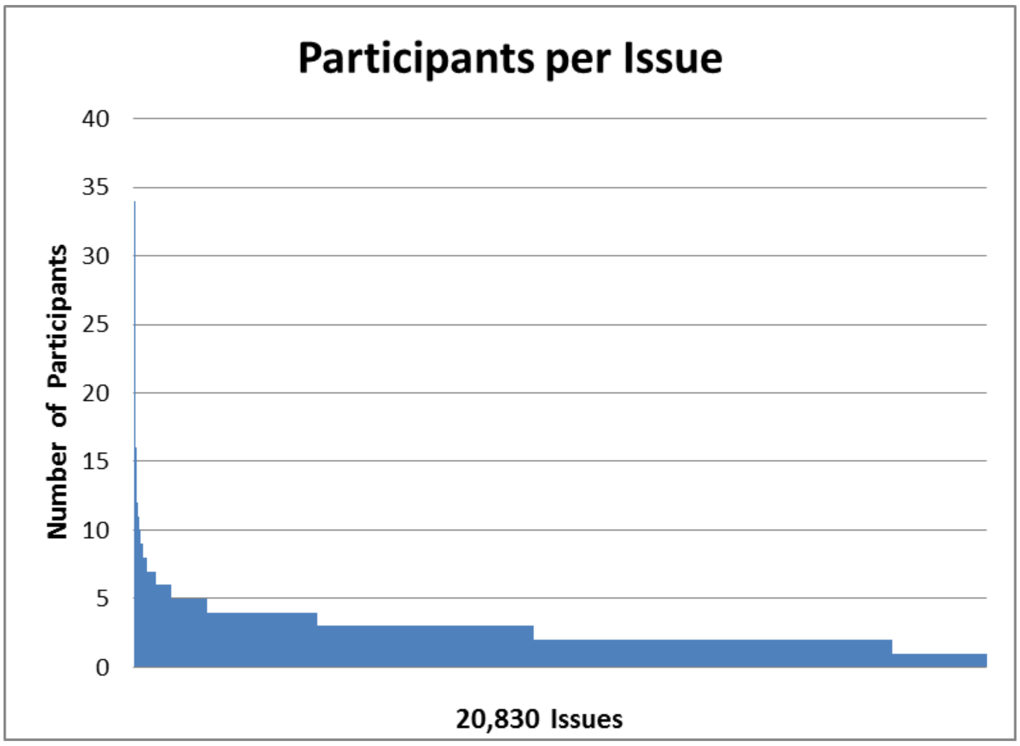


Figure 15 Participants per Issue

4.2.4.4 Issue Types: Bugs, New Feature Requests and Invalid Issues

A bug is generally thought of as a type of error that has manifested in software. Bugs are unintended and are unwanted insofar as they either prevent some feature of the software from working properly or possibly worse, change existing behaviour in an adverse way. However, not all issues in the tracker are true bugs. Many are feature requests or suggestions for improvements of existing features. For example, a teacher wishing to allow students to peer-assess each other's work in a particular way might submit an issue to the tracker requesting this functionality.

New feature requests, existing feature improvements and bugs can be grouped under the generic term *issues*. However, because bugs are generally the most numerous type of issue, the terms *bug* and *issue* are often used interchangeably. For example the "Moodle bug tracker" might more properly be called the "Moodle issue tracker". Moreover, the classification of an issue into a bug or feature request is often contested. The lack of a specific feature that a teacher wishes to use may seem to them like a fault in Moodle. In their eyes it may be a grave problem that is preventing a particular pedagogy, which may feel to them very much like a bug. For a developer however this feature has never existed so may fall into a "nice-to-have" rather than a "need-to-have" category. This is a simplified example but illustrates that the definition of a bug is not always fixed, given the different perspectives of the participants. Nonetheless once an issue is submitted in the tracker, it must be defined (by the submitter) as either a bug or a feature request/improvement. The assignee can then change this designation (most usually by re-classifying a bug as a feature request) after which it will generally remain fixed for the continued life of the issue.

There are other ways that the assignee can classify an issue, such as a duplicate of an existing bug, as previously mentioned, or they may report that they cannot reproduce the bug or simply say that the issue is not a bug. We can broadly group these classifications under the heading of *invalid issues*. For the purposes of this study we can define an *invalid issue as one which will never be fixed until such time as an assignee recognises its validity*. Generally speaking once an issue has been declared invalid it will remain so. An important point to note is that once an issue is deemed to be invalid it is closed and also that this classification may be a contested concept i.e. the closure of an issue, as demonstrated in the calendar and negative marking issues, does not mean that it has yet been resolved to the satisfaction of everyone.

It is unsurprising that non-assignees contribute more invalid issues than assignees. They are in general less experienced and knowledgeable users of Moodle (and of course assignees are the ones who generally decide whether bugs are valid or not.) This is illustrated in a breakdown of invalid issue submissions in Figure 16 below.

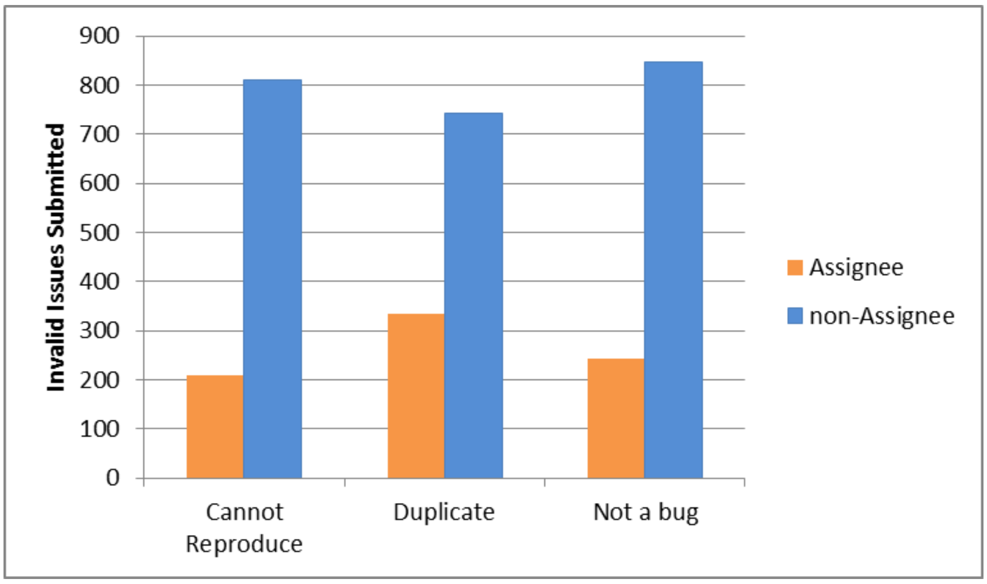


Figure 16 Invalid Issues Grouped by Submitter Type

As can be seen from Figure 16 above, assignees submit less invalid issues than non-assignees and the difference, calculated by chi square test, is statistically significant χ^2 (2, N = 3184) = 36.11, p = 1.441^{-8}.

If we return to valid issues, we can see that they are more likely to be fixed if they are classified as bugs rather than as request for new features/improvements (though many feature and improvement requests are nonetheless fixed). The graph in Figure 17 below illustrates how bugs are more often successfully resolved than feature requests. This success is significant χ^2 (1, N = 14119) = 1278.9, p = 4.42^{-280}.

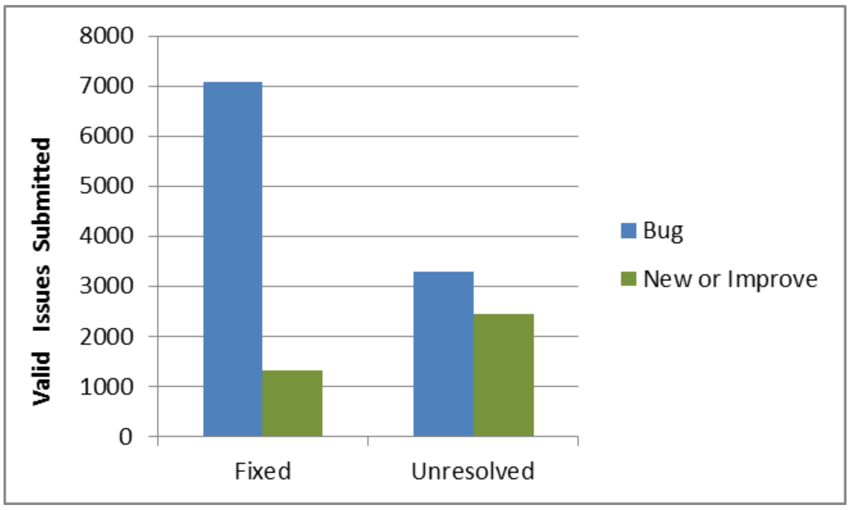


Figure 17 Bugs versus New/Improved Feature Resolutions

Lastly we will look at the proportion of valid issues submitted by assignees that are eventually fixed, with the starting hypothesis that assignee submitted issues are fixed more than non-assignee ones. As is clear from Figure 18 below this is the case i.e. assignee submitted issues constitute a greater portion of those issues that were fixed than non-assignee submitted issues and this is significant χ^2 (1, N = 14119) = 1162.5, p = 8.57^{-255}. Although this difference exists, it should nonetheless be noted that a large amount (3,457) and a sizeable proportion (24%) of all valid issues were submitted by non-assignees and were subsequently fixed.

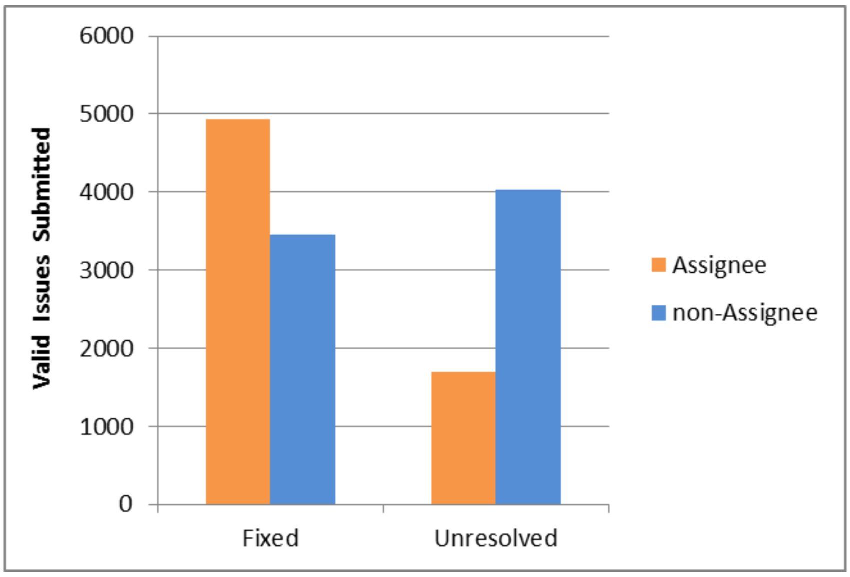


Figure 18 Eventual Resolutions of Issue Submission by Assignees versus Non-Assignees

4.2.5 Summary

We have seen that the tracker participant within our sample frame population is diverse with a wide geographic spread. It is composed of highly educated cohorts who have professional affiliations to Moodle HQ, Moodle Partners or other ICT consultancies, schools and universities. Within these organisations participants have professional identities of teachers, lecturers, educational technologists, researchers, managers, software developers or other ICT professionals. From a high level we can hence view the tracker as comprising different communities. To contextualise these people and the issues they are involved in we examined three sample issues that illustrated important concepts such as how participants negotiate to attempt to fix issues (and that even whether an issue is valid, or whether it has been truly fixed, can be negotiable and contentious concepts).

The statistical picture of the tracker presented demonstrates that a small number of contributors accounted for the submission of most issues but also that the sizeable wider community submitted one or a small handful of issues each. This shows that some people were very active in the tracker, whilst others were much less so, being only periodic or sporadic users, ones whose interest perhaps lay only in one or a very small number of specific issues. Likewise with voting, a handful of issues attracted a lot of votes while most issues attracted none. A similar story was found with participants per issue, though the overall variation was smaller in this case. The non-assignee group of users are more likely to submit issues that are subsequently classed as invalid, and valid issues they do submit are less likely to be fixed than those submitted by their assignee counterparts. Hence we have determined that assignees are more influential and active (in general) than their non-assginee counterparts. This broad conception of these two groups provided a basis from which to examine the core community practice more closely and so it is to issue resolution, through the eyes of its participants, to which we will now turn.

4.3 What Factors and Related Processes are Important in the Resolution of Issues in Moodle?

The second research question sought to explore the factors and related processes that are involved in the resolution of issues in the Moodle tracker. In theoretical terms this is where meaning is negotiated and knowledge brokered in the community. The resolution of an issue is the production of new knowledge.

Over thirty factors that participants believed important to issue resolution were identified (fully listed as Appendix D). These views form a belief matrix of the community, with commonalities but also contrasts. The factors were divided into three groups (codes), in the second coding phase, which revolved around: the submitter, the assignee and the code submitter. (As illustrated in Appendix C a fourth code of *inertias* was also identified here and this is dealt with later under research question four). Two of the groups represented by these second level codes were based on participant role following from the data analysis of the first question where the roles of relative community insiders (assignees) and outsiders (non-assignees i.e. ordinary submitters of bugs) were conceptualised.

Thus, the first group includes those factors that lie most within the *submitter*'s gift or domain. The second group are factors that relate closely to the *assignee/maintainer*. The third set of factors relate to the situation when *a submitter has code* to contribute i.e. they have written a patch or have commissioned the writing of one. Having a patch (one's own code) increases the potential influence domain of the submitter, giving them extra factors. Hence, although there are overlapping elements, these three groupings can be seen to have both chronological and ordinal aspects. If a submitter does not attain enough of the factors within the basic submitter domain, then the phase where the assignee becomes important may never be reached (e.g. if a submitter is rude, the assignee may ignore the issue). Similarly a submitter with a patch may be able to bypass some of the factors within the basic submitter domain (e.g. they don't need to be so polite). These three domains are represented in Figure 19 below:

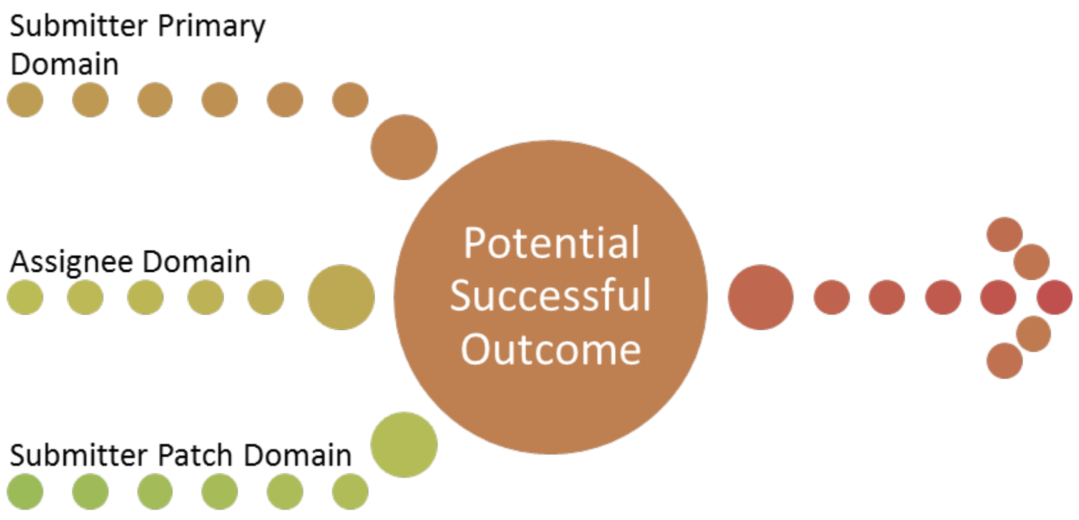


Figure 19 Resolution Factor Participant Domains

The following section discusses in detail the factors related to successful issue resolution within each of these three groupings. After dealing with the main factors in the context of submitters, assignees and code submitters, the factors were then analysed as processes. The arranging of the codes into graphical displays constituted part of the analysis that attempted to uncover the processes that linked these factors.

4.3.1 What Factors are Important to Issue Resolution in the Submitter's Primary Domain?

4.3.1.1 Information: Exchanging Story Steps

All interviewed participants stressed the quality of information in an issue submission as being important to its successful resolution. Precision and clarity were common attributes of what was deemed good information whilst vagueness or brevity were often characteristics of poor information and even too much detail could be as bad as too little. For a bug fix this information refers to a description of the issue, including what it looks like to the end user and also what happened to cause the issue. The steps leading up to the issue (its cause) allow the assignee to contextualise and diagnose the problem. Valued information may be characterised in proscriptive and abstract terms, usually with reference to clarity, as highlighted below with emphasis added:

> …sensible bug reports that are *clear* and easy to understand in *clear* language with easy steps to reproduce… (Interview M10).

> I think that the biggest factor yeah is having a *clear* bug report in the beginning (Interview M19).

Inaccurate information is the opposite of accurate information and it is frustrating because it is incomplete:

> If someone says oh I can't do when I go on the page it's all blue and I can't see the red bits y'know, they've not told you what page, what Moodle version**,** what they clicked on, how they got to that page, what they expected to happen (Interview M10).

The steps that led up to an issue and are hence needed to reproduce it featured strongly in assignee depictions of good bug report descriptions. Although these steps are described as informational – "numbered points, you-know, do this then do that, then do that" (Interview M10) there are hints that there may also be more tacit exchanges at work:

> …those are all factors that help bugs getting fixed. If your bug looks really not interesting or even very hard to understand what the problem is then it will stay there for a very long time (Interview M17).

What this last quote shows is that a bug may or may not look “interesting”. It hints that the writing of the report is part of a narrative exchange where the submitter must sell their story to the assignee. One participant, whilst emphasising clarity, also stressed the word communication and framed the issue in terms of the submitter’s need or want:

> …a lot of times it’s a matter of clearly communicating what the issue is […] I think that’s one aspect that kinda contributes to why certain issues don’t get resolved […] like the, what the user’s reporting or wanting isn’t clear enough (Interview M11).

Clear information is often professed to be very important by participants in disembodied and rational terms but it is also important to “communicate” what you “want”, tell a story via “steps” and to make that story not only understandable but “interesting”.

One non-assignee also reported feeling sometimes disadvantaged in communicating clear bug reports because English was his second language and claimed that this was not such an issue for bug submitters with established reputations but was a problem for newcomers: “It’s just a matter of experience. Sometimes people think you are – you have – not enough skills to judge some of the things you are talking about” (Interview M12).

4.3.1.2 Responsiveness of the Submitter

When the initial information is inadequate the issue may languish, or the assignee may prompt the submitter for more or better information. At this point *responsiveness* on the part of the submitter comes into play:

> Typically you’d get something filed and then, sort of about half the time it might be all-right, and half the time you got to ask them and then just ignore it until they respond (Interview M15).

The participant quoted above will ignore issues until the submitter has responded. The requirement of the response may be as much about the information required, as it is a test of how much the submitter wants the issue fixed as measured by their willingness to engage. Essentially, responsiveness thus contains elements of creating rapport and establishing a positive identity for the submitter as the below case (where the submitter also has a proposed code fix) illustrates:

> 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. Sometimes you-know you read something, and you write back to them and ask them to clarify something or fix something, and you don't hear back and it just hangs there for months and we are not going to push it if they don't want to push it either way (Interview M20).

Although responsiveness was perceived as important in establishing an issue and putting it on the assignee's agenda ("ignore it until they respond"), it is also a factor that must be sustained throughout the process for increased chance of success.

Although it was not touched upon in the interviews, other data sources confirm that a response can be made even to an apparently final decision – for instance in deciding whether a bug is valid or not, or when a bug is closed or not. We have seen the latter example in the negative marking and calendar issues in section 4.2.3 above, where responses were made after the issues had been officially closed. We do not know whether these responses will lead to a successful outcome or not (or even how we would precisely define that outcome given that issue closure is in many ways the horizon of our research question). However, the negative marking issue was reclassified from being "not a valid bug" to being a valid one after a participant responded to the maintainer which shows that decisions can be contested if the submitter is willing to engage.

Moodle HQ Development Manager Michael du Raadt also highlighted this important aspect of issue closure:

I did a bit of a clean-up at the beginning of the year that I think was productive because a lot of issues had been sitting there for over a year and no one had even touched them and they weren't related to a supported version anymore. So we shut down about a thousand issues **yeah,** yeah. Again it's sort of like saying: 'we are never going to get to this**.** So let's be realistic about it, and if this is still a problem let us know' (Interview with Michael du Raadt, 2012).

In such cases the onus is on the issue submitter or supporter to come forward and make the case as to why their issue should be re-opened. The definition of the validity of an issue now becomes whether someone cares about it as demonstrated by their *responsiveness*. It is also worth noting that although issues are generally thought of as involving a set number of interested parties, they can also be acted upon by a person who is interested in issues in general, as du Raadt's closing of thousands of issues shows.

4.3.1.3 Validity of the Issue

Once an issue is reported someone must determine if it is valid or not. Validity is an etic term we can use to explore the classification of issues in the tracker. This is a very fundamental decision as anything classed as invalid – termed “not a bug” when reified in the tracker itself – will never be resolved (unless that classification changes). Determining issue validity is not always straightforward and may be time-consuming. The cost of this time can be a frustration, a detractor from the time available for “real work” on valid issues. This is part of the reason that the quality of the initial submission information is deemed so highly, as it may allow quick discernment of issue validity.

A commonly cited source of invalid issues is a lack of knowledge on the part of the submitter as to how Moodle works. For instance someone may request something that actually already exists:

> Somebody yesterday created a tracker issue and a forum issue about wanting a certain feature. Now they didn’t know that that feature also actually existed elsewhere. So – and that’s part of the problem – so people may ask for something that already exists (Interview M7).

Established tracker users believe that these types of mix-ups should ideally be sorted out in the discussion forums (not the tracker).

The other main issue invalidity reason is issue duplication, which as one participant points out, can be easily made:

> 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).

Duplicates are partially a consequence of the volume of issues in the tracker. Although there is a search function, and issue submitters are encouraged to first search to see if their issue already exists, this is not straightforward. For instance people may describe the same issue using different language. Also, if a duplicate does get into the tracker, it may not be immediately obvious to potential assignees that it is one:

> And again that's just the difficult job of knowing what all the issues are in the Tracker **the amount of information that's in there** yeah, yeah and to be able to link similar issues together and to have in your head – when a new one comes in – to think: oh yeah that is the same issue that I triaged two months ago and to find that other issue and to link them together, you know it's, it's a big job (Interview M6).

There are also grey areas around the definition of validity and it may be a contested concept for assignees and non-assignees as the two following contrasting views attest. The first excerpt describes a dispute over a definition of an issue and its closure from an assignee perspective. The second quote gives a non-assignee perspective of issues being closed that they feel are still valid and should remain open.

> I did get one the other day who's – I dunno – he got really, got really shirty 'cause I closed his issue 'cause it wasn't a bug and I said well you know actually this isn't – it was a feature request – well actually you couldn't really do that and I closed it. Anyway then he got annoyed 'cause, well, to start he hadn't understood and so on which is kind of irritating (Assignee view).

> I think one problem that I've had recently is that I've found things in version two and there's already an issue raised for version one point nine, and I think that one point nine gets ignored and I've struggled a bit, to, to encourage people to add two as an affected version so that people will look at it. Because if they don't have two as an affected version then I get a problem of them actually… taking that seriously. They are quite often closing the issue and somebody will come along and say 'don't close it' and then I'll say add 'it affects two' to every single one and quite often that's the only way I get them to do it (non-Assignee view).

This last excerpt also points to a possible factor involving Moodle version which no other respondent mentioned. The participant believes that issues relating to Moodle 1.9 are being neglected relative to those of Moodle 2.

4.3.1.4 Behaving Correctly: Submitter and Normative Behaviour

An unexpectedly common factor that participants believed to be important in successful issue resolution was the behaviour of the submitter (seven interviewees mentioned this in some context and two went into some detail). Participants could be strong in their view that certain normative behaviours should be expressed, such as politeness and patience. Most participants prefaced any comments around this by saying that the majority are well behaved: "really nice people who do their best for Moodle as the project and then you can feel that you are 'on the same ship'" (Interview M2); or "most people are really agreeable and understanding" (Interview M19).

Examples were given of submitters who are impatient and impolite: “Some people might see a feature request as a feature demand. It's not. ‘I want this stuff and when is this being done?’” (Interview M7). The assignee below outlines how their perception of submitter behaviour influences which issues they will choose to work on and this view was not uncommon amongst assignees in general:

> But there are few that arrive and have a very… feel like they [inaudible] than anyone else and that their issue is a bigger deal than anyone else and it should be fixed and it’s not appropriate that someone’s been y’know that they haven’t had a response in one week or they haven’t heard anything in two days and they get very upset about that and expect volunteers to jump to and in a lot of those cases I’m more to ignore it completely than to come and help. 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 ha, ha (Assignee view).

The assignee quoted above makes reference to “volunteers”, which some of the assignees may be. Assignees are often paid to work on Moodle as part of their job but volunteer help outside of that, or may manage to include some issues into their workload that are not directly work-related. Some participants make this link between volunteerism and behaving correctly on the part of the submitter. Submitters are not paying for a service, and ergo should mirror the “good” behaviour of the volunteer. Although there is no direct monetary price to getting an issue resolved by a volunteer there is a behavioural one – as the volunteers are acting ostensibly selflessly they can demand this behaviour.

Newcomers must make an effort to understand the “ethos of the community” and for one participant they may require induction into 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). Part of this type of behavioural assignation can be understood in identity creation and this is something we will return to later.

4.3.1.5 Getting Attention: Votes, Comments, etc.

The case for an issue being fixed should get stronger when more people are in favour of its resolution. This is an intuitive notion and the tracker includes a straightforward mechanism – voting, to allow users to express their support for an issue. As we have seen however, votes can be an imperfect measure of how many people are actually interested in an issue. So it may not be surprising then that participants had differing views on the importance of voting. Several believed the number of votes to be an important factor. One respondent appeared to have a good understanding of the actual distribution of issues (as shown earlier) professing the view that vote counts less than 20 were not relevant. Moodle HQ Development Manager Michael du Raadt also cited 20 votes as a level at which the number of votes might come into play and 100 votes as being “hard to ignore” (Interview with Michael du Raadt, 2012). He made a distinction between critical numbers of votes for new features, which would need to be higher, than those for bug fixes.

Martin Dougiamas also mentions votes as being important, saying that the “most voted” for issues are ones that will be prioritised. He also introduces a note of caution about how high numbers of votes may come about:

> Sometimes the issues that have been around a long time have a lot of votes simply just from longevity. So they’ve had a vote every week for six years and you know then there is an issue that was just filed last week which has twenty-thirty votes on it **wow** so that’s like a hot button issue **okay**. But, if you look in the voting history, you can sometimes see block voting. So a company like [Moodle Partner Company] for example will – they did, they don’t do it anymore because we told them off – they would tweet around or on their internal chat: ‘Just go look – this issue is bothering us. Everyone go vote for it’. So you get this block vote. So pure numbers is taken with a grain of salt (Interview with Martin Dougiamas, 2013).

Most of the fifteen interviewees who mentioned votes, attempted to address the question of the importance of voting by reference to the mind-set of the assignee or Moodle HQ (particularly Michael du Raadt) i.e. by speculating on their point of view. One assignee respondent however appeared to view this from a more causal perspective i.e. rather than considering what the assignees might or might not think, they looked at what actually happened, in their experience. This led them to conclude that voting was not particularly important:

> I think having lots of people involved [is important]. I’m not sure that the voting really has that much effect but commenting I think does that’s…if you get lots of people commenting, and if you get quite influential people commenting, yeah then you’ve got a chance (non-Assignee view).

Aside from voting and commenting there is a range of other ways that attention can be brought to an issue, such as through blogging or tweeting about the issue, or by contacting someone directly. The Moodle.org forums are also relevant if “people post in the forums and they mention a tracker issue” (Interview M5). The forums may provide a link (in all senses of the word) to the issue in the tracker, alerting people to its existence there, and in turn prompting them to log into the tracker and vote or comment on it.

This overall process can be summed up as *getting attention on an issue* – an emic phrase which became the code for this factor. One participant described the start of the issue lifecycle as when “a user would submit an issue, then there’s a process of *trying to get attention to the issue* [emphasis added]” (interview M3).

4.3.1.6 Type of Issue: Bug versus New Feature

As we have already seen above, a valid issue is statistically more likely to have been fixed if it was classified as a bug rather than a request for a new feature or an improvement of an existing feature. This is a stated policy of Moodle HQ (Interview with Michael du Raadt, 2012) and is well believed by participants i.e. no interviewee expressed a contrary view. A requested new feature, although it may happen, will likely take time, or significant effort on the part of the participant, and their chances will be increased if they can provide the code themselves (as we will examine later). One participant sums this up in outlining that a new feature development “would normally take someone with the enthusiasm to do something or someone with the funding to do it” (Interview M4).

4.3.2 What is the Process of Issue Resolution in the Primary Submitter Domain?

So far we have explored factors believed important to issue resolution that are closely tied to the role of the submitter, which goes towards answering the first part of our second research question. The second and related part of our question concerns the *process* of issue resolution. In the coding several of the factors were themselves process codes (provid*ing* information, gett*ing* attention, exchang*ing* story steps etc.) but the second of the research questions is seeking to examine the *overarching* process of issue resolution. Consequently the factors identified from the initial stage of coding, and grouped during the second stage (in the case here under *submitter*) were conceptualised as a process. This was achieved by arranging the factors iteratively into visual displays using the Microsoft Visio application. This allowed the factors to be modelled as elements of potential processes. As we tell the story of how issues progress, in narrative form, we can now also map this progress graphically. The diagram in Figure 20 below shows prospective pathways to issue resolution considering the submitter primary domain factors that we have looked at thus far:

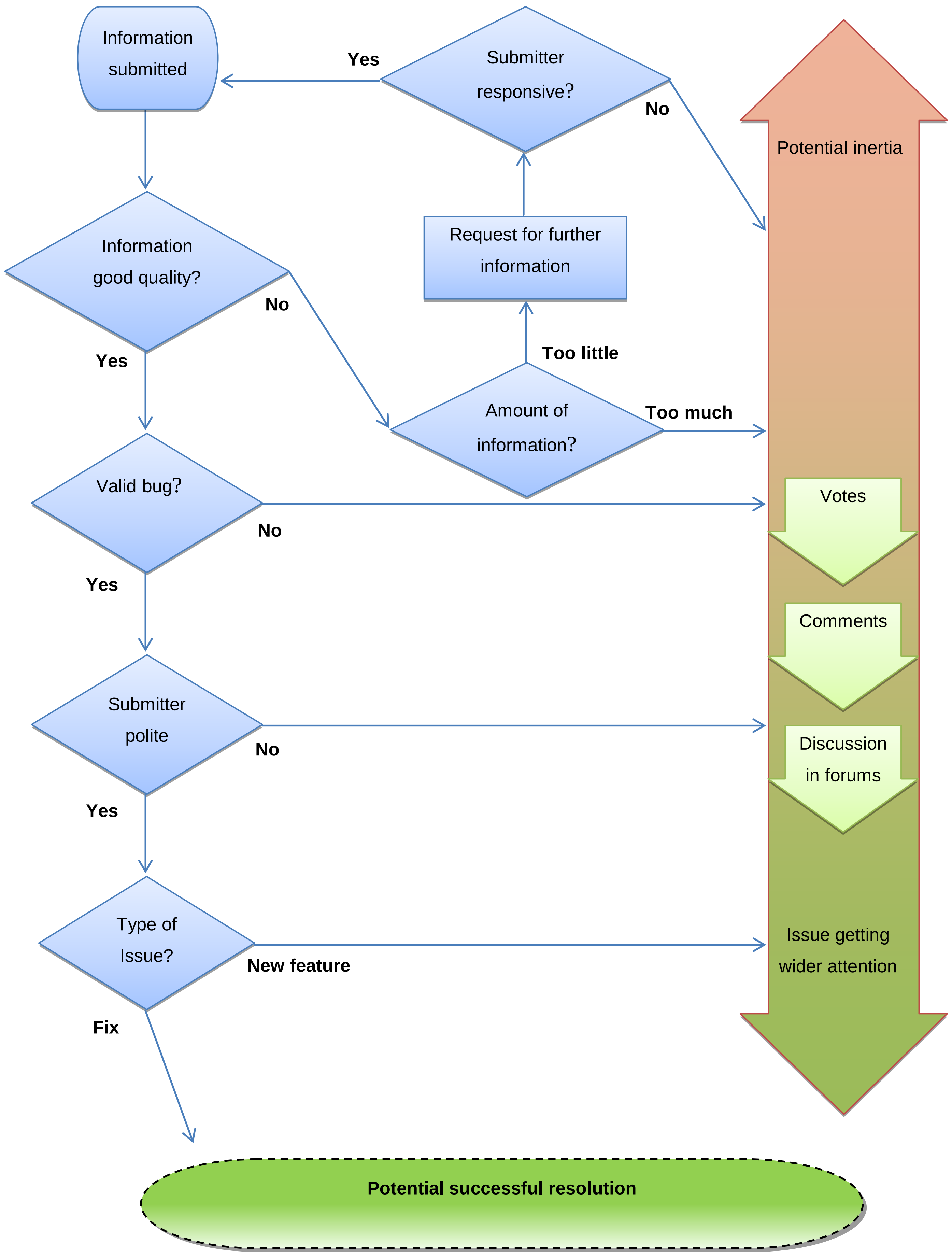


Figure 20 Paths through Submitter Primary Domain

According to Miles and Huberman (1994) there is no standard or even dominant method for in-case display of data in qualitative analysis. The chart above is derived principally by using the flow chart method (Neustadt, 2006) where diamonds represents decisions, or potential branches, in the path and arrows emanating from the diamonds denote the alternative routes that may be taken based on the answer to the branch question. The flow chart method, if most strictly followed, would have led to a very convoluted, and also brittle, chart claiming to describe completely determinable and mechanistic processes. Hence this display type was adapted to encompass elements of network display types which are useful for "combining 'process' and 'variable' analyses" (Miles & Huberman, 1994, p. 160). Hence Figure 20 above should be read not as showing factors as strict prerequisites of each other, but instead as indicating a relative primacy of certain factors and including the proviso that these factors exist in a fluid system and do not always follow a logical progression. To express this, the arrow stretching from top to bottom along the right-most side of the chart represents a probability continuum that moves from less likelihood of success at the top towards more likelihood at the bottom. If an issue fails a test in a diamond it is often shown as then going directly into this arrow. This does not mean that the issue can no longer progress through the next steps in the flow chart but rather that its chances of doing so have diminished. Each step represented in the process is thus a probabilistic rather than a deterministic one.

In Figure 20 *Paths through Submitter Primary Domain* above the first thing to happen is that information is submitted as part of a bug request opened by a submitter as indicated by the rectangle in the top left corner. If this information is not deemed to be good quality information we move to the amount of information diamond. From here if the amount of information is deemed to be too much we then move to the continuum arrow on the rightmost side. What this means is not that this issue will never be resolved but that its chances are lessened. However, on this continuum arrow on the rightmost side we can see the three factors of votes, comments and discussion in the Moodle.org forums. These factors i.e. "getting attention on an issue" (interview M3) may or may not come into play. If these factors do come into play then the issue is more likely to be successfully resolved, an outcome which is represented by the oval at the bottom of the diagram. If they do not i.e. the issue receives little further attention, it is more likely to languish i.e. come under an inertia. This is represented by the top of this arrow which indicates that the issue is moving in the opposite direction of a possible successful resolution, the issue is not moving at all and may "sit there for a long time" (Interview M5) until it "kind of dies a death being open" (Interview M85).

Although this domain represents factors the submitter is considered to have input or even influence over, and here we are taking submitter to be a non-assignee and a comparative outsider to the system, it does simplify a submitter's influence *relative to the assignee*. For instance, many of the concepts in the decision diamonds are, it is arguable, primarily determined by the assignee e.g. whether the submitter is being polite, whether the information is of good quality, whether the bug is valid, whether the issue is a fix or a request, etc. Some of these factors may be relatively clear cut, such as whether the issue is a fix or a new feature, whilst others, such as whether the submitter is behaving correctly, may be more in the eye of the beholder and greater subject to an assignee's whim. However, what distinguishes the submitter domain from the assignee domain, which we will examine next, is that, power disparities aside, decisions here are more contestable. Knowledge may be negotiated here. Taken from this perspective, politeness and responsiveness may be two important items in the submitter's armoury during this contest.

4.3.3 What Factors are Important to Issue Resolution in the Assignee Domain?

Thus far we have examined factors that are important for an ordinary submitter to give their issue a momentum that might lead to a successful resolution. If they have success meeting some or all of these factors, they may then encounter other factors over which they have less control. These factors are examined below in terms of the assignees' role and their part in the process.

4.3.3.1 Inactive Component Leads

A prominent factor that may affect an issue is the presence of a suitable assignee. Many areas of Moodle have a designed "lead" or "maintainer" e.g. Tim Hunt for the Quiz, David Mudrak for the Workshop, Dan Marsden for SCORM (an interoperability standard). The maintainer/lead is the obvious assignee for the issue. However some areas of Moodle have no such obvious guardian. This may happen when the person who originally worked on a particular part of Moodle has now left the project. The calendar import issue for example may have been complicated by the fact that one of the original developers of the calendar functionality was no longer working on Moodle. There may in effect be orphaned parts of Moodle.

One participant cites the presence of an active (and sympathetic) component lead as one of the most important issue resolution success factors:

> One [factor] is an existing and sympathetic component lead. It's probably one of the biggest factors if there's no component lead for that component then it's… you've got a potentially a bit of an uphill struggle**.** They are working on it. I've seen a lot of progress eh – in terms of – eh they are tightening things up and Michael du Raadt is I think seems to be very much in charge of going through all open issues and triaging them […] (Interview M10).

As highlighted above Michael du Raadt may intervene at this point and triage the issue which includes potentially assigning it to someone else. This is an act of brokerage that we will return to in the examination of roles under research question three. The implication here is that a broker may link an issue to a prospective assignee but that this results in a looser coupling between assignee and issue, and thus a lesser issue resolution likelihood than in the case with a canonical lead. Or, more accurately, it may lengthen the time to a prospective fix. One participant assigned an issue for a module for which they are not a maintainer professed not to have "time to look at it [now] so I'll leave it in the email queue and if I happen to have time someday I'll happen to have a look" (Assignee view).

4.3.3.2 Receptiveness of the Component Lead

The personality or the mood at a particular time of the component lead may be an important factor. The assignee to an issue may be "receptive" or "sympathetic" (Interview M10). Participants reported this as it applied to particular issues and one participant also described this in terms of assignees' inherent personalities, saying that they believed some to be less receptive than others (particularly where code patches were involved) (Interview M5). There is a link here to the behavioural expectations that assignees make of submitters which we have discussed already, where assignees claimed that well-mannered participants had better chances of having their issues progressed.

One non-assignee expressed the view that the (assignee) Moodle community should be more receptive in general terms, portraying a sense of disconnectedness between both sides:

> I think they should do more like communicating, not just here is the solution and just take it but more like – I don't know how to say it – to give the people who contribute the issues just more the feeling that they could, that they do a good job and that the Moodle core community needs those people **yeah** you know what I mean? (non-Assignee view).

It would appear that behavioural norms required of the project may not always be reciprocal or at least that there is a disparity whereby an issue submitter must behave in a correct manner, though the assignee may not always choose to do so. Sometimes this may be implied, such as a respondent who expressed the view that it takes a “brave person” to report a bug. One assignee claimed that assignees could sometimes be intimidating:

> It can be daunting for new developers. Core developers know each other for a long time. They can argue without using, how should I say it, very flowery language (Assignee view).

A non-assignee, in giving their description of a successful issue lifecycle, lists two main steps – the first being to describe the issue well and the second as to be ready to “really to fight with the developer” (non-Assignee view). This may be apparent in the calendar import issue discussed above, where a submitter of a fix is forthright in their views and forceful in their attempt to progress the issue (which is ultimately resolved). This suggests that politeness and patience on the part of the submitter is not the best policy in all cases (though, perhaps unsurprisingly, putting your case forcefully as a non-assignee was never promoted by assignee interviewees).

4.3.3.3 Canonical and Non-Canonical Issue Classifications by Assignees

We have looked at two canonical issue classifications already: whether a bug is a fix or whether it is a new feature request, with the former being given a higher priority. These are canonical concepts because they have designated fields in the tracker for their representation and are widely used terms in Moodle documentation. A related canonical concept is whether the issue is deemed a “critical” one or not. This was cited by many interviewees, with any security implication being a prominent reason why it would be critical to fix an issue. Other reasons could include loss of some significant functionality. Criticality could be a more abstract classification of an issue than whether it is a fix or a feature request. It has a reified representation in the tracker, where a flag can be added to an issue indicating its level of criticality. The highest of these levels is “blocker” implying that no progress can be made until this issue is dealt with.

Criticality may be related to the level of interest in an issue such that it represents “a block to say multiple Moodle partners and all their clients and that’s a lot, a lot of people” (Interview M7). Equally it may be the case that an issue can be critical “even if there are only a few people who have actually reported it" (Interview M16) which would mean that issue criticality can be determined even if an issue has not generated a lot of attention.

Other factors that affect an assignee's decision to work on an issue are more abstract and non-canonical such as whether the issue is "annoying". An annoying bug may be "nasty" and "get on people's nerves" (Interview M9). Similarly issues may be deemed "interesting" (Interview M17) by assignees and there appear to be cases where someone may feel a compulsion to fix a particular issue. The reason they wish to fix such an issue is defined in more emotional terms than a critical issue may be.

Another factor that may enter an assignee's decision making process is whether an issue represents a "big win" which we can define as *the effort of developing the fix or feature being low relative to its pay-off.* As one participant put it in relation to an area of Moodle for which they are not responsible:

> I mean if I spot something, and it's really obvious, then I may just do the fix myself because it's not much trouble (Assignee view).

The phrase *big win* was used in the context of issue complexity by another participant:

> It really depends on how complicated. If it's something simple its much, and it's something simple with a big win, it's more likely to get picked up (Interview M10).

New features may not be big wins as they may require significant effort to develop, whereas fixes may require less. Conversely, the concept of big wins can also help explain why some issues that have generated a lot of attention take a long time to resolve i.e. that the effort required is non-trivial. The submitter has an important role in getting their submission right so that it may constitute a big win but ultimately an assignee must make a judgement on imperfect information (because they cannot be certain how long a fix will take them). Hence this represents a part of the issue lifecycle that may be hard to predict and where the assignee plays a key role.

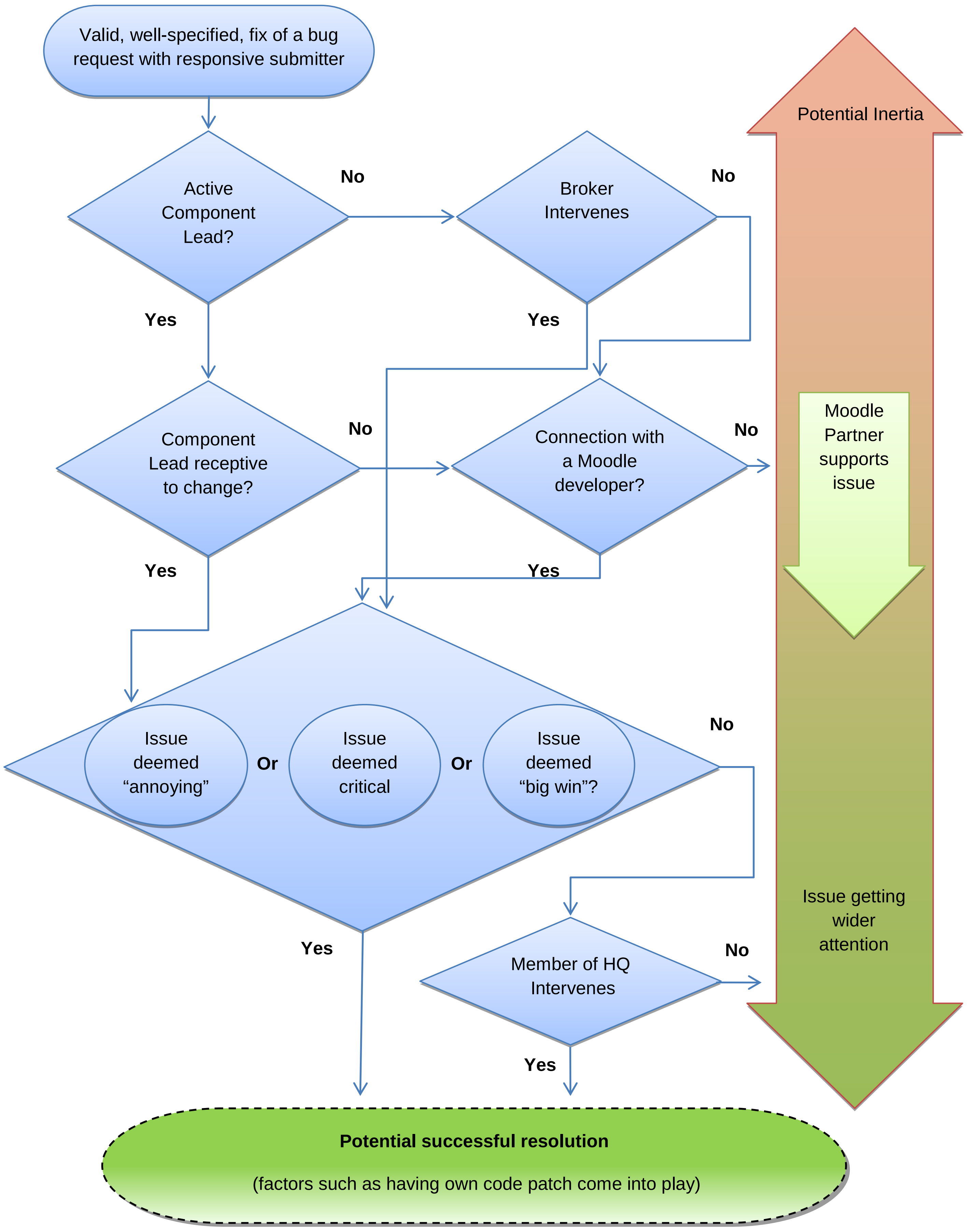
4.3.3.4 *Influence of Moodle HQ and Moodle Partners*

Moodle Partners' influence was believed to be strong by many of the interviewees. There is a special flag in the tracker that a Moodle Partner can add to an issue to signify their interest in it that a non-Moodle Partner cannot. (Although one Moodle Partner employee interviewed was not aware of this). A range of views emerged from interviewees on Moodle Partners and their role in the process. Most participants believed that their involvement made an issue more likely to be resolved for two key reasons: because they support Moodle financially but also because they can represent a proxy for a large number of end users e.g. a Moodle Partner may have a number of universities and companies as clients who use Moodle. The project may give “priority treatment to an issue that a partner has identified because that means that their users are telling them we need this” (Interview M3). However it was also acknowledged that the presence of a Moodle Partner did not mean that an issue would always be resolved.

Unsurprisingly Moodle HQ was deemed by participants to have a large influence on issue resolution. Martin Dougiamas and Michael du Raadt were cited by interviewees as having important roles in the process. Dougiamas was believed to be a decision maker on important or contentious issues and he confirmed that he would get involved if developers “are split or there is a difficult decision” (Interview with Martin Dougiamas, 2013). Most importantly however Moodle HQ may intervene in some way to progress an issue that might have otherwise have reached a dead-end.

4.3.4 What is the Process of Issue Resolution in the Assignee Domain?

Now that we have discussed factors believed important to issue resolution through the prism of assignee/maintainer influence we can arrange them in a display showing them as a process of possible pathways. *Figure 21 Assignee Domain Issue Resolution Process: Paths to a Fix* below, is a graphical depiction of the issue resolution factor codes that were coded to assignee during the second coding stage. It outlines factors, and the related process, that participants believe are important in potential issue resolution paths where assignees play an important role:



Valid, well-specified, fix of a bug request with responsive submitter
Active Component Lead?
No
Yes
Broker Intervenes
No
Yes
Component Lead receptive to change?
No
Yes
Connection with a Moodle developer?
No
Yes
Issue deemed "annoying"
Or
Issue deemed critical
Or
Issue deemed "big win"?
No
Yes
Member of HQ Intervenes
No
Yes
Potential successful resolution
(factors such as having own code patch come into play)
Potential Inertia
Moodle Partner supports issue
Issue getting wider attention

Figure 21 Assignee Domain Issue Resolution Process: Paths to a Fix

The first element of the chart in Figure 21 above represents a “valid, well-specified, fix of a bug request with responsive submitter”. This may be thought of as the output of the previous chart Figure 20 *Paths through Submitter Primary Domain* i.e. the submitter has done everything within their gift successfully to this point. The dominant process in the assignee domain chart is the large diamond in the centre. This represents the assignee as decision maker. It emphasises the strength of their role in deciding whether a bug is annoying, critical or a big win. Important decisions are often “up to the maintainer” (Interview M12). We could also add whether a bug is valid to the decisions that the assignee/maintainer may make – although we have put this in the primary submitter domain, it could also sit here. Adding these decisions together we can conceptualise the assignee as an important decision maker with a powerful role. One participant described the assignee’s decision-making domain as; “One – deciding whether it's a bug or not; two – whether it's a feature request or not; and three whether it will happen or not” (Interview M7). The last of these actions illustrates well the perceived power of the assignee.

4.3.5 What Factors are Important when the Submitter has their own Code?

If the submitter has the capacity to develop some code to fix the issue (known as a patch), or commission one, they may have an advantage over an ordinary submitter. With their own code patch the submitter may be able to bypass many of the factors in the Submitter Primary Domain process. Submissions with patches differ from ordinary submissions because a good patch may contribute to a “big win” i.e. if there is less effort required on the part of the assignee/component maintainer because the code is already written and they may be more willing to help progress the issue. If there is no obvious active assignee, being able to propose your own code, may be almost the only way to ensure a timely resolution of an issue.

4.3.5.1 Creating Solutions in Code

For a patch to contribute to the resolution of an issue it must satisfy certain criteria. The first may seem an obvious one – that the patch fixes the issue. This is the base requirement. Then a distinction must be made between a patch that fixes the issue and a patch that fixes the issue in a way that is acceptable to Moodle HQ. There are various reasons that a solution may not be deemed acceptable, for example it may not provide much actual benefit or that benefit may not extend much beyond the submitter i.e. the submitter may wish to use Moodle in a very specialised or atypical way. Instead a good solution “should be helpful to the general public” (Interview with Martin Dougiamas, 2013). Alternatively a better solution might exist that solves the problem in a more “elegant way” or that solves that problem but also some other related problem at the same time. The reasons why a solution may not be deemed acceptable are therefore not straightforward to determine.

In addition to considering the solution in the abstract, there are also particular specific ways it must be implemented. There are detailed prescriptive canonical conventions for writing code. In a similar manner to styles and referencing rules that must be followed in an academic work, all code submitted to Moodle must follow detailed coding guidelines. Just as a badly placed comma in an in-text citation may spell disaster for an academic writer, so too a Moodle code submitter must be vigilant about how and where they place each character in their program:

> We really care about how code looks because we look at each other’s code constantly […] We make quite a big deal about coding guidelines and occasionally there’ll be half an hour discussion about whether there should be a colon in this position or not and it gets a bit mad but, you know, we do like to make it read readable and consistent (Interview with Martin Dougiamas, 2013).

Although there are detailed canonical rules around how to write code, these rules are evolving all the time (as implied by the quote above). One respondent was exasperated that some rules were “nowhere written down”:

> They expect you to provide everything completely conforming to the to the agreed standard sometimes even to agreed standards that haven’t even been written down [...] I asked several times, what are these requirements? What is wrong? And no-one would answer until I realised that they all use a code-checking tool provided by the Open University, which complains about certain, which somehow informs you about certain rules being violated but if you check with this tool and it conforms, then they will accept it. So no-one has written down the rules, but they all rely on this tool **okay so you’ve managed to figure that out** yes, yes. So there are a number of these unwritten rules about comments that you make when committing to Git and so you spend a lot of time in the beginning getting used to it (Interview M14).

Getting your code into the correct format requires ‘determination’ and ‘perseverance’ (Interviews M5, M14). There are practices of the group that “haven’t been written down” (Interview M14). The above excerpt provides an interesting counterpoint to a view that the rules to be followed for code submission are explicit. For instance Interviewee M10 first answered the question about the issue lifecycle by quoting from the Moodle documentation directly before they were prompted, in a follow-up question, for their “own experience” of the issue lifecycle in an attempt by the researcher to uncover non-canonical representations.

Reference to canonical descriptions such as technical standards, to the ways things “should” be done, can be a stratagem employed by an assignee who cannot, or does not want, to progress an issue. One participant described how a submitter may be asked to submit a patch themselves. If they do not know how to create a patch they may receive a “reply with a url for the wiki ‘how to make a patch’”, the participant knows this tactic “because I am used to give that kind of reply, you-know in my [own] work as well as in other projects” (Interview M12).

4.3.6 How Does Having Your Own Code Affect the Process of Issue Resolution?

Even if a submitter has a proposed code patch, the assignee may decide to write it themselves instead. Or, perhaps, the patch will go in with the proviso that the submitter commits to maintain this piece of Moodle into the future. We will return to these two scenarios but first we can show the processes involved in the domain arrived at by the submitter where they have the wherewithal to proffer their own solution to a problem in code:

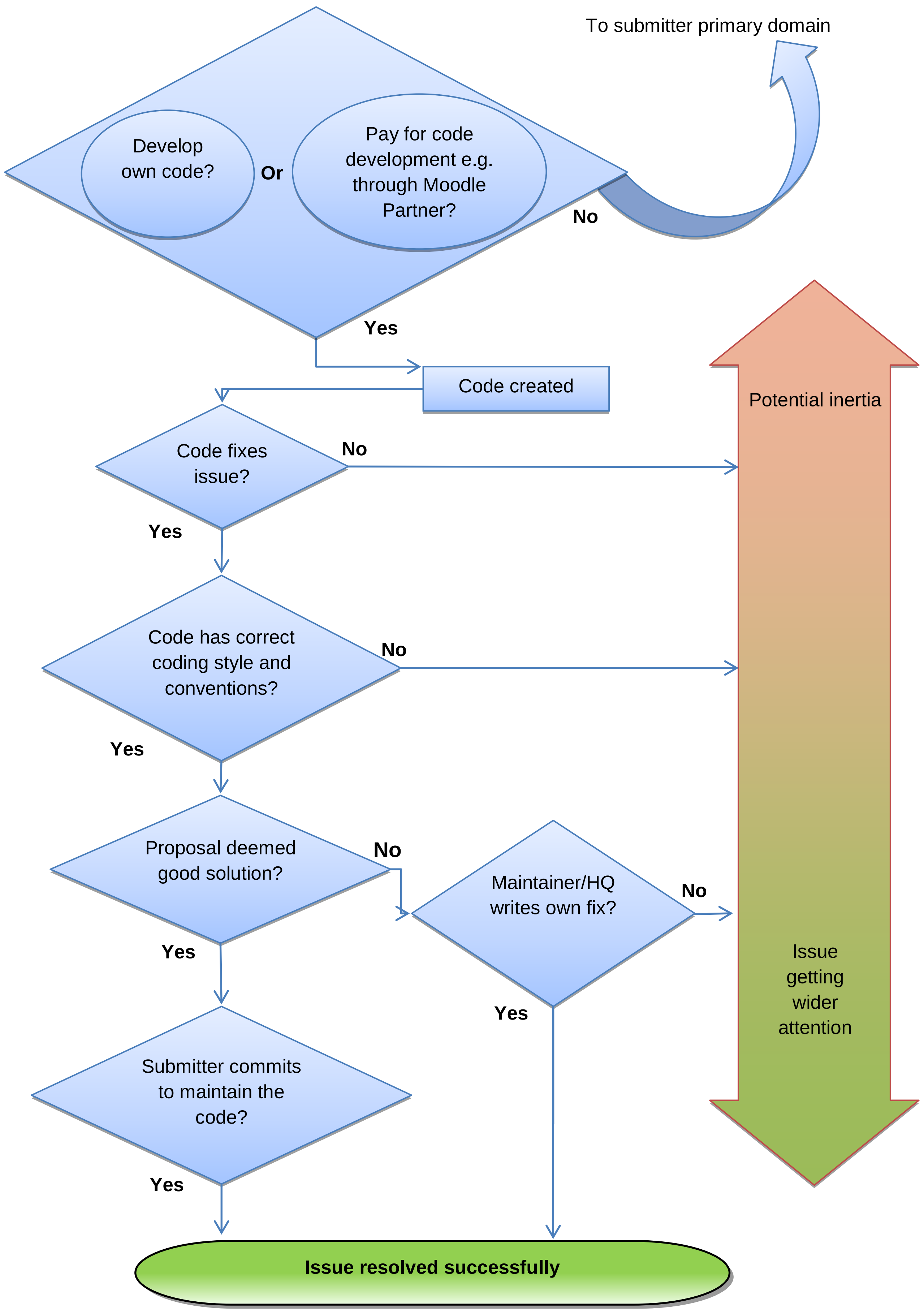

To submitter primary domain
Develop own code?
Or
Pay for code development e.g. through Moodle Partner?
No
Yes
Code created
Potential inertia
Code fixes issue?
No
Yes
Code has correct coding style and conventions?
No
Yes
Proposal deemed good solution?
No
Maintainer/HQ writes own fix?
No
Yes
Issue getting wider attention
Yes
Submitter commits to maintain the code?
Yes
Issue resolved successfully

Figure 22 Submitter Code Domain

Figure 22 Submitter Code Domain above shows possible paths that may be taken for a submitter who has coded a representation of their problem and offered that code as a resolution to the issue. Factors that may be more under the control of the submitter (such as whether the code actually fixes the issue or is in the correct format) are closer to the top of the chart. Factors such as whether the proposal is deemed a good one may be further outside of the submitter’s influence and are closer to the bottom of the chart. These important decisions in the leftmost columns of diamonds of Figure 22 are made mostly by Moodle HQ, although a non-Moodle HQ component lead may also have a say and be involved in reviewing the code. Ultimately however all code must go through a final Moodle HQ based audit.

4.3.6.1 What is the final part of the Issue Resolution Process?

We have now examined almost all factors uncovered during the interviews that participants believed important in leading up to successful issue resolutions. To examine the final resolution itself we can narrow the outcome to:

- whether a submitter gets their own patch accepted
- whether they get their proffered solution accepted but not their code
- or, whether neither their code nor their solution manage to resolve the issue

4.3.6.2 Code and Solution Get into Moodle

In scenario one a patch may eventually make its way into the core or canonical Moodle. This code has been written by the submitter, or perhaps by someone on behalf of a submitter, but it has come from outside Moodle. As we come to this stage in the process it makes less sense to talk about assignees and even component maintainers because they are not writing the code (the submitter is) but also because, according to processes instituted around the end of 2010, Moodle HQ is involved in a very systematic process for all code acceptance into Moodle. Even a component lead submitting their own code must submit to the processes of Peer Review, Integration and Testing by Moodle HQ.

This integration process was not in place for the entire duration of the case study. Hence interview data and issues examined here are also reflecting an older (pre-December 2010) code submission process which was much less formal and where assignees were effectively given a large amount of trust to put code into Moodle without that code necessarily undergoing rigorous scrutiny or conformance tests. In addition, although the integration process in itself could be worthy of another level of analysis, it is sufficient for our purposes to use the *approval of Moodle HQ* as a general term that can cover the Integration, Testing and Peer review process. The main point to note about this process is that "new code receives multiple reviews by different people before it is included into the core Moodle code" (Moodle.org, 2013a). That is, once a component maintainer has assented to the code patch that the submitter is proposing (termed Peer Review in canonical terms) it is then subject to a further review by Moodle HQ (Integration and Testing).

There may exist shortcuts, for trusted component maintainers or assignees, through some of these processes. For instance, they may be able to themselves bypass peer review if they are granted certain rights (known as pull requests). Or they may be able to sometimes bend the rules:

> If I think I know exactly what's going on I'll do the fix and submit it straight to integration. I won't ask anyone else to peer review it because it adds more time to the system. So if it's something simple and I think I really know what's going on y'know 'cause I'm supposed to be a peer reviewer for most stuff I kind of yeah feel I …That's slightly naughty but anyway but sort of pragmatic (Interview M5).

4.3.6.3 Solution but not Code Gets into Moodle

Another possible scenario is that Moodle HQ, or perhaps the component lead if they are not a member of HQ, takes it upon themselves to write the fix, even though someone else has proposed a patch. Depending on the submitter's motivation and outlook this may be an almost equally desirable outcome to having their own code accepted. Some assignees expressed the view that even if a patch was badly written it could nonetheless provide a good guide for their preferred solution, or more aptly provide a guide to exactly what problem the submitter was trying to solve. In the case outlined below the submitter's idea but not their code gets into Moodle:

> Sometimes you contribute a patch and then someone of the core developers decides to resolve it in a completely different way. That happens of course. This causes duplicate work but maybe it's not avoidable (Interview M14).

If the submitter's main motivation was the resolution of the problem then they may be satisfied with this. Although as we will cover later, a submitter may see the ability to write code as being important to their identity, so having your code discarded in favour of someone else's (as hinted at in the last quote) may not be ideal.

4.3.6.4 Patch Inertia: Neither Code nor Solution Get in

There is also a third alternative possible where neither code nor solution gets into Moodle. There are two broad possible reasons this may happen. One is that the submitter no longer needs to get their code in. The momentum of the trajectory for a potential successful issue resolution may diminish once a patch is arrived at by the submitter. Once a patch has been developed it may be applied to a local Moodle installation only. The person has now solved their immediate problem or resolved their need. A trade-off must now be made between the value *to them* of submitting this code to canonical Moodle and the effort required to do so.

Another possibility is that the patch languishes for unknown reasons, perhaps because it has not come to the attention of the right person, or because of the sheer volume of issues: “It often took quite a while, or even infinity until someone would eh actually pick up [my patch] and integrate it” (Interview M14). Some respondents cite luck and randomness as factors, but they highlight the workload of assignees due to the relentless sea of new issues that are submitted every day: “there are only so many developer hours and they’ve got to prioritise things” (Interview M10).

4.3.7 Issue Resolution Factors and Processes Summary

The exploration of the successful issue lifecycle was carried out by extracting all of the potential success factors from the analysis of the interviews. A sub-section of these factors were then grouped according to those most within an ordinary submitter’s gift or domain. The next set of factors was examined through the lens of the assignee, or component maintainer, as these are the ones lying most within his/her domain. A third category of factors was identified in terms of a submitter who has the means to themselves write code, and here we also examined the final steps of how innovations to the Moodle codebase may be made. Flow charts were used to organise the success factors into processes and to illustrate possible pathways. Participant voices, by direct interview quotation, were used to earth these concepts in the reality of community members’ experiences and where an affirming or disconfirming data source was needed references to the Moodle.org discussion forums or the tracker were admitted. This section addressed the research question concerning the factors that are involved in successful issue resolution and the related processes involved in decision-making and meaning negotiation. However, these questions still exist in a relatively disembodied way from the actors involved; hence the next stage is the examination of the community itself through the identities of its members and their sub-groups. This will allow us to further explore who the members of the Moodle bug tracker are and how people become involved in this community.

4.4 What are the Key Identities and Roles of the Participants?

4.4.1 Introduction

Different people play important different roles in Moodle and we have made an exploratory examination of the characteristics of participants as a whole such as where they work, their professional roles and their educational backgrounds. The Moodle tracker community is comprised of a diversity of people such as teachers, software developers, lecturers and educational technologists. Examination was also made of the characteristics of participation by assignee insiders and non-assignee outsiders e.g. how inner community members contribute more on average and to greater effect than fringe ones. This constituted a broad preparatory exploration of the roles of the community but not much about their identities. This third research question must also be addressed with reference to data from the participants themselves (the interviews), to explore how they perceive their own involvement. This question seeks to examine the identities of participants and the impact or interplay of these with the roles they occupy – for instance how self-professed identities or projected identities may impact on what an individual may do within the community.

4.4.2 Identities

We have explored some aspects of individual identity in the tracker profile sample, through educational backgrounds and employment; and also implicitly during the examination of issue resolution success factors by partitioning people into assignees, non-assignees, Moodle Partners and Moodle HQ members. A number of the interview questions also asked participants directly about various aspects of their individual identities. The first question on the interview schedule was an invitation for the interviewee to describe how they came to be involved in Moodle. The closing question asked participants to explain what Moodle meant to them. A probe to the first question asked participants about the particular skills and experiences they felt they had that were important to working on Moodle. In addition, parts of answers to other questions were also coded as identity markers.

4.4.3 Developer Identities

A little over half of interviewees identified themselves strongly as a programmer or software developer (which in common usage is shortened to simply “developer”), two for example made reference to learning to program as children. One interviewee, when asked what skills allowed them to be involved in Moodle, conveyed as much about how they felt about programming in how little they said: “well being able to write good code goes without saying” (Interview M1). This ability to write *good* code is important to the identity of several participants. These people 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). One participant described their involvement with Moodle in terms of their love of programming:

> Well I could say now it’s a job. I get paid for it, but.. I enjoy it! I like the code. I like the work (development). I’ll be honest: I get quite a buzz from seeing people look at my code and go ‘that’s cool’ and yeah I enjoy it. I enjoy the neatness of, I just enjoy coding I like the neatness of producing some things (Interview M10).

Other evidence of this was seen in developers paid to work on Moodle who also worked on other non-paid Moodle activities as “hobbies” or a component lead who referred to the components they developed as their “toys”.

A recurring theme amongst developers in the interviews was the need to “scratch an itch” (Interview M1), that they felt almost compelled to try and fix or solve a problem. Dougiamas described Moodle’s genesis in this way: “I just had this need to build this thing to prove to myself that I could do it” (Interview with Martin Dougiamas, 2013). As we have already seen bugs can be “annoying” or “interesting”, implying an intrinsic motivational impetus to their resolution. This aspect of tinkering and building, which open source enables, is important to these participants’ individual identities. As is learning, which one participant put vividly via a gastronomic metaphor:

> 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).

Participants who identified as developers also reported positive identifications of interacting with their peers and in particular portrayed their *excitement* in this interaction:

> 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! Petr's doing something on security and I know him cool. That kind of stuff excites me (Interview M9).

A person's status as a developer in the community is confirmed by their reputation:

> Petr Skoda is a valuable person. He writes very beautiful code. Let's say I'm one of the fans for code poetry and let's say code has its own aesthetic and looking at the way you divide the solution in small pieces of code. So, basically, Petr is one of the most well regarded developers (Interview M12).

4.4.4 Teacher Identities

Interviewees who had a background in teaching or lecturing did not necessarily primarily identify with that role although some did. Several participants for example had backgrounds in teaching but had moved into software development. Although it was a secondary identification these people still believed their teaching background to be important to their function within Moodle by giving them a particular insight into the end users (teachers and students). They believed their background in teaching gave them a perspective into how people "actually use Moodle […] from an educational point of view, rather than just purely technical point of view" (Interview M8).

Interviewees who were active teachers cited teaching as playing a primary role in their participation in the Moodle community. The activity of teaching may be seen for these people as a form of mediation between the worlds of Moodle's technical intricacies and of education. One participant describes their role in this mediation by identifying as a teacher:

> Also I really like – and again this is a bit, this comes from being a teacher – I really like breaking difficult things down to make them less complex into step by step sequences so that people can understand them (Interview M16).

Another participant describes their role as a teacher as being the foremost attribute they bring to the Moodle community, comparing activity in the tracker and the discussion forums to teaching with the rhetorical question that: "…in the forums, answering questions – that's a real teacher thing to do isn't it?" (Interview M19). Although it may be convenient to divide participants up into teachers and developers, the situation is often more complex. Everyone who works on Moodle is undoubtedly some form of "educationalist". Developers who do not consider themselves teachers may nonetheless work in universities, or possess PhDs or otherwise demonstrate knowledge of pedagogical beliefs and idioms such as of giving feedback:

> Or, you know, this bit of code doesn't follow the coding guidelines. Here's the link, you just need to tidy it up. So that's the standard practice for giving good feedback. Give the feedback on the work they have done and have them correct it, not a personal attack (Interview M1).

4.4.5 Brokers

During the examination of issue resolution above, a participant cited Michael du Raadt's role in looking at issues that may not have an active or obvious component maintainer (owner). As part of the issue "triage" process Michael du Raadt may act as a *broker* and try and find a potential assignee:

> I'll go: 'I know, you know Andrew has been working in that area recently, even though he is not the component lead' **okay, yeah.** I'll pipe it to him, 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 in general connected individuals, though any participant could also be one. Indeed, active community members, such as assignees, may be able to progress an issue that they are not assigned to by dint of their connections. One participant describes how "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" (Interview M18) and this may be explicit:

> 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).

Other participants identified themselves as mediators between technological and commercial or business realms rather than educational ones. One participant described themselves as being greater than the sum of their parts for being: “50% technology, 50% training and 50% business”, a personality composition they believed key to their role as broker between software and business communities: “I’m a bridge between the use case and the systems” (Interview M7).

One common and very relevant brokerage act is that of filing bug reports on behalf of someone else:

> But sometimes, people actually just don’t sign up for tracker accounts and it’s up to… then hopefully some *kindly soul* in the forums will spot it and write it up as a bug report and that often happens (Interview M10 emphasis added).

Some component leads cited these “kindly souls” as fulfilling an important function. These may be key members of the community such as Helen Foster. She is an example of a specialist broker. Although she has an interest in usability aspects of Moodle she also encourages or reports tracker issues relating to any area of Moodle herself on behalf of others (Interview with Helen Foster, 2012). By contrast, Interviewee M12 files issues on behalf of users mostly only for the particular module that they are interested in and will also sometimes go on to propose patches for these issues. There is one other type of broker that is important here, that of someone who inducts new members into the community and who may also eventually confer a status upon them as an accepted member of an inner layer. This is an issue we will turn to later in the context of community joining trajectories.

4.4.6 Inside Moodle Community and Identity

Many participants – unsurprisingly in particular those with most involvement – described the Moodle community in positive terms. Moodle’s open source nature is seen as a cause for good, as is its educational function. Terms such as “friendly” and “very nice people” were common. Many respondents reported positive feelings of being part of a community coming from their involvement with Moodle even for those whose involvement was relatively episodic and solely online: “as I was saying, it means like I’m part of team part of a community” (non-Assignee view). Participants may associate *their function* within the community as an active member with positive feelings towards Moodle: “Well it feels an honour really to be part of such a global community and it makes me pleased to feel that I can help people" (Interview M16).

The success of Moodle is something that several participants displayed a positive identification with and investment in:

> I mean it's been amazing now Moodle has really, really grown in popularity. I don't think anyone eh least of all Martin himself would have imagined how incredibly popular it is and how, you know, you just keep hearing across the Internet these enormous Universities and other institutions moving from Blackboard to Moodle and like Moodle just keeps on going from strength to strength (Interview M19).

Some participants were also expressive of a deeply held view that Moodle served a wider purpose and that they had a duty to play a role in the service of the project, as a selection of excerpts can best illustrate:

> It still amazes me that universities trust their basically their livelihood on top of this thing, you know, that's like a lot of people that are depending on us. […] it's a moral duty. You feel morally responsible (Interview M20).

> So, for me, Moodle is a part, or a piece of, my hope for more open education. I definitely believe in the values put forth through open source software about collaboration, working together, creativity, very much trying to focus on, you-know, what can we do as a society to cooperate and produce things that are of value to the common good (Interview M3).

> Yeah, so this is a kind of a service thing I guess to the world. Everyone has gifts and this is a way of me sort of being able to use what I can do to benefit people (Interview M18).

These types of views were perhaps unsurprisingly very concentrated in members of Moodle's core community though they did also exist for some non-assignees and more peripheral members also.

4.4.7 Outside Moodle Community and Identity

Although the sample bias of people who are pre-disposed to Moodle may be apparent, some more neutral views were also evident from interviewees. Some people worked on Moodle simply because it was part of their job and did not consider themselves "Moodle purists", such that they claimed they would have little compunction about using alternative software to Moodle. Two such participants had an identity that is primarily rooted outside of the Moodle community in the business sphere. One participant saw particular development practices of the Moodle programmers as being derived from the Linux computing tradition and alien to his own ways of doing things which were based around Windows. This participant also thought Moodle was too focused on education rather than on the corporate enterprise which was his primary domain. As such Moodle did not fit his needs and he left the community (though still expressing a positive view of his experience in it).

4.5 How do Educators Come to Participate in the Inner Community of Moodle Development?

4.5.1 Introduction

The fourth research question asks how participation in the Moodle tracker comes about. The identities and roles we have outlined were established by individuals who entered the community at some time. We will now examine these entries as trajectories, ones which are related to activities within the community. In so doing we will also look at the corollary of trajectories: interties. Just as participation can have a momentum – such that change can be affected and knowledge produced – so too can activity halt. Identities are again relevant here and this section will bring together much of the themes that emerged from the analysis of the preceding questions.

4.5.2 Trajectories

One trajectory is particularly important – that of joining and then entering a successive inner layer of the Moodle community via brokers (see Table 1 Key Theoretical Concepts for brokers and brokerage). The brokerage discussed so far has involved a strengthening of connections: a person who has made their way to the forums already has established a connection, this is then strengthened by a broker who files an issue on their behalf or an issue is elevated by a broker bringing it to a developer’s attention on behalf of someone else. In these cases a broker makes use of existing connections in the community. Another important brokerage is the establishment of a new member of the community and it takes a special type of broker or maker to do this. Several participants described how they owed part of their involvement in Moodle to particular individuals such as receptive component leads. The key aspect of this relationship is that the maker has the power to confer particular rights on the participant and hence induct them into an inner layer of the community. One participant describes their induction into Moodle:

> He does a lot of stuff outside of Moodle helping people all the time. He’s just a really nice guy. He’s also a very busy guy and when I was talking to him he was telling me how, you-know, I wanted to sort of find out how why hasn’t latest version been accepted, or something

> like that, and he said ‘sorry I’ve been busy doing other things’. So I started to go ‘I’ve got a bit of time on my hands maybe I could help?’(Interview M18).

The establishment of rapport is illustrated here and also the importance of the personality of the component lead. Once a trust is established through demonstration of the participant’s ability, they may offer to do more and at this point they may be granted rights to become an assignee or a maintainer. One of the capabilities that may be granted is that of making pull requests, something which allows developers to get code that they write more quickly into the HQ review process:

> 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. Yeah, so in that was kind of an acknowledgment of the things that I had submitted had been recognised as being a valuable contribution (Interview M6).

This process is “informal” (Interviews with Helen Foster, Michael du Raadt and Martin Dougiamas). The inductee must have proven their ability to understand the proscribed practices of writing and submitting work. They must also participate peripherally in the forums, tracker, developer chat to find out who the makers are, as this process is not formalised. There is no canonical track (although some were being established just at the time that interviews were being conducted). One interviewee described their happiness at how they had just recently become an assignee. They achieved this by sending Helen Foster a private message through Moodle and asking to be granted the access rights they needed i.e. this was a relatively informal process.

One particularly important broker and maker is Martin Dougiamas. Many of the interviewees made positive identifications with him. For example, one participant describes Martin’s role in decision making processes:

> In [the open source project] Apache if something needs to get escalated, in a sort of dispute process, it sort of ends up in a committee that votes and in Moodle if something gets escalated it ends up at Martin making an autocratic decision but which, if you’ve ever met Martin, that doesn’t concern you at all, because Martin’s very good at listening to people and considering all sides and making decisions that turn out to be right (Interview M1).

Another interesting data point here is the opinion of someone who had left the project and was hence potentially more free to express a negative opinion, but professed rather to still “owe a lot to Moodle and Martin (and I’m still friends with Martin)” (Interview M5).

Martin Dougiamas himself gives perhaps an illustrative account of his role as a broker and inductor when describing his role in an early phase of the project:

> Most people would say that [the current code standards and submission process] is a good thing because in the early days I was fairly lax and I was really about encouraging, you-know I was more social and psychologically trying to encourage everyone to be open, and you-know making relationships with people and you-know: 'thanks man that's really cool!' y'know 'maybe fix that but otherwise just put it in let's do it. Put it in.' And it's much more flat and but a lot of the weirdness of Moodle, 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 and then, maybe it wasn't quite how I would have done it, but I wanted just to be open and accepting (Interview with Martin Dougiamas, 2013).

One early participant describes his direct influence upon them: "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 describes a similar induction to Moodle:

> It was a really nice and easy to understand system and the community was really small and Martin, the lead developer, was there constantly on the forums and if you had a question he was there so it was… It was very nice to work with (Interview M17).

Dougiamas describes the effort it took to sustain this level of interaction that early inductees to the community were experiencing:

> 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, 2013).

We have seen cases above where code and/or the idea behind a proposed patch get accepted into Moodle's codebase and we can also add a third case where the code, idea and *the person* get accepted by Moodle HQ:

> If it's a biggish thing if the owner commits to maintaining it , that really helps **Oh?**. Yeah, if they say, 'look I'm really keen to submit this thing and I want to maintain it for future releases' – that helps. We like those people (Interview with Martin Dougiamas, 2013).

A submitter's commitment to maintain code after it has been submitted may help them to get that code accepted. A person's role within Moodle may hence increase or becomes more defined due to their work.

Other non-programming acts may be equally important in participating in the community such as:

- Answering questions in discussion forums
- Filing or helping improve bug tracker reports
- Offering to test bug fixes
- Working on translations of Moodle into other languages
- Working on themes (styles) for Moodle
- Or otherwise being a “helpful community member”

For example a member of the community may be prized for “coming in […] doing good work” such as improving the quality of bug reports by helping “pry information out of the bug reporter” (Interview M5). They start doing this work “unofficially”.

Such helpful activity is generally unpaid and carried out by “kindly souls” (Interview M10) perhaps as a “hobby”. However it may lead to paid activities. Two interviewees worked as teachers in schools before joining Moodle HQ including Helen Foster:

> I was pretty active on Moodle.org posting to the forums quite a lot and Martin Dougiamas contacted me and asked whether I could help setting up the documentation for Moodle […] We didn’t have any documentation at all [at the time] so we set up this wiki and I was spending my weekends and evenings helping with that. And the following year I was relocating to Belgium […] and Martin offered to employ me to work remotely so that’s how I started in Moodle HQ (Interview with Helen Foster, 2012).

It is also worth noting that although this case study is confined to the tracker, many of these activities whilst rooted in tracker issues, can happen in other places, such as the discussion forums or even by one to one interactions via email or Moodle message.

Several participants from the tracker sample for instance had written books on Moodle. Although the books in themselves may not have generated large revenues, they do build the reputation and identity of the writer in the community:

> When [Michael du Raadt] put his hand up when I said I was looking for a development manager, I went well, someone who has written a book on Moodle plugins would probably be a pretty good development manager (Interview with Martin Dougiamas, 2013).

One interviewee secured a work visa and then residency in a particular country based on their work in Moodle i.e. as a highly skilled specialist. Several participants described their journey into working in Moodle HQ as one where a hobby or something they loved doing developed into a full-time career after meeting Martin Dougiamas, who then made them a job offer. Many of these people worked remotely for Moodle HQ, though Michael du Raadt's trajectory into Moodle involved a physical journey from one side of Australia to the other with his young family leaving a tenured university career behind. His journey involved a return to a career as a developer when he took on the role of Moodle Development Manager, following a career as a teacher and academic which was itself preceded by a career in software development (Interview with Michael du Raadt, 2012). Another participant described a similar trajectory from developer to teacher and then a return to developer with a Moodle Partner, after becoming involved in Moodle development while teaching.

In another case a new member of Moodle HQ was described as having identified strongly with Moodle and evangelised Moodle in their previous workplace before taking up their Moodle HQ role:

> There was Dan Potowlski who was at LUNS until very recently he has just moved to Perth to work for Martin but the interesting thing is that he has managed to leave this legacy of these two people he has converted to be very active contributors to Moodle core code (Interview M1).

Patience and commitment over time are important traits for developing software which can be a slow process that includes learning all of the community intricacies, even those that "are nowhere written down" (Interview M14). Persistence in the face of setbacks is also necessary:

> Other people might chip in and say well actually this isn't gonna fix the problem because you, you've failed to take into account something else somewhere else. That happened to me at least once where I submitted probably two or three fixes for the same issue before I got the right one (Interview M6).

The above participant took criticisms of his proposed fix on the chin, treating it as valuable data with which to improve his proposed solution until it was eventually accepted. This is just one of the legitimate activities that a peripheral community member can be engaged in. An important point is that it takes time to enter the Moodle community such as to affect change.

4.5.3 Non-Identities

The establishment of an identity in Moodle as part of an inward trajectory implies that there are non-identities or distrusted identities and there is much evidence of these:

> The users that are asking for help with any specifics I'll respond and say please ask on the Moodle forum and I'll keep an eye on it and otherwise they might need my help but if someone has regularly been contributing or helping I'm likely to respond to them (Interview M4).

In the above excerpt someone is identified by their regular contribution and help. Establishing an identity has a particular significance in the online world where identity might be counterfeited. A particular mechanism known as badges serves to establish the reputations of important Moodle community members in the forums. Specific symbols called badges appear beside users' names in the forums to indicate their length of membership and the level and currency of their activity:

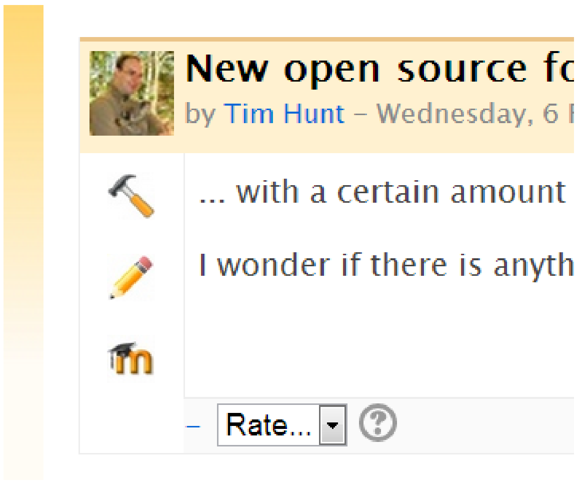

New open source fc
by Tim Hunt – Wednesday, 6 I
... with a certain amount
I wonder if there is anyth
Rate...

Figure 23 Badges in Discussion Forums

The three icons of a hammer, a pencil and the letter M wearing a mortarboard in Figure 23 above, signify that this forum poster is a Moodle Developer, a Documentation Writer and a Particularly Helpful Moodler respectively.

One participant referred directly to identities they mistrusted because of their recent origin:

> I am seeing queries on the Tracker and in the forum, questions or bug reports, asked by an account that is created a week ago and whether it is someone within the community, who wants to create this and not have it tied to themselves because, don't forget, if you ask a question and which is, or report something which you think is a bug or whatever, you want to be pretty noisy about it. You may not want to sully your name with that (Interview M7).

4.5.4 Inertias

Although vectors whereby a person joins the Moodle Community are important because they are related to the momentum of "driving a change to Moodle" their opposite also exists: inertias. An inertia may exist for instance where a participant has conflicting identities from membership of different communities. When a person has developed their own patch they have now solved a problem of their immediate community i.e. that of their "day job". Ideally they might like to submit that patch to Moodle so that their solution would be available to everyone who uses Moodle. However they may cite the effort required to submit that patch to canonical Moodle as being too onerous:

> I remember for example the feedback had some kind of bug that I posted them in the Tracker but I didn't know when they will be fixed so for me it was quicker because I knew how to change it so it was quicker to make it myself (non-Assignee view).

This participant finds it easier to write their own patch and fix their local issue rather than going through the Moodle tracker issue resolution process. Any participant who can do this must weigh it against the effort of not only writing the patch itself but also of maintaining it. Locally applied patches to Moodle will decay as Moodle grows and they must be maintained and reapplied over time. For example, OU policy has been described as seeking to lessen their dependence on these local customisations and rather to have all of their developments either well modularised, so they can easily be applied and reapplied, or contributed back into the core or canonical Moodle (Hunt, 2010; Marshall, 2011). However, these types of decisions are complex, not least because a local Moodle developer may have a significant part of their identity, and their job, tied up in customising Moodle. For example, a developer may be employed by a university or a Moodle Partner to help maintain local customisations (innovations) of Moodle. If these innovations were transferred into core Moodle, this aspect of the person's role might be redundant.

Although the tracker's canonical purpose is to fix bugs it may be a valuable source of patches (non canonical workarounds):

> I found [the tracker] really, really useful for searching so if you get a problem it's really easy to go and search and then you quite often find that somebody else has already reported it or there is already a [patch] (Interview M11).

This may be a relatively satisfactory solution. A patch may represent a workaround where, as the above non-assignee puts it, that although the issue will “usually generally take a bit longer, and be a bit awkward, but it will work so it will pacify my users at their end” (Interview M11).

Although this can be an inertia that prevents change to the code of core Moodle and concomitantly prevents the person from entering into the inner community of Moodle, it may affirm them in the wider tracker community:

> You can search the issues and find things that people have submitted, patches that they’ve submitted, and then you can use them even though they are not in the core yet. So yeah [it is] really useful. You get that *sense of community* from it (Interview M11).

Many customisations to Moodle can be developed as plugins. These are, in software engineering terms, *loosely coupled* to the core system and in theory “pluggable”. To use an analogy, a plugin can be thought of as adding an extra appendage. A patch by contrast must be written into the heart of Moodle, a delicate operation not to be undertaken lightly and which requires long-term monitoring. As Moodle evolved, a conscious decision was made to increase the capacity of plugins and lessen the need to write patches for those who wished to adapt canonical Moodle to their particular needs. However, some parts of Moodle cannot now be changed due to decisions taken early in Moodle’s development which had unforeseeable consequences. Parts of Moodle are now so embedded within the system that has evolved around them, that they must stay that way:

> It’s impossible to change now I mean we just can’t. We just can’t go to those core decisions anymore. That’s part of it. [Moodle] is definitely growing and evolving in different ways, at different speeds. There are strata of code in Moodle of different ages you-know (Interview with Martin Dougiamas, 2013).

This represents another type of inertia: where structure dictates behaviour. Moreover, complex engineering decisions can be compounded by issues of the identities entangled in them. The decision of the Moodle developers to adopt a development tool known as YUI, for example, was arrived at by “looking at all the evidence”, and arriving at a “consensus” (Interview with Martin Dougiamas). However, these particular types of tools were in relative infancy at the time and although YUI proved fit for purpose, it went on to become eclipsed amongst developers more widely by a rival development tool known as JQuery:

> Now of course, you-know, every developer that walks in the door knows JQuery. It's become the de facto thing and there's no way to predict that and we can't just suddenly switch to jQuery now, it would mean rewriting too much. So sometimes the decision is made for you (Interview with Martin Dougiamas, 2013).

This issue is the subject of various discussions in the Moodle forums. Software developers who are new to Moodle wish to use the JQuery tool, but Moodle HQ does not allow it and this has led to some contentious debate. This is evident in the below section of Moodle documentation:

> The official JavaScript library for Moodle is YUI. *That may not be your favourite, but it's the one that was chosen after careful research, so live with it* (Moodle.org, 2013b emphasis added).

By the time of writing the above reference had been removed from the documentation. The explanation in the wiki history for its deletion is given as "Make the YUI section less offensive". The evident feeling of the original documentation writer, before it was made "less offensive", is a demonstration that these can be emotive issues. Although existing code structure may often dictate the direction of Moodle development, there are also cultural structures at work, shared practices of developers that help constitute their community, which may be difficult to change.

4.5.5 Identity, Roles and Participation Summary

The preceding sections examined the identities and associated roles of the Moodle community members and analysed how educators participate in such a community to effect change. Examination was made of the community 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 are defined as people collide with a foreign world, possibly taking them to a standstill or as far as they can go from our perspective of how changes are driven to Moodle. For example, a participant may not necessarily want to have an issue fixed in core Moodle code, and rather a patch may be all they need for their own local community (two communities aims are not always aligned). Or, these people may continue a trajectory into Moodle, revising their identity such as by carving out a new role within the Moodle community and even joining Moodle HQ. In these latter cases they ultimately also affect a change to Moodle itself by negotating new meanings in it through successful issue resolution.

4.6 Chapter Summary

This chapter examined the factors involved in issue resolution. They are many and varied, and have been grouped in three ways: as those over which an ordinary submitter has most influence; where the say of the influential assignee and component lead affects the decision; and where a submitter has the ability to produce code themselves. At all stages of the related processes, decisions are made, meaning is negotiated and new knowledge generated (resolution) or existing knowledge maintained (inertia). The development and non-development of Moodle comes about through the interaction of different types of participant – teachers, developers and brokers. These identities themselves are not fixed and it is often through the change of an identity, as one negotiates further access into the community, that change to the Moodle code artefact is also affected. Sometimes however, community identities can be a countervailing force to change.

5 Discussion

5.1 Introduction

In this chapter we will situate the findings outlined in the last chapter. Research findings of themselves are of little significance until they are contextualised. In pursuit of this task the results of the analysis will be compared to results from comparable and related research by both revisiting topics covered in the literature review and admitting new research literature that has now become relevant (for a summary of the main concepts/themes of the literature review see Table 1 Key Theoretical Concepts above). Linking this study to the wider literature will be carried out to determine whether findings here confirm or contradict other research. In the case of the latter, the question must then be asked as to why any such findings might have novelty and whether (and in what contexts) we can trust them or expect them to hold. In addressing the fourth of our research questions in the previous chapter we drew upon the results of the first three questions, which in itself constitutes some form of discussion or contextualisation of findings. This process is elaborated in this chapter where a more discursive approach is utilised. The main differentiator between this and the preceding chapter will be minimal recourse to the primary data of this study and more to secondary sources i.e. research literature. In particular we will return to a theme of Chapter Two that contends that the life of a community of practice is only seen in its members' mutual engagement in shared practices and hence that it "evolves in organic ways that tend to escape formal descriptions and control. The landscape of practice is therefore not congruent with the reified structures of institutional affiliations, divisions and boundaries. It is not independent of these institutional structures but neither is it reducible to them" (Wenger, 1999, p. 118). We will now begin to discuss the Moodle tracker's unique landscape of practice.

5.2 Revisiting the Research Problem

Before evaluating and discussing the findings it is useful to revisit the research problem itself and the questions posed in order to tackle it. This study set out to discover how the open source educational software of Moodle is developed by participants of its bug tracker by concentrating on the specific practice of bug tracker issue resolution. The following thesis statement was formulated:

How educators participate in developing open source educational software: The case of the Moodle Bug Tracker Community

In unpacking this statement a first exploratory question looked at the participants in the bug tracker itself:

1. *What are the characteristics of participants in the Moodle bug tracker and the issues they engage in?*

The next question sought to probe deeply the defining canonical practice of the community – that of issue resolution. It did this through an elucidation of believed success factors to the processes involved:

2. *What factors and related processes are important in the resolution of issues in Moodle?*

The third question examined the identities of participants and the impact or interplay of these with the roles they occupy:

3. *What are the key identities and roles of the participants?*

Lastly, the themes of the first three research questions were tied back to the overarching research question and the concept of trajectory into or within the community was explored:

4. *How do educators come to participate in the inner community of Moodle development?*

In this chapter we will discuss the findings previously presented as a result of the examination of each of these questions respectively.

5.3 Characteristics of Tracker Participants and the Issues they Engage in

Yin (2009) defines a case study as a research strategy involving an empirical investigation of a contemporary phenomenon within its real life context. Miles and Huberman (1994, p. 27) use the term "site" as it "reminds us that the 'case' occurs in a specified physical and social *setting.* We cannot study individual cases devoid of their context in a way that a quantitative researcher often does" (Miles & Huberman, 1994, p. 27). There was no physical setting for this study. Virtual settings however can be defined and described with no less accuracy and the first task of the research was to enumerate and elucidate the two main constituents of the study's "site": the *participants* of the Moodle bug tracker between January 2009 and February 2011 and the bug tracker *issues* that they engaged in during this period of time.

This research task was intended to act as a foundation for the remainder of the research e.g. one of its functions was in uncovering the makeup of the community which informed the interviews' design and implementation etc. Moreover it also played an important role in setting the stage for telling the story of the Moodle tracker. This is important in qualitative research and case studies where narrative is valued as "an ancient method and perhaps our most fundamental form for making sense of experience" (Flyvbjerg, 2006, p. 222). One of the aims of this research was to explore and describe the Moodle tracker community through the lens of the practice of bug fixing and this exploratory and elucidative imperative was addressed in part by the first research question.

5.3.1 Communities Defined by their Members' Backgrounds

The Moodle tracker is a diverse place. It includes women, men, school teachers, university lecturers, learning technologists, developers, managers, researchers, consultants, IT systems administrators and more. It includes people who appear there every day and those who only visit once.

The participants within the tracker were found to be distributed in locations among countries with advanced economies in Australasia, Europe and North America. English was the first language of 63% of these countries as it was for 15 (75%) of the eventual interview participants. Moodle itself appears to be deployed in a wider geographic spread of countries with developing countries, such as Brazil, appearing to be underrepresented in the tracker community to date. The geographical distribution of the community members broadly accords with those from the literature (Crowston et al., 2012) with a couple of notable exceptions. The first is that Australasia is generally much less prominent in open source projects accounted for here by the location of Moodle HQ in Australia and early adoption of Moodle in New Zealand (Costello, 2013). The second is that central and southern Europe are strongly represented relative to Northern Europe with the exception of the UK which is the dominant country, in part explained by the presence of the OU.

It is instructive to compare the countries of origin data with official figures for overall usage of Moodle worldwide at approximately the same time which comes from a keynote address by Martin Dougiamas in the Japan Moodle Conference in 2011 (Dougiamas, 2011):

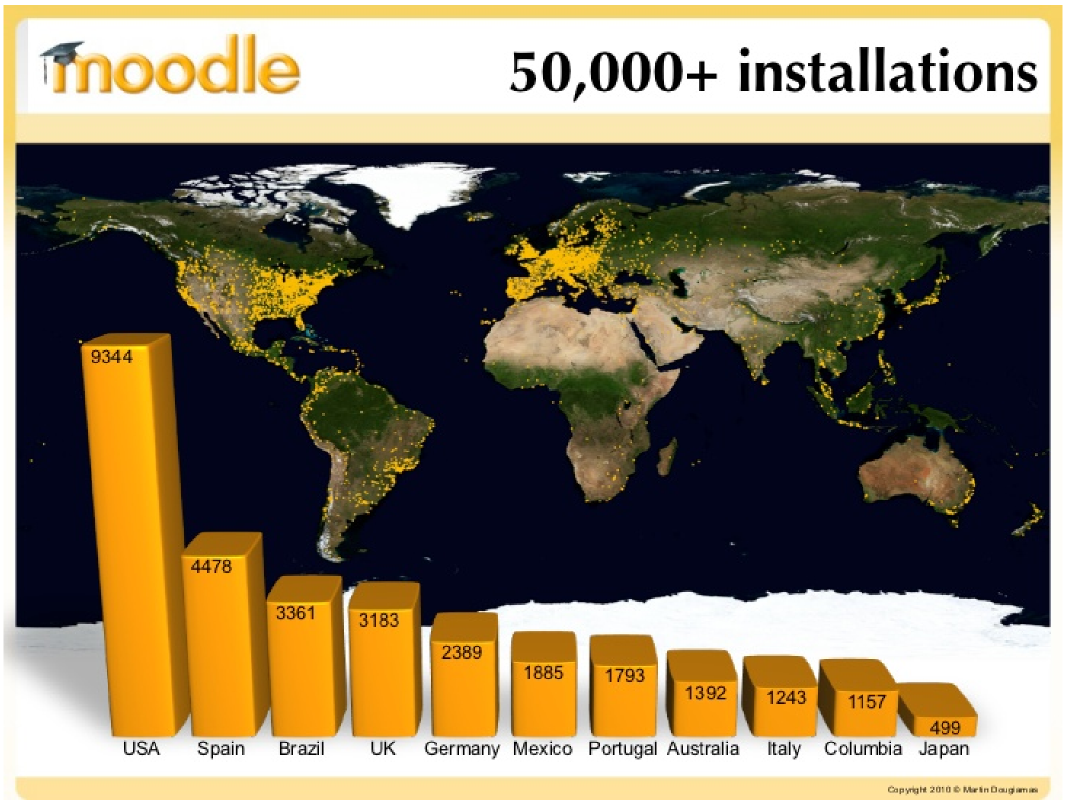


Figure 24 Countries using Moodle in 2011

Brazil and Columbia are represented as prominent countries for Moodle usage. However they have yet to be represented proportionately in the tracker community. Partly there may be a lag effect here. It may be that these countries are still at a more basic level of their use of Moodle i.e. it may be their first VLE and their need to customise it to their requirements will be less than their need to understand and make use of its existing functionality. The increase in usage in countries such as Brazil will have important implications for the tracker, as communication and information were key factors in bug issue resolution processes and language is central to this. We will assess later in this chapter the implications of English being the second language of tracker users, but here it is enough to highlight that Moodle's wider global community, from which the inner tracker core is gradually suffused, is a vast and changing one that is become increasingly heterogeneous and rich.

31% of the participants sampled, for whom data was available, were employed in universities and almost two fifths in either a second or a third level educational institution. (Employees in schools should make up a larger share than the 8% detected in the sample, as they may be less likely to have a professional profile on the web as a university employee.) Over 50% of participants were some form of ICT professional with 40% clearly identified as developers.

However lecturers, teachers, research students and educational technologists accounted for a quarter of occupations and were all represented amongst the assignees.

What does this tell us and why is it important? Determining the different backgrounds and demographic profiles of participants as performed here is not a standard task in open source software research, where it is more common to classify people by the form and level of their participation in a project (Mockus et al., 2002). Where backgrounds are accounted for in this type of research it is usually to determine whether they are a volunteer or a paid company employee (Shah, 2006; Allen, 2009; Ma et al., 2010)). It was important here however because of our theoretical perspective that considered the Moodle bug tracker as a boundary object. As we have explored previously a boundary object sits between two realms such that “each social world has partial jurisdiction over the resources represented by that object, and mismatches caused by the overlap become problems for negotiation” (Star & Griesemer, 1989, p. 412). In our case we have the social world of the Moodle end users such as teachers and lecturers on the one hand and the core assignee developers of Moodle on the other – the worlds of teaching and of coding are joined by the tracker where they must negotiate to fix bugs. Hence this study chose to portray the different backgrounds and different professional realms that constitute the tracker community.

In this light we sought to frame an aspect of our study in a way than is not typical in this research tradition, by directing some of the research towards the *identity of the community members*, paying particular attention to the roots of their outside or “global” identity (Wenger, 1999). We took the approach suggested by the Dalle et al. (2008) study of the Firefox web browser, the end artefact of an open source community of practice which was reaching a vast community of users many of whom, it is reasonable to suppose, had very little idea of (nor perhaps interest in) its provenance. This study postulated the most peripheral participants possible of an open source project (whom they termed vividly “Mom and Pop”) as broadly non-digital/technical natives, and explored this further by contextualising the members of this layer. It is argued here that the increasing ubiquity of the end products of open source software is leading to an increasing heterogeneity and richness of their communities, for example that now include more women and more people from diverse backgrounds whose foreign expertise and voices vitalise and sustain them.

Perhaps the most relevant point here however is that we have evidence of very specialised communities as shown in the roles and educational qualifications of the participants. There are broadly educational realms comprised for example of primary school teachers, secondary teachers, or university lecturers (who might also interact with educational technologists and academic researchers). These realms, and their common cause of education, are becoming increasingly technologically specialised. Indeed Moodle itself is evidence of this – the VLE becoming "as important to the identity of a university as a library" (Costello, 2014). As communities, such as the education community, and its particular subsets, develop in the modern digital age they may spawn open source communities so that they can model their increasing complexity in software. The presence of so many teachers and educators, even peripherally involved in the tracker, shows that Moodle actively engages its end users in ways which proprietary products where the code is a commercial secret will not. Moodle is not simply being deployed in schools and universities but is *being co-created in them* also. The implication here is that educators will increasingly have to engage with highly complex and specialised socio-technical cultures in order to fully express their own growing specialisation and complexification. Because of this increase in specialisation, intermediation will become increasingly important. Educational mediators, in the case of the Moodle tracker, may be designated – for example a University may employ learning technologists to this end. Or equally, intrinsically motivated teachers and lecturers may more spontaneously come to act as mediators. These mediators may not necessarily be lecturers in computing or even STEM subjects as this case shows. Rather it may be their very skills as mediators that are important as we will further examine.

5.3.2 Communities Defined by their Members' Practices

Lastly in our exploratory phase of establishing the boundary of the case study, we performed a descriptive statistical analysis of tracker issues themselves. This established broadly that a certain core of the community is more influential than its more peripheral members. As a first step, we framed the layers themselves showing that, as is often to be expected in representing distributions of social phenomena (Shirky, 2003; Anderson & Andersson, 2006; Ko & Chilana, 2010), the community is skewed with a small cluster of very highly active contributors at one end of the scale representing a core. For instance, we saw that a small number of users submit a high proportion of issues to the tracker. These people represent a group of super-users. Moreover, the minimum *participant per issue* was one, and represented 11% of all issues. Clearly this type of issue – which a submitter submits and then assigns to themselves which involves no other participant – shows that the tracker is a space being utilised in very different ways by very different groups. Assignees are using the tracker in this case in an introspective way. They instinctively turn to the tracker not necessarily to explain an issue to someone else per say (though it has that function), nor to petition for its resolution, but rather to remember and record it, before fixing it themselves. They are utilising a boundary object here in a fashion that is instinctual, as a tool of practice with which they are intimately familiar. This usage is very far removed from the poorly legible first bug report of the most peripheral outsider.

As well as using it instinctively insiders are using the tracker intensively. We saw that during the period of the case Petr Skoda submitted 1,457 issues. However, if one were to pick an issue at random (the mode), it would most likely be that that issue's reporter had only ever submitted a single issue. A core of super-users is intensively engaged in practices and activities in a space that they share with other users who are causal, episodic and transient in their participation. This is a key finding because the *scale* of this gulf between core and peripheral participant practice is a proximal cause of brokerage, of mediators, and why and from whence these mediators emerge, as we will see.

The relatively crude conceptual partition of the core and outer communities in the tracker became more useful in looking at issue outcomes and issue classifications (themselves linked to outcomes) for assignees compared to non-assignees. It was unsurprising that non-assignees had statistically significant levels of influence over issues that were less on average than assignees, as this is clear from previous research (Crowston & Howison, 2006; Jensen & Scacchi, 2007; Ko & Chilana, 2010; Izquierdo-Cortazar et al., 2011). These peripheral users submitted more issues that were classed as invalid and less issues that were eventually successfully resolved. However, they did submit 24% of all issues that were valid and were subsequently fixed. This is very close to the 21% found by Ko and Chilana (2010) in their examination of the same metric of non-assignee contribution to fixed bugs in the Mozilla community. Their conclusion is that the "value to be obtained from open bug reporting repositories is primarily in recruiting and retaining talented developers and reporters, and not in deriving value from the masses" (Ko & Chilana, 2010, p. 1). Although the value of the bug tracker as a boundary object that helps bring new members into the core of the community is something that is agreed with here, the quantification of the contributions of the peripheral users as minimal is not. Almost one in four (or one in five in the case of Ko and Chilana (2010)) still represents a significant proportion of valuable fixes contributed by the outside community. Moreover, although they may appear less valuable in relative terms from inside the core, these fixes are highly valuable to the outsider educators who have reported them, whose needs have been met and addressed after they took the time to report a bug that was affecting their teaching practice.

The contribution of this peripheral type of participant to communities is hence very important. This study bears out the findings of other projects that outsiders are more likely to submit duplicate bugs (Ko & Chilana, 2010; Zimmermann et al., 2010). It is also clear from this study however that duplicates are canonically defined labels applied by the core community and moreover, are not as harmful as they may appear. They can also sometimes be a valuable source of information as per the interesting but not widely reported finding of Zimmermann et al. (2010) who showed that duplicates help provide alternative descriptions of bugs that can ultimately help resolve them and that despite its negative connotation this "duplication" does not cause a great deal of extra work.

The non-assignees are vital as they represent an almost umbilical link between the tracker core and the Moodle user on the outermost layer of the onion (Crowston & Howison, 2006). Without them the community could not function, would be completely introspective. This finding helped answer a question that motivated this research from the outset. Does an individual university lecturer, who finds and reports a bug in Moodle, have a chance of that bug being fixed? In Chapter One we narrated the story of a lecturing colleague of mine for whom I (as a mediator from the university educational world) had reported a bug in the tracker. The bug was leading to incorrect results in negatively-marked, multiple choice tests. We followed the story of this bug in Chapter Three, where we saw that its resolution "only took 6.5 years" (Hunt, 2011) and its fixer Tim Hunt later became one of the study's interviewees. This is a unique bug with its own story. Its resistance to being fixed marks it as a "superbug" (Dalle & den Besten, 2007) attested to by the amount of comments, votes and participants that it attracted. Hence we can now answer the question of whether peripheral educators can influence the evolution of Moodle in the affirmative, as they submitted 3,457 (24%) of valid issues during the case that were fixed. However we must also add the caveat that many issues they submit are never fixed, that the factors that are marshalled against bug resolution are many and varied. We will next discuss these factors.

5.4 Importance of Non-Canonical Factors in Issue Resolution

The second research question sought to examine the factors that can lead to issue resolution and how those factors relate to each other as processes. Enumerating factors was itself deemed to be an important task in the design of this research question. As we have seen in the literature review in Chapter Two, many studies have focused on trying to find a determining factor or factors to issue resolution. A limitation of some studies identified at this stage was the tendency to fail to consider possible confounds, and it was shown that factors cannot necessarily always be neatly manipulated in quantitative formulae, but rather are often less divisible aspects of complex stories and processes (Hooimeijer & Weimer, 2007; Aranda & Venolia, 2009). For example, an outlying factor detected here almost incidentally – that Michael du Raadt closed thousands of issues systematically at one time – is something that might not have been detected in a purely statistical analysis of the database, and even if it were, there would be no explanation as to why this happened. A related concept was identified in one study (Francalanci & Merlo, 2008) where bug fix dates were found to be closely correlated to release deadlines. In this section, we will discuss those factors detected here that impact on issue resolution, which are non-canonical in nature i.e. not part of the formal sanctioned or written accounts of work practices.

It was argued in the literature review that one of the richest bug resolution models reviewed (Hooimeijer & Weimer, 2007) was derived from narrative accounts of bug-fixing. In the Findings chapter we enumerated further factors that will help inform such models (collated in Appendix D Factors Believed Important to Issue Resolution). For example, whether the submitter is *responsive* to requests for further information about their submission, was found to be one of the factors in assignees' decision making about whether to work on an issue or not. This is not a prominent theme in the literature (one very notable recent exception being the work of Breu et al. (2010)). Moreover, in the analysis this factor was seen as one of the prime or leading factors appearing in the very first decision diamond of Figure 20 Paths through Submitter Primary Domain, on page 114 above.

Another factor which assignees reported to be important in their decision to progress an issue was their perceived politeness of the submitter. Detecting and modelling rude behaviour might prove difficult, but accounts of bug fixing that fail to consider this factor will be limited if the assignee participants of this study are to be believed. However, we might need to somehow modulate this against the contrary view of one non-assignee 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 (and we saw this approach employed to arguably good effect in the Calendar Import issue by a non-assignee).

Perhaps the complexity of this aspect of bug resolution is why it does not appear prominently in the literature i.e. that it represents an awkward question to ask and answer systematically, relative to many other questions that can be asked of bug trackers. Bug-fixing has been conceived of as a complex social process (Sack et al., 2006) and so we should not be entirely surprised that it is subject to strong non-predicable (because they are non-rational) influences. The case that interdisciplinary approaches – characterised for example by a mix of psychology and computer science – are best in examining the complex social mechanics of bug fixing has been made (Ducheneaut, 2005; Sack et al., 2006) and is borne out by this study. Moreover further research is required in this area specifically through using approaches that gather the views of participants on issue resolution. There is a genuine paucity of data when considering this research field as a whole, as only a few studies (De Souza et al., 2003; Freeman, 2007; Aranda & Venolia, 2009; Zimmermann et al., 2010) have employed this approach, even though one recently showed 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). Even within this minority tradition, interviews as a data-gathering technique in eliciting these participant views are less likely than surveys i.e. research designs are mainly fixed so may not be flexible enough to react to, and hence properly capture, unexpected information that emerges.

Another social aspect of bug fixing that militates against computational and predictive models of issue resolution was detected here in the role of votes. On the face of it votes would seem one of the most accurate ways to quantify the demand for an issue to be fixed. Votes have a very clear reified form in the tracker as a simple number. As we have seen, many participants reported the belief that votes were important, with one notable non-assignee contrarian who was dismissive of their importance. However it was detected in the bug tracker analysis, when addressing the first research question, that submitters often did not vote for their own issue which alone should allow us to caution that votes are not a true measure of aggregate views in favour of a given issue's resolution. The interview with Martin Dougiamas identified a pertinent example of a particular Moodle Partner attempting to "game" the voting system. He described that votes often needed to be taken with a "pinch of salt" by assignees (in this case Moodle HQ). One way of conceiving the importance of votes to issue resolution, and indeed possibly almost any other factor, is as being similar to a predictor of a stock price. We could consider the Moodle bug tracker as similar to a market where factors are "priced-in" to issue resolution. Where someone discovers a way to game the system there will be a counter-balance whereby the assignees will then start to discount that factor. In this way a bug tracker can be thought of as an inherently unpredictable system. Indeed its non-predictability is one of its important functions because it ensures that its ownership is distributed insofar as limited agency can be exercised over it.

Numerical factors such as vote counts are easy to game. Comments on an issue are less so. To create a comment takes time and comments from "non-trusted identities" can be easily identified. (This difference between the scalability of commenting and voting can be seen by contrasting *Figure 15 Participants per Issue* with *Figure 14 Votes per Issue* above which showed participants/comments per issue to be more evenly distributed than votes per issue.) This may provide a possible explanation for why comments emerged as one of the likeliest predictors of issues resolution in the literature (Hooimeijer & Weimer, 2007; Panjer, 2007; Giger et al., 2010). As mentioned in the findings chapter, a limitation of this study was the inability to systematically count the comments due to the structure of the data in the JIRA bug tracker database. Hence we cannot address this question in directly comparable terms, although it is clear that the findings of this study do nothing to refute the hypothesis that commenting is correlated with issue resolution. We may point out however that correlation does not necessarily imply causation. Indeed, from the issues examined here, and the views of their participants, there are certainly many cases where comments are a side-effect of the bug-fixing that is going on.

We do know from this study that the content of the comments and who is doing the commenting is important (Interview M11), particularly in light of the strong theme of assignee influence found in the statistical analysis of the tracker. Moreover, we also discovered that participants per issue (Figure 15), was one of the least skewed and least non-uniform of the distributions that were found amongst aggregate measures of issue attributes. We defined a tracker participant (simply from its reified representation in the tracker database) as someone who submits an issue, comments on an issue, or is assigned to an issue i.e. voters are not included. Indeed it might be possible to model comments per issue by subtracting the submitters and assignees from participants per issue (or even just subtracting the average submitters and assignees if this was not possible). Either way this would give us a much lower figure, unsurprisingly, than votes per issue and affirms what other research suggests (Hooimeijer & Weimer, 2007; Panjer, 2007; Giger et al., 2010) – that certain activities such as taking the time to talk about an issue by commenting on it in the tracker do not scale dramatically and hence are invested with a certain authenticity that is denied more scalable activities such as voting. Simulating hard work is not easy and comments may represent evidence of real engagement which is important to issue resolution.

Coming from the interview analysis, a closely related theme to the correlation of comments with issue resolution was *responsiveness*. Responsiveness was an important factor identified that could prove more systematically quantifiable than politeness. This would be a relatively more straight-forward factor to express in a bug resolution model such as to detect responsive submitters, who are prompt with replies and follow-up information requests, which the assignees in the Moodle tracker studied here believed critical to issue resolution. Any analysis of this factor is absent from the literature apart from one relatively recently published study (Breu et al., 2010) which has given a thorough and detailed analysis of the power of responsiveness of submitters in the Mozilla and Eclipse communities.

Relatedly, inactive component leads should also be detectable from assignee response times, although as these are likely to be a small enumerable population, this information could as easily be determined by simply asking some assignees. It would follow that rich models of bug resolution are best composed of parameters derived from both narrative accounts and statistical data (for example perhaps survival analysis in this case for which see Bird et al. (2007) or Dalle and Besten (2007)). This study highlights important issue resolution factors such as responsiveness and politeness gleaned from qualitative analysis of the stories of participants that are either missing or greatly underrepresented in the existing research literature in this area.

The concept of a “big win” (the relative effort required to fix an issue) was uncovered as another important non-canonical issue resolution factor. Research into the development of the Firefox browser (Hooimeijer & Weimer, 2007) also found a link between being able to understand a bug problem from its description (and hence estimate its complexity) and the time to its resolution. In other words it found a link between “big wins” and the readability of the initial bug report hypothesising (and attempting to show) that “bugs that are more difficult to understand will be more difficult to deal with and will be addressed later” (Hooimeijer & Weimer, 2007, p. 36). *Readability* of bug submission text was computationally estimated by Hooimeijer and Weimar (2007) by using a relatively basic algorithm that analysed word and sentence structure. A possible problem with their method is that it may not account well for participants for whom English is a second language compared to alternative algorithms (Crossley et al., 2008). This study found language was an issue for interview participants and also that a significant amount of the sampled tracker participants (38%) came from non-English speaking countries (including five of our interviewees). We also highlighted Moodle's growth in countries such as Brazil. Putting this research together with that of the literature then, we can say that issue resolution in Moodle will likely face increased linguistic challenges.

Hooimeijer and Weimar's (2007) *readability* concept, also well studied by Zimmermann et al. (2010), maps to our issue resolution factor of *information* (and clarity of information). However, not only was clarity of the language deemed important under the category of *information* but, in addition, the overall coherence of the message, the ability to tell a story that included all the relevant data was shown as vital. This theme in the coding was labelled *exchanging story steps.* Clarity is key but so is communication. Formulating this as a process we saw that when the initial information was deemed poor, responsiveness and politeness then became critical. Moreover it was found that selling the idea to the assignee was important in this context. Although giving the exact steps to reproduce the problem was salient, so too was contextualising it with other aspects of the submitter's identity: responsiveness, politeness, command of English, ability to relay information and overall their ability to narrate their personal story whilst positioning themselves communally. Overall then we have a picture of a cluster of factors that have many non-canonical aspects, insofar as they are not part of the official or documented bug submission process. These factors emerged in the stories that the participants told and are part of a complex "shared repertoire" of practices that have tacit and implicit aspects.

A prominent theme in our analysis of issue resolution factors was canonical and non-canonical depictions of artefacts and practices of the community. As we saw in Chapter Two this conception is drawn from Brown and Duguid (1991):

> …reliance on espoused practice (which we refer to as canonical practice) can blind an organization's core to the actual, and usually valuable practices of its members (including non-canonical practices, such as "work-arounds"). It is the actual practices, however, that determine the success or failure of organizations (Brown & Duguid, 1991, p. 41).

In the assignee domain, *issue classification* could be canonical (reified in the tracker) – such as whether a bug was "critical" or a "blocker" – but even then we noted that this could be a subjective (or what Wenger (1999) terms a "negotiable") act. On the clearly non-canonical side, assignees identified issue classifications that affected their practice, which included whether issues were "interesting" or "annoying". This certainly points to another latent factor that may not be accounted for in the literature – what we could term a class of bugs that have some property of being *intrinsic motivators* to their own fixing. "Come and fix me!" such bugs would say. It seems that more work would be needed to establish the impact of this influence upon developers. We have highlighted here that software developers have strong intrinsic motivations for what they do in this community, and this is in congruence with wider research. In addition it is postulated here that motivational impetus may apply at the level of individual bugs in subtle ways that have not been hitherto examined.

In examining the factors important when submitters have their own code, more complex examples of non-canonical work practices were found, ones representing embedded cultural knowledge, such as when assignees attempted to follow rules that "were nowhere written down" (Interview M14). This is an example of how "practice can be guarded just as it can be made available" (Wenger, 1999, p. 121) even in a community defined by the openness of its key artefact, the Moodle source code. Although outsiders must ferret out information and comply with complex and sometimes opaque requirements, insiders may sometimes take "naughty" shortcuts (Interview M1). This points to a power disparity and a similar one can be seen where politeness is demanded of submitters from assignees who themselves may "argue without using flowery language" (Interview M5). From this perspective the negotiation of new meaning promised of communities of practice, in our case solving the problem encapsulated by a bug tracker issue, may at times be subsumed by the perpetuation of existing social orders (Fox, 2000; Roberts, 2006).

However, the "informal fabric" of the community (as Wenger (1999) refers to non-canonical practices) was nonetheless evident in this study as one that could add value and do real work. So, for example, the bug tracker could be a source of "workarounds" (Interview M11). The word *patch* itself means a temporary or interim solution. The canonical function of the tracker is to transform patches into sustainable solutions and integrate them fully into the Moodle codebase. For many users however, the patches are good enough in themselves. As one non-assignee put it, although there is work involved in applying the patch it will "pacify my users at their end" (Interview M11). Patches are not an ideal solution but they may well "satisfice" a term used by Schwartz (2002) in showing how the attractiveness of sub-optimal solutions is increased when the effort of attaining the optimal outcome is forgone.

Overall then one of the most important findings made here was the relative importance of non-canonical factors. Canonical factors are *reified*, clear to see, documented and well measured and weighted in the literature, but they do not tell the full story. Meaning can be usefully described as a duality or interplay between reification and participation (Wenger, 1999; Hildreth & Kimble, 2002). When reification is too strong, knowledge becomes hard (Hildreth & Kimble, 2002). When an issue is resolved in the tracker, it gets a status of *Closed* and a resolution of *Fixed*. We saw tracker participants commenting after an issue had been resolved in one of these ways attempting to reopen it. Here commenting is the participatory form of meaning where participation is "essential to repairing the potential misalignments inherent in reification" (Wenger, 1999, p. 64). This is a point to which we will return.

5.5 Identities and Roles

In communities of practice personal identities are often viewed in relation to the community identity, and the interplay of these becomes a source of the community's dynamism. Much research concentrates on personal identity in open source communities, often via the motivations of participants as we saw in Chapter Two and, in the context of such analysis this study also investigated this area. The research schedule probed participants on the background to their involvement and what meanings the project held for them. Two identities in particular that related to specific societal roles/professions were found: those of programmer or software developer (usually shortened to simply "developer") and teacher. The more abstract role of mediator or broker also emerged as important in the analysis. At an outer level we then explored identifications (and the consequent construction of identities) with the community of Moodle itself and by contrast identifications outside of Moodle. Here we will discuss these findings critically in the context of our single-case study of Moodle.

5.5.1 Developers

Participants in this study who were (software) developers reported deeply intrinsic motivations for engaging in programming, likening it sometimes to a compulsion to "scratch an itch" (Interview M1). This in an idiom that may be traced to Raymond: "Every good work of software starts by scratching a developer's *personal* itch [emphasis added]" (Raymond, 1999, p. 30). We saw Martin Dougiamas describe how he felt compelled to start developing Moodle. We can place him in the mythology of the community as the archetypal figure of the lone first programmer who goes on to become the "benevolent dictator" (Raymond, 1998;

Dougiamas, 2007). The obsessional nature of software development is well attested to in the literature, perhaps best illustrated by the study that analysed levels of sleep deprivation incurred by participants working long hours on open source projects (for which it seems almost incidental to mention they were not being paid) (Lakhani et al., 2002). The members of the Moodle community interviewed here are devoted coders, several of whom reported to having been writing code since childhood. They are the “inveterate tinkerers” of Orr’s (2006) photocopy repair technician community of practice. In identity theory the core of a person’s being may “remain unconscious and untapped until it is discovered during the course of engaging in activities that resonate with it” (Schwartz, 2001, p. 33). The primal or ego identity is hence safest “where it is grounded in activities” (Erikson, 1974, p. 107). In our case we identified a subgroup of people within the Moodle tracker who found a deep form of self-expression in writing software. They referred to coding in primal terms such as in referring to parts of Moodle as their “toys” and they portrayed a deep care and passion for what they were doing: “we really care about how code looks” (Interview with Martin Dougiamas, 2013).

The time comes of course for one’s code to be reviewed by someone else, leaving the relative safety of this activity which a person may believe is core to their identity (Waterman et al., 1995). In the open source software community literature this community validation of one’s ability to program is classed as one where participants are intrinsically motivated to operate (Lakhani et al., 2002; Krishnamurthy, 2006) or more often can be conceived of as an *internalised extrinsic motivation* (Roberts et al., 2006; Von Krogh et al., 2012). Thus the motivation for peer validation whilst not as clearly intrinsic as a love of programming for its own sake, is nonetheless self-regulated rather than externally imposed (Ryan & Deci, 2000). We saw how developer participants in Moodle admired and looked up to fellow community members, even those they did not know personally, but simply knew through their code and/or their reputation as coders. Core developers were admired for their “code poetry” and knowing them was in itself considered “exciting”. For non-assignees and assignees alike this was a strong motivation for their involvement in Moodle and so the case could be made that core developers arguing or acting impolitely, might be more acceptable on this basis i.e. the cultural capital that a developer accrues from their demonstrated, or more accurately reputed, abilities gives them certain behavioural license (Côté, 1996).

5.5.2 Teachers and Brokers

Another identity encountered during the case study was that rooted in the activity of teaching. We saw in the tracker sample profile that teachers and lecturers are well represented in the tracker and also that these are not necessarily teachers in Computer Science or even STEM subject areas. A number of the interviewees spoke about how teaching informed their practice within the Moodle community. Answering queries from users of the discussion forums for example was seen as a teaching type of activity. Teachers reported that by virtue of their role they had a view of Moodle from the perspective of people who "actually use Moodle". Another important theme that emerged here was of teachers as mediators or brokers, the task of teaching as one of translating between ontological worlds, such as by "breaking difficult things down to make them less complex […] so that people can understand them" (Teacher view). This task of explaining Moodle and how it works can best be performed by those who "actually use it" (Teacher view) because it entails what Brown et al. (1991) term "situated learning" where "tasks are embedded in a *familiar* activity" (Brown et al., 1989 emphasis added) i.e. familiar to both teacher and student.

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. This is not easy to study as there no way to tell from a bug tracker itself whether someone has filed an issue for someone else or not. We could not, for instance, systematically count these types of proxy issue submissions although, it might be possible to manually identify *some* of them by reading issue descriptions (because the submitter will sometimes put a link to the discussion forum thread where the bug has been originally raised in the issue description when they submit it). It is hence difficult to quantify how pervasive this form of proxy issue opening is. This is relevant as it may affect many types of models of bug resolution e.g. many bug fixing metrics (including ones generated here) take the submitter as an unproblematic parameter. 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 others after reading the discussion forums dedicated to the particular area of Moodle they were interested in. In the case of Helen Foster however, although she was interested in the area of Usability 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 identification with. We could see her brokerage as one rooted in her identity as teacher – she *specialises as a broker* – whereas Interviewee M12 is *a specialised broker* whose activity may be more closely related to their identity as developer.

Brokers are important to projects because they may provide “a buffer between developers and peripheral users” (Crowston & Howison, 2006). This should become more significant as a project matures and 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 attempt to “pipe” them to potential, suitable fixers. Moodle Documentation Fairy – so called after the Moodle forum avatar of Mary “Moodle Fairy” Cooch – was a new Moodle HQ role that emerged towards the end of this research[1]. Developing documentation is an important and legitimising activity for members of open source communities (Jensen & Scacchi, 2007). Software developers may enjoy (whether unfairly or otherwise) a reputation for not relishing, and consequently not sufficiently engaging in, the writing of documentation to explain the fruits of their coding; however explanation *is* at the core of teaching. We should hence be unsurprised that a teacher was elevated to this role in Moodle HQ.

Many of the brokers in Moodle such as those who participated in the discussion forums also claimed a strong desire to help people and to be part of a sharing community. Sharing, helping and being part of community which is built upon free software formed part of a complex of positive feelings participants professed to have about Moodle. We will return to this issue in the context of inertias and momentums of the community but the next theme to be discussed is one which ties together many of the concepts discussed so far, that of entering and progressing through the community.

5.6 Participating in the Inner Community of Moodle Development

5.6.1 Trajectories Based on Gift-Giving of Code

[1] Whimsical titles are not uncommon in open source and can be considered part of its culture e.g. Mitchell Baker, “Chief Lizard Wrangler” of Mozilla.

In this research we have posited the Moodle bug tracker as a boundary object which facilitates the skilled knowledge group of the core tracker community by providing a barrier that simultaneously serves to include and exclude outsiders. It does this by providing the framework through which learning and acculturation can happen for outsiders, through the process of their legitimately participating at the periphery. Now we will examine how in practice this can come about and what it means to enter this particular community i.e. what is the reason for such a trajectory and where might it lead.

Regularly contributing valued code was found to be a primary way for a participant to demonstrate their helpfulness to the Moodle community. This is also well shown in existing research (Mockus et al., 2002; Von Krogh et al., 2003; Crowston & Howison, 2005; Herraiz et al., 2006; Ma et al., 2010). This phenomenon was apparent in more than one of the research sub-questions. For instance, code contribution quality appeared in three of the factors of the process depicted in Figure 22 Submitter Code Domain – namely whether the code fixes the issue, is correctly written, and is deemed a good solution.

Researchers have explained the behaviour of writing and contributing code as a form of "gift giving" (Raymond, 1999; Bergquist & Ljungberg, 2008) as a way by which "successful participants progressively construct identities as software craftsmen" in a process that is "punctuated by specific rites of passage" (Ducheneaut, 2005, p. 323). A particular trajectory became apparent here in the analysis of the factors and processes of issue resolution. As these factors were grouped, one significant class became those that come into play when the submitter has their own code. The end of this process involved either: the code being accepted; Moodle HQ rewriting the code; or, *both the person and the code being accepted*. This last possibility was one where the submitter was designated with certain rights and responsibilities within the community so that they could maintain that part of Moodle for which they submitted code into the future. Broadly this trajectory represents an entry into the community by crossing the threshold defined in this case as the assignee/non-assignee division. This phenomenon has been studied as we have seen in Chapter Two by examining specific sequences of activities which may lead to a participant joining a community, or if the community is complex, an inner layer of that community (Von Krogh et al., 2003; Crowston & Howison, 2005; Jensen & Scacchi, 2005; Crowston & Howison, 2006; Herraiz et al., 2006). We also saw in Chapter Two however that while attempts to model these joining trajectories have been successful in small projects (Von Krogh et al., 2003), where joining may appear to follow a "script", there seems to be a range of possible forms that these trajectories can take and that they are not completely predicable in a large and mature project such as Moodle (Herraiz et al., 2006).

5.6.2 Trajectories Based on Brokerage

Code contributions and other related tasks are important for joining the inner core community but advancing on this conception are studies that 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 here. As has been shown being a “helpful community member” was a strong emergent theme relating to a participant’s establishment in the community. We saw that “helpfulness” had reification in the Moodle discussion forums, where long-standing and active members could earn a “Particularly *helpful* Moodler” badge (emphasis added) that would appear below their avatar. Establishing this identity does not require any coding and many other tasks were found to fall into this category, such as writing documentation, creating styles or themes for Moodle, testing Moodle, translating Moodle into other languages 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. As we noted, Moodle changed its code submission process during the period of the case, which introduced a very formal and explicit layer of checks and reviews that code submission requests should undergo. This may make it harder in the future for novice or hobbyist programmers to join communities as more professional expertise is required, however new jobs and related roles will also emerge in documenting, styling, explaining, translating etc.

Gaining a reputation for being “helpful” is important no matter what task a contributor is engaged in. As discussed previously, one undeniably helpful task that Moodle appears either to have a unique emphasis on, and one that has not been studied before, is filing bug reports on behalf of others. This role is non-canonical. It is not accounted for in the bug tracker system i.e. there is no way to record that the bug filer is not the bug discoverer. Crucially it is reliant on the goodwill of the broker and hence we can conceive of it as, in the emic terminology of the community, “particularly helpful” behaviour. This was well described by the participant who related how some community members never went beyond discussing their issue in the forums. Such submitters never bothered to create a tracker account and explicitly submit the issue, and their problem would hence languish without the intervention of a broker. As one participant put it: “hopefully some *kindly soul* in the forums will spot it and write it up as a bug report” (Interview M10, emphasis added).

The tracker and the officially documented bug filing and resolution instructions are the canonical and reified accounts of bug fixing. Bug stories may emerge in the noisy and highly participatory (i.e. un-reified) atmosphere of the discussion forums. *Kindly souls* that engage in the discussion forums move between that realm and the inner layer of the tracker seamlessly. They may also improve the quality of bug reports by taking it upon themselves to “pry information out of the bug reporter” (Interview M5). Their participation is what softens the hard knowledge that has been reified in these boundary objects (Hildreth & Kimble, 2002). Kindly souls provide a key link because it may take a “brave person to submit a bug in the tracker” (Interview M18) i.e. the tracker is a boundary object designed as much to exclude as to include. These participations may themselves in turn become reified into official tasks and roles (bug triage, Moodle Community Manager) where what once were unpaid activities become formal salaried positions, or functions of them, within Moodle HQ itself.

In addition to code contributions and other related work it was also noted that certain behaviours were believed by the community to be important as entry requirements. Loud newcomers were likely to be ignored and “determination”, “patience” and “politeness” on the part of submitters were non-canonical requirements of the task at hand within the tracker such as issue submission and resolution. Perseverance and patience are viewed as important because some “agreed standards haven’t even been written down” (Interview M14) or, conversely, a core member may respond to a much less knowledgeable outsider in an unhelpful way *precisely by* referring them to the standards which are in the form of dense technical documentation. Moodle is no different to other open source projects or socio-technical cultures in this respect where instructions may not be provided “since figuring out how to do all of this can be considered part of the ‘entry requirements’” (Krishnamurthy, 2005). It should be noted however that this is not official Moodle policy and there are detailed guidelines for encouraging involvement with Moodle at all levels including step by step instructions for how newcomers can get involved with programming Moodle. However the practices of the community are rapidly evolving and, as is argued here, the canonical accounts of these practices are in themselves grist for the participation that must be engaged in.

Politeness and helpfulness are important to entrants because a key task is in establishing a rapport with a connected person in the community. Many models of community joining emphasise specific sequences or quantities of tasks that lead to entry into an inner layer of a community (Von Krogh et al., 2003; Crowston & Howison, 2005; Krishnamurthy, 2005; Crowston & Howison, 2006; Herraiz et al., 2006; Bird et al., 2007; Ko & Chilana, 2010). In addition here, we also note behavioural aspects to joining, and one key theme that emerged was of building a rapport with a connected individual. For example, one participant described their induction to an inner layer as being related to their identifying strongly with a broker: "He does a lot of stuff outside of Moodle helping people all the time. He's just a really nice guy" (Interview M18). This interviewee went on to become an important member of the community and later to work for Moodle HQ itself.

We have previously seen that a developer can identify with other developers and that this is related to the latter's reputation and role: "Wow, these are the core developers!" (Interview M9). This concurs with what we know from the literature as outlined in Chapter Two (for a particularly pointed example see Stewart (2005)). However, this study has shown that an individual can have an impact for other reasons - that they can have a reputation for generally "helping people". This dimension of the behavioural aspects of individuals and their ability to create rapport could add to their reputation as coder or indeed, we can suppose, it could compensate for it. Little research has been completed in this area though there is related work that suggests further research would be worthwhile. For instance it has been shown that rapport and trust are important within communities once members *have already been established*, a relevant example being the work of Guo et al. (2010).

Martin Dougiamas was cited as a particularly important member of the community for his role as a connected broker helping users into inner layers of the community. The portrayals of him by other members suggest a charismatic individual who inspires loyalty. We also saw in his own account of the development of Moodle, suggestions that his role as a mediator may have been at least as important as his coding skills for Moodle's genesis. He described at one point for instance a trade-off between growing the project socially and accepting code that was not optimal. Dougiamas was at this point building an inclusive (flat) architecture according to participatory principles as much as engineering ones. The tension between control and growth in evolving open source projects by encouraging or constraining participation has been explored by West and O'Mahony (2008) and they use the term "participation architecture" to describe how projects engage outsiders and allow them to contribute. There is of course a trade-off involved here as any interface or boundary must provide both security (the capacity for defence) and at the same time openness (the capacity for co-operation) (Whitworth, 1998). It is certainly plausible that the increase of rules, checks and code reviews as Moodle matured made the project more stable and improved the quality of its code submissions but it also raised the bar and made participation more difficult, required that more skills and processes be learned by contributors. If a project is growing this is affordable and necessary (as the community can be fussy and turn prospective contributors away if they cannot keep up). This is an area that deserves further research so as to examine the implications of the increasing professionalism and complexification of code submission in mature projects. The full effects of this would need to be ascertained in light of the view that "restriction of participants' task autonomy should be negatively associated with their intrinsic motivations to participate" (Roberts et al., 2006, p. 988). Although this last statement was made in the context of paid versus unpaid activities, it would be worth exploring its application to motivations to engage in tightly versus loosely specified tasks. As Grabher (2002) has noted, software projects as they mature may focus more on "sedimenting knowledge" through incremental improvements at the expense of innovation and creativity.

We have outlined here something of the individual stories of how people came to participate in Moodle; how their motivation was linked to their personal identities. For instance we saw a developer become a teacher and then move back into development again with Moodle – a form of homecoming to their developer identity perhaps. These stories may not be relevant to the study of many open source projects and are not common. A notable exception is Freeman (2007, p. 50) who sought to explore "dynamic […] and content-sensitive aspects" of community-joining impetus and produced interesting case studies of individuals' motivations that were the product of "changing objects and personal histories prior to and during participation". However this research was very much focused on the joiners and did not consider, as we did here, the effect of brokers who induct these newcomers (indeed there was no mention of them in the study). The role of the brokers who may have particular abilities at bringing in new members deserves further study, not least because it is an underexplored phenomenon but also because as projects mature and become more specialised, intermediaries become more important. The literature review did not highlight this aspect but widening the scope of the literature search, in the light of the findings, did yield a confirming research agenda. Some relevant research is concentrated in the Collaborative Innovation with

Customers (CIC) subsection of the Knowledge Management research field, in large part corralled from the open source literature but which makes interesting and very apt claims[2]:

> As firms seek to capture more product design and technical insight from outside the firm, the nature of in-house researchers and teams may change as well. In-house researchers will increasingly serve as liaisons or linking pins with external entities, and it is likely that their own job designs and required skill sets will correspondingly shift (Greer & Lei, 2012, p. 77).

People who are particularly effective at the boundary of their community or at internal boundaries between community layers become crucial as communities grow and new layers emerge. The layers that a community gains are composed of the reified sediments of the artefacts of practice and of the rituals that emerge from a shared repertoire. To make their way through this hard knowledge, new members must negotiate new meanings. Brokers are there to help infuse this process, to facilitate participation simply by being helpful and receptive. That is their key skill. It is not complex or technical but it is hard to fake (and sometimes hard to come by for the bewildered newcomer).

5.6.3 Group Identity and Community Entry

Buying into the ethos of the community is also important. There are cultural and behavioural norms that must be learned. As a project matures, its community begins to have its own established mythology and in-jokes e.g. Moodle Documentation Fairy. Moreover many members of the Moodle community have a strong sense of wider mission as we detected in Chapter Four. There are many with a strong ideological identification with Moodle's purpose and correlated strong positive identifications and feelings towards the community. Not all participants share these feelings, it must be noted, though they were very prominent among core community members. This ties with existing research on motivations for participating in open source software projects. One particularly relevant study for example used surveys to elicit participants' ratings of the leadership effectiveness, interpersonal relationship and community ideology of a project (Xu et al., 2009). What our study tells us however is something of the unique ideology of the Moodle Community itself. It suggests that individual projects may have unique ethoi. It is reasonable to suppose that we will not find the

[2] There is of course important work in this area as we have seen in Chapter Two (Von Krogh, 1998; Chesbrough, 2003) that looks at open source software development as a form of "open innovation" but not in detail at the particular role and identities of intermediaries, and as follows the complexities inherent in their evolutionary involvement.

motivation of one participant here that “Moodle is a part, or a piece of, my hope for more open education” (Interview M3) in most open source communities.

5.6.4 Momentums and Inertias

Trajectories are twofold. There is the lifecycle of an individual bug and also the larger but related arc of how someone may enter a community, be acculturated to it and learn to be a member through participating in the periphery in legitimate activities. The legitimacy of these activities as we have seen may be well known and well described, or they may be implicit normative behaviours and only tacitly attained. Moreover, an individual may blaze their own trail and beget their own role within the project. In the case of Moodle this could lead to a fulltime job. Every large trajectory is comprised of many smaller vectors. We looked at these vectors in detail as the processes of issue resolution. One code that emerged (from emic terminology) in the analysis was *driving a change to Moodle.* This encompasses several factors that must be attained in order for a successful change to occur. What happens once this change has been effected is another matter. Research suggests that users who have a very specific need may leave the project once that need has been met (Roberts et al., 2006; Shah, 2006).

We should not be surprised then to have discovered that committing to maintain an issue can be a factor in getting the fix accepted into Moodle. Individual contributors may be bent upon solving some particular problem that is vexing their own local community of students, fellow teachers, or clients. The Moodle core community however are interested not just in the end of the journey of *that* bug, but in a whole class of bugs like it. For the Moodle project the ideal path to resolution of an issue is one that brings a person with it. This explains why enculturation is so important to communities of practice, acting as a binding agent to attract and retain participants. Hence, for the health of the community and the project overall, it may be necessary to reject certain fixes, and that though it may seem counter-intuitive, the fixing of bugs is not always the purpose of the tracker.

There are other obvious inertias. Just as a member of an outside community may be focused on expediting a very specific issue, they may also use the tracker as a source of knowledge to either attain or develop a patch of their own. Again these “workarounds” (patches), which may appear in the tracker, can ostensibly work against the tracker’s canonical purpose of issue resolution. They may be “good enough” as a solution for members of one particular section of the community but inadequate for the Moodle core and hence be unavailable to the majority of Moodle users. This type of inertia is one of a misalignment of two communities. For the patch-user the outcome is good enough (Schwartz et al., 2002) and may indeed be optimal if the effort and skill required to take and apply this patch in their local community gives them capital there. Moreover, finding usable patches in the tracker gave one user a "sense of community" in and of itself (Interview M11). The definition of community i.e. of what is best for Moodle and of what the outcome of issue resolution can or should be is a contested concept. This could be seen to problematise our whole question of issue resolution, because the successful outcome of an issue may not necessarily be its designation as "fixed" in the tracker. An entire research lineage is predicated on this assumption (Strate & Laplante, 2013) and of course it has great utility but there are also interesting side-effects of the tracker (such as for example, unapproved/unofficial patches) that may in some cases militate against issue resolution but nonetheless provide successful outcomes for some participants.

The final example of an inertia detailed in Chapter Four did not fit neatly with the previous evidence insofar as it did not revolve around tracker issues in general but one particular instance of an issue in the development of Moodle, one that came up in the interview with Martin Dougiamas and which was triangulated with evidence from the Moodle documentation wiki. This issue was concerned with tool adoption by the developers and the struggle within the core community to resist calls for the introduction of a new tool (JQuery) at the expense of an existing one (YUI). Identity becomes relevant here. Similar to the findings of Raymond (1999) and Stewart (2005), a Moodle developer's identity is, as we have seen, defined by their skill in using a particular tool to write good code. Suggestions that they are using the wrong tool, a redundant tool, or that they should rewrite their code using a new tool are direct affronts to that identity, and hence to the legitimacy of the group. The aggressive tone in the documentation displays contemptuousness for outsiders who are demanding the introduction of the new tool to Moodle. This tone in itself may be a touchstone for the identity of the core community, or at least it is safe to assume that the existing tool is a key part of the culture and the "shared repertoire" (Wenger, 1999) of the community. Although the community identify with the tool and its use, "the very process of identification constrains negotiability" and hence "membership is both enabling and limiting of identity. It is a resource and a cost" (Wenger, 1999, p. 207). It may very well be, as Dougiamas outlined, that there is little real choice. It may be that the existing tool has been so deeply embedded into existing work practices that the effort of its excision and replacement would be so great as to negate any potential benefits. No matter what the reason for this inertia, it represents an example of a blocker which only a very great amount of negotiation over a sustained time might unlock and thus shows that specific issue resolutions can have one overwhelming impeding factor i.e. it confirms the existence of superbugs (Dalle & den Besten, 2007). This is the type of issue outlier that rarely fits neatly into the research literature which focuses on average (i.e. predictable) outcomes.

A related example from the literature may be useful here to contextualise this. Bug 213186 from the Firefox web browser concerned a request to change the explanation text given to browser users for cookies from the irreverent "delicious delicacies" to something more meaningful (Dalle et al., 2008). This issue took a long time to resolve and there was considerable debate between those in favour of retaining the "geek humour" and those who felt that it was actually misleading to many users of the browser who did not know what cookies were. This is a rather trivial example and also an exceptional one (most tracker issues do not involve in-jokes) but the important point is that the joke is only humorous by presupposing an outsider group who are not in on it, who cannot get it because they do not have access to the shared cultural repertoire of the community.

What these examples of the resistance to the introduction of the new tool to Moodle and the Firefox cookies show, is that inertias can have very dominant cultural components. That is not to say that everything is determined culturally. An outsider may come to the tracker and propose (politely) a fix or a feature for Moodle that constitutes a "big win", which attracts the attention of the potential fixer and causes them to act and resolve it. Fixing issues is after all the avowed purpose of all of those who participate in the tracker, but not at any cost. The fix may also need to affirm the community and its members.

5.6.5 Discussion Conclusion

This case study has examined, analysed and portrayed Moodle's bug tracker through the lens of the practice of issue resolution and how participants come together to engage in this practice. Moodle is an open source community and as a member of this class of phenomena, it can draw on a rich, voluminous and growing body of research literature to be explained. However it also has many unique aspects; for example its situation within the world of education. For this reason we conducted a single-case study that would explore, elucidate and describe it in the hope of laying its constituents open for others to research further (Yin, 2009). Simple derivation of factors believed important to issue resolution became a key task. This task is one that is often overlooked in the literature and hence one where this study found aspects not widely reported elsewhere. Our use of the communities of practice model allowed us to analyse these factors in the context of boundary objects and around the theme of canonical and non-canonical descriptions of work practices.

For example, the behaviour of the submitter is not something that is generally reified or codified in the models of how bugs should be fixed. This phenomenon is not completely unknown in the research literature around open source communities. For example Ko and Chilana (2010) provide some interesting examples of aggressive and demanding bug submission requests. However it is rarely accorded the attention it is due, nor its full implications examined. The behaviour of submitters is important not simply because they are asking someone to do something for which they may be seen to be giving very little in return – after all it should be in the interest of assignees to improve Moodle, to fix it and make it better. It is also important because every bug report is on some level an implicit criticism of Moodle, a critique of the existing order. The reporter has found a flaw, an explicit deficiency. They are pointing out this flaw in the Moodle source code, one of the key shared artefacts around which the universe of Moodle revolves and one into which developers, brokers, testers and core community members have poured the soul of their efforts for years.

For developers in particular, the code of Moodle may have a deep personal significance. They may love the look of good code and love looking at code; it may be “poetry” to them and parts of Moodle they have built may be their “toys”. In order to improve this code they must elicit feedback from end-users. The must invite criticism on their creation. This feedback will come from primary school teachers, secondary school teachers, university lecturers, business consultants, learning technologists and systems administrators. These end users, way out on the periphery, will likely have no appreciation of poetry; will probably not even know it exists.

Other researchers have judged the relative contribution of non-assignees to be minor upon finding that “the masses” *only* reported 21% of all fixed issues (Ko & Chilana, 2010). This figure was 24% for non-assignees in Moodle. If we consider the ever lengthening distance between the outer periphery of Moodle and the code lines of its deepest core as it grows and complexifies, we can instead marvel that *so much* of Moodle’s contributions are suggested by those who never do the hard work of then writing the code to implement them. Almost a quarter of the DNA of Moodle that is accounted for by the tracker has been actively shaped by people from outside the community, or more accurately from those only loosely coupled to it, from those where its fringes blend into the peripheries of other distinct communities such as schools and universities. In doing so they have traversed a process that is comprised of some combination of the over thirty factors that we found here to be potentially important to issue resolution. They have managed to negotiate a new meaning, however small, of Moodle that will now apply to the millions of users of Moodle spread throughout the world.

There is no defined prescription given here for how educators can get their fix or feature request into Moodle. To be sure they can be polite and be mindful that they are treading in a cherished creation; they can be responsive and attempt to establish an immediate identity of "helpfulness"; they may wish to seek out and enlist the help of a mediator, of someone who cannot help them directly but who can show them the ropes, who can translate their request and bring it to the attention of the right person; they can employ someone to write the code if they have the resources to do this; and, when all else fails, a forceful approach may then be worth resorting to. However, there are no guarantees that any or all of this will work. The particular part of Moodle they wish to affect may be orphaned, may have an endemic structural or cultural resistance to change. There is not much way to know for certain if a particular issue can be resolved, though much can be gleaned if one is willing to commit to a sustained engagement, to venturing further into the fold of the tracker community. In doing this more can be learned of the unwritten rules of issue resolution, some of which have been glimpsed here (although it is certain that others have eluded us).

Another way to posit this is to consider the theoretical possibility of a guaranteed path to a Moodle bug fix, of a secret formula that one could employ that would guarantee that one's fix would be implemented. If this guaranteed path to a fix became known to someone, they could have all of their issues fixed. They would be in control of Moodle. Or, if everyone knew this formula then everyone would have their issues fixed (even if issue A required Moodle to be blue while issue B required Moodle to be red). This hypothetical situation cannot of course exist. We saw how voting was partially distrusted for this very reason – there can be no completely knowable path to a fix, no easy way. There must always be uncertainty and engagement must always be required. And so Moodle only evolves. It can never be entirely directed, its essence remains distributed. This perhaps partly explains why particularly key members may move to Moodle HQ – this may represent the community attempting to wrest some of its agency, to bring those people who appear to have special knowledge or influence over bug fixing closer to the core.

It also casts some inertias in a different light. A developer, for example someone tasked with maintaining a version of Moodle installed in a university, may have a “good enough” solution that works locally but one which never makes its way back into the Moodle core codebase. However this is not necessarily negative. At the outset of the study this researcher wondered why more local innovations did not make their way back upstream to their parent, back into canonical Moodle so that they could be made available for the benefit of everyone (Costello & Johnston, Forthcoming). Now however it is clear that mutated versions of Moodle, which live in the server rooms of universities or under the desks of ICT teachers in small schools, are a positive thing as they represent another form of the distributed ownership of Moodle. They make the Moodle ecosystem richer, more heterogeneous and stronger precisely by resisting Moodle HQ. Martin Dougiamas himself, as one of the successors of Orr’s (2006) “inveterate tinkerers” who rebelled against the closed source code of WebCT to build Moodle, would no doubt concur.

This non-participation is important. This study sought to examine how educators participate in developing open source educational software using Moodle as a case study. It has elucidated how this happens in Moodle, what some of the unique characteristics of it may be as a community and shone a light on how educators come to participate in the inner core such as: by becoming designated issue assignees in the tracker, perhaps by moving to Moodle HQ itself; or indeed most simply by affecting a change to the Moodle code by contributing an issue that gets fixed. However even this seemingly simple work quantum of fixing a single issue is not all it appears to be. Not every fix is good for Moodle. Each participant of the mutual engagement required to negotiate a new meaning for Moodle at this atomic level has a unique identity, has goals derived from both global and local communities. They may very well dispute how a fix should be made, whether it is necessary, or even its eventual canonical status as “fixed”. We should not be disappointed that this creative social furnace, where the classrooms of the future are being forged, should sometimes prove resistant to having its fundamental practice neatly defined. Notwithstanding this resistance this study has told some of *the story of this practice*, of the tale of those engaged in the pursuit of bug tracker issue resolution in open source educational software, and of how educators participate in the development of Moodle.

6 Conclusion

6.1 Summary of Findings

This study had its own trajectory. In a manner not dissimilar to that of someone with a particularly “annoying” and intractable bug on their hands it began with a clear impetus. Indeed, in more than one sense, it began with a bug. A bug in the Moodle Quiz was the start of the story of this research. There also appeared at the outset to be a “bug” in the literature, a small piece missing. The question of how educators could fix Moodle bugs that impacted their teaching practice had not been examined. This spawned a research process, the formulation of a Doctoral research proposal, the search for a supervisor etc. and so began a long journey. Along the route a research approach was adopted and a study designed, evidence marshalled analysed and weighted. Crucial data became the stories told by the bug fixers themselves whether in hotel rooms in between sessions of a Moodle conference or in Skype conversations held during the most convenient overlap of two time zones. The text of these interviews became the input of a coding process that fractured and sorted the data meticulously so that meanings could then be constructed from it by the researcher.

Specific themes were identified as relevant and analysed in the literature review of open source communities such as the motivation and identity of contributors, joining processes and some specific approaches and examples of building models of bug resolution. In the light of this study’s findings, models of bug resolution may be further categorised as those which account for *de facto* as well as *de jure* processes, which look at non-canonical accounts of practice in addition to those of a legitimatised and explicit canon of the same. According to Wenger (1999), both exist in a duality, that is, neither can exist without the other and indeed actual practice is the oil in the system, is that which softens hard or reified and canonical knowledge. Commenting in bug trackers is an obvious example that is clear in the literature and affirmed here. Taking this concept further we found and elaborated upon a more abstract concept of *responsiveness* which surprisingly is not examined closely in the literature with one notable exception provided by Breu et al. (2010). As we ventured further into the spectrum of non-canonical constituents of the bug fixing process we found other abstract concepts such as the behaviour of the submitter. Although it may seem obviously important that a submitter be polite if they wish to have their bug fixed, this aspect is curiously not well represented in existing models of related practice. Another factor is quite intriguing and not so obvious: that a bug may be “interesting” or “annoying” to an assignee i.e. specific bugs may act as *intrinsic motivators* in their own resolution.

The non-canonical factors identified above do not appear to be unique to Moodle. Other findings of this study by contrast fulfilled the aim of painting a picture of Moodle’s unique culture and ethos. The professional identities that make up Moodle are a heterogeneous group drawn from educational and computing worlds. Sometimes it is difficult to distinguish where one of these worlds ends and the other begins. An argument could be made that a coder of Moodle may have more influence upon education than a theorist famous in pedagogical research. Moreover, if developers are teachers then the teachers that can navigate the Moodle tracker are also developers. Tracker participants live all over the (developed) world with a high Antipodean and European representation. Relatively speaking many of them are women. This mix of identities and roles contributes to the particular culture and ethos of the community, one that is also defined by the philosophical ideals of open source and open education, and normatively by the behaviour of helpfulness.

Perhaps the most significant contribution of this research was in its exposition of the roles of brokers within the Moodle tracker. Specific individuals were found to be crucially important to the participatory architecture of the community. Its founder and several of its key participants occupy positions that create vital social linkages. They may induct new members or they may mediate the process of bug-fixing itself. Filing a bug report on behalf of someone else is a hitherto undocumented aspect of the practice of issue resolution in open source communities. It may well be particularly related to the ethos, form and identities of a particular community of teaching and helping. Alternatively it may simply be something that is lying in wait of other researchers in other communities at a similar stage of maturity and with a similar interface to specialist outside worlds which those communities exist to serve. Either way, mediation acts - such as proxy bug reporting and improvement - are not adequately accounted for in existing bug resolution models. They may be difficult to determine programmatically, to derive from statistical analyses of bug tracker databases, but they are clear here in the narrative accounts of the participants.

Lastly, it was theorised that formulae predictive of bug resolutions may, to temporarily take an extreme philosophical position, be impossible and that this can be seen as an important function of the system itself i.e. it has a property that makes it always resistant to being fully known. Even if bug fixing could be predicted with a high degree of probability such knowledge would be problematic to usefully deploy because fixes are not always benign or desirable. Some unfixed or duplicate bugs have as an important a role to play as fixed ones do. The cost of a fix may very well be traded, in practice, for some other non-canonical solution.

6.2 Potential Implications and Future Research

This study was informed by a social constructivist philosophical perspective. The researcher relied on the participants' views and these were then interpreted through analysis to generate meanings. Such an approach does not purport to yield objective truths as discussed in Chapter Two. Equally however, alternative approaches may be perfectly right about the wrong things if they act from overly positivist positions. Studies can produce quantitative models of boundary objects such as trackers, which exhibit high degrees of rigour and statistical significance, but they may only be considering canonical factors, those most explicitly exposed by the database or the official documentation and not considering other significant hidden evidence. This has important implications for future research. There is great potential to know more about tracker practices through interviews or indeed through ethnographic studies that employ more direct observation and even participation (both of which could be carried out virtually as well as, or in addition to, being physically co-located with participants).

Specific research questions that could build upon this study include ones that would model certain non-canonical and undervalued aspects of the bug resolution process. A useful study could start with an approach that sought participants' views on a focused topic such as the behaviour of the submitter. These views could be analysed to form behavioural models of submitters. A well-defined model could in turn lend itself to testing by survey using a greater number of participants than could be practically interviewed. Such a model could alternatively (or additionally) be triangulated with bug tracker database data. Another research avenue that might prove fruitful here would be a similar mixed-methods approach starting with collecting rich accounts of participants' views of affective factors that assignees may consider when choosing or evaluating issues (e.g. whether they are "interesting" or "annoying"). Related work (Aranda & Venolia, 2009), which focused on obtaining participant views of the readability of bug issues (a remarkably under-utilised research strategy in this field), could provide a good basis for this.

A key area of future research that would build upon this study would investigate further the role of brokers, particularly ones that specialise in the altruistic task of filing and improving bug reports on behalf of strangers, the *kindly souls* that mediated between the discussion forums and the bug tracker in Moodle. Do these actors occur comparably in other projects with the same intrinsic motivations? How pervasive are they and what is the extent of their work? These are questions whose answers would contribute much to the research field of open source software communities. Many studies of boundary object bug trackers conclude with recommendations of proscriptions to help improve the database software. It is tempting to do the same here, to suggest ways of capturing responsiveness for example, many aspects of which are easily definable; or to design some process to make the filing of bugs easier, the accompanying descriptions clearer; and maybe even of putting the submitter into a calm and polite frame of mind. Instead however, what is most recommended here for bug tracker communities is that they invest instead in brokers; that they try to identify the areas where two hands grasp but miss each other at a boundary and put a mediator at that spot that can hold both hands. It may almost seem perverse to suggest adding people to a system instead of trying to automate it. This may seem to go against the typical grain of process improvement until it is borne in mind that any system as it becomes more complex, has a need for specialists. Just as Neolithic farmers gave rise to the writers and accountants necessary to record their food surpluses, so too any social system as it grows will differentiate and give rise to roles defined by meta-activities. A community that relies on global sources for a quarter of its contributions will need brokers to explain, mediate and induct people from these wider worlds.

There are lessons here for outside communities too. Universities and schools would do well to cultivate and retain expertise in educational technology, specifically in the skills necessary to contribute to projects like Moodle. The trend to outsource as much of a university's functions as possible, so that it focuses only on its core competencies, may make economic sense for individual institutions but the long term effects of this are uncertain. Wenger (1999) gives us the salutatory tale of the technicians, who were a specialist form of insurance claims processor. At first they worked embedded with ordinary claims processors but later the company decided to pool them all together in one office separate from the offices of the ordinary claims processor teams. At this point relationships between the two sets began to deteriorate and insurance claims that could have been handled by the ordinary processors, had they still been working cheek by jowl with the technicians, began instead to be routinely referred to the technicians. The value of having experts in Moodle working in schools and universities should not be underestimated for similar reasons. They provide a mediating link, a lifeline that tethers an educational institution to Moodle, gives it a stake in the creation and re-creation of what is becoming more and more a part of their critical infrastructure.

Having a stake in Moodle and not outsourcing all aspects of its development, running or maintenance are important tasks that universities who use it should be engaged in. These are times perhaps of great tumult in education. The rise of MOOCs has thrown the university into something of an existential crisis (Daniel, 2012). How serious this disruption will prove remains to be seen but there is a concern that a power grab could be afoot and a dominant player or players could come to dominate the third level educational landscape. The organisation at the reins, be it Udacity, Coursera or some other entity, could conceivably come to control access to education. A not dissimilar case could also be made that the centralisation of power in the hands of Moodle is a real prospect, given that there are few viable choices of VLE now for large institutions. Educational institutions are increasingly reliant on VLEs to fulfil their educational missions and so it is certainly arguable they should not abrogate all responsibility for the development of these same VLEs (benign and open as Moodle may ostensibly be).

It is hard however for universities to see the big picture or even if they do see it to pay for it. Few institutions would be sufficiently large as to be able to plough their own furrow such as the OU does with its contribution to the development of Moodle. Rather governments might have a role to play in creating consortia dedicated to open source efforts that were deemed critical to educational institutions in their jurisdictions. The critical point is that the development of educational technology should not simply be left to the experts, to Moodle HQ and the Moodle Partners. It is good for the health of the ecosystem that a system can evolve and mutate, that Moodle can be evolved and experimented upon outside of its core community, that in a sense it can spawn alternative cores.

Although the barriers to entry as a programmer to Moodle became higher as the project specialised and its construction became more professional it is nonetheless still possible for educators to affect change within Moodle as has been shown here. Moreover, the evidence presented in this thesis may help educators to better understand the Moodle community and be more influential within in. It is possible for anyone to tinker with Moodle. All that is required is an impetus to do so. Moodle is still free to play with. While we have just touched on ominous headwinds that may be facing education we can finish on a note of hope: Moodle in its open source form still affords one of the most basic and joyous premises of education – play.

Bibliography

Agar, M. (1991) The right brain strikes back. IN: Fielding, R. and Lee, R. eds. Using Computers in Qualitative Research. London, UK: Sage Publications. pp 181-194.

Aksulu, A. and Wade, M. (2010) A comprehensive review and synthesis of open source research. Journal of the Association for Information Systems. vol. 11, no. 11, pp 576-656.

Alier, M., Casa, M. and Piguillem, J. (2010) Moodle 2.0: Shifting from a Learning Toolkit to a Open Learning Platform. IN: Lytras, M.D., Ordonez De Pablos, P., Avison, D., et al eds. Technology Enhanced Learning. Quality of Teaching and Educational Reform. Berlin: Springer. pp 1-10.

Allen, W. (2009) Boundary Objects in Hybrid Commercial/Open-Source Software Development Firms. Philadelphia, PA: Drexel University.

Anderson, C. and Andersson, M.P. (2006) The long tail. New York: Hyperion

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

Ardichvili, A., Page, V. and Wentling, T. (2003) Motivation and barriers to participation in virtual knowledge-sharing communities of practice. Journal of knowledge management. vol. 7, no. 1, pp 64-77.

Ashforth, B. E., and Mael, F. (1989). Social identity theory and the organization. Academy of management review, vol. 14, no. 1, pp 20-39.

Barab, S., Kling, R. and Gray, J.H. (2004) Designing for virtual communities in the service of learning. New York, NY: Cambridge University Press.

Barham, A. (2012) The Impact of Formal QA Practices on FLOSS Communities–The Case of Mozilla. IN: Hammouda, I., Lundell, B., Mikkonen, T. and Scacchi, W. eds. Open Source Systems: Long-Term Sustainability. London, UK: Springer. pp 262-267.

Bathelt, H., Malmberg, A. and Maskell, P. (2004) Clusters and knowledge: local buzz, global pipelines and the process of knowledge creation. Prog.Hum.Geogr. vol. 28, no. 1, pp 31-56.

Bazeley, P. (2007) Qualitative data analysis with NVivo. London, UK: Sage Publications.

Becker, G.S. (2009) Human capital: A theoretical and empirical analysis, with special reference to education. Chicago: Univ. of Chicago Press.

Benford, R. D. and Snow, D. A. (2000) Framing processes and social movements: An overview and assessment. Annual review of sociology. vol. 26, pp 611-639.

Bennett, S. (2011). Report for AUT university LMS review group May 2011: Learning management systems: A review. Auckland: AUT.

Bergquist, M. and Ljungberg, J. (2008) The power of gifts: organizing social relationships in open source communities. Information Systems Journal. vol. 11, no. 4, pp 305-320.

Bezroukov, N. (1999) Open source software development as a special type of academic research First Monday [Online]. vol. 4, no. 10, Available from: http://firstmonday.org/issues/issue4_10/bezroukov/index.html [Accessed 17/12/2009].

Bird, C., Gourley, A., Devanbu, P., Swaminathan, A. and Hsu, G. (2007) Open borders? immigration in open source projects. IN: Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on. Minneapolis, MN, 20-26 May 2007. IEEE.

Braun, V. and Clarke, V. (2006) Using thematic analysis in psychology. Qualitative research in psychology. vol. 3, no. 2, pp 77-101.

Brenner, M.E. (2006) Interviewing in educational research. IN: Green, J., Camilli, G. and Elmore, P. eds. Handbook of complementary methods in education research. Mahwah, NJ: Lawrence Erlbaum Associates. pp 357-370.

Breu, S., Premraj, R., Sillito, J. and Zimmermann, T. (2010) Information needs in bug reports: improving cooperation between developers and users. IN: Proceedings of the 2010 ACM conference on Computer supported cooperative work. Savannah, GA, Feb 6–10. ACM.

Brown, J. S. and Adler, R. P. (2008) Open education, the long tail, and learning 2.0. Educause review. vol. 43, no. 1, pp 16-20.

Brown, J. S., Collins, A. and Duguid, P. (1989) Situated cognition and the culture of learning. Educational researcher. vol. 18, no. 1, pp 32-42.

Brown, J. S. and Duguid, P. (1998) Organizing knowledge. Calif.Manage.Rev. vol. 40, no. 3, pp 91-111.

Brown, J. S. and Duguid, P. (1991) Organizational learning and communities-of-practice: Toward a unified view of working, learning, and innovation. Organization Science. vol. 2, no. 1, pp 40-57.

Browne, T., Jenkins, M., and Walker, R. (2006). A longitudinal perspective regarding the use of VLEs by higher education institutions in the United Kingdom. Interactive Learning Environments, vol. 14, no.2, pp 177–192.

Brunet, P. M. and Schmidt, L. A. (2007) Is shyness context specific? Relation between shyness and online self-disclosure with and without a live webcam in young adults. Journal of Research in Personality. vol. 41, no. 4, pp 938-945.

Carr, W. and Kemmis, S. (1986) Becoming critical: Education, knowledge, and action research. London, UK: Routledge.

Charmaz, K. (2006) Constructing grounded theory: A practical guide through qualitative analysis. London, UK: Sage Publications.

Chesbrough, H. W. (2003) The Era of Open Innovation. MIT Sloan management review. vol. 44, no. 3, pp 35-41.

Chiu, C. M., Hsu, M. H. and Wang, E. T. G. (2006) Understanding knowledge sharing in virtual communities: An integration of social capital and social cognitive theories. Decis.Support Syst. vol. 42, no. 3, pp 1872-1888.

Christensen, C. M., Horn, M. B., and Johnson, C. W. (2008) Disrupting class: How disruptive innovation will change the way the world learns. New York, NY: McGraw-Hill.

Clifford, W. K. (1879) The ethics of belief. Lectures and essays. vol. 2, pp 177-211.

Cole, J. and Foster, H. (2007) Using Moodle: teaching with the popular open source course management system. Sebastopol, CA: O'Reilly Media.

Cole, R. E. and Lee, K. G. (2003) From a firm-based to a community-based model of knowledge creation: The case of the Linux kernel development. Organization Science. vol. 14, no. 6, pp 633-649.

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

Costello, E. (2014) Opening up to Open Source: Looking at how Moodle was adopted in Higher Education Open Learning. vol. 28, no. 2, pp. 187-200. Available from: http://dx.doi.org/10.1080/02680513.2013.856289 [Accessed 14/05/2014].

Costello, E. (2012) Smart schools = smart economy: intelligence equation or textspeak policy? Trinity Education Papers. vol. 1, no. 1, pp 72-93.

Costello, E. and Johnston, K. (Forthcoming) Moodle from VLE to PLE. IN: Siemens, G., Downes, S. and Kop, R. eds. Personal Learning Environments and Personal Learning Networks. 1st. Athabasca: Athabasca University and the National Research Council of Canada.

Côté, J. E. (1996) Sociological perspectives on identity formation: The culture–identity link and identity capital. J.Adolesc. vol. 19, no. 5, pp 417-428.

Cox, A. (2005) What are communities of practice? A comparative review of four seminal works. J.Inf.Sci. vol. 31, no. 6, pp 527-540.

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

Creswell, J.W. (2009) Research Design: Qualitative, Quantitative, and Mixed Methods Approaches. London, UK: Sage Publications.

Crossley, S. A., Greenfield, J. and McNamara, D. S. (2008) Assessing text readability using cognitively based indices. Tesol Quarterly. vol. 42, no. 3, pp 475-493.

Crotty, M. (1998) The Foundations of Social Research: Meaning and Perspective in the Research Process. London, UK: Sage Publications.

Crowston, K. and Howison, J. (2006) Assessing the health of open source communities. Computer. vol. 39, no. 5, pp 89-91.

Crowston, K. and Howison, J. (2005) The social structure of free and open source software development. First Monday [Online]. vol. 10, no. 2-7, Available from: http://firstmonday.org/ojs/index.php/fm/rt/printerFriendly/1478/1393 [Accessed 12/03/2009].

Crowston, K. and Scozzi, B. (2004) Coordination practices for bug fixing within FLOSS development teams. IN: Proceedings of the First International Workshop on Computer Supported Activity Coordination (CSAC 2004). Porto, Portugal. April 13-14, 2004.

Crowston, K., Wei, K., Howison, J. and Wiggins, A. (2012) Free/Libre open-source software development: What we know and what we do not know. ACM Computing Surveys (CSUR). vol. 44, no. 2, pp 7.

D'Antoni, S. (2009) Open Educational Resources: reviewing initiatives and issues, Open Learning. vol. 24, no. 1 pp. 3-10 . Available from: http://dx.doi.org/10.1080/02680510802625443 [Accessed 14/05/2014].

Dalle, J. M., den Besten, M. and Masmoudi, H. (2008) Channelling Firefox Developers: Mom and Dad Aren't Happy Yet. Open Source Development, Communities and Quality. vol. 275, pp 265-271.

Dalle, J. and den Besten, M. (2007) Different bug fixing regimes? A preliminary case for superbugs. The International Federation for Information Processing. vol. 234, pp 247-252.

Daniel, B., Schwier, R. A. and McCalla, G. (2003) Social capital in virtual learning communities and distributed communities of practice. Canadian Journal of Learning and Technology/La revue canadienne de l'apprentissage et de la technologie. vol. 29, no. 3

Daniel, J. (2012) Making Sense of MOOCs: Musings in a Maze of Myth, Paradox and Possibility. Seoul: Korea National Open University.

De Souza, C.R., Redmiles, D., Mark, G., Penix, J. and Sierhuis, M. (2003) Management of interdependencies in collaborative software development. IN: Empirical Software Engineering, 2003. ISESE 2003. Proceedings. 2003 International Symposium on. Rome, Italy, 30 Sept -1 Oct. IEEE.

DiBona, C. and Ockman, S. (1999) Open sources: Voices from the open source revolution. Sebastopol, CA: O'Reilly Media.

Dougiamas, M. (2011) Keynote Address. IN: Proceedings of Moodle Moot Japan 2011. Kochi, Japan, February 22-23. Kochi: Kochi University of Technology.

Dougiamas, M. (2007) OSS Watch - Moodle: a case study in sustainability [Online]. Available from: http://www.oss-watch.ac.uk/resources/cs-moodle.xml [Accessed 07/07/2009].

Dougiamas, M., and Taylor, P. (2003). Moodle: Using learning communities to create an open source course management system. IN: Proceedings of the EDMEDIA 2003 Conference, Honolulu, Hawaii, June 23-28, pp 171–178

Dougiamas, M., and Taylor, P. C. (2002). Interpretive analysis of an internet-based course constructed using a new courseware tool called Moodle. IN: Proceedings of the Conference of HERDSA, Perth, Australia, July 7th. Available from http://online.dimitra.gr/sektrainers/file.php/1/MartinDougiamas.pdf [Accessed 07/03/2010]

Ducheneaut, N. (2005) Socialization in an open source software community: A socio-technical analysis. Computer Supported Cooperative Work (CSCW). vol. 14, no. 4, pp 323-368.

Dron, J. (2006) Any color you like, as long as it’s blackboard. IN: Proceedings of E-Learn 2006 World Conference on E-Learning in Corporate, Government, Healthcare, & Higher Education. Honolulu, Hawaii. Oct 13-17

Erikson, E. H. (1950). Childhood and Society. New York, NY: Norton

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

Ess,C. and Jones (2012) Ethical decision-making and Internet research: Recommendations from the AOIR ethics working committee [Online]. Available from: http://aoir.org/documents/ethics-guide/ [Accessed 16/01/2012].

Farmer, J. and Dolphin, I. (2005) Sakai: eLearning and more. IN: EUNIS 2005-Leadership and Strategy in a Cyber-Infrastructure World. Manchester, UK, 21-24 June.

Flyvbjerg, B. (2006) Five misunderstandings about case-study research. Qualitative Inquiry. vol. 12, no. 2, pp 219.

Fontana, A. and Frey, J.H. (1994) Interviewing - the Art of Science. IN: Denzin, N. and Yvonna, L. eds. Handbook of Qualitative Research. London, UK: Sage Publications. pp 361-377.

Fox, S. (2000) Communities Of Practice, Foucault And Actor-Network Theory. Journal of Management Studies. vol. 37, no. 6, pp 853-868.

Francalanci, C. and Merlo, F. (2008) Empirical analysis of the bug fixing process in open source projects. Open Source Development, Communities and Quality. vol. 275, pp 187-196.

Freeman, S. (2007) The Material and social dynamics of motivation. Science Studies. vol. 20, no. 2, pp 55-77.

Gardler,R. and Hanganu (2012) Governance Models [Online]. Available from: http://www.oss-watch.ac.uk/resources/governanceModels [Accessed 12/12/2012].

Geertz, C. (1973) The Interpretation of Cultures: Selected Essays. New York, NY: Basic Books.

Ghezzi, C., Jazayeri, M. and Mandrioli, D. (2002) Fundamentals of software engineering. Upper Saddle River, NJ: Prentice Hall.

Ghosh, R.A., Glott, R., Krieger, B. and Robles, G. (2002) Free/libre and open source software: Survey and study. University of Maastricht, The Netherlands: Maastricht Economic Research Institute on Innovation and Technology.

Giger, E., Pinzger, M. and Gall, H. (2010) Predicting the fix time of bugs. IN: Proceedings of the 2nd International Workshop on Recommendation Systems for Software Engineering. Capetown, South Africa, May 4, 2010. ACM.

Gilbert, L. S. (2002) Going the distance:'closeness' in qualitative data analysis software. International Journal of Social Research Methodology. vol. 5, no. 3, pp 215-228.

Gillespie, J. (2000) Communities of Practice: the buzz and the buzzword. Journal of Industrial Teacher Education. vol. 37, no. 2, pp 96-100.

Glaser, B. G. (2008) Conceptualization: On theory and theorizing using grounded theory. International Journal of Qualitative Methods. vol. 1, no. 2, pp 23-38.

GNU.org (2007) The GNU General Public License [Online]. Available from: http://www.gnu.org/licenses/gpl.html [Accessed 11/02/2012].

Gobo, G. (2008) Re-conceptualizing generalization: Old issues in a new frame. IN: Alasuutari, P., Bickman, L. and Brannen, J. eds. The Sage Handbook of Social Research Methods. 1st. London, UK: Sage Publications. pp 193-213.

Gomm, R. (2004) Social research methodology. New York, NY: Palgrave Macmillan.

Government of Ireland. (2003) Data Protection (Amendment) Act. Dublin: Stationary Office.

Government of Ireland. (1998) Data Protection Act. Dublin: Stationary Office.

Grabher, G. (2002) Cool projects, boring institutions: temporary collaboration in social context. Reg.Stud. vol. 36, no. 3, pp 205-214.

Greenhow, C. and Robelia, B. (2009) Informal learning and identity formation in online social networks. Learning, Media and Technology. vol. 34, no. 2, pp 119-140.

Greer, C. R. and Lei, D. (2012) Collaborative Innovation with Customers: A Review of the Literature and Suggestions for Future Research. International Journal of Management Reviews. vol. 14, no. 1, pp 63-84.

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

Guest, G., Bunce, A. and Johnson, L. (2006) How many interviews are enough? An experiment with data saturation and variability. Field methods. vol. 18, no. 1, pp 59-82.

Guo, P.J., Zimmermann, T., Nagappan, N. and Murphy, B. (2010) Characterizing and predicting which bugs get fixed: An empirical study of Microsoft Windows. IN: Software Engineering, 2010 ACM/IEEE 32nd International Conference on. Cape Town, South Africa, 2-8 May 2010. IEEE.

Hammersley, M. (1992) What's wrong with ethnography?: Methodological explorations. New York, NY: Routledge.

Hawkins, B. L., and Rudy, J. A. (2008). EDUCAUSE core data service: Fiscal year 2007 summary report. Educause. Available from: http://net.educause.edu/ir/library/pdf/pub8005.pdf [Accessed 06/02/2011].

Hayes, N. and Walsham, G. (2000) Competing interpretations of computer-supported cooperative work in organizational contexts. Organization. vol. 7, no. 1, pp 49-67.

Herraiz, I., Robles, G., Amor, J.J.É., Romera, T. and González Barahona, J.M. (2006) The processes of joining in global distributed software projects. IN: Proceedings of the 2006 international workshop on Global software development for the practitioner. Shanghai, China, May 23. New York, NY: ACM.

Hertel, G., Niedner, S. and Herrmann, S. (2003) Motivation of software developers in Open Source projects: an Internet-based survey of contributors to the Linux kernel. Research policy. vol. 32, no. 7, pp 1159-1177.

Hildreth, P. M. and Kimble, C. (2002) The duality of knowledge. Information Research. vol. 8, no. 1, pp 27-38.

Hooimeijer, P. and Weimer, W. (2007) Modeling bug report quality. IN: Proceedings of the twenty-second IEEE/ACM international conference on Automated software engineering. Atlanta, GA, Nov 05 - 09, 2007. New York, NY: ACM.

Hunt,T. (2011) MDL-1647 [Online]. Available from: https://tracker.moodle.org/browse/MDL-1647?attachmentOrder=desc [Accessed 06/02/2011].

Hunt,T. (2010) Herding Cats [Online]. Available from: http://tjhunt.blogspot.com/2009/08/herding-cats.html?showComment=1254171812177#c6707299409006449057 [Accessed 08/08/2010].

Ibert, O. (2004) Projects and firms as discordant complements: organisational learning in the Munich software ecology. Research Policy. vol. 33, no. 10, pp 1529-1546.

Ihaka, R. and Gentleman, R. (1996). R: A language for data analysis and graphics. Journal of computational and graphical statistics. vol. 5, no. 3, pp 299-314.

Izquierdo-Cortazar, D., Capiluppi, A. and Gonzalez-Barahona, J. M. (2011) Are developers fixing their own bugs?: Tracing bug-fixing and bug-seeding committers. International Journal of Open Source Software and Processes (IJOSSP). vol. 3, no. 2, pp 23-42.

Jensen, C. and Scacchi, W. (2005) Modelling recruitment and role migration processes in OSSD projects. IN: Proceedings of 6th International Workshop on Software Process Simulation and Modelling. St. Louis, MO, May 14-15.

Jensen, C. and Scacchi, W. (2007) Role migration and advancement processes in OSSD projects: A comparative case study. IN: Software Engineering, 2007. ICSE 2007. 29th International Conference on. Minneapolis, MN, May 19 - 23. IEEE.

Jordan, B. (1996) Ethnographic workplace studies and CSCW. Human Factors in Information Technology. vol. 12, pp 17-42.

Jordan, B. (1989) Cosmopolitical obstetrics: Some insights from the training of traditional midwives. Soc.Sci.Med. vol. 28, no. 9, pp 925-937.

Jung, C.G. and Hull, R.F.C. (1968) Psychology and alchemy. Princeton, NJ: Princeton University Press.

Kimble, C., Hildreth, P. and Wright, P. (2001) Communities of practice: going virtual. Knowledge Management and Business Model Innovation. pp 220-234.

Knapik, M. (2008) The qualitative research interview: Participants' responsive participation in knowledge making. International Journal of Qualitative Methods. vol. 5, no. 3, pp 77-93.

Ko, A.J. and Chilana, P.K. (2010) How power users help and hinder open bug reporting. IN: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. New York, NY: ACM.

Krafft, F.M. (2010) A Delphi study on the influences of innovation adoption and process evolution in a large open source project: The case of Debian. Limerick: University of Limerick.

Krishnamurthy, A. and O'Connor, R.V. (2013) An Analysis of the Software Development Processes of Open Source E-Learning Systems. IN: Fergal, M., Rory, V., O'Connor and Richard, M. eds. Systems, Software and Services Process Improvement. Dundalk, Ireland: Springer. pp 60-71.

Krishnamurthy, S. (2006) On the intrinsic and extrinsic motivation of free/libre/open source (FLOSS) developers. Knowledge, Technology & Policy. vol. 18, no. 4, pp 17-39.

Krishnamurthy, S. (2005) About Closed-door Free/Libre/Open Source (FLOSS) Projects: Lessons from the Mozilla Firefox Developer Recruitment Approach. Upgrade. vol. 6, no. 3, pp 28-32.

Krishnamurthy, S. (2002) Cave or community?: An empirical examination of 100 mature open source projects. First Monday [Online]. vol. 7, no. 3, Available from: http://firstmonday.org/ojs/index.php/fm/article/view/960 [Accessed 03/11/2009].

Kuzel, A.J. (1992) Sampling in qualitative inquiry. IN: Crabtree, B. and Miller, W. eds. Doing qualitative research. Newbury Park, CA: Sage Publications, Inc. pp 31-44.

Lakhani, K., Wolf, B., Bates, J. and DiBona, C. (2002) The Boston Consulting Group Hacker Survey. IN: O'Reilly Open Source Conference. San Diego, July 24 2002. O Reilly.

Lakhani, K. and Wolf, R. (2003) Why hackers do what they do: Understanding motivation and effort in free/open source software projects. Cambridge, MA: MIT Sloan Working Paper.

Lakhani, K. R. and Von Hippel, E. (2003) How open source software works: "free" user-to-user assistance. Research policy. vol. 32, no. 6, pp 923-943.

Lave, J. (1988) Cognition in practice: Mind, mathematics and culture in everyday life. New York, NY: Cambridge University Press.

Lave, J. (1977) Cognitive consequences of traditional apprenticeship training in West Africa. Anthropology & Education Quarterly. vol. 8, no. 3, pp 177-180.

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

Lerner, J. and Tirole, J. (2003) Some simple economics of open source. The journal of industrial economics. vol. 50, no. 2, pp 197-234.

Lincoln, Y. S. and Guba, E. G. (1979) The only generalization is: There is no generalization. Case Study Method.London, UK: Sage Publications. pp 27-44.

Lofland, J. and Lofland, L.H. (1995) Analyzing social settings. Belmont, CA: Wadsworth.

Loxley, A. and Seery, A. (2008) Some philosophical and other related issues of insider research. IN: Sikes, P. and Potts, A. eds. Researching Education from the Inside: Investigations from Within. London, UK: Psychology Press. pp 15-30.

Lynn, L.E., Heinrich, C.J. and Hill, C.J. (2002) Improving governance: A new logic for empirical research. Washington DC: Georgetown University Press.

Ma, X., Zhou, M. and Mei, H. (2010) How developers participate in open source projects: a replicate case study on JBossAS, JOnAS and Apache Geronimo. IN: Proceedings of the 1st International Workshop on Replication in Empirical Software Engineering Research.

Marcia, J. E. (1966) Development and validation of ego-identity status. J.Pers.Soc.Psychol. vol. 3, no. 5, pp 551-558.

Markus, M. L. (2007) The governance of free/open source software projects: monolithic, multidimensional, or configurational? Journal of Management and Governance. vol. 11, no. 2, pp 151-163.

Marshall, S. (2011) How to change Mooodle: Working with Moodle HQ. IN: Proceedings of Moodle Moot UK 2011. London, 19-20 April. UCL.

Mason, M. (2010) Sample size and saturation in PhD studies using qualitative interviews. FQS [Online]. vol. 11, no. 3, Available from: http://www.qualitative-research.net/index.php/fqs/article/view/1428 [Accessed 22/09/2010].

Maxwell, J.A. (2004) Qualitative research design: An interactive approach. London, UK: Sage Publications.

McLure Wasko, M. and Faraj, S. (2000) "It is what one does": why people participate and help others in electronic communities of practice. The Journal of Strategic Information Systems. vol. 9, no. 2–3, pp 155-173. Available from: http://www.sciencedirect.com/science/article/pii/S0963868700000457 [Accessed 13/03/2009].

McMullin, B. and Munro, M. (2004) Moodle at DCU. Dublin, Ireland: Dublin City University.

Merriam, S. B. (1985) The Case Study in Educational Research: A Review of Selected Literature. Journal of Educational Thought. vol. 19, no. 3, pp 204-217.

Miles, M.B. and Huberman, A.M. (1994) Qualitative data analysis: An expanded sourcebook. London, UK: Sage Publications.

Mockus, A., Fielding, R. T. and Herbsleb, J. D. (2002) Two case studies of open source software development: Apache and Mozilla. ACM Transactions on Software Engineering and Methodology (TOSEM). vol. 11, no. 3, pp 309-346.

Moodle.org (2013a) Moodle Development Procces [Online]. Available from: http://docs.moodle.org/dev/Process [Accessed 03/02/2013].

Moodle.org (2013b) Moodle JavaScript Documentation [Online]. Available from: http://docs.moodle.org/dev/index.php?title=JavaScript_guidelines&curid=747&diff=38076&oldid=38075 [Accessed 03/02/2013].

Moodle.org (2013c) Moodle Usage Statistics [Online]. Available from: https://moodle.org/stats/ [Accessed 07/10/2013].

Mysirlaki, S. and Paraskeva, F. (2012) Leadership in MMOGs: A Field of Research on Virtual Teams. Electronic Journal of e-Learning. vol. 10, no. 2, pp 223-234.

Neustadt, A. (2006) UML 2 and the Unified Process. New York, NY: McGraw-Hill.

NFQ (2013) National Framework For Qualifications [Online]. Available from: http://www.nfq.ie/nfq/en/FanDiagram/nqai_nfq_08.html [Accessed 05/02/2013].

Nissenbaum, H.F. (2010) Privacy in context: Technology, policy, and the integrity of social life. Stanford, CA: Stanford University Press.

Noble, D. (1989) Digital diploma mills: The automation of higher education. First Monday. vol 3, no 1, Available from: http://uncommonculture.org/ojs/index.php/fm/article/view/569/490 [Accessed 14/05/2014]

Norman, D.A. (2002) The design of everyday things. New York, NY: Basic Books.

O'Mahony, S. (2007) The governance of open source initiatives: what does it mean to be community managed? Journal of Management and Governance. vol. 11, no. 2, pp 139-150.

Oezbek, C., Prechelt, L. and Thiel, F. (2010) The onion has cancer: some social network analysis visualizations of open source project communication. IN: Proceedings of the 3rd International Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development. Cape Town, South Africa, May 01 - 08. ACM.

O'Mahoney, S. and Ferraro, F. (2007) The emergence of governance in an open source community. Academy of Management Journal. vol. 50, no. 5, pp 1079-1106.

Oreg, S. and Nov, O. (2008) Exploring motivations for contributing to open source initiatives: The roles of contribution context and personal values. Comput.Hum.Behav. vol. 24, no. 5, pp 2055-2073.

Orr, J. (1990) Celebrating Identity: War Stories and Community Memory in a Service Culture. IN: Middleton, D.S. and Edwards, D. eds. Collective remembering: Memory in society. Newbury Park, CA: Sage Publications. pp 187-201.

Orr, J. E. (2006) Ten years of talking about machines. Organ.Stud. vol. 27, no. 12, pp 1805-1820.

Orr, J.E. (1996) Talking about machines: An ethnography of a modern job. Ithaca, United States: Cornell University Press.

OSI (2012) History of the Open Source Initiative [Online]. Available from: http://opensource.org/history [Accessed 11/26/2012].

Panjer, L.D. (2007) Predicting eclipse bug lifetimes. IN: Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on. Minneapolis, MN, 20-26 May. IEEE.

Polanyi, M. (1967) The tacit dimension. New York, NY: Anchor Books.

Popper, K.R. (1968) The open society. London, UK: Routledge.

Popper, M. and Lipshitz, R. (2000) Organizational learning mechanisms, culture, and feasibility. Management learning. vol. 31, no. 2, pp 181-196.

R Development Core Team. (2013) R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing.

Raymond, E. (1998) Homesteading the noosphere First Monday [Online]. vol. 3, no. 10, Available from: http://catb.org/esr/writings/homesteading/homesteading/ [Accessed 13/08/2009].

Raymond, E. S. (2003) How to become a hacker. Database Network J. vol. 33, no. 2, pp 8-9.

Raymond,E. S. (2004) Open Minds, Open Source [Online]. Available from: http://www.catb.org/~esr/writings/analog.html [Accessed 3/6/2009].

Raymond, E. S. (1999) The cathedral and the bazaar. Knowledge, Technology, and Policy. vol. 12, no. 3, pp 23-49.

Reason, P. (2003) Pragmatist Philosophy and Action Research Readings and Conversation with Richard Rorty. Action Research. vol. 1, no. 1, pp 103-123.

Resnick, L. B. (1987) The 1987 presidential address: Learning in school and out. Educational Researcher. pp 13-54.

Roberts, J. A., Hann, I. and Slaughter, S. A. (2006) Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the Apache projects. Management Science. vol. 52, no. 7, pp 984-999.

Roberts, J. (2006) Limits to communities of practice. Journal of management studies. vol. 43, no. 3, pp 623-639.

Robles, G., Scheider, H., Tretkowski, I. and Weber, N. (2001) Who is doing it? Research on Libre Software developers. Berlin, DE: Berlios.

Robson, C. (2002) Real world research: a resource for social scientists and practitioner-researchers. Blackwell Oxford, UK: Wiley.

Rogers, C.R., Dorfman, E., Gordon, T. and Hobbs, N. (1965) Client-centered therapy: Its current practice, implications, and theory. Boston, MA: Houghton Mifflin.

Rorty, R. (1982) Consequences of pragmatism: Essays, 1972-1980. Minneapolis, MN: University of Minnesota Press.

Ryan, R. M. and Deci, E. L. (2000) Intrinsic and extrinsic motivations: Classic definitions and new directions. Contemp.Educ.Psychol. vol. 25, no. 1, pp 54-67.

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). vol. 15, no. 2-3, pp 229-250.

Saldaña, J. (2009) The coding manual for qualitative researchers. New York, NY: Sage Publications.

Salmon, P. and Riessman, C.K. (2008) Looking back on narrative research: An exchange. IN: Andrews, M., Squire, C. and Tamboukou, M. eds. Doing Narratative Research. London, UK: Sage Publications. pp 78-85.

Schwartz, B., Ward, A., Monterosso, J. (2002) Maximizing versus satisficing: happiness is a matter of choice. J.Pers.Soc.Psychol. vol. 83, no. 5, pp 1178-1197.

Schwartz, S. J. (2001) The evolution of Eriksonian and, neo-Eriksonian identity theory and research: A review and integration. Identity: An International Journal of Theory and Research. vol. 1, no. 1, pp 7-58.

Sclater, N. (2008) LargeScale Open Source ELearning Systems at Open University UK. Educase [Online]. vol. 1, no. 12, Available from: http://ccblog.typepad.com/weblog/files/ERB0812.pdf [Accessed 03/03/2009].

Scott, P., Castaneda, L., Quick, K. and Linney, J. (2009) Synchronous symmetrical support: a naturalistic study of live online peer-to-peer learning via software videoconferencing. Interactive Learning Environments. vol. 17, no. 2, pp 119-134.

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

Shah, S. K. (2006) Motivation, governance, and the viability of hybrid forms in open source software development. Management Science. vol. 52, no. 7, pp 1000-1014.

Shirky,C. (2003) Power Laws, Weblogs, and Inequality [Online]. Available from: http://www.shirky.com/writings/powerlaw_weblog.html [Accessed 19/03/2009].

Simons, H. (1980) Towards a Science of the Singular: Essays about Case Study in Educational Research and Evaluation. Care occasional publications, Centre for Applied Research in Education Norwich. vol. 10.

Stake, R.E. (1999) The art of case study research. London, UK: Sage Publications.

Stake, R. E. (1978) The case study method in social inquiry. Educational Researcher. vol. 7, no. 2, pp 5-8.

Stamps, D. (2000) Learning Is Social. Training is Irrelevant. *IN* Lesser L., Fontaine M., A. Slushe J. A., eds. Knowledge and Communities. Woburn, MA: Butterworth-Heineman. pp 53-72.

Star, S. L. and Griesemer, J. R. (1989) Institutional ecology,translations' and boundary objects: Amateurs and professionals in Berkeley's Museum of Vertebrate Zoology, 1907-39. Soc.Stud.Sci. vol. 19, no. 3, pp 387-420.

Stevens, J. (2009).Applied multivariate statistics for the social sciences. New York, NY: Taylor & Francis.

Stewart, D. (2005) Social status in an open-source community. Am.Sociol.Rev. vol. 70, no. 5, pp 823-842.

Stewart, K. J. and Gosain, S. (2006) The impact of ideology on effectiveness in open source software development teams. Mis Quarterly. pp 291-314.

Strate, J. D. and Laplante, P. A. (2013) A Literature Review of Research in Software Defect Reporting. Reliability, IEEE Transactions. vol. 62, no. 2, pp 34-51.

Strauss, A. and Corbin, J. (1994) Grounded theory methodology: an overview. IN: Denzin, N.K. and Lincoln, Y.S. eds. Handbook of Qualitative Research. London, UK: Sage Publications. pp 273-285.

Torvalds, L. and Diamond, D. (2001) Why open source makes sense. Educause Review. vol. 36, pp 70-77.

Trinity School of Education (2009) Research Ethics in the School of Education [Online]. Available from: http://www.tcd.ie/Education/ethics/ [Accessed 03/05/2009].

Von Krogh, G. (1998) Care in Knowledge Creation. Calif.Manage.Rev. vol. 40, no. 3, pp 133-153.

Von Krogh, G. and Spaeth, S. (2007) The open source software phenomenon: Characteristics that promote research. The Journal of Strategic Information Systems. vol. 16, no. 3, pp 236-253.

Von Krogh, G., Spaeth, S. and Lakhani, K. R. (2003) Community, joining, and specialization in open source software innovation: a case study. Research Policy. vol. 32, no. 7, pp 1217-1241.

Von Krogh, G., Haefliger, S., Spaeth, S. and Wallin, M. W. (2012) Carrots and rainbows: Motivation and social practice in open source software development. MIS Quarterly. vol. 36, no. 2, pp 649-676.

Waterman, A. S. (1988) Identity status theory and Erikson's theory: Communalities and differences. Developmental Review. vol. 8, no. 2, pp 185-208.

Waterman, A. S. (2004) Finding someone to be: Studies on the role of intrinsic motivation in identity formation. Identity. vol. 4, no. 3, pp 209-228.

Waterman, A., Kurtines, W. and Gewirtz, J. (1995) Eudaimonic theory: Self-realization and the collective good. IN: Kurtines, W. and Gewirtz, J. eds. Moral development: An introduction. New York, NY: Allyn & Bacon. pp 255-278.

Weick, K.E. (1995) Sensemaking in organizations. New York, NY: Sage Publications.

Weiss, C., Premraj, R., Zimmermann, T. and Zeller, A. (2007) How long will it take to fix this bug? IN: Mining Software Repositories, 2007. ICSE Workshops MSR'07. Fourth International Workshop on. Minneapolis, MN, 20-26 May. IEEE.

Wells, A. S., Hirshberg, D., Lipton, M. and Oakes, J. (1995) Bounding the case within its context: A constructivist approach to studying detracking reform. Educational Researcher. vol. 24, no. 5, pp 18-24.

Wenger, E. (2000) Communities of practice and social learning systems. Organization. vol. 7, no. 2, pp 225-246.

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

Wenger, E. (1998) Communities of practice: Learning as a social system. Systems Thinker. vol. 9, no. 5, pp 2-3.

Wenger, E., McDermott, R.A. and Snyder, W. (2002) Cultivating communities of practice: A guide to managing knowledge. Cambridge, MA: Harvard Business Press.

West, J. and Lakhani, K. R. (2008) Getting clear about communities in open innovation. Industry and Innovation. vol. 15, no. 2, pp 223-231.

West, J. and O'Mahony, S. (2008) The role of participation architecture in growing sponsored open source communities. Industry and Innovation. vol. 15, no. 2, pp 145-168.

Whitworth, B. (1998) The web of system properties: a general view of systems. SIGCSE Bull. vol. 30, no. 4, pp 46-50.

Whyte, W.F. (1993) Street corner society. Chicago: Univ. of Chicago Press.

Williams, D., Caplan, S. and Xiong, L. (2007) Can you hear me now? The impact of voice in an online gaming community. Human Communication Research. vol. 33, no. 4, pp 427-449.

Williams van Rooij, S. (2011). Higher education sub-cultures and open source adoption. Computers & Education, vol. 57, no. 1, pp 1171–1183.

Xu, B., Jones, D. R. and Shao, B. (2009) Volunteers' involvement in online community based software development. Information & management. vol. 46, no. 3, pp 151-158.

Yin, R.K. (2009) Case study research: Design and methods. London, UK: Sage Publications.

Yin, R. K. (1981) The case study crisis: some answers. Adm.Sci.Q. vol. 26, no. 1, pp 58-65.

Yoder, A. E. (2000) Barriers to ego identity status formation: A contextual qualification of Marcia's identity status paradigm. J.Adolesc. vol. 23, no. 1, pp 95-106.

Zimmermann, T., Premraj, R., Bettenburg, N. (2010) What makes a good bug report? Software Engineering, IEEE Transactions on. vol. 36, no. 5, pp 618-643.

Appendix A Interview Schedule

Overview

The format of the interviews was semi-structured. Some of the questions were tailored to the participant. Some questions were skipped if the participant had already covered the topic in the answer to another question. There were also other impromptu questions asked of participants based on unanticipated but relevant themes that arose in the interviewee answers.

Interview Questions

[[CONSENT]]
I am going to record this interview and transcribe the recording. I am going to keep the interview audio and text transcription securely and destroy them after my research has finished. Anything you tell me I will treat as confidential. Anything I use in my thesis will be anonymised and aggregated with other data. I won't associate anything you say with any information that exists in the public domain such as to make you identifiable. If I publish anything from this research I will maintain your anonymity.

This is just so you feel as comfortable as possible and that you answer as freely as you would like to. Don't answer a question if you don't want to and feel free to end the interview at any time. I will email you a written copy of this consent and also a copy of the transcript of this interview to be sure you are happy with it. Do you have any questions or are you willing to consent to the interview?

[[PREAMBLE]]
I say this bit to everyone I interview for consistency so I'm just going to read it to you here if you don't mind. My name is Eamon Costello. I work in the Distance Education centre in Dublin City University and am studying towards a doctorate in Education. In my doctoral research I am interested in how educational software is built, taking Moodle as a case study, and specifically how people interact via the Moodle Bug Tracker to help build Moodle.

PROMPT*: More details about me if appropriate*

[[IDENTITIES AND INWARD TRAJECTOIRES]]

1. To start I'd like you to tell me a little bit about your background and how you came to be involved with Moodle

    ○ PROMPT: *your professional background?*
    ○ PROMPT: *your educational background?*

2. You are a contributor to the Moodle community. What do you see as the key skills and experience you that allow you to contribute?

3. Can you tell me what the level of your interaction with the Moodle Bug Tracker is?

    ○ PROMPT: e.g. *Were you a commenter, voter, reporter, or assignee?*

4. How many issues have you contributed to?
5. What issues are you interested in and why?
    ○ PROMPT: *Are you interested in a range of issues/features etc. or is there one main thing you are interested in?*

[[ISSUE RESOLUTION FACTORS AND PROCESSES]]

6. Can you describe what you think the typical lifecycle of an issue would be in the Moodle Tracker?
    ○ PROMPT: *In your own experience?*
7. What factors do you think might lead to the successful resolution of an issue?
    ○ PROMPT: *e.g. the issue gets fixed?*
8. How would you describe your influence on the resolution of issues in the Moodle Tracker?

    PROMPT: *How would you describe your influence on the initiation of a new feature/bug fix?*

[[ROLES]]

9. How would you describe the role of [[*group to which respondent is NOT a member i.e. assignee or non-assignee*]] in Moodle Bug tracker issues?
10. How did you find the experience of being involved in a Moodle tracker issue?
    ○ PROMPT: *What was your experience of the other participants?*
11. It seems to me that some issues close but not necessarily to the satisfaction of everyone involved. Why is this?
12. Some issues can stay open for a very long time. Why do you think this?

[[CLOSING QUESTION – COMMUNITY IDENTITY]]

13. What does Moodle mean to you?

[[OTHER]]

14. Have you anything else you would like to add or anything you wish to ask me?

Appendix B Interview Solicitation Email Template

The format of this email was tailored in some cases such as if I had met the prospective interviewee or someone had recommended them. Follow-up emails generally ensued – to gently prompt for a response or to try to firm up a tentative offer to participate. There were five non-responses and two people who declined. Three others tentatively offered to participate but their interviews never materialised.

Dear XXXX,

I have come across your name in the Moodle Bug tracker and know that you have been involved in the development of Moodle. The development of open source educational software is a big interest of mine. As part of research I am doing I would love to do a Skype interview some time and ask you a bit about your experiences of working on Moodle.

The interview would inform a doctorate with Trinity College Dublin that I am studying towards which looks at how Moodle, as an open source project, is built. Specifically, I am looking at the interface between Moodle core developers (those that can be assigned to bugs) and the wider Moodle community who contribute requests, fixes or vote on issues in the Moodle Bug Tracker.

In recognition of you giving your time, I will donate five euro to the charity Village Reach for the interview. Village Reach is an international charity carrying out a range of work in the developing world and has been independently reviewed and recommended (GiveWell.org).

Your interview will be anonymous and below is an ethics/privacy statement. I will email you a list of the questions in the interview beforehand. The interview would last about 30 minutes and then you are done. I can do any time of day when you might have a half hour to chat. Would you be willing to do this? I look forward to hearing from you either way.

Kind Regards,

Eamon

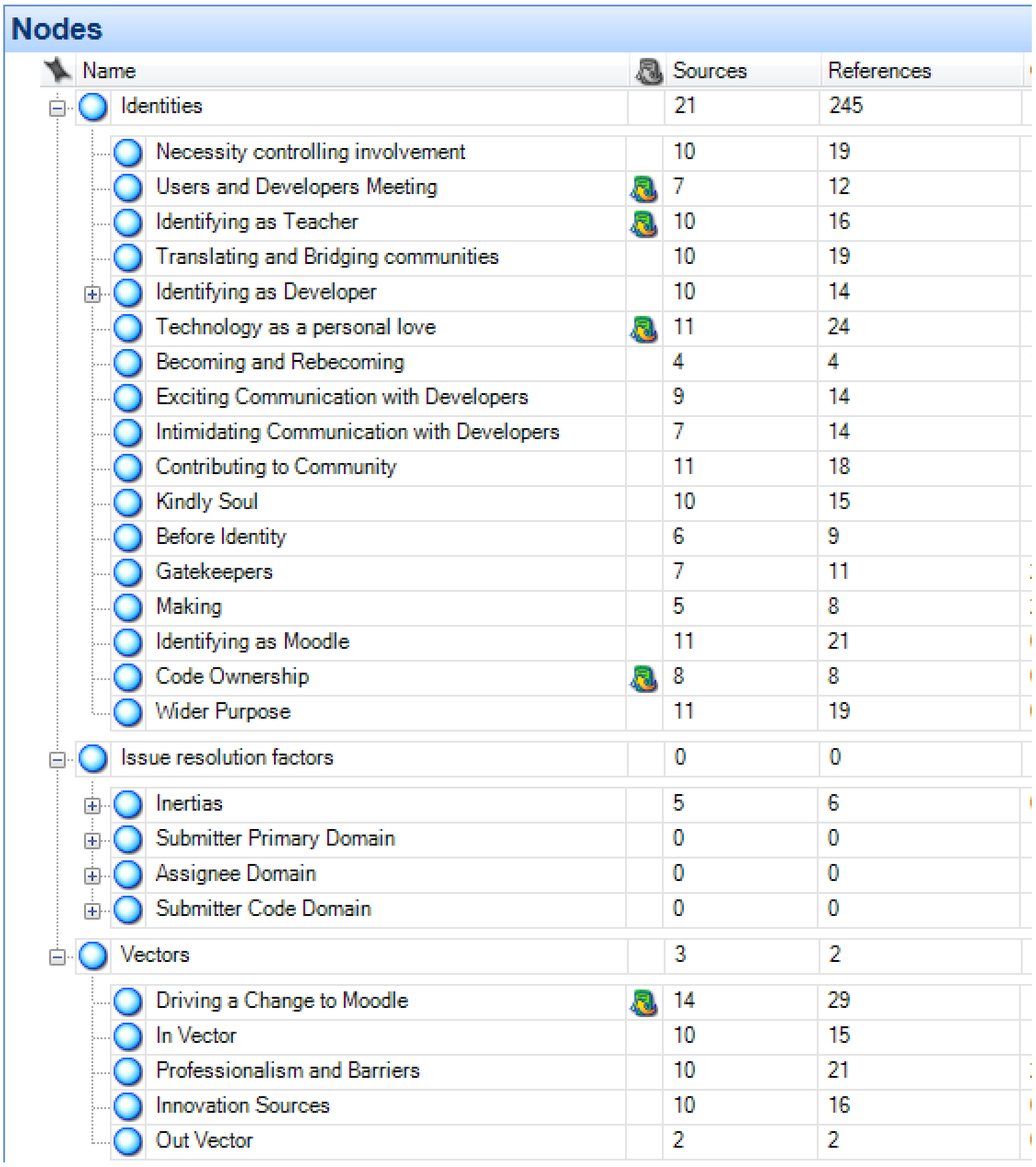
[[Ethics/Privacy Statement]]

I am going to record the interview and transcribe the recording. I am going to keep the interview audio and text transcription securely and destroy them after my research has finished. Anything you tell me I will treat as confidential. Anything I use in my thesis will be anonymised and aggregated with other data. I won’t associate anything you say with any information that exists in the public domain such as to make you identifiable. If I publish anything from this research I will maintain your anonymity.

This is just so you feel as comfortable as possible and that you answer as freely as you would like to. Don’t answer a question if you don’t want to and feel free to end the interview at any time. I will email you a copy of the transcript of the interview to be sure you are happy with it.

Appendix C Examples of Development of Coding Scheme

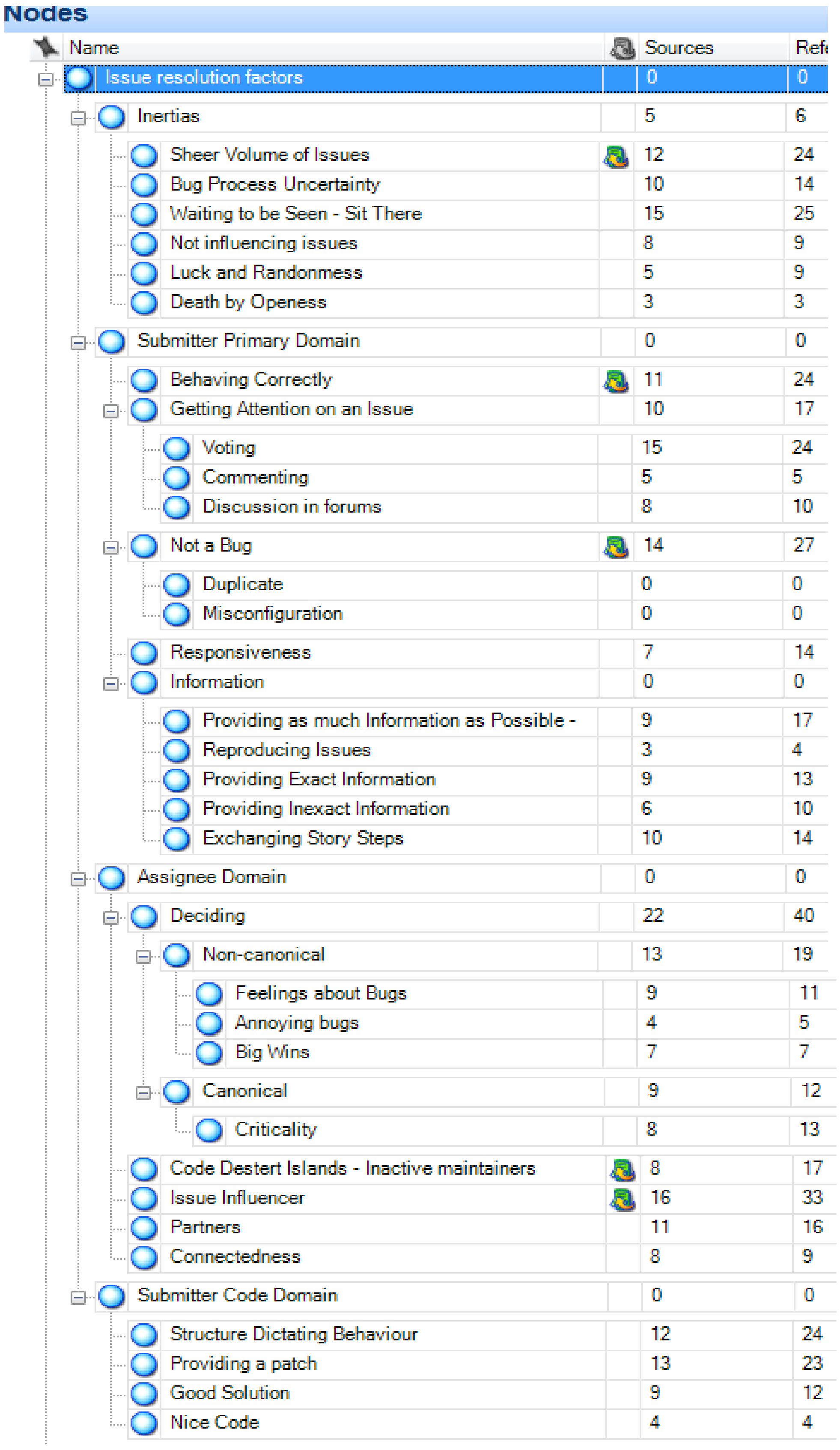
Snapshot of Top Level Coding:



Nodes

Name		Sources	References
Identities		21	245
Necessity controlling involvement		10	19
Users and Developers Meeting		7	12
Identifying as Teacher		10	16
Translating and Bridging communities		10	19
Identifying as Developer		10	14
Technology as a personal love		11	24
Becoming and Rebecoming		4	4
Exciting Communication with Developers		9	14
Intimidating Communication with Developers		7	14
Contributing to Community		11	18
Kindly Soul		10	15
Before Identity		6	9
Gatekeepers		7	11
Making		5	8
Identifying as Moodle		11	21
Code Ownership		8	8
Wider Purpose		11	19
Issue resolution factors		0	0
Inertias		5	6
Submitter Primary Domain		0	0
Assignee Domain		0	0
Submitter Code Domain		0	0
Vectors		3	2
Driving a Change to Moodle		14	29
In Vector		10	15
Professionalism and Barriers		10	21
Innovation Sources		10	16
Out Vector		2	2

Snapshot of Issue Resolution Success Factors Coding:



Nodes

Name	Sources	Refe
Issue resolution factors	0	0
Inertias	5	6
Sheer Volume of Issues	12	24
Bug Process Uncertainty	10	14
Waiting to be Seen - Sit There	15	25
Not influencing issues	8	9
Luck and Randonmess	5	9
Death by Openess	3	3
Submitter Primary Domain	0	0
Behaving Correctly	11	24
Getting Attention on an Issue	10	17
Voting	15	24
Commenting	5	5
Discussion in forums	8	10
Not a Bug	14	27
Duplicate	0	0
Misconfiguration	0	0
Responsiveness	7	14
Information	0	0
Providing as much Information as Possible -	9	17
Reproducing Issues	3	4
Providing Exact Information	9	13
Providing Inexact Information	6	10
Exchanging Story Steps	10	14
Assignee Domain	0	0
Deciding	22	40
Non-canonical	13	19
Feelings about Bugs	9	11
Annoying bugs	4	5
Big Wins	7	7
Canonical	9	12
Criticality	8	13
Code Desert Islands - Inactive maintainers	8	17
Issue Influencer	16	33
Partners	11	16
Connectedness	8	9
Submitter Code Domain	0	0
Structure Dictating Behaviour	12	24
Providing a patch	13	23
Good Solution	9	12
Nice Code	4	4

Appendix D Factors Believed Important to Issues Resolution

These factors were derived from the first coding cycle of the interview transcripts

Providing good information about the issue – including a clear description	**Providing steps** to reproduce the issue
Behaving correctly – being polite and not demanding	**Being responsive –** following up any queries quickly on a submitted issue
Discussing the issue in the Moodle.org forums	**Voting on an issue** (number of votes)
Commenting on an issue (number of comments)	**"Getting attention on an issue"** – by talking to someone about it, using email, jabber, social media etc.
Is a valid bug – not a misunderstanding of how Moodle works or a duplicate of another issue	**Is a bug fix –** a fix has a higher priority than a feature request (may also be a "bigger win" as below)
Criticality – issue is considered "critical" (such as security bug)	**"Big win" –** effort of developing the fix is low relative to its pay-off (see *Providing a usable patch* below)
Bug is "annoying" – deep feeling of developer that bug must be fixed	Issue supporter is a **Moodle Partner**
Luck	**Time –** as time goes on an issue is more likely to be fixed
Intervention of a "kindly soul" –	**Presence of a component**

a broker who helps the issue along (or submits it on behalf of someone else)	**maintainer –** someone with active ownership of the code
Connectedness – knowing the (right) person to communicate with	**Influence of the component maintainer –** maintainer has large say
Outlook of the maintainer – some maintainers more receptive than others	**Influence of Moodle HQ –** Moodle HQ has large say
Influence of Michael de Raadt – Moodle HQ Development manager	**Influence of Martin Dougiamas** – Martin has final say
Monetary investment – willing to pay a Moodle Partner to fix the issue	**Providing a patch –** writing code for the fix that should at least help explain the problem
Providing a usable patch – solves problem well and conforms to canonical programming style	**Providing a good solution –** solution that fixes a real problem in an acceptable way
Willingness to become component maintainer – Willing to take ownership of code after it has been submitted	**Willingness to test** a potential fix of the issue

Appendix E Sample Excerpts from Interview Transcripts

Interview Sample A

[Interview Start]

So we have, one of our main priorities at the moment is three week turnaround of feedback **okay** so anything that relates to that is really high priority for us so anything that makes that feedback faster and better so audio and [[inaudible]] download **okay** so that is really major for us at the moment.

Okay very interesting – did you say three week turnaround of feedback? Yeah **and do you have any kind of ehm I know some places in the UK seem to do some review ehm what do you call it … blind review or double marking or anything like that of your assignments or?**

Anonymous marking is something we want **anonymous marking is something you are looking at as well okay?** Yeah

Ehm I'll ask you we want to be able to, sorry – we want to be able to measure ehm the performance of academics as regards feedback as well, we have got no way of actually fining out if they are complying with that three week turnaround so that is another thing that we are really interested in.

Oh I see some kind of reporting mechanism or…

So you can run a report and see who is, who is late with their feedback or who hasn't… yeah **okay and is it both marking and feedback that you want to measure or is it just the marks just that the correct marks are there?**

No the quality of the feedback is really high priority as well because that is really high priority for our students **okay and how do you measure that?**

Ehm we are not at the moment it is just something that, we have another team that deals with that kind of thing. We've got what we call Learning Development **right** they will work out the process and the requirements and then come to me **I see okay** depending on what we are after **okay so you work closely with them?** I do yeah.

Our students are also, one of the highest priorities for students as well is comparison, they want to see how they rank in the class **okay so they want to get a picture of the distribution** yeah **okay.**

Ehm can you describe for me XXXXX what you think the typical lifecycle of an issue would be in the Moodle Tracker from your experience and it doesn't have to be ehm what … what is says it is in Moodle or what is the official version but what your, more or less what your impression of it is?

Ehm my impression of it is that, well so far in my experience they don't go very far ehm you are encouraged… when you immediately submit an issue, you get, I get a response straight away within 24 hours eh it is usually a standard response **uh hum ehm** quite often asking for a fix because I usually raise the issue before I fix it just in case somebody else is working on it **okay.** So anything that comes to my attention I will put it on there first, and then if I do fix it I'll add the fix **uh hu** and they encourage you [[inaudible]] but in my experience most of the things that I have raised are still open **uh hum** They are just sitting there and I get the feeling that ehm the priority has been set at least for the next year and they won't do that much. All the little things have been side-lined – they kind of already know what they want to do. **Okay** And there are so many things that are being majorly rewritten. I just feel that there is a lack of momentum at the moment

[[Interview continues]]

Interview Sample B

[Interview Start]

You've comment voted, reported on issues or have you ever been can you be assigned to an issue?

Eh yes and no **okay** let's say one of the big issues that I had in writing down things in Moodle Tracker **yeah** is that the core developers team **yeah** is very, very eh closed **yeah** let's say probably they have too much contributors **oh** so basically they need to filter the things **okay yeah** so you need to spend a big amount of time **yeah** to let them see you're your issues **yeah** find give value to your reasons **yeah** so basically **it takes** one of the things that you need to know when you get in core, get in touch with the core developer team **yeah** is that you need to spend a big amount of time **okay yeah** to take the reasons to give value **yeah** to your work because I'm let's say I'm a PHP developer **yeah** from several years **yeah, yeah** and I'm a senior developer say but I'm not as senior as probably as the people inside the core team you need to get a reputation. **Yes, yes absolutely yeah, yeah – you need to win their trust.**

Yeah, so basically what I found is that XXXX which is the contributor in eh the person in charge to maintain the XXXX module eh has found in me a valuable how can I say person to get in touch with **yeah** sometimes if he ask me through the Tracker to validate the eh the issue that someone open within the XXXX module - **He wants you to test that an issue is valid?** No, no not only if the issue is valid, if there are, if how can I say it eh… how can I explain it I have no quick example so let’s say eh, sometimes **yeah** people ask for features **okay** or claims for bugs **and he wants to know if it’s a real bug sorry?** It’s something related to, **yeah** I had a big reputation within the XXXX module **yeah** and let’s say I ehm I can send you just a reply I have sent today basically [sends link] **yeah great** just to let you understand that the eh working for an open source project is just a matter of time and reputation **okay** before also, also skills because reputation is made up on skills **skills yeah** whatever the skills are not also for developing **yeah** because contributing to an open source project means documentation, testing **yeah yeah** I’m a developer and I'm used to, to write functional specifications and so on **yeah yeah** so basically the contribution is made upon several parts so basically if you look at the Tracker you see also the QA part **yeah** of the tracker as well as the implementation part of the tracking issues in Moodle so if you look at these [sends me link] **yeah** eh… parts you will see that **yeah, okay in the…? Moodle forums okay** this person XXXX has **yeah** wrote a private message to me to ask me to eh.. some information about this issue regardless of what the amount they have done is telling to him. So they are **So this a PHP version issue not a not an issue per Moodle per say** exactly but if you look at the, at all of the thread, YYYY who is the main maintainer has already replied **yeah, yeah** but [he/she], XXXX has written to me a private message recalling to another thread **yeah** because he saw some of my replies in other threads **okay** and asked me to give an opinion and I privately reply to him **yeah** that I am used to making my replies in a public way **right yeah.** So basically every time someone is asking for an opinion on an issue I always ask to write it down in the community **exactly.**

No-one is paying anything for my time **yeah.** One of the strange things is that eh.. someone is maybe could make some money in eh helping people I can **yeah** guarantee that no-one has can give has … eh… oh shit has **he he** given me one euro for my work **okay yeah** and also in the past for [*OPEN SOURCE PROJECT X]* **yeah**, **yeah** the work in providing patches and hacking **yeah** but just for fun all for ehm…. Eh.. **the joy of doing it yeah** and also for eh… testing my skills **okay yeah build your skills yeah.** This kind of approach helped me in being really confident in my work **yeah yeah** and also in my life even if my life is not open source software **he, he excellent –** kind of eh a way of living. I.. to be confident **yeah** and the way I use to increase my confidence is to get in touch with the things and eh… try any kind of challenge.

Okay can I ask you another question yeah yeah **XXXXX which is** yeah Otherwise I will talk for many **good! Good! He he** basically drive this interview because I will **right, right eh.. I want to ask you roughly how many issues you would have contributed to in the Tracker and you talked about [*X Module]* and are there other issues in the Tracker you are interested in or is it just]*X Module]*?**
[Interview continues]