Comparing Software Development Models: Structural Problems in the Cathedral and Bazaar metaphors.

Raymond Elferink
RayCom
raymond@raycom.com

David Griffiths
Universitat Pompeu Fabra
david.Griffiths@upf.edu

Sara Zondergeld
RayCom
sara@raycom.com

Abstract

This paper builds on Levesque's identification of five issues holding back the Open Source Software (OSS) movement, and proposes that increased involvement by users in the development process can help address them. Metaphors have a valuable role in explaining to users what OSS is, and what their role involves. The dominant metaphors are the Cathedral and the Bazaar, proposed by Raymond. These metaphors have been valuable in identifying two approaches to software development. The metaphors also generate confusion, in part because of widely differing interpretations. A more structural reason for confusion is the use of vehicles which are not assigned to the same domain in the two metaphors. The negative consequences of this regarding comparison of approaches to software development are described. Adapted metaphors are proposed which facilitate comparison: the process which constitutes and maintains a Bazaar as an institution, and the process which constitutes and maintains a Shopping Mall.

1. Introduction

The authors of this paper are members of the SIGOSSEE Project Working Group on Organizational and Management Issues in Open Source Software (OSS). The project supports the use of both Free and Open Source Software in education, and is funded by the European Commission in the program 'Preparatory and innovative actions - eLearning Initiative'. Details of the project are available at *www.ossite.org*.

One of the issues on which we are principally focusing in our working group is the role of the end user in the software development process. It is now widely accepted that there is a role for users in the development process. We observe, however, that the majority of software development projects are conceived of as being carried out by developers only, with the end user simply acting as a consumer. Users are often involved, but principally as subjects in usability trials, and are still often seen as part of the external environment in which the software operates.

In OSS development projects, on the other hand, systems are often "custom built", with end users taking on a more complex role, acting as design consultants, providing information on requirements and feedback on the proposed solutions, or even becoming full co-designers and members of the development team. This paper is an initial step towards clarifying the role of the user in OSS development, and so to making this more effective. Our focus is on the metaphors used to describe the development process, and in particular Raymond's widely known Cathedral and Bazaar metaphors. This may not seem a central issue to those who use formal and controlled development processes, but we propose that in the OSS environment, where development is widely distributed and formal processes and control are weak, the metaphors which are used to describe the roles of the actors in the development process are of critical importance. We describe a number of problems with the dominant metaphors, examine their implications for the understanding of the role of users, and propose an alternative formulation.

2. Models of development in OSS and proprietary software

The main differences between the models of development used in OSS and proprietary software are in the areas of *user participation* and the degree of *formality*. Yamauchi [1] describes the Open Source development model as an informal development process. It is mainly collaborative, carried out by volunteers, and communication is asynchronous and decentralized, mainly based on e-mail, web fora and chat. These characteristics contrast with commercial development models. In that context the developers are employees, the development model is fixed and the communication is often centralised and synchronous, facilitated by meetings and informal chats in the office.

According to O'Reilly [2], integration of users as co-developers is a distinguishing characteristic of Open Source development models. He argues that by integrating the user into the development process, the software can be conceived of as a service, which evolves in response to the real needs of the users and not as a particular solution to a concrete problem.

3. Addressing constraints on the OSS model of development

Levesque [3], identifies five fundamental issues holding back the OSS movement, which lead to most of the public feeling that OSS is not accessible to them.

The five issues Levesque distinguishes are:

1. User interface design: the user interface of OSS projects is most of the time neglected, it is often not considered by the programmer as the `real` work;
2. Documentation: most of the time there is no documentation, there is no contractual responsibility. Mostly through bulletin boards or chat logs it is possible to get, often non-documented, answers to questions;
3. Feature-centric development: in many OSS projects, the fun stuff, the features, get more attention than the fundamental, mundane work;
4. Programming for the self: programmers tend to build their programs in a logic which is intuitive to themselves. As a result, OSS projects are mainly targeted at other programmers, although the programmers think they are open to everyone;
5. Religious blindness: OSS programmers are often blind to what proprietary software has to teach them. Every concept, even a good one, is automatically rejected.

We propose that at least three of these issues (1,2 and 4) could be addressed by involving users more closely in the development process. For this to be achieved, developers would have to welcome users into the development team and invest time to overcome the communication gap that clearly exists between the two parties. Similarly users would have to accept their responsibility and participate actively in the development team, working together with developers to identify their needs in terms of software functionality and to ensure usability by partaking in interface and workflow design.

This present paper is the first in a series of articles and web-based discussions in which we will research how far such user inclusion in software development can be achieved, and what interventions are necessary to facilitate and structure communication within this geographically dispersed, asynchronous and interdisciplinary cooperation.

In thinking about these issues, the first obstacle we came upon was the multiplicity of views on the nature of Open Source Software, and its counterpart, proprietary software. These are strongly influenced by the metaphors chosen to describe them, and in particular by Raymond's metaphor of the Cathedral and the Bazaar[4].

3. Why focus on the Cathedral and the Bazaar?

If OSS is to be more widely adopted, both in education and in other markets, users need to be helped to understand what it is, and how their role is different from that of the user of proprietary software. In introducing these unfamiliar concepts metaphors are very valuable.

In 1997 Eric S. Raymond wrote an essay for the Linux Kongress, entitled The Cathedral and the Bazaar, which later grew into a book. It has been enormously stimulating for many people who are thinking about the issues surrounding Open Source Software, including the authors of this paper. The metaphor which he introduced in the title refers to different ways of creating software, and it has shaped much of the discussion of Open Source Software production. Raymond's work has also been subject to extensive interpretation and criticism, not all of which has been equally constructive and well founded. Indeed it seems that the central metaphors have taken on a life of their own, leading to a range of mutually exclusive interpretations and opinions which were never envisioned by the author. Many alternative formulations have been offered, but few are analyzed in any depth. Because of this, it is worth examining the structure and implications of the metaphors more closely, and assessing degree to which they help elucidate the nature of the Open Source Software community. This task is undertaken by this paper.

4. How the metaphors were intended

In ´The Cathedral and the Bazaar´ Raymond describes two methods of Open Source Software development:

1. The traditional model of a finished product developed by a closed group of programmers with no beta released before its time;
2. The approach pioneered by Linus Torvald of widely delegated development of incremental

improvements to software which is released as early and often as possible ‘to the point of promiscuity´.

The metaphor introduced for the first method is the building of a cathedral ´carefully crafted by individual wizards or small bands of mages working in splendid isolation.’ [4]. The aspects which Raymond identifies as being significant for the metaphor are its awe inspiring scale and internal coherence.

The metaphor for the second method is the activities of a ´great babbling bazaar of differing agendas and approaches´ [4]. The aspects identified as significant for the metaphor are the many autonomous voices, views and goals which yet make up a single functioning entity.

These metaphors have proved to be very valuable in stimulating thought about the two approaches to software creation. Cathedral building is a good metaphor for a large software product produced by an effort coordinated by a single central planner, and a bazaar successfully suggests multiple actors working together with a high degree of autonomy. Despite this apparent clarity, there has nevertheless been a substantial degree of confusion as to what the metaphors actually mean.

5. How the metaphors are often used

People have used Raymond’s metaphors in ways which are quite different from what he intended and explicitly stated in his paper. In the first place while Raymond was discussing OSS development, the Cathedral approach is often identified with proprietary software, while the bazaar approach is associated with OSS. This holds true in many cases, but other aspects of reinterpretation are more confusing.

5.1 The Cathedral metaphor

In some cases the focus is placed on the cathedral as a building or institution, rather than on the process of building a cathedral, which Raymond specifically refers to.

This generates a number of associations which were not originally intended. Rather than representing a process, the Cathedral becomes associated with monolithic proprietary software applications. The cathedral as an institution also has strong value associations. For some it is an oppressive medieval institution which ensures its survival as a working system by imposing “one true view”, suggesting a parallel with dominant commercial software development companies. For others the cathedral represents all that is good in the world, suggesting security, peace and moral authority.

5.2 The Bazaar metaphor

The bazaar has also been understood in a number of ways. In the first place it is seen as Raymond intended, as a social structure and its interactions. In other cases it is referred to as a flexible and adaptable architectural structure. It can also be seen as an institution, with all its rules of membership, enabling structures, roles, etc. As a building or an institution the bazaar would not refer to the software creation process but rather the software itself. In conversation it tends to slip from one role to the other without the participants being clear about how they are using the metaphor.

6. Comparing the Cathedral and the Bazaar

Raymond’s original metaphors have been useful in stimulating and organizing thinking about Open Source Software, and have been widely adopted, so it may be counterproductive to propose completely new metaphors. The confusion surrounding Raymond’s metaphors is, however, extremely intense. A trawl of the Web for references to the Cathedral and the Bazaar turns up a myriad of interpretations of the Cathedral, including as Christianity with a capital C [5], as Microsoft [6], and as capitalism and Apple [7], while the bazaar has been associated with Islam [5] and Communism [8]. No doubt whatever metaphors we propose will also be subject to misinterpretation, but we believe that there are structural problems with the pair of metaphors chosen by Raymond which tend to exacerbate this inevitable misuse.

According to Lakoff “the essence of metaphor is understanding and experiencing one kind of thing in terms of another” [9] p.5. As most people are not used to thinking of different approaches to software design, and do not have a clearly defined framework for this domain, Raymond’s two metaphors help them approach the subject. Each of the metaphors is effective in itself and provides insight into the phenomenon which they refer to.

According to the Encyclopedia Britannica [10] a metaphor is composed of “the tenor referring to the concept, object, or person meant, and the vehicle being the image that carries the weight of the comparison.”. The metaphors which Raymond establishes have not only been used to elucidate the tenors (approaches to software design), but also to compare them. Thus the vehicles (activities of a bazaar and cathedral building) are compared, and from the observed distinctions people hope to find insights into the differences between the two tenors (approaches to software design). For this comparison to be productive the vehicles should, like the tenors, be elements from the same domain. If this is not the case then the differences between the domains confuse and overwhelm the insights which may gleaned from the comparison, and modify the interpretation of the individual metaphors.

As we describe above, Raymond is clear about the aspects of his vehicles which are significant for his two metaphors. The significant aspects of Cathedral building are its awe inspiring scale and internal coherence, while the significant aspects of the Bazaar are the many autonomous voices, views and goals which yet make up a single functioning entity. The common factor between these two vehicles is that they both have distinct types of social interaction, and to this extent they are comparable. However, the comparison breaks down if we try to compare other aspects of software development. For example different approaches to software development all work towards a comparable objective, i.e. an application. Raymond's two vehicles have very different objectives. Cathedral Building results in the production of a magnificent and complete edifice, while the Bazaar provides a social environment within which sales persons can carry out their work. Thus, while the two tenors of Raymond's metaphors are in the same domain (software production), the two vehicles as he defines them are clearly not. In order to make a productive comparison the reader needs to change the vehicles so that they belong to the same domain. In discussion the vehicles are usually interpreted as both belonging to one of two main alternative domains:

a) *Buildings and institutions*: The title "The Cathedral and the Bazaar" invites readers to use as vehicles the Cathedral and the Bazaar themselves, both belonging to the domains of buildings or institutions. This domain is then compared with the domain of software products. This goes against Raymond's explanation of the meaning of his metaphors, which is explicitly on processes.

b) *Construction processes*: the reader can focus on the building of the Cathedral as a process, and compare it with the process of creating a Bazaar (rather than the social aspects of the activity of a Bazaar indicated by Raymond). In this case the domain of construction processes is compared with the domain of software production processes.

Both these interpretations have provided useful insights, but because the choice of domain is not usually made explicit, it is often difficult to interpret the many discussions which use the Cathedral and the Bazaar as a starting point. Thus we conclude that when using Raymond's metaphors as a basis for comparison of software development processes it is important to specify to which domains the vehicles are being assigned.

7. Balancing the metaphors

Our experience of discussing software development using Raymond's metaphors suggests that no matter how aware conscientious one is about clarifying the domain of the vehicles, the structural imbalance means that confusions creep into the discourse. Consequently we believe it would be valuable to find vehicles which permit consistent comparison of both production processes and software, without the confusion caused by shifting domains. We therefore now embark, with some trepidation, on providing an alternative to Raymond's formulation, joining a motley crew of academics, programmers, and other denizens of the Web who have proposed alternatives and variations, such as planeteria [11], town councils [12], libraries [13], Baptist Church-Social Bazaar [5] and so forth. We do not suggest that our alternative is true, in the sense that it provides a better representation of reality. Our hope is that it will prove to be productive, rather than predictive, and that the use of corresponding domains for the vehicles will make it more effective than Raymond's original version in terms of generating stimulating ideas and parallels.

We find that the Bazaar provides a strong metaphor, because it captures well the openness and fragmentation of the Open Source community. Our misgivings about the metaphor are that the activity of a Bazaar does not have a product, and so does not belong to the domain of construction processes. Moreover, the principal activities of a bazaar are those of salesmen. This has serious implications, because it suggests an identification between these sales people and coders, who are then seen as the sole actors in the software development process, producing and maintaining software and providing it to users. This is a gross distortion, maintaining the splendid isolation ascribed to the cathedral builders. In software design the product is not the source code, but rather what the program does. Users are not directly interested in the elegance of the source code, but rather in the degree to which the application meets their needs, and the source code is no more and no less than the means of achieving this. Accordingly Open Source Software is built by many different actors taking different roles, including users, system administrators, developers, accountants, marketeers, etc. All these actors need good outcomes, which address their own needs and goals.

8. A revised pair of metaphors

We suggest that we revise the vehicle by focusing on the bazaar not as a production process, but rather as an institution. The ¨great babbling Bazaar of differing agendas and approaches¨ that Raymond mentions is then not the Bazaar itself, but the process which constitutes and maintains the Bazaar as an institution. This not only allows discussion of the Bazaar as a vehicle for a product, but invites discussion of the constitution and maintenance of the Bazaar as a vehicle for a production process. This provides a much richer metaphor for the activities of the open source software community,

This adaptation would give us a balanced pair of metaphors in the sense that we now have two products, and two implied production processes. But this does not necessarily mean that the metaphors work well at a detailed level. In order to maximize the opportunities for generating potential insights we need to maximize the richness of the potential comparisons. Consequently we require metaphors for software production in which both production methods create comparable products, with comparable properties and comparable functionality. The bazaar and the cathedral are quite different entities in function and characteristics, and so a comparison of their detailed structure does not generate a lot of insights into software production.

In order to facilitate productive comparisons we propose to use a Shopping Mall as a substitute for the Cathedral, because it has a comparable function and similar characteristics to a Bazaar while maintaining the associations of careful planning and grand scale of the Cathedral. In doing this we also avoid the counterproductive positive and negative associations of the Cathedral mentioned above. While many people may have a preference for shopping at a Bazaar or a Mall, the vast majority use both on one occasion or another, and recognize positive and negative aspects in both.

9. Conclusion

Raymond’s metaphors of the Cathedral and the Bazaar are so well known and firmly established that it is hard to carry out the task of our working group (analyzing and discussing the role of the user in OSS software development) without reference to them. While Raymond is clear about his interpretations, others who use them have been less so, and many alternative interpretations and metaphors have been generated.

In our analysis we point out that the Cathedral and the Bazaar is not one metaphor but two, and that each metaphor is itself composed of a tenor (the concept referred to, in our case software production), and a vehicle (carrying the weight of the comparison). The metaphors as defined by Raymond are valuable, although we note that the real world institution of the Cathedral generates strong positive and negative associations in different people, which tends to distort interpretation. Confusion is, however, generated when Raymond’s metaphors are used to seek insight into comparisons between approaches to software development models. This is because the two tenors identified by Raymond are from the same domain (software production methodologies), but the vehicles he defines are not (the scale and coherence of a building, and the voices and goals of social institution).

From this analysis we draw three conclusions:

a) much of the confusion surrounding interpretation of the metaphors is due to lack of clarity as to the domains to which the vehicles belong (e.g. is a cathedral referred to as an institution or a building project)

b) Raymond’s metaphors were intended to help identify approaches to software development. If they are extended beyond this function to facilitate *comparisons* between approaches, then it is important to specify to which domains the vehicles are being assigned.

c) In order to facilitate potential insights into the comparison of approaches to software development it is valuable to use metaphors in which the vehicles are comparable items from the same domain. Those which we have adopted are the process which constitutes and maintains a Bazaar as an institution, and the process which constitutes and maintains a Shopping Mall. With these adapted metaphors we seek to give form to more structured discussions, which can refer to both software development processes and products. We also encourage a more rational discourse, with less emotional baggage carried over from the vehicles chosen.

10. References

[1] Yutaka Yamauchi, et al. Collaboration with Lean Media: How Open-Source Software Succeeds. in Computer supported cooperative work. 2000. Philadelphia.

[2] Tim O'Reilly. Lessons from open-source software development. in Communications of the ACM. 1999.

[3] Michelle Levesque, Fundamental issues with open source software development. First Monday, 2004. 9[4).
[4] Eric S. Raymond, The Cathedral and the Bazaar. version 3 ed. 1997.
[5] Michael Turner, WHEN A BAZAAR IS NO LONGER A BAZAAR.
[6] Interview with Jim Ready from Montavista. 2004.
[7] Jeff Lewis, The Cathedral and the Bizarre, in www.macopinion.com. 2000. p. published as column under Mac Skeptic.
[8] Nikolai Bezroukov, A Second Look at the Cathedral and Bazaar. First Monday, 1999. 4(12).
[9] George Lakoff and Mark Johnson, Metaphors We Live By. 1980, Chicago and London: University of Chicago Press.
[10] Encyclopedia Britannica Premium Service, Tenor and Vehicle.
[11] April Corioso, The Cathedral, the Bazaar, and the Planetarium. 2002.
[12] Alan Cox, Cathedrals, Bazaars and the Town Council. 1998.
[13] Ken Krechmer. Cathedrals, Libraries and Bazaars. in the Association of Computing Machinery (ACM) Symposium on Applied Computing SAC March 10 - 13, 2002. 2002. Madrid.