ABSTRACT
A growing body of empirical research has examined large, successful open source software projects such as the Linux kernel, Apache web server, and Mozilla web browser. Do these results extend to small open source efforts involving a handful of developers? A study of the OpenEMR open source electronic medical record project was conducted, with the goal of understanding how requirements are elicited, documented, agreed, and validated in a small open source software project. The results show that the majority of features are asserted by developers, based on either their personal experience, or knowledge of users’ needs. Relatively few were requested directly by users. Validation and documentation took the form of informal discussions via the project’s developer mailing list. These results are consistent with an earlier study of the Firefox web browser, suggesting that there is a common open source requirements approach that is independent of project size.

Categories and Subject Descriptors
D.2.1 [Software Engineering]: Requirements/Specifications; J.3 [Life and Medical Sciences]: [Medical Information Systems]

Keywords
Open Source Software, Electronic Medical Record, Electronic Health Record, Software Requirements

1. INTRODUCTION
Open source software development remains an important subject of study for both researchers and practitioners. A growing body of empirical research has examined large, successful open source software projects such as the Linux kernel, Apache web server, and Mozilla web browser. In particular, evidence suggests that, rather than eliciting needs from users, documenting those needs in a specification, then analyzing and validating the specification, open source developers frequently assert the existence of requirements, sometimes by supplying a complete implementation [29] of a feature (or bug fix). Requirements are typically documented informally, and validation happens through discussion among the project’s developers, rather than with its users [34].

Do these results extend beyond the “banner” projects to small open source efforts involving a handful of developers?

This paper presents a case study of OpenEMR, an open source project developing electronic medical record (EMR) software. The traditional strength of open source software development has been Internet infrastructure and programmer tools [28]. OpenEMR is interesting because it is a successful project in a much different, highly specialized domain, with a narrowly-focused end-user population.

The study’s goal is to understand how requirements are elicited, documented, agreed, and validated in a small open source software project. The study follows the approach of an earlier study of the Firefox web browser [33].

The results show that, similar to Firefox, the majority of OpenEMR requirements are asserted by developers; a few were proposed by users; none were influenced by competing products. Validation of requirements is informal and typically involves an on-line discussion among developers. Documentation is likewise informal, consisting mainly of archived discussions. A surprising number of the contributors to the OpenEMR project are also end-users, in this case medical practitioners such as doctors or clinic administrators, who use the product in their practices. The implication for software development in general is that developers can be a significant source of innovation.

The remainder of this paper is organized as follows: the next section presents background and an explanation of the method employed in the study; next are the results, and a discussion of their implications; the last section presents conclusions and future directions.

2. METHOD
In a previous study, we examined requirements elicitation in the Firefox web browser project [33]. This study traced the origin and initial implementation of fourteen features comprising release 2.0 of Firefox.

The study showed that the majority of features (ten of fourteen) were asserted by developers, a small minority (three of fourteen) features were proposed directly by users, and one feature was influenced by a competing product. Of those features asserted by developers, four were deemed appropriate based on the developers’ own experience, three were justified by the developer’s knowledge of users’ needs, and three were asserted by implementations, through Firefox’s extension mechanism.

We applied the same method used in the Firefox study to a project...
in a different domain with a narrow population of end users. The project – OpenEMR – develops electronic medical record software for use by medical practitioners to keep track of their patients’ medical treatment. We would expect this population to have relatively few members who are also skilled programmers, when compared to users of open source products like Firefox or Apache.

We start with following research questions:
1. Who proposes requirements in OpenEMR?
2. How are requirements validated?
3. How are requirements agreed?
4. How are requirements documented?

The method comprised the following steps:
1. Identify the set of features delivered for OpenEMR release 2.8.0, up to and including release 2.9.0.
2. Select a subset of these features for examination.
3. Examine Internet resources related to OpenEMR, such as archives of discussion forums, the OpenEMR issue database, the OpenEMR “tracker” on Sourceforge.net, and other online forums, to discover when the feature was first proposed, and what role the person proposing the feature played (such as user or developer).
4. Determine the initial implementation of the feature (prototype by a developer, patch submitted to the tracker, or enhancement committed directly to the codebase).
5. Categorize the requirement as asserted by a developer, either from his or her personal experience or knowledge of user needs; proposed by an end-user, for example by posting a request to one of the discussion forums, or filing a bug report or “Request for Enhancement” in the issue database; or derived from features found in competing products.

The release notes for release 2.9.0 and the minor releases preceding it since 2.8.0 list over one hundred new features [19–22]. Items such as “Many other bug fixes and minor improvements” and “Performance optimizations” that represent minor maintenance activities were excluded from consideration. From the remaining features that represent new functionality, thirteen were selected for detailed investigation; these are listed in Table 1. Each of these was traced to its first mention in a project forum or archive.

The author was identified, either by signature in the case of discussion forum postings where the author included a signature, or user identifier when no signature was included. Using this information, the author’s role in the project was determined by cross-reference to other project artifacts such as notes posted by the same author to other forums, the project description [17], and the historical essays about OpenEMR [27]. From this information, the feature was classified according to the categories defined above.

The investigation of each feature was performed by the second author and checked by the first author. In the case of disagreement, differences were discussed and a final determination agreed.

3. RESULTS

The results (see Table 2) of this study confirm those of the Firefox study: the majority of features were asserted by developers, and the majority of those assertions were by core developers.

Six requirements were asserted based on developers’ experience and knowledge. An additional four were asserted from developers’ first-hand knowledge of user needs, for example when a client of a developer who is hired to install or enhance OpenEMR on a consulting basis expresses some need for his or her specific practice. The remaining three features were proposed directly by users.

3.1 Research Question Answers

Following are observations specific to the four research questions listed in Section 2.

3.1.1 Who proposes requirements for OpenEMR?

Nine distinct individuals proposed the thirteen features studied. Four are consultants whose business is installing and tailoring OpenEMR for practitioners. Three are system administrators, who oversee OpenEMR installations at clinics or private practices. One of these is a core developer who also has significant medical training and is currently studying to become a nurse. The remaining two are health practitioners who use OpenEMR in their practice. One is a medical doctor who enhances OpenEMR for his own practice, and contributes these enhancements to the project; the other is a tech-savvy chiropractor who is an enthusiastic OpenEMR user that contributes many issue reports to the developer forum.

3.1.2 What roles do contributors play?

Ten of the thirteen features (77%) were asserted by developers. Of these, six were asserted by a core developer, individuals who have commit access to the source code repository, and are active contributors of code; four of these were proposed by one individual, a consultant whose business is installing and tailoring OpenEMR for clients.

OpenEMR is unusual in that some of the most active contributors, in terms of both participation in discussion forums, and actual code contributed, are not programmers; at least two health care professionals are enthusiastic participants in the project, to the extent that they have hired programmers at their own expense to develop new features for OpenEMR. One feature was proposed (and two implemented) by one such “core user” – a medical doctor in the Netherlands who has hired programmers to tailor OpenEMR for medical practice in the Netherlands.

In all, eight of the thirteen features were proposed by either core developers or core users. Only two were proposed by active users who were not part of the core team.

3.1.3 How are proposed requirements agreed and validated?

In OpenEMR, requirements are agreed and “validated” through discussions among the developer community about the merits of a proposed feature. These discussions are usually brief and tend to
Table 2: Source and classification of OpenEMR Requirements

<table>
<thead>
<tr>
<th>#</th>
<th>Description</th>
<th>Requirement Source</th>
<th>Contributor Role</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Claims generation integrated into OpenEMR - no more FreeB tracker</td>
<td>tracker [23]</td>
<td>core developer</td>
<td>asserted, from experience</td>
</tr>
<tr>
<td>2</td>
<td>Active Directory and LDAP support</td>
<td>dev. forum [14]</td>
<td>developer</td>
<td>asserted, from experience</td>
</tr>
<tr>
<td>3</td>
<td>Patient pictures may be displayed in the demographics summary</td>
<td>dev. forum [3]</td>
<td>active user</td>
<td>contributed by user</td>
</tr>
<tr>
<td>4</td>
<td>New customizable printable super-bill</td>
<td>dev. forum [11]</td>
<td>developer</td>
<td>asserted, from knowledge of user needs</td>
</tr>
<tr>
<td>5</td>
<td>SMS Reminder feature</td>
<td>dev. forum [24]</td>
<td>active user</td>
<td>contributed by user</td>
</tr>
<tr>
<td>6</td>
<td>UTF8 encoding support</td>
<td>user. forum [25]</td>
<td>developer</td>
<td>contributed by user</td>
</tr>
<tr>
<td>7</td>
<td>Fixes for security vulnerabilities</td>
<td>dev. forum [10]</td>
<td>core developer</td>
<td>asserted, from experience</td>
</tr>
<tr>
<td>8</td>
<td>New optional frame-based workflow</td>
<td>dev. forum [9]</td>
<td>developer</td>
<td>asserted, from knowledge of user needs</td>
</tr>
<tr>
<td>9</td>
<td>Added PHP5 compatibility</td>
<td>dev. forum [4]</td>
<td>developer</td>
<td>asserted, from experience</td>
</tr>
<tr>
<td>10</td>
<td>Added experimental X12 835 remittance processing for primary claims</td>
<td>dev. forum [1]</td>
<td>core developer</td>
<td>asserted, from knowledge of user needs</td>
</tr>
<tr>
<td>11</td>
<td>Added second date to the patient encounter form for date of onset or hospital admission</td>
<td>dev. forum [1]</td>
<td>core developer</td>
<td>asserted, from knowledge of user needs</td>
</tr>
<tr>
<td>12</td>
<td>Support for deleting immunizations</td>
<td>dev. forum [5]</td>
<td>core developer</td>
<td>asserted, from experience</td>
</tr>
<tr>
<td>13</td>
<td>Option to use date of service as invoice date</td>
<td>dev. forum [8]</td>
<td>core developer</td>
<td>asserted, from experience</td>
</tr>
</tbody>
</table>

Sometimes, requirements are not validated at all; they get proposed, then an implementation appears sometime later. For example, the following statement appeared as part of an active user’s “wish list” posting to the development forum:

18. Integrated patient photograph on first screen that is pulled up on the patient. Flow= Staff uses a web-cam attached to their workstation to take a picture of the patient, which is then attached to their chart. The next person to open their chart sees the patients picture when they first open the chart. [3]

Two years later, a developer posted a note announcing the ability to include patient photographs.

This feature is also an example of how many requirements are agreed: someone proposes a new feature, and someone implements it later. If the implementation does not introduce problems to the rest of the code, it is incorporated into the product. Thus, agreement is obtained implicitly from a working implementation.

In contrast to Firefox, which uses the Bugzilla issue database as the primary discussion forum, the most common venue for announcing and discussing new features for OpenEMR is the OpenEMR developers forum on Sourceforge.net. Eleven features were proposed and discussed in this forum. Both developers and users contribute to the forum; while there is also a users forum, it is primarily a means for end-users to seek help in installing and configuring OpenEMR.

In this aspect, OpenEMR is similar to the Apache web server project, in which the developer mailing list is the primary vehicle for discussing enhancements [31].

3.1.4 How are agreed requirements documented?

OpenEMR has three types of requirements documentation: the OpenEMR roadmap, part of the OpenEMR Wiki [18]; tracker en-
tries, posted to the OpenEMR “Feature Requests” tracker on Sourceforge.net; and entries posted to the OpenEMR discussion forums on Sourceforge.net.

The OpenEMR roadmap was originally created by one of the core users to serve as a way to record features that needed funding to be implemented; it has since evolved toward a more comprehensive requirements document [15].

Postings to the developers discussion forum are the most common way of documenting requirements: as in the example above, a developer (or user) posts an idea for a new feature to the forum, occasionally with some evidence of its usefulness to end users. Other participants post comments, and the resulting collection forms the permanent public documentation of the requirement.

A good example is the entry in the developers forum describing Feature 4 (“New customizable printable super-bill”):

I want to create a template for a printable super bill customized for a practice.

This is a form that I will print out and hand the doctor.

Our old billing system does this.

This is for offices where the doc is not coding in directly. He is used to checking off on a printed superbill and we key in the data from that.

Im just not really sure how to to about designing something like that OR if there is already something like that in openemr.

It should have on it the patient’s name and some demographics and notes at the top.

Then section 2 will have our most common procedures printed out with boxes next to them, so the doctor can check off what applies.

Then section 3 will have our most common 50 or so diagnoses that the doctor can circle.

Optionally a section 4 would have payment boxes on it to record patient payment.

How do I create such a form?

First I think I would have to add a new table or tables for

1. Most commonly used procedures. I want it categorized by Office Visits, Consults, Surgical Procedures, Other, etc...
2. Most commonly used diagnoses.
3. I want the ability to go and change this form adding or deleting codes [11].

This posting, from a developer, documents the feature and provides some justification for the requirement.

Another developer proposed a solution:

Hi,

We do a similar thing with superbill in our office. Basically what I did was to scan our old superbill to an image and wrote a small perl utility which grabs details of tomorrow’s appointments from the openemr database and overlays on the scanned image. I run it as a cron job at noon everyday to print for the staff to pull charts and prepare. We are still in paper chart mode.

If you are interested please email me at [email address removed] [12]

A third developer elaborated on the proposed solution:

You could add a link to the existing superbill page which would open a new browser window/tab with a printable version that meets your criteria. This way, you could leverage existing code and probably not have to add a table. I am thinking of something similar to printable links elsewhere in the program, like in reports and patient report [13].

3.1.5 Summary

It is interesting to compare these observations with those of the earlier Firefox study.

In both the Firefox and OpenEMR projects, the majority of requirements are asserted by developers. Of a combined total of twenty-seven features studied from both projects, twenty (74%) were asserted by developers, six (22%) were contributed by users, one was influenced directly by a competing product. Table 3 compares these observations.

OpenEMR does not have an extension mechanism like Firefox does, so none of the OpenEMR features examined were asserted by implementation as was the case with two of Firefox’s features.

Both Firefox and OpenEMR eschew formal requirements validation in favor of discussion of the merits of an idea among the developer community; only one instance of formal validation - a usability study of Firefox’s tabbed browsing feature [16] - was observed in either case.

The Firefox project employs more formal documentation for requirements: the majority of proposed enhancements are documented in the Bugzilla issue database, and each release has a Product Requirements Document, that summarizes the functionality of each feature, and refers to the relevant Bugzilla entry if it exists [32]. In contrast, OpenEMR relies primarily on informal postings to the developers discussion forum to document requirements. In summary, we observed the following:

- Developers are the source of the majority of requirements for OpenEMR;
- More than half of the requirements were proposed by participants outside the core team.

This study confirms an important aspect of the open source approach, that seems to be independent of the project’s domain: user participation as contributors to the project appears to be a necessary facet of open source development [28, 31]. It also confirms the hypotheses that open source projects have a small core of developers, who are also users of the software [30].

3.2 Limitations
There are some limitations to this study that might affect the validity of the results. First, we assume that features delivered in a given release represent actual agreed requirements. This has two implications:
1. Only features actually completed by the release deadline are considered; any features agreed to be requirements but still under development were excluded from the study.
2. Requirements asserted by developers, or proposed by users, that were rejected by consensus of the community were not considered.

The first issue is mitigated by the observation that some of the features delivered in a given release were likely allocated to an earlier release, but were not completed in time. Thus, the features considered for this study represent a mix of requirements planned for the release, and features delayed from a previous release.

Our assumption is that features delivered are functionality that the project views as a requirements, whether or not they represent the true needs of stakeholders. So this study is an attempt to characterise the process of requirements elicitation, but not necessarily its accuracy in identifying true requirements. However, the fact that OpenEMR has continued to evolve and grow its user base indicates that the project is delivering features that users need, so the process seems to produce valid requirements in most cases.

Second, the features studied were selected after an initial search of OpenEMR archives revealed there was likely enough evidence to determine the origin of the feature. As such, there could be some bias in the set of features studied toward requirements that received public discussion. We have no evidence to suggest that the features studied are not representative of the whole, but neither have we attempted to verify this fact.

Third, we did not contact any of the project’s participants to verify our assessment of the artifacts. There is evidence that some communication among the developers occurs via private email; thus it is possible that some discussion of requirements occurred before the artifact we used as evidence of initial proposal was created.

Finally, the observations presented above apply to a small project in a narrow domain; while they are entirely consistent with earlier studies of larger projects in other domains, and thus contribute to a growing body of evidence regarding open source requirements processes, we cannot claim that these results represent open source software projects in general.

4. CONCLUSIONS

In general, participants in open source projects show a strong sense of engagement in, and ownership of, the project, as the following excerpt from a posting by one of the OpenEMR core users illustrates:

Linux has benefited me a great deal and I want to give something back to the Linux community.

... But neither do I want my work to only enrich the pockets of a for-profit business.

... I have donated servers and colocation space to the OpenEMR community. I try to support new users to the best of my ability, because I remember how lost I was when I started. I have installed OpenEMR a lot and feel helping others get started is something I can do

Commercial firms engaged in global software development are challenged with forming a unified effort from distributed, culturally distinct teams. The open source approach to requirements elicitation places a great deal of power, and responsibility, to shape the final product in the hands of programmers; could this sense of empowerment contribute to the enthusiasm and sense of purpose they have for the project?

In domains where innovation is key to remaining competitive, developers may be an under-appreciated source of new ideas. Firms like 3-M, Genentech, and Google have recognized this potential, giving developers significant time (15-20% of the work week) to pursue personal projects of their own choosing. The open source approach, in which developers contribute not only code, but ideas, may represent another approach to innovation by “inventing requirements [26].”

5. ACKNOWLEDGMENTS

This study described herein was performed while the first author was a visiting professor, and the second author was a student, at California Lutheran University. We would especially like to thank the students of the CS582 Open Source Software Development seminar during the winter term at CLU, who provided the stimulus to conduct this study.

We would also like to thank Sarah Beecham at Lero for her insightful comments.

References


