Chapter 13

Strategies for Improving Open Source Software Usability: An Exploratory Learning Framework and a Web–based Inspection Tool

Luyin Zhao
Rutgers University, USA

Fadi P. Deek
New Jersey Institute of Technology, USA

James A. McHugh
New Jersey Institute of Technology, USA

ABSTRACT

The Open Source Software (OSS) movement has had enormous impact on how software is created and continues to attract interest from researchers, software developers and users. A factor that may be inhibiting OSS from achieving greater success is usability, a fundamental characteristic to user acceptance of software. Motivated by the uniquely user-driven nature of the open source model and the extensive user base that participates in OSS projects, the authors propose an exploratory learning method and an associated web-based inspection environment that enables non-experts to contribute to open source usability inspection. This tool uses pattern-based usability guidelines to help identify usability knowledge during inspection. The method emphasizes outlining and exploration features which the authors have formally evaluated and the results of which are described. Data collected from a qualitative study indicates positive impact of the proposed method in helping end-users inspect software and achieve better results in discovering usability problems.

DOI: 10.4018/978-1-60960-513-1.ch013
INTRODUCTION

The Open Source Software development model is distinguished from traditional software development models by its distributed, collaborative developer participation, its use of highly public distributed peer review, the common use of frequent development releases, and the use of open licensing arrangements (principally via the General Public License). Open invitation to developers is a cornerstone of the open source paradigm, and the distinction between the developers and users of software may be fluid. While numerous studies have examined the OSS model from a development point of view, there has been much less concern with open source usability and how users can effectively improve software usability.

It is questionable as to whether OSS projects devote adequate resources to usability improvement, and whether there is a positive correlation between the OSS development model and better software usability (Nichols & Twidale, 2003) (Twidale & Nichols, 2004). Widely recognized motivations for participating in OSS development include personal satisfaction or technical needs, increasing one’s technical skills, as well as a desire to enhance one’s technical reputation in the broad open source community. These motivations and the culture of the community have potential implication for usability. For example, it is often said that “when programmers produce open source software, since they are largely “scratching their own itch”, they tend to produce software for themselves and to be content with a (programmer-oriented) user interface” (Pemberton, 2004). Furthermore, since algorithmic or functional problems have greater value in the developer ‘reputation market’ than contributions to usability (Nichols et al., 2002), it is more difficult for usability proponents to establish and promote the legitimacy of their focus vis-à-vis the functionality perspective of expert-users (Twidale & Nichols, 2004).

User involvement at the level of identifying software bugs or suggesting new features has been a major factor in OSS success. However, this kind of participation has been less effective in recognizing “soft” interests like usability in applications programs, and even less so at accommodating the kind of user diversity which is the objective of fine usability engineering as described by (Shneiderman, 1998). On the other hand there is a kind of inherited deep usability knowledge that is captured by the OSS model precisely because the software is often designed based on a “following the tail-lights” approach. The idea of this OSS strategy is to imitate successful existing systems’ functionality and their external shells. Assuming these systems have sound usability characteristics, then the imitative system should acquire many of these characteristics by default. Despite this, Messerschmitt (2004) cogently argues that software (usability) defects expose a fundamental weakness in the open source paradigm itself: an inability to adequately grasp the needs of application users. He argues that for most application users, critical usability information comes from marketers, salespeople, customer service, testers, and program managers who play a crucial role in surfacing, clarifying, and understanding the real needs of end users. These users are typically unable to participate in OSS development. Consequently their perspectives may be marginalized with the result that most OSS projects lack crucial direct usability information from these users.

The empirical study by Zhao and Deek (2005) contrasted the usability problems identified using a formal usability test method (protocol analysis) versus those submitted through voluntarily user participation. The study indicated that:

- Only a small portion of the usability problems found by formal usability testing were reported by users contributing through the regular OSS project bug reporting channel and subsequently addressed by the project developers. In other words, the existing
Related Content

Social Acceptability of Open Source Software by Example of the Ubuntu Operating System
www.igi-global.com/chapter/social-acceptability-of-open-source-software-by-example-of-the-ubuntu-operating-system/120935?camid=4v1a

Motivation of Open Source Developers: Do License Type and Status Hierarchy Matter?
www.igi-global.com/article/motivation-open-source-developers/41949?camid=4v1a

Analyzing OSS Project Health with Heterogeneous Data Sources
www.igi-global.com/article/analyzing-oss-project-health-heterogeneous/68151?camid=4v1a

A Perspective on Software Engineering Education with Open Source Software
www.igi-global.com/article/a-perspective-on-software-engineering-education-with-open-source-software/101203?camid=4v1a