

Comparing API Design Choices with Usability Studies: A Case Study and Future Directions

Jeffrey Stylos¹, Steven Clarke² and Brad Myers¹

¹ Carnegie Mellon University,
5000 Forbes Ave, Pittsburgh, PA, USA
{ jsstylos, bam }@cs.cmu.edu

² Microsoft,
1 Microsoft Way, Redmond, WA, USA
stevencl@microsoft.com

Abstract. There are more APIs than ever, and designing APIs that are usable by their target audience is difficult. Work at Microsoft has demonstrated that running controlled usability studies with participants from different personas and analyzing the results of these studies using the cognitive dimensions framework is effective at identifying and preventing usability problems in APIs. This paper presents a generalization of that technique in the form of usability studies of common API design choices. By studying a single design choice from multiple perspectives, we can generalize our results to any API that might be affected by the design choice. We show the feasibility of our approach with an initial study on whether or not to require constructor parameters, and present our current and planned work to study more API design choices.

1 Introduction

An important and challenging programming activity is using application programming interfaces (APIs), frameworks, toolkits and libraries. Programmers implementing new functionality need to figure out which APIs to use and how to combine them [10]. Programmers reading or modifying code have to understand how existing code that calls APIs works, what assumptions the code makes, and how to change or add to the code without breaking these assumptions. These tasks are increasingly challenging with the growing number and size of APIs; there are many options of which frameworks to use and current frameworks have tens of thousands of classes and methods.

One way to help this problem is to design APIs that are more usable by their target audience. Well designed APIs can be more quickly and effectively used and can lead to fewer programming bugs. However, the usability of an API can be difficult for an API designer to predict ahead of time, and difficult to fix after releasing an API, because of compatibility concerns.

Running a usability study of a specific API before it is released, using programmers representative of the target audience and the most common programming tasks that the API is intended to support, is an effective technique for detecting non-obvious usability issues with APIs and improving the usability of the released APIs [4]. However,

