**Assessing Programming Language Impact on**

**Development and Maintenance: A Study on C and C++**

(Bhattacharya et. al, 2011) proposed a novel methodology which controls for development process and developer competence and quantifies how the choice of programming language impacts software quality and developer productivity. They conduct a study and statistical analysis on a set of long-lived, widely used, open source projects such as, Firefox, Blender, VLC, and MySQL. The key novelties of the study are: (1) the only consideration of the projects which have considerable portions of development is in two languages, C and C++, and (2) a majority of developers in these projects contribute to both C and C++ code bases. Their methodology lays a solid foundation for future studies on comparative advantages of particular programming languages. They used several criteria for selecting their test applications. First, since they are interested in long-term software evolution and pursue statistically significant results, the applications had to have a long release history. Second, applications had to be sizable, so they can understand the issues that appear in the evolution of realistic, production-quality software. Third, the applications had to be actively maintained by a large number of developers. Fourth, the applications had to be used by a wide number of users who report bugs and submit patches. Their research hypothesis includes the following: (1) C++ is replacing C as a main development language, (2) C++ code is of higher internal quality than C code, (3) C++ code is less prone to bugs than C code and (4) C++ code requires less effort to maintain than C code. They test these hypotheses on large data sets to ensure statistically significant results. Their conclusion is that their analyses demonstrate that applications that start with C as the primary language are shifting their code base to C++, and that C++ code is less complex, less prone to errors and requires less effort to maintain. Below are the graphs of each of their hypotheses depending on the metrics they used to obtain results:

**Code Distribution**

Hypothesis (H1A): C++ is replacing C as a main development language.

Metrics: To test this hypothesis, they study how the percentages of C and C++ code change as an application evolves. They measure the eLOC (lines that are not comments, blanks or standalone braces or parentheses) of C and C++ code for each version using the Resource Standard Metrics (RSM) tool. If the hypothesis holds, they should find that C++ percentages increase over an application’s lifetime.