

A Classification Analysis of the Success of Open Source Health Information Technology Projects

Evangelos Katsamakas, Fordham University, USA

Balaji Janamanchi, Texas A&M International University, USA

Wullianallur Raghupathi, Fordham University, USA

Wei Gao, Fordham University, USA

Forthcoming: International Journal of Healthcare Information Systems
and Informatics, 2009, 4(4)

ABSTRACT

As the number of open source software (OSS) projects in health care grows rapidly, researchers are faced with the challenge of understanding and explaining the success of the open source phenomenon. This paper proposes a research framework that examines the roles of project sponsorship, license type, development status and technological complements in the success of open source health information technology (HIT) projects and it develops a systematic method for classifying projects based on their success potential. Drawing from economic theory, a novel proposition in our framework suggests that higher project-license restrictiveness will increase OSS adoption, because organizations will be more confident that the OSS project will remain open source in the future.

We apply the framework to a sample of open source software projects in health care. Our findings suggest that although project sponsorship and license restrictiveness influence project metrics they are not significant predictors of project success categorization. On the other hand, development status, operating system and programming language are significant predictors of an OSS project's success categorization. We discuss research and application implications and suggest future research directions.

Keywords: *health care, health information technology (HIT), license restrictiveness, open source software (OSS), project sponsorship,*

INTRODUCTION

“Rapidly rising health care costs and an epidemic of inferior health care quality over the past decade” (Brailer, 2005) call for an urgent and aggressive adoption of health information technology (HIT). HIT has the potential to transform the health care industry by increasing productivity, reducing errors and costs, facilitating information sharing and improving the

quality of healthcare services (Brailer, 2005), effectively transforming the healthcare system. Yet, adoption of HIT has been slow and appears to lag the effective application of IT and related transformations seen in other industries (Goulde et al , 2006).

With the renewed urgency to adopt HIT, open source approaches are gaining attention (Goulde et al, 2006, Kantor et al, 2003, McDonald et al, 2003, Raghupathi & Gao, 2007). For example, under development in Europe is the open source project Care2X, an application with four components: hospital information system, practice management, a central data server and a health exchange protocol. The software is distributed under the GPL license. Another initiative, OpenEHR, funded primarily by the U.S. Department of Health and Human Services, is an open source application that will support health record exchange and access control services in rural Mendocino County, California. These and other similar initiatives have the potential to create low cost tools for physicians.

On a larger scale, government agencies (the predominant payers of health care bills) are looking for open source to meet their primary objectives of lowering costs and enabling connectivity. Canada Health InfoWay, funded by federal and provincial grants, started an open source initiative in 2005 to develop software that hospitals and HIT developers could use to ensure the reliable exchange of patient health records among various entities. The U.S. government already has placed its VistaA integrated hospital software package in the public domain to provide adopters with open source software (Goulde et al, 2006).

The most significant open source health care application is OpenVista, the open source version of Vista, developed and used by all medical centers of the U.S. department of Veterans Affairs. The Vista software and its EMR module can be purchased for \$25.00 or less¹, are open source by virtue of the Freedom of Information Act, and are being actively marketed by new vendors. Other open source applications include TORCH, a web-enabled EHR application

¹ Data as of Oct 2007.

believed to be usable in single practitioner offices and scalable to multi-site practices. Written in an interpreted language, TORCH is therefore operating system independent. Another clinical medical records type application is tkFP, which was implemented using a number of languages including C, C++, Python and Perl. OSCAR, an application from McMaster University, Canada, comprises several modules including an electronic patient record system, billing, referrals and secure messaging. The system requirements include Linux, Java2 SDK, MySQL and Jakarta Tomcat. GnuMED is yet another EMR built using a cross platform WxPython GUI and the Postgres relational database. FreeMed, on the other hand, uses the popular LAMP (Linux, Apache, MySQL and PHP) platform, to provide web browser-based interface.

These advances suggest that the open source development approach is a viable means to developing HIT applications. Considering these activities, OSS, itself a transformative force in the software industry, may have a significant role in this hoped-for HIT revolution, potentially affecting the development and adoption of HIT and the strategic positioning of HIT vendors. For example, a recent joint venture of Red Hat Inc. and McKesson Corp. is aiming to push IT further into U.S. health care through open source software (Babcock et al, 2007), thereby intensifying competition between Linux and Microsoft Windows (Economides and Katsamakos, 2006).

And yet while several applications have been reported in the literature—primarily in the bio informatics field (see (Raghupathi & Gao, 2007) for a comprehensive review of OSS in healthcare)—hardly any rigorous studies exist to advance the understanding of OSS development in health care. For example, we do not have sufficient insight into the current level and speed of development of OSS in different types of healthcare organizations, and the factors that influence development and adoption.

Therefore, it is important to identify the characteristics and factors that influence software development and adoption in HIT, explain the forces behind them (e.g. sponsorship, licensing, technologies used), evaluate the effect of potential policies, and suggest the targets of such

policies. To that end, this paper is the first rigorous quantitative study based on objective data. A detailed analysis of open source development is one of the most overlooked aspects of HIT literature. Several HIT applications, including electronic medical record systems, are listed on the SourceForge web site, a good starting point for a comprehensive study of OSS in health care. The rest of the paper is organized as follows: section 2 describes the research framework for our study, drawing on work in the OSS and information systems fields. Section 3 describes the methodology and section 4 discusses the results. Finally, section 5 discusses the scope, limitations, conclusions and future research directions of our study.

RESEARCH FRAMEWORK

The primary objective of this research is to classify open source HIT projects into distinct groups based on their success and to explore the antecedents of those groups. Prior research in information systems proposed project metrics and identified antecedents of project success (Crowston et al, 2006, Crowston et al, 2006, Stewart et al, 2006), but there is limited research on classification frameworks that would provide more insight into open source projects (English et al, 2007). For instance, Crowston & Howison (2006) discuss the need to explore the community of developers, leaders, and active users behind OSS to make decisions regarding software viability and suitability for user needs. They suggest looking at sponsorship (as a measure of success). In addition, understanding a project's life cycle and its developers' motivations is a critical basis for the open source community's impact on a project's success. Crowston et al. (2007) provide empirical evidence regarding the management practices of OSS teams. Specifically, the authors identify how OSS teams organize their work (focusing in particular on practices for assigning work), how these practices differ from those of conventional software development groups and thus suggest what others might learn from OSS communities.

Crowston et al. (2003, 2006) identified measures that could be applied to calculate the success of OSS projects based on a brief review of the literature, a consideration of the OSS

development process, and an analysis of the opinions of OSS developers. They suggest that the development of success measures for OSS is important for two reasons. One, such measures would be useful to OSS project managers in assessing their projects. In many cases, third parties sponsor OSS projects so measures would help sponsors estimate a return on their investment. Two, OSS is an increasingly visible and copied method of systems development.

Drawing on prior literature in OSS and information systems we identify the relevant metrics of project success in HIT such as downloads and activity (Crowston et al, 2006, Stewart et al, 2006) and combine them with two extended metrics of success namely, the project rank and participation to create project clusters. Upon the creation of clusters, we explore the antecedents of such clusters. In addition to the effects of project sponsorship and license type, our framework focuses on the effect of such explanatory variables as development status, intended audience and technological factors (database environment, operating system, and programming language) on project success.

Our research framework is shown in **Figure 1**. The primary independent variables are *project sponsorship* and *license type*. Other variables considered in the framework are development status and technological variables such as programming language, operating system and database environment. The primary dependent variables are the success measures namely, *activity, downloads, rank, and participants*. Clusters of projects are derived using cluster analysis based on these dependent variables. Once the clusters are identified, we also examine *project sponsorship, license type*, and the other independent variables as antecedents of these clusters. Below we describe the main variables and provide the theoretical justification for the research hypotheses. Our choice of variables is consistent with previous studies on open source project success (e.g. Lerner et al, 2005, Stewart et al, 2006). More detailed descriptions can also be found on the Sourceforge website.

=====

Figure 1 about here

=====

Dependent variables

Project rank: As per the SourceForge website, project rank measures the rank of a project within SourceForge database. The measure captures information about traffic, communication and development of each project.² Traffic reflects downloads and visits to project page. Development reflects commits to CVS repository and age of last release. Communication reflects tracker, mailing list and discussion forum activity.

Downloads: This metric measures downloads' of a project's code from the project's page, as reported on SourceForge web site. Downloads reflect the popularity of a project to users and is also a proxy of use. Note that downloads are also captured in the *rank* dependent variable.

Communication and development activity: It refers to communication activity (tracker, mailing list, and discussion forum activity) and development activity (commits to CVS repository). Note that these are also captured in the *rank* dependent variable. Typically all projects are ranked on a percentile basis; the higher the percentile the greater the activity.

Participants: As reported on SourceForge and in the context of our research framework, the participants metric refers to developers that participate in the project, not the end users. Since we are indirectly capturing the user participation from downloads metric and activity on the project website, focusing on developers' involvement is considered more important here.

Stewart et al. (2006) comment that project success in the context of OSS projects is a concept that varies in meaning across projects and stakeholders. Different stakeholders view success differently and are influenced by time, need, use, risk management, and a multitude of other similar context specific variables. Given the nature of OSS projects, where work is

² For more information see: http://sourceforge.net/forum/forum.php?forum_id=465092

performed free of charge by voluntary developers without rigid deadlines or implementation schedules, traditional metrics of “on time and within budget completion” or “revenue generation” are not appropriate to measure the success of these projects. Alternative non-traditional metrics have emerged as indicators of success in OSS projects. These metrics may reflect the perspectives of particular stakeholders or they may have been explored in prior research. For example, (Crowston et al, 2003) submits that success or lack thereof is indicated by volume of traffic on the project web site, quantity of code downloads, and the number of people monitoring project activity. The attraction of voluntary developers to join and contribute to an ongoing OSS project, too, is an indicator of OSS project success, as argued by (Stewart et al, 2006).

Independent variables

Sponsorship: A project is sponsored when it is initiated and/or actively supported by a health care organization or a firm providing health related software. We draw from economic theory in proposing that sponsorship increases a project’s likelihood of success. This effect of sponsorship on success may occur because of the provision of resources such as non-volunteer developers, code (Henkel, 2006), or tools. Commitment to a process that is otherwise self-organizing as well as the signaling effect that attracts other developers and users imply that sponsorship should increase the likelihood of project success. Jeppesen and Frederiksen (2006) find that innovative users who contribute to business-hosted communities are either hobbyists or they are responsive to firm recognition. Sponsorship is a categorical variable.

The hypothesis related to sponsorship is as follows:

H1: Project sponsorship is positively related to higher probability of project being classified into a cluster of more successful projects

License type: A software license defines the use, modification and distribution rights assigned to users. The invention of GPL (General Public License) by the Free Software Foundation was followed by a large number of open source licenses³. The major licenses among them are GPL, LGPL, BSD, MIT and the Mozilla Public License. Compared to closed (proprietary) licenses, GPL provides users with the right to use, modify and redistribute software. There are three main types of licenses (Lerner et al, 2003, Nelson et al, 2002), namely, strong copyleft (highly restrictive, such as GPL); weak copyleft (restrictive, such as LGPL); and non-copyleft (non-restrictive, such as BSD). Highly restrictive licenses are less likely to be usurped by an organization that takes the open source code, modifies it, and then commercializes the result. Prior research (Lerner et al, 2002) examined the choice of open source license and found that restrictive licenses are used for projects targeted to end-users rather than developers, and for projects attractive to consumers, such as games. Earlier research has also argued that projects with restrictive licenses should attract more contributors (Lerner et al, 2002, Stewart et al, 2006) but fewer users because of the restrictions and license uncertainty (Stewart et al, 2006).

We have a novel interpretation of the role of licenses in open source development. We propose that a more restrictive license is positively related to higher user downloads. An organization adopting open source at the outset of a project, perceives benefits if it is assured that the project will remain open source in the future. On the other hand, users may perceive that projects with less restrictive licenses will not remain open source in the long run. Stricter licenses are convincing indicators that these projects will not get usurped and will remain open source in the future. Health care organizations, the predominant users of health software listed on SourceForge, usually are not interested in commercializing open source code, and for various reasons should find this assurance appealing (for instance, they can avoid commercial vendor

³ The Open Source Initiative website lists more than 50 approved licenses complying with the open source definition (see <http://www.opensource.org/licenses/>, accessed August 2, 2006)

lock-in). Projects with higher restrictiveness should also attract developers interested in protecting the openness of their work in the future.

The hypothesis pertaining to license restrictiveness is as follows:

***H2:** The higher the license restrictiveness, the greater the probability of the project being classified into a cluster of more successful projects*

Development status: This variable captures the software development status (e.g. pre-planning, alpha, beta, etc.). The development status pinpoints stages of the lifecycle of software development and should affect the success metrics of a project. It stands to reason that the project activity at various stages of development of an OSS project is bound to be varied. Since project success metrics (rank, downloads and activity percentile) all depend heavily on the activity of the project, implicit in our logistics regression of clusters is the hypothesis that development status does have a positive impact on the project classification into successful cluster. Formally, the hypothesis related to project development status is,

***H3:** The more advanced the project development status the higher the probability of project classification into a cluster of more successful projects*

Technological complements: We also explore the relationship between each of programming language (PL), operating system (OS), and database environment (DB) and project success measures. The motivation for this comparison is that these technologies are complements to the project output in the sense that output requires a PL and is deployed in a DB/OS environment. Therefore, these technologies are likely to affect the success metrics of a project. For instance, a project targeting a popular OS or DB environment may increase its success potential. Likewise a project using a popular PL in the health domain should attract developers easily as well as organizations that will use this particular PL to customize the OSS. So formally, the hypothesis related to project technological complements is,

H4: Projects associated with successful technological complements are more likely to be classified into a cluster of more successful projects

METHODOLOGY

Data collection

We searched SourceForge for projects using the various keywords pertaining to health, medical, and bioinformatics applications. This search identified 607 projects related to HIT. We then excluded all indirectly related projects, such as those pertaining to pure medical sciences and medical devices. This filter narrowed the field to 258 projects. We excluded 79 of these on the basis of their not being considered typical HIT as per Institute of Medicine classification of HIT applications. An additional 5 projects were deleted because of duplication. The final sample of 174 projects was considered mainstream HIT falling as they did into such categories as health record systems, health office support, and utilities (such as interoperability). In addition, we gathered from the Internet sponsorship information on each project and integrated this data into the SourceForge dataset. A Java program was written to extract data from the web pages of each of the 174 healthcare open source projects. All extracted data were stored in a CSV (Comma Separated Values) text file that could be loaded easily into other applications, such as Excel and SPSS, for further analysis.

Data preparation and transformation

The variables in the research framework were coded appropriately to fit our analysis. For example, project licenses were coded as highly restrictive, restrictive and non-restrictive. Three variables that had over 15% missing data were dropped from the dataset and not considered further. SVN Repository Commits (82.7%) SVN Repository Reads (83.2%) and Mailing lists (25.7%) were the three variables that were dropped from the dataset. With regard to other variables, missing values were replaced with “0” or the median of the population (which incidentally was “0”).

A large number of variables we studied had “severe positive” skew distributions. To reduce skewness, those variable values were transformed using “Inverse” transformation. Typically, inverse transformation produces values that are ranked in reverse order. It is not difficult to visualize this transformation: 10 becomes 0.1, 100 becomes 0.01, and so on. While 10 is less than 100 ($10 < 100$), the resulting 0.1 is greater than 0.01 ($0.1 > 0.01$). So we used INVERSE REFLECT transformation. In other words, we computed the inverse and then reflected by subtracting the inverse value from one (“1”). So an INVREF transformation of 10 results in 0.9 (or $1 - 0.1$) and INVREF transformation of 100 produces a 0.99 (or $1 - 0.01$). The resulting numbers were ranked in the same order as they were originally. This retention of original ranking of transformed variable values made interpretation of subsequent results less confusing. With the INVREF transformation, the severity in the skewness was reduced but not removed altogether. However, the subsequent statistical processes were not overly sensitive to moderate levels of skewness, so the results are meaningful as well as useful.

Descriptive statistics

Table 1 presents an overview of the project statistics over the past 12 months (the 12-month mean is the value for each metric by project). The mean column represents the grand mean, or the mean of each project’s 12-month mean. In a few cases, data was available for fewer than 12 months; they may have been registered within the past year. All the projects were active as of May 2007.

=====

Table 1 about here

=====

The mean of activity percentile for the projects is 71.84, a positive indicator for the average activity. The average number of developers is 4, but there are projects with as many as 110 developers. It is interesting to note that activity percentiles range from a low of 16.31 to a high of 99.86. The total pages in a project ranges from 14.67 to 58,007.

Table 2 presents the descriptive statistics for the main independent and dependent variables. The most common type of license is restrictive, followed by highly restrictive and non-restrictive. The Highly Restrictive and Restrictive licenses do have some overlap, which can be ascertained easily as follows: The mean of Non Restrictive licenses is 0.17, that is, 17% of the projects belong to the Non Restrictive license type. Therefore, 83% (100 - 17) are of the Restrictive type. The means of Highly Restrictive and Restrictive license types add up to 1.12 (0.47 + 0.65) for a total of 112%; therefore, 29% (112 - 83) of the licenses fall under both Highly Restrictive and Restrictive categories.

=====

Table 2 about here

=====

The prefix DB_ stands for Database Environment, IA_ for Intended Audience, OS_ for Operating System, PL_ for Programming Language, and DS_ for Development Status. To facilitate useful insight and easy interpretation, the dummy variables under each of the categories with these prefixes were first sorted in the order of descending mean values. Then we generated correlations. For example, DB_Unspecified (mean = 0.5805) is listed at the top followed by DB_OS (mean = 0.3448), DB_NOS (mean = 0.0862), and DB_Other (mean = 0.0517) in that order for the Database Environment category. This implies 58% of the projects had not specified the database environment. Additionally, approximately, 34% had employed Open Source (OS)

database technologies, while 8% of projects used Non-Open Source database technologies. Approximately 6% of projects employed two or more database technologies concurrently. (Overlapping classifications can be spotted easily when, as in this case, the total of mean values for classifications exceeds unity.) Additional insights include the fact that the mean of Sponsorship (0 = No; 1 = Yes) is 0.37, implying 37% of projects had sponsors and the remaining 63% did not. As for the intended audience classification, 38.51% of the projects targeted Industry, Science, Organizations and Research (ISOR), while 27.59% targeted Developers. Because the intended audience categories are mutually exclusive, the sum of their mean adds up to unity. While independent operating systems were preferred by 32.18% of the projects, 41.38% employ Java as the preferred programming language. These observations and others in **Table 2** would be of interest to such OSS stakeholders as developers, sponsors, and users.

However, one factor limited our statistical analysis: some projects had missing data or reported none under various dummy variables, currently classified at DB_unspecified, IA_unspecified, OS_unspecified, PL_unspecified, and DS_unspecified. If certain data values had been reported for those projects, some of the results could potentially change.

ANALYSIS AND DISCUSSION OF RESULTS

The steps included the following: first, we incorporated select success metrics (see figure 1) as dimensions of cluster analysis to identify the project cluster; and second, we used logistic regression to analyze the antecedents of project participation in each cluster.

Cluster analysis

Given that our model includes several types of variables including continuous (Downloads, Activity percentile), categorical (Dev_status, License Restrictiveness), and binary (Sponsorship yes/no, other dummy variables), it became necessary to employ two-step clustering unless we could find some transformations to change all of our data into continuous data types. Our solution for this research was to combine cluster analysis and logistic regression. First, we used

cluster analysis to group projects into more successful and less successful groups. Then, binary logistic regression was used to understand the impact of attendant independent variables and complementary factors on the increase or decrease in the probability of each project being classified into either of the designated groups.

For the first step, we used two-step clustering to create clusters in SPSS. We let the system create the best number of clusters. Because the focus of clustering is to demarcate projects into successful or otherwise, we specified three criteria: viz, downloads, rank, and activity percentile for the creation of the clusters. These three dependent variables were chosen primarily because each of them is an alternative measure of project success in different perspectives (Crowston et al, 2003, Stewart et al, 2006]. Dependent variable “developers” is left out of cluster creation process since the prior research didn’t conclusively find association with developer participation and project success (Krishnamurthy). Typically, downloads and activity percentile are positively associated with the success of projects, while rank is negatively associated with the success since lower ranks denote greater success. **Table 3** summarizes the cluster distributions for system-picked (two clusters) and user- specified (three clusters).

=====

Table 3 about here

=====

There is no difference in the first cluster for the system-picked or user-specified cluster creations. It is clear that Cluster 1 (which gives similar results under both processes) is distinct compared to the rest of the data. Consider the following plots of confidence intervals of three key characteristics of clusters that we used as criteria in creating the clusters. **Figures 2a, 2b, and 2c**

show that “downloads” for the first cluster is the main characteristic that differentiates that cluster from the other two in the data set.

=====

Figure 2a about here

=====

=====

Figure 2b about here

=====

=====

Figure 2c about here

=====

The reason for creating two alternate sets of clusters — first a set of two “best clusters” selected by the system and then a set of three “user-requested” clusters — was to compare the sets for developing possible insights. One important discovery was immediately evident: the first cluster remained the same with each approach. This finding suggests the occurrence of a natural cluster on the prescribed dimension viz. the chosen indicators of project success. Descriptive statistics of the best clusters picked by the system are presented in **Table 4** below.

=====

Table 4 about here

=====

The mean values of downloads and activity percentile for cluster 1 at 0.9562 and 83.5743 were higher than those of cluster 2 at .0000 and 57.3941, respectively. Similarly the mean of rank for Cluster 1 at 27742.9737 was substantially lower than the rank of Cluster 2 at 74021.8789 (because the lower ranks indicate higher success). Downloads, Activity Percentile and Rank indicate the predictable behavior because they were used as the basis for defining the clusters in the first place.

It's significant to note that the sponsorship mean for Cluster 1 is a 0.47, indicating that 47% of projects were sponsored. However, only 27% of Cluster 2 projects had sponsors. This finding supports the framing of hypothesis H1. Restrictive and Highly Restrictive license types recorded a higher mean for the Cluster 1 than Cluster 2, supporting the framing of hypothesis H2 that the higher the license restrictiveness the greater the chance a project will be classified as successful. It is interesting to note the mean of Non-restrictive licenses for Cluster 1 is lower than that of Cluster 2, consistent with other findings.

To summarize, Cluster 1 encompasses the most successful open source projects in HIT. These projects are characterized by relatively high downloads, high rank, and more developers. Cluster 1 projects are also more likely to be sponsored, and they have more restrictive licenses. These observations are consistent with our research framework.

=====

Table 5 about here

=====

Table 5 presents the descriptive statistics for the three-cluster definition.

The combination of the means of the first three variables in the table defines the centroid of each cluster. The mean of total bugs is higher for successful projects than it is for less successful projects. If one infers that the improvements to projects are based on total bugs reported, then we

can surmise that reporting of more bugs indicates higher activity levels and better quality patronage. Sponsorship and license restrictiveness across the three clusters were generally consistent with prior research findings.

Logistic regression

According to Mertler & Vannatta (2002), “logistic regression has the same basic purpose as discriminant analysis—the classification of individuals into groups.” They go on to elaborate that “logistics regression seeks to identify a combination of IVs (independent variables)—which are limited in few, if any, ways—that best predicts membership in a particular group, as measured by a categorical DV (dependent variables).”

One advantage is that no assumption need be made that the predictors are normally distributed, linearly related, or have equal variances within the groups. Accordingly, we do not specifically screen the data for normality, linearity or homoskedasticity in preparation for the logistic regression. Further, since we have used “inverse reflect” transformation on the continuous variables to facilitate other statistical models, and most of the predictors are either categorical or binary, we have effectively avoided the problems with outliers. A preliminary multiple regression was performed to examine multicollinearity among the predictor variables and revealed the tolerance for all variables to be greater than 0.2, the recommended tolerance as per Field, (Field, 2005).

As explained under cluster analysis section, Downloads emerged as the single most dominant factor in the creation of clusters. We left “downloads” out of the binary logistic regressions so that we might understand the impact of other predictors. Since the system picked only two clusters as best clusters, we limit the logistic regression discussion to the two clusters picked by the system. Instead of a single categorical variable “Development Status” (Dev_status) on a scale of 1-7 (denoting Planning, Pre-Alpha, Alpha, etc.), we coded binary 0/1 for each development

status stage. Similarly, we coded binary dummy variables for the intended audience and programming language and other categorical variables as discussed above.

A Backward Stepwise Binary Logistic Regression was conducted to determine the independent variables that are significant predictors of the classification of projects into best cluster categories. The regression results indicate that the overall model of 11 predictors and a constant is significant in distinguishing between “successful” and less “successful” projects. (- 2 Log likelihood = 153.774; χ^2 (11) = 85.576; $p < .0001$). The model correctly classified 81.6% of the cases. Regression coefficients that are significant in the equation that predicts the cluster membership are presented in **Table 6** below. Since the Wald statistic is considered to be very conservative and by adopting a liberal significance level ($p < .05$ or $p < .1$), nine of the 11 variables are found to be significant contributors to predicting the project category.

=====
Table 6 about here
=====

Hypotheses testing

The results obtained from a logistic regression are somewhat different from the other types of regression equations in that, what is predicted in a logistic regression is the probability of a case being classified into a category rather than the value of a DV. The odds ratio or the Exp (B) indicates increase (or decrease if the B value is negative) in odds of being classified in a category when the predictor variable increases by 1. Therefore, the Exp (B), the odd ratio for programming language (PL_CPlusPlus) at 7.736, indicates that for an increase of 1 unit (in this case the flip of 0 to 1 of the dummy variable) there is 7.736 times likelihood of the project being successful for every 1 time of likelihood of project being unsuccessful.

Surprisingly, sponsorship is not indicated at all as a significant predictor of a project success. Therefore, Hypothesis H1 doesn’t find support. However, non restrictive license does

appear as a significant factor having an effect on project classification. Our hypothesis concerning the project licensing was that the higher restrictive licenses lead to project success. To support this hypothesis, one would like to have seen highly restrictive license obtaining a higher Exp (B) than restrictive license and non restrictive license's Exp (B) values respectively. But that was not the case here. So hypothesis H2 also fails to find support.

Three development status levels, including in order Production Stable, Multiple and Beta, have high odds ratios for indicating greater influences of those variables in influencing the probability of the project classification. This evidence provides support for hypothesis H3.

Programming language (PHP, C++), operating system (Proprietary) are predictors of project classification. Providing support for H4, these findings suggest that the success of a project is related to the availability of complementary assets, such as programming skills of developers and operating systems employed by users. Thus, technological factors such as choice of programming language and choice of target operating system strongly influence project success, and should be carefully chosen by project leaders.

Project leaders should carefully analyze and understand the impact of these variables (factors) and their tradeoffs. To summarize, while sponsorship encourages developer participation and higher activity in a project (based on past research findings), it does not guarantee the translation of these positive effects into higher downloads or a higher rank for the project. It is surprising that sponsorship did not influence project success. With regards to license restrictiveness, while it attracts more downloads and consequently results in a higher project rank and higher activity percentile (based on past research findings), we found in Table 6 that license restrictiveness does not guarantee the project classification into the successful projects cluster. This last finding is somewhat inconsistent with the increased downloads and higher activity percentile.

Several inferences can be drawn. While project sponsorship and license restrictiveness had significant influence on project success metrics, they did not directly impact project classification as successful or less than successful. Project development status indeed finds a prominent place in the logistics regression results. This suggests that the stages of development status have significant impact on project classification. Further, it is noted that programming language and operating system also have significant impact on project classification.

CONCLUSIONS AND FUTURE RESEARCH

This study proposed a research framework that explains open source project success and developed a method of classifying open source HIT projects. That identification of project classes provides useful insights to all OSS stakeholders in terms of project success and the drivers of that success. The study illustrates the usefulness of this approach in the context of HIIT projects, while future research can leverage this method to other open source settings. Interestingly, development status, programming language (PHP, C++), and operating system (proprietary) are predictors of project classification. These findings suggest that the success of a project is related to such complementary assets as programming skills of developers and operating systems used by users. Thus, not only legal/social factors (such as license, organizational sponsorship) but also technological factors (such as choice of programming language and target operating system) strongly influence project success. Leaders of future projects should carefully consider the tradeoffs between these variables.

Before we emphasize the contributions, a number of limitations should become explicit. Since data from SourceForge was gathered at a specific time, this study is a snapshot in time. We recognize, too, that not all open source HIT projects are registered with SourceForge; many are registered at Freshmeat and at other related web sites. And many high profile projects maintain their own developer sites. Another limitation is that some projects may have outdated or erroneous data in their listings, not to mention those projects for which there was missing data.

We assume that the HIT-related projects found on SourceForge, given the sites popularity and the large number of projects and developers registered there, are representative of the overall open source movement in health care.

The study makes a number of important contributions. First, we use cluster analysis to identify groups of successful and unsuccessful projects on SourceForge and find predictors of participation in each group. This systematic approach can benefit future studies that attempt to identify different types of projects in other domains. Second, we develop a theoretical framework that examines the role of technological complements, project sponsorship, development status and license type in the pattern of open source development projects and we test related hypotheses. Drawing from economic theory, a novel proposition in our framework suggests that higher project-license restrictiveness will increase OSS adoption, because organizations will be more confident that the OSS project will remain open source in the future. Third, we demonstrate how open source development may be better understood in the context of a specific domain—healthcare, and we provide insights on the status of open source development in that domain.

Project sponsors, such as firms or organizations, too can benefit from our insights. These findings have the potential to help sponsors identify projects worthy of their time and resource investments, whose success would enhance both brand recognition and market presence. Further, the programming language, database technology, and operating system preferences of developers and users of open source software projects are useful information to IT firms related with these technologies.

Regarding HIT, future research should consider the open source development dynamics (Katsamakos et al, 2007) in the HIT context. The impact of OSS on HIT diffusion is another area worth investigating. A time series analysis and longitudinal studies may provide more sophisticated insights into the OSS development process.

Future research might consider a study that compares generic OSS (e.g. projects listed on SourceForge and Freshmeat) and those developed in-house (e.g. bioinformatics applications). Detailed case studies of important development projects should provide a richer understanding of open source development in healthcare. A related problem to be examined is the adoption of open source software by healthcare organizations. While OSS applications development in health has great potential, the research framework, classification approach and findings presented here may be applied to other industries and organizations. But clearly open source development, especially with regard to health care, is a growing field. This is good and timely news given the need for HIT, wherein lies the opportunity to transform an entire industry.

REFERENCES

Babcock, C., & McGee M. K.. (2007) "Microsoft Vs. Open Source Moves Into Health Care," *InformationWeek*, 1128, 31-32.

Brailer, D.J.(2005) "Economic Perspectives on Health Information Technology," *Business Economics*, 40(3), 6-14.

Crowston, K., & Howison J.. (2006) "Assessing the Health of Open Source Communities," *Computer*, 39(5), 89-91.

Crowston, K., Annabi, H., & Howison J.. (2003) "Defining Open Source Software Project Success," *Proceedings of the 24th International Conference on Information Systems*, 1-14.

Crowston, K., Howison, J., & Annabi H.. (2006) "Information Systems Success in Free and Open Source Software Development: Theory and Measures," *Software Process Improvement and Practice*, 11(2), 123-148.

Crowston, K., Li, Q., Wei, K. N., Eseryel, U.Y., & Howison J.. (2007) "Self-organization of Teams for Free/Libre Open Source Software Development," *Information and Software Technology*, 49(6), 564-575.

Economides, N., & Katsamakas E.. (2006) "Two-Sided Competition of Proprietary vs. Open Source Technology Platforms and the Implications for the Software Industry," *Management Science*, 52(7) 1057-1071.

English R., & Schweik C. (2007) "Identifying Success and Tragedy of FLOSS Commons: A Preliminary Classification of Sourceforge.net Projects," in: *IEEE International Conference on Software Engineering*, Minneapolis, Minnesota.

- Field, A.(2005) *Discovering Statistics using SPSS*, Sage Publishing, Thousand Oaks, CA.
- Goulde, M. and Brown, E. (2006) "Open Source Software: A Primer for Health Care Leaders', Forrester Consulting.
- Henkel, J. (2006) "Selective Revealing in Open Innovation Processes: The Case of Embedded Linux," *Research Policy*, 35(7), 953-969.
- Jeppesen, L.B., & Frederiksen L.. (2006) "Why Do Users Contribute to Firm-hosted User Communities? The Case of Computer-controlled Music Instruments," *Organization Science*, 17(1), 45-63.
- Kantor, G. S., Wilson, W. D. and Midgley, A. (2003) "Open-source software and the primary care EMR," *Journal of the American Medical Informatics Association*, 616.
- Katsamakas, E., & Georgantzas N. (2007) "Why Most Open Source Development Projects Do Not Succeed?," in: *Proceedings of IEEE International Conference on Software Engineering*, Minneapolis, Minnesota.
- Krishnamurthy, S. "Cave or Community? An empirical examination of 100 mature open source projects," *First Monday*, 7(4). http://www.firstmonday.dk/issues/issue7_6/Krishnamurthy/index.html.
- Lerner, J., & Tirole J. (2002) "Some Simple Economics of Open Source," *Journal of Industrial Economics*, 50(2), 197-234.
- Lerner, J., & Tirole J.. (2005) "The Scope of Open Source Licensing," *Journal of Law Economics & Organization*, 21(1), 20.
- McDonald, C. J., Schadow, G., Barnes, M., Dexter, P., Overhage, J. M., Mamlin, B. & McCoy, J. M. (2003) 'Open source software in medical informatics-why, how and what', *International Journal of Medical Informatics*, 69, 175-184.
- Mertler, C.A., & Vannatta R.A.. (2002) *Advanced and Multivariate Statistical Methods* 2nd edition, pp 313, Pyrczak Publishing, Los Angeles, California.
- Nelson, M., Sen, R., & Subramaniam C.. (2006) "Understanding Open Source Software: A Research Classification Framework," *Communications of the AIS*,17.
- Raghupathi, W., & Gao W.. (2007) "An Eclipse-based Development Approach to Health Information Technology," *International Journal of Electronic Healthcare*, 3(4), 433-452.
- Stewart, K. J., Ammeter, A. P., & Maruping L. M.. (2006) "Impacts of License Choice and Organizational Sponsorship on User Interest and Development Activity in Open Source Software Projects," *Information Systems Research*, 17(2) 126-144.

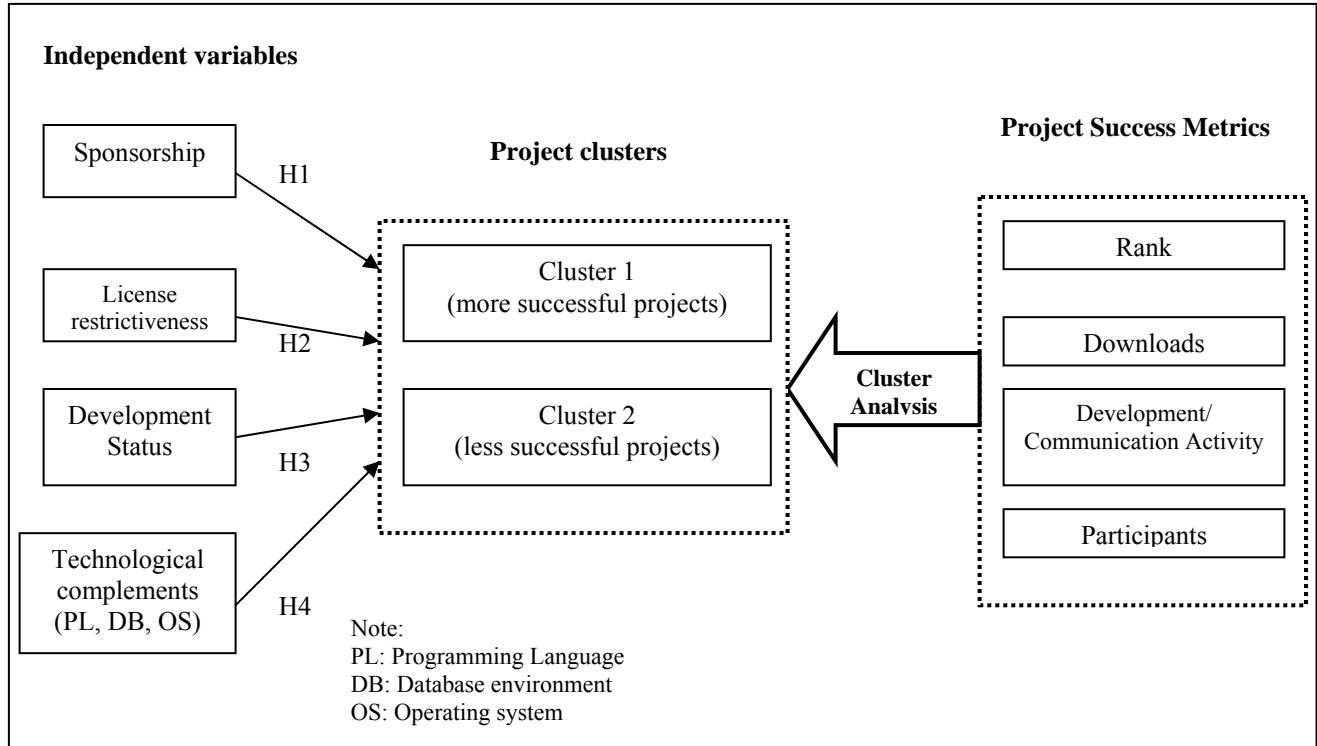


Figure 1: Research Framework

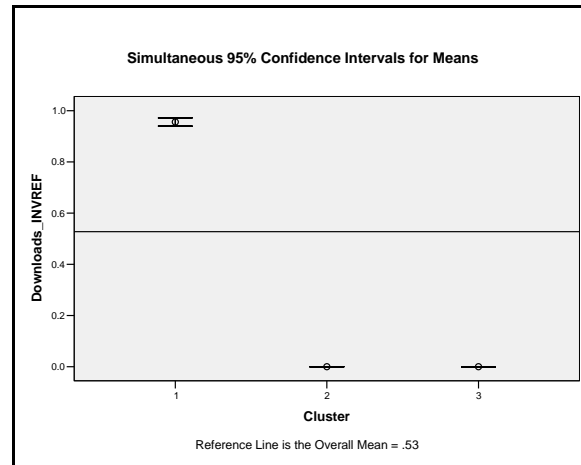


Figure 2a: Simultaneous 95% Confidence Intervals for Mean value of Downloads by the Cluster Numbers

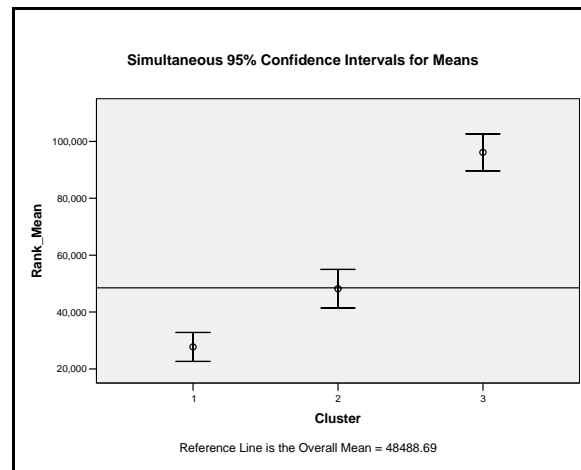


Figure 2b: Simultaneous 95% Confidence Intervals for Mean Value of Project Rank by the Cluster Numbers

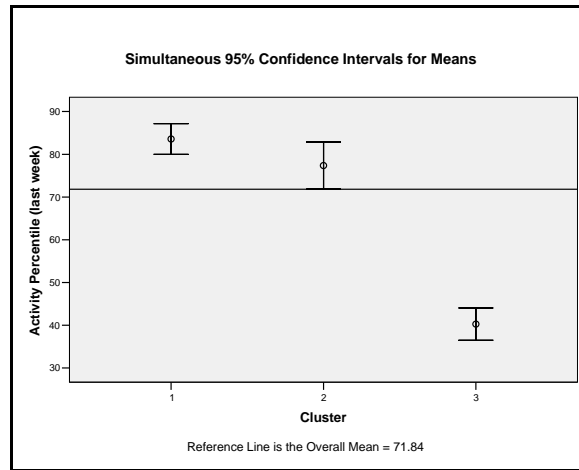


Figure 2c: Simultaneous 95% Confidence Intervals for Mean value of Project Activity Percentile by the Cluster Numbers

Table 1: Project Statistics

Project Activity metrics	Minimum	Maximum	Mean	Std. Deviation
Developers	0	110	4.09	9.267
Activity Percentile (last week)	16.31	99.86	71.8383	22.28214
Forum Messages	.00	3973.00	28.2414	301.15120
Mailing Lists	.00	16.00	.5747	1.69072
Open Bugs	.00	37.00	1.0862	3.94449
Total Bugs	.00	72.00	3.1379	10.88529
Open Support Requests	.00	18.00	.3161	1.52743
Total Support Requests	.00	18.00	.4138	1.76053
Open Patches	.00	3.00	.0287	.27261
Total Patches	.00	3.00	.0402	.29168
Open Feature Requests	.00	23.00	.9425	3.33611
Total Feature Requests	.00	58.00	1.4885	6.06205
Total Pages	14.67	58007.42	1473.55	5609.67
Down loads	.00	10234.25	182.13	859.86
Project Web Hits	0	40825	1580.63	5369.411
Tracker opened	.00	8.75	.15	.83
Tracker closed	0	6	.08	.552
Forum Posts	0	15	.21	1.534
Rank (Mean)	123.92	141852.50	48488.69	33776.82

Table2: Descriptive Statistics for Independent and Dependent Variables

	Variable (N=174)	Mean	S D
1	Activity Percentile (last week)	71.8383	22.28214
2	Developers_INVREF	0.6423	0.17427
3	Downloads_INVREF	0.5275	0.47923
4	Rank_Mean	48488.6898	33776.823
5	Restrictive	0.65	0.479
6	HighlyRestrictive	0.47	0.501
7	NonRestrictive	0.17	0.379
8	Sponsorship (0 No 1 Yes)	0.37	0.484
9	DB_UNSPECIFIED	0.5805	0.49491
10	DB_OS	0.3448	0.47668
11	DB_NOS	0.0862	0.28148
12	DB_OTHER	0.0517	0.22211
13	IA_ISOR	0.3851	0.48801
14	IA_DEV	0.2759	0.44824
15	IA_UNSPECIFIED	0.1667	0.37375
16	IA_ENDUSERS	0.0862	0.28148
17	IA_GOVNP	0.0345	0.18299
18	IA_AEU	0.023	0.1503
19	IA_EDU	0.023	0.1503
20	IA_CS	0.0057	0.07581
21	OS_Independent	0.3218	0.46853
22	OS_UNSPECIFIED	0.2759	0.44824
23	OS_MIXED	0.1782	0.38375
24	OS_PROPRIETARY	0.0977	0.29777
25	OS_OPENSOURCE	0.0632	0.24406
26	OS_PORTABLE	0.0517	0.22211
27	OS_OSX	0.0057	0.07581
28	OS_IND_WINCE	0.0057	0.07581
29	PL_JAVA	0.4138	0.49393
30	PL_Misc	0.2184	0.41435
31	PL_Unspecified	0.2069	0.40625
32	PL_PHP	0.1724	0.37883
33	PL_C	0.092	0.28979
34	PL_Python	0.069	0.25413
35	PL_CPlusPlus	0.0575	0.23341
36	PL_Perl	0.0575	0.23341
37	PL_PLSQL	0.046	0.21004
38	PL_MUMPS	0.0287	0.16754
39	PL_VB.NET	0.023	0.1503
40	PL_TcL	0.0115	0.1069
41	PL_XSL	0.0057	0.07581
42	DS_Unspecified	0.2241	0.41822
43	DS_ProdnStable	0.1667	0.37375
44	DS_Beta	0.1494	0.35754
45	DS_Planning	0.1322	0.33967
46	DS_Multiple	0.1034	0.30542
47	DS_PreAlpha	0.0977	0.29777
48	DS_Alpha	0.0977	0.29777
49	DS_Mature	0.023	0.1503
50	DS_Inactive	0.0057	0.07581

Table 3. Cluster Distribution

	Best clusters picked by system			Three cluster request result		
		N	% of Total		N	% of Total
Clusters	1	96	55.2%	1	96	55.2%
	2	78	44.8%	2	36	20.7%
				3	42	24.1%
		174	100.0%		174	100.0%

Table 4: Descriptive Statistics for Key Variables of Best Clusters

BEST CLUSTERS: NUMBER 1				BEST CLUSTERS: NUMBER 2		
	N	Mean	Std. Deviation	N	Mean	Std. Deviation
Downloads_INVREF	96	.9562	.06377	78	.0000	.00000
Rank_Mean	96	27742.9737	20445.14634	78	74021.8789	29164.16388
Activity Percentile (last week)	96	83.5743	14.42206	78	57.3941	21.81324
TotalBugs_INVREF	96	.2728	.38090	78	.0489	.19589
Sponsorship 0 No 1 Yes	96	.47	.502	78	.24	.432
HighlyRestrictive	96	.51	.503	78	.42	.497
Restrictive	96	.74	.441	78	.54	.502
NonRestrictive	96	.16	.365	78	.19	.397
Developers_INVREF	96	.6808	.18848	78	.5948	.14236

(Note:the combination of the means of first three variables, Downloads_INVREF, Rank_mean and Activity Percentile define the centroid of the cluster).

Table 5: Descriptive Statistics for Key Variables of Three Cluster Definition

	Cluster 1: Successful			Cluster 2: Moderately Successful			Cluster 3: Least successful		
	N	Mean	Std. Deviation	N	Mean	Std. Deviation	N	Mean	Std. Deviation
Downloads_INVREF	96	.9562	.06377	36	.0000	.00000	42	.0000	.00000
Rank_Mean	96	27742.9737	20445.14634	36	48198.2929	16237.47063	42	96156.3811	16878.40076
Activity Percentile (last week)	96	83.5743	14.42206	36	77.3694	13.08993	42	40.2724	9.83260
TotalBugs_INVREF	96	.2728	.38090	36	.0646	.22475	42	.0354	.16894
Sponsorship 0 No 1 Yes	96	.47	.502	36	.22	.422	42	.26	.445
HighlyRestrictive	96	.51	.503	36	.44	.504	42	.40	.497
Restrictive	96	.74	.441	36	.58	.500	42	.50	.506
NonRestrictive	96	.16	.365	36	.28	.454	42	.12	.328
Developers_INVREF	96	.6808	.18848	36	.6145	.14858	42	.5779	.13632

(Note: there is no difference in the profile of the first cluster compared to the first cluster under best clusters picked by the system).

Table 6: Regression Coefficient Obtained Under Binary Logistic Regression

Variable	B	Wald	Df	Sig.	Exp(B)
DS_ProdnStable	1.933	9.645	1	.002	6.912
DS_Planning	-3.403	8.233	1	.004	.033
TotalBugs_INVREF	-2.289	7.739	1	.005	.101
PL_PHP	-1.618	7.535	1	.006	.198
OS_Proprietary	1.334	7.397	1	.007	3.797
DS_Multiple	1.930	5.334	1	.021	6.888
DS_Beta	1.272	4.068	1	.044	3.569
PL_CPlusPlus	2.046	3.157	1	.076	7.736
NonRestrictive	-.939	2.783	1	.095	.391

Dependent variable: Cluster Number (1 or 2)