Open Source Project Categorization Based on Growth Rate Analysis and Portfolio Planning Methods

Stefan Koch and Volker Stix
Vienna University of Economics and Business Administration
Institute for Information Business
Augasse 2-6
1090 Vienna/Austria
{stefan.koch, volker.stix}@wu-wien.ac.at

Abstract. In this paper, we propose to arrive at an assessment and evaluation of open source projects based on an analysis of their growth rates in several aspects. These include code base, developer number, bug reports and downloads. Based on this analysis and assessment, a well-known portfolio planning method, the BCG matrix, is employed for arriving at a very broad classification of open source projects. While this approach naturally results in a loss of detailed information, a top-level categorization is in some domains necessary and of interest.

1 Introduction

The adoption and evaluation of open source projects has gained increasing interest, both from an academic and business perspective. This has led to the development of assessment schemes like OpenBRR (Open Business Readiness Rating), Open Source Maturity Model, QSOS by Atos Origin, OpenBQR [8] and similar achievements. Most of these approaches are based on detailed scoring of open source products, and aggregation using some form of weightings. While some consider that features of the underlying community form an important part of an evaluation, this is not generally acknowledged. In addition, while in some approaches the use of real data, both on community and the software product itself is planned, some rely on personal rating or data entry of many features.

The approach we will present here is intended to be coupled with a repository of repositories on open source project data [7]. In this context, an enormous amount of data is collected, but this might prove to create additional problems with users of such a service, who might not be able to put it to use. Too much information could effectively hamper their use of the system for rather simple evaluation and adoption tasks or decisions. Therefore, providing a top-level, aggregate classification scheme is necessary. In this paper, we will propose to base such an effort on well-known portfolio planning techniques, especially the BCG matrix. For constructing the axis of such a matrix, several possibilities exist, but in our case we will argue for applying the results of growth rate analysis.

Please use the following format when citing this chapter:

Koch, S. and Stix, V., 2008, in IFIP International Federation for Information Processing, Volume 275; *Open Source Development, Communities and Quality*; Barbara Russo, Ernesto Damiani, Scott Hissam, Björn Lundell, Giancarlo Succi; (Boston: Springer), pp. 375–380.

2 Growth Rate Analysis

For performing the proposed analysis of open source software projects, the information contained in software development repositories will be used. These repositories contain a plethora of information on the underlying software and the associated development processes [2]. Studying software systems and development processes using these sources of data is very cost-effective and does not influence the software process under consideration. In addition, longitudinal data is available, allowing for analyses considering the project history. Depending on the tools used in a project, possible repositories available for analysis include source code versioning systems, bug reporting systems, or mailing lists.

In open source software development projects, repositories in several forms are also in use, in fact form the most important communication and coordination channels. Therefore only a small amount of information can not be captured by repository analyses because it is transmitted inter-personally. As a side effect, the repositories in use must be available openly and publicly. Therefore open source software development repositories form an optimal data source for studying the associated type of software development. Currently, efforts are underway to consolidate information from diverse sources, in building RoRs (repositories of repositories) [7]. In this pa-per, we propose to build on this infrastructure using appropriate techniques.

For characterising the past development, and also gain an understanding of possible future developments, growth rates can be computed for several aspects like source code, contributors, bug reports, mailing list postings or downloads. All of these might give some insight, while of course the growth in size (of source code) is most often cited (software evolution). For computing and characterising the growth rates, the following methodology will be adopted. This is taken from a prior study of one of the project participants [4] on growth in size.

The first step is to analyse whether a linear or other growth pattern is present in the data. To this end, both a linear and a quadratic model are computed for each project, taking the size in lines-of-code S as a function of the time in days since the first commit t, which is used as project start date, and using one month as time window. Therefore model A is formulated simply as SA(t) = a * t + b and model B as SB(t) = a * t2 + t *b + c. The necessary parameters are to be estimated using regression techniques. As a next step, it is necessary to explore whether the growth rate is decreasing over time. This can be done by analysing the second derivative of the quadratic model *SB(t)'*, or directly the coefficient of the quadratic term a.

The sharp distinction between two groups of projects might prove too inflexible. A new group is therefore introduced representing linear growth in contrast to sub- and super-linear rates. This group is defined as those projects having either a better fit for the linear than the quadratic model, or a coefficient of the quadratic term between -0.1 and 0.1, thus being very near to zero. This allows for arriving, for each project and each aspect of interest, at a classification for the evolutionary behavior as being either sub-linear, linear, or super-linear.

3 Portfolio Planning and the BCG Matrix

Portfolio planning methods have been applied in strategic decision making for over 20 years [1,9] although they have little theoretical support. They are presented in the literature as diagnostic aids and as prescriptive guides for selecting strategic options [5]. The general idea is to classify positions of products along two dimensions to form a matrix: attractiveness of the market and ability of the product to compete within that market, and to derive insights into strategic actions in this way. Managers often neglect to use a rational economic approach, instead applying un-structured judgmental processes. They may base their decisions on power or emotional factors, which might lead to many of their decisions as being irrational. Thus, portfolio planning methods, such as the BCG matrix, may lead managers to make decisions that are less irrational.

Maybe the most well-known portfolio planning method is the Boston Consulting Group (BCG) method [3], the most widely used portfolio method in US firms [1]. It is based on measuring market attractiveness by market growth rate, and it assesses the firm's ability to compete by its relative market share. The BCG matrix assumes a causal relationship between market share and profitability. It is based on product life cycle theory that can be used to determine what priorities should be given to different products. To ensure long-term value creation, a company should construct a portfolio using products that contains both high-growth products in need of cash inputs and low-growth products that generate cash. Each of the two axes is normally divided into a high and a low portion, resulting in four different quadrants. Each quadrant is assigned both a catching name and a general strategy (see also Figure 1):

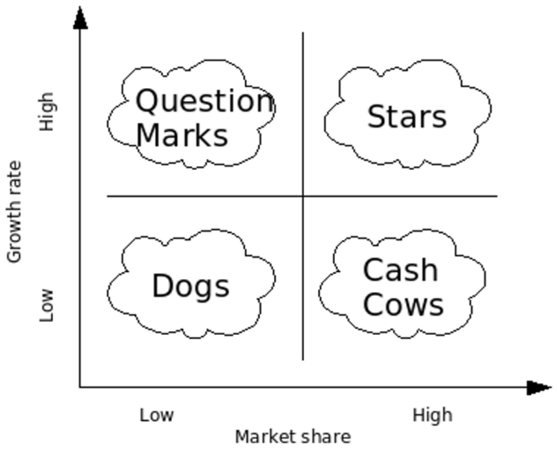

Fig. 1. BCG matrix

Stars are located in the high growth and high market share area. Normally, the cash flow is rather balanced or even, but a position in stars should be maintained. Cash Cows are placed in low growth area coupled with high market share. Profits and cash generation will generally be high, with relatively small investment due to low growth, translating into a very desirable type. Dogs are placed in low growth and low market share quadrant, which are normally associated with a de-invest strategy. Question Marks are enjoying high growth but low market share, resulting in demand for cash but low profits. This kind of product over time might turn into either star or dog, so careful considerations is advised to invest or liquidate.

A review of previously published evaluations of the BCG matrix can be found in [6]. Actual practical use of the BCG matrix is often found to be inhibited by difficulties in measurement of market growth rates and relative market shares. The results are highly sensitive on these measurements. As a result, different matrix methods are likely to yield different recommendations for the same situation.

4 Open Source Matrix and Classification

In this paper, we propose to adopt the BCG matrix approach for classifying open source projects. There are two main aspects to discuss and decide: The construction of the axes, and results of the classification. For constructing and measuring the axes, we propose to use the results of a growth rate analysis. The growth rate of an open source project is constructed using the growth rate in source code (which equals the software evolution viewpoint of software engineering research) and the growth rate of developers. For market share, we propose to use growth rates of bug reports and downloads. Bug reports normally are associated with usage of a product, especially by interested individuals, but might also signal a product with problems.

As each growth rate of the four types used here is classified in one of three steps, conversion into a single measure needed. We propose to use the mean of both rates, with source code respectively downloads taking priority in ties. Using this approach, an open source project matrix can be constructed, with the standard names having been replaced by possible release numbers (see Figure 2).

As can be seen, the classification results in four possible types of projects. This has to be translated into strategies. We need to differentiate between two possible uses, the first one being simple adoption, in which a company or individual wants to decide on which project within a given domain to adopt for a certain task. In the second case, a company wants to build a portfolio of projects. Possible reasons for this include a business model based on a range of software, cooperation with the open source world within a given area, or an application in marketing. Also a company that might pursue a development based on open source, but wants to keep open to several projects might pursue this idea. This leads to the following strategies as-signed to the different types of projects.

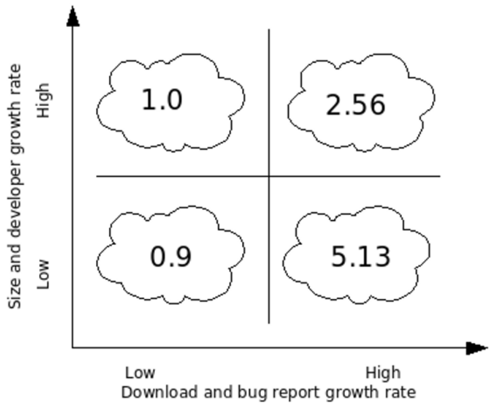
Size and developer growth rate
High
Low
1.0
2.56
0.9
5.13
Low
High
Download and bug report growth rate

Fig. 2. Open source project matrix

1.0 (Question Marks) currently enjoy huge growth in size and participants, but have not achieved widespread adoption yet. Therefore they might become quite successful, but might fail. For a separate adoption decision, these projects pose considerable risk, while adding 1.0 projects to a portfolio might be interesting.

2.56 (Stars) enjoy both considerable growth and adoption. This makes them interesting candidates for any portfolio selection decision, and candidates for a singular adoption consideration.

5.13 (Cash Cows) have somewhat stabilized in their code and developer growth, but have achieved widespread adoption. This means that normally a mature solution has been found, with less emphasis on introducing new functionalities. This makes these projects prime candidates for consideration for a single adoption decision, and an interesting candidate for portfolio selection. On the other hand, there might be the need for maintenance at a later stage, maybe due to technological changes, but the community is not that active any more.

0.9 (Dogs) have neither huge growth nor adoption, meaning that they prove to be of interest to neither adoption or portfolio considerations. These projects could be termed as failed.

5 Conclusion and Future Research

In this paper we have argued for constructing a top-level open source project classification based on growth rate analyses of several project aspects, and using portfolio planning techniques, especially the BCG matrix. Given current efforts of constructing the databases necessary to support the kind of analyses that form the basis of this approach, pursuing this road seems worthwhile. Portfolio planning approaches have been in business use since several decades, and despite some shortcomings, have provided value and are in wide-spread use. Given that software adoption decisions on organisational levels are often decisions made on a management level, adopting these common approaches might prove beneficial for communication between technical evaluation level and decision authorities.

In future research, an empirical evaluation of the proposed approach would be very important. This would include performing the necessary analyses based on a database, and presenting the results of categorization to decision-makers. There are several possible ways of refinement of the approach that could be pursued: On the one hand, the construction of the axes could be discussed by adding or substituting aspects of projects or communities. On the other hand, there are other portfolio planning approaches besides the BCG matrix that could be explored.

Acknowledgments

This work is partially supported by the Free / libre / open source software metrics and benchmarking study (FLOSSMETRICS), funded by the European Commission under con-tract number FP6-033982 (www.flossmetrics.org).

References

1. Armstrong, J.S., & Brodie, R.J. (1994) Effects of portfolio planning meth-ods on decision making: experimental results. International Journal of Re-search in Marketing, 11(1):73-84.
2. Cook, J.E., Votta, L.G., & Wolf, A.L. (1998) Cost-effective analysis of in-place software processes. IEEE Transactions on Software Engineering, 24 (8):650-663.
3. Day, G. (1986) Analysis for strategic market decisions. St. Paul: West.
4. Koch, S. (2007) Software Evolution in Open Source Projects - A Large-Scale Investigation. Journal of Software Maintenance and Evolution, 19(6):361-382.
5. Kotler, P. (1984) Marketing Management. 5th ed. Englewood Cliffs, NJ: Prentice-Hall.
6. Morrison, A. & Wensley, R. (1991) Boxing up or boxed in? A short history of the Boston Consulting Group share/growth matrix. Journal of Marketing Management, 7:105-129.
7. Sowe, S.K., Angelis, L., Stamelos, I., & Manolopoulos, Y. (2007) Using Repository of Repositories (RoRs) to Study the Growth of F/OSS Projects: A Meta-Analysis Research Approach. In Open Source Development, Adoption and Innovation, pp. 147-160, Boston: Springer.
8. Taibi, D., Lavazza, L., & Morasca, S. (2007) OpenBQR: a framework for the assessment of OSS. In Open Source Development, Adoption and Innovation, pp. 173-186, Boston: Springer.
9. Wind, Y., & Mahajan, V. (1981) Designing product and business portfolios.