The Journal of Systems and Software 128 (2017) 164–197

Contents lists available at ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Source code metrics: A systematic mapping study

Alberto S. Nuñez-Varela[*], Héctor G. Pérez-Gonzalez, Francisco E. Martínez-Perez, Carlos Soubervielle-Montalvo

Facultad de Ingeniería, Universidad Autónoma de San Luis Potosí, Manuel Nava No.8, San Luis Potosí 78216, Mexico

A R T I C L E   I N F O

Article history:
Received 10 June 2016
Revised 16 January 2017
Accepted 27 March 2017
Available online 5 April 2017

Keywords:
Source code metrics
Software metrics
Object-oriented metrics
Aspect-oriented metrics
Feature-oriented metrics
Systematic mapping study

A B S T R A C T

Context: Source code metrics are essential components in the software measurement process. They are extracted from the source code of the software, and their values allow us to reach conclusions about the quality attributes measured by the metrics.
Objectives: This paper aims to collect source code metrics related studies, review them, and perform an analysis, while providing an overview on the current state of source code metrics and their current trends.
Method: A systematic mapping study was conducted. A total of 226 studies, published between the years 2010 and 2015, were selected and analyzed.
Results: Almost 300 source code metrics were found. Object oriented programming is the most commonly studied paradigm with the Chidamber and Kemerer metrics, lines of code, McCabe's cyclomatic complexity, and number of methods and attributes being the most used metrics. Research on aspect and feature oriented programming is growing, especially for the current interest in programming concerns and software product lines.
Conclusions: Object oriented metrics have gained much attention, but there is a current need for more studies on aspect and feature oriented metrics. Software fault prediction, complexity and quality assessment are recurrent topics, while concerns, big scale software and software product lines represent current trends.

1. Introduction

Software metrics are an essential aid to the software measurement process. Software measurement is a task that is carried out during all the phases of the software development process. During this process, many intermediate or final software products are developed and measured by software product metrics. One of those products is the program's source code, which is part of the final software system and is measured to assess its quality. Source code metrics are a type of product metrics that focus on measuring the source code of a system. This measurement is achieved by mapping a particular characteristic of a measured entity to a numerical value (Lanza and Marinescu, 2006), taking in consideration that for some cases that value can be also categorical or ordinal.

Source code metrics such as Lines of Code (LOC), McCabe Cyclomatic Complexity (Mccabe, 1976) and the Halstead Software Science Metrics (Halstead, 1977), are widely known and have been studied for many years. LOC is usually mentioned to have been first used in the late 1960′s (Fenton and Neil, 1999), and the McCabe and Halstead metrics in the late 1970′s. Those metrics were the most studied during a period of time that spanned from the 1970′s to the beginning of the 1990′s, when object oriented metrics became popular thanks to the research and metrics sets proposed by Chidamber and Kemerer (1994) and Li and Henry (1993). Source code metrics are a very important component for the software measurement process, and are commonly used to measure and improve the quality of the source code itself. Additionally, these metrics are being used in a wide variety of applications and experiments related to source code in general (i.e. fault prediction, testing, refactoring) to assess the overall quality of the software. Moreover, source code metrics are gaining importance due to their applications in software product lines and programming concerns. In order to accomplish the different measurement needs, many source code metrics, designed from different programming paradigms, have been proposed, studied and validated throughout the years, with new metrics and studies being proposed constantly. This pace, along with the large body of research that source code metrics provide, makes it difficult for researchers to keep track of the current state and trends of source code metrics. This pa-

[*] Corresponding author.
E-mail addresses: alberto_snv@hotmail.com (A.S. Nuñez-Varela), hectorgerardo@yahoo.com (H.G. Pérez-Gonzalez), eduardo.perez@uaslp.mx (F.E. Martínez-Perez), csober22@gmail.com (C. Soubervielle-Montalvo).

http://dx.doi.org/10.1016/j.jss.2017.03.044
0164-1212/© 2017 Elsevier Inc. All rights reserved.

per presents a systematic mapping study on source code metrics. It aims to provide a clear understanding of the state of the art on the use of source code metrics and their empirical studies (i.e. case study, experiments, and experience reports). All programming paradigms, currently being researched, are considered as part of the study. It is also important to identify the elements that allow the case studies to be replicated, which is a necessary attribute in a scientific work to evaluate its validity. These elements include the metrics used, systems measured, and how the metrics' values are obtained.

The main contribution of this review is that it focuses solely on source code metrics, which no other secondary study found does. It involves all four currently measured paradigms, and aims to study and analyze source code metrics while taking in consideration the different relevant elements that are part of the source code measurement process, such as the systems measured, programming languages and how metrics are extracted. The findings in this review allow us to establish the state of the art and current trends of source code metrics, and to provide a supplementary website containing relevant information about the topic for researchers and practitioners (information about this website can be found after the conclusions section). This review can be used as a common reference or a starting point for new research. The review is supported by the analysis of 226 primary studies, a quantity of studies not found in any other software metrics related secondary study. From the analysis of those studies, the results indicate that the Chidamber and Kemerer metrics, along with lines of code and cyclomatic complexity, are the most studied metrics, and that fault prediction and complexity are recurrent quality topics of interest for the research community. The metrics are extracted automatically using a wide variety of metrics tools, and open source systems written in Java are the most common measured types of systems. Almost all research is based on empirical studies of already defined metrics, only a few studies propose new metrics.

This paper has the following structure. The systematic mapping study process is detailed in Section 2. Results of the study and discussion are presented in Section 3. Related work is presented in Section 4. The threats of validity for this study are discussed in Section 5, and Section 6 presents the conclusions obtained from the review. The appendices presented in the paper are listed as follows: Appendix A - Selected primary studies, Appendix B - Quality assessment and Appendix C - Object oriented metrics.

2. Systematic mapping study

A systematic mapping study is a type of secondary study designed to provide an overview of a research area through classifications and counting research contributions (Petersen et al., 2015). This is achieved by answering broad research questions that identify the state of the art in the research area. A systematic literature review, another type of secondary study, answers particular research questions that require an in-depth analysis. The systematic mapping study was chosen since our goal is to answer broad questions about the overall domain of the research topic, and not specific aspects of it. The study is performed according to the guidelines for conducting secondary studies proposed by Kitchenham and Charters (2007). According to the authors, a secondary review process consists of three steps: planning, conducting, and reporting the review. In the following sections, each of the three steps applied to our review are presented. Fig. 1 depicts the overall review process with the associated sections as presented in this paper.

2.1. Planning the review

The planning of the review process is described below.

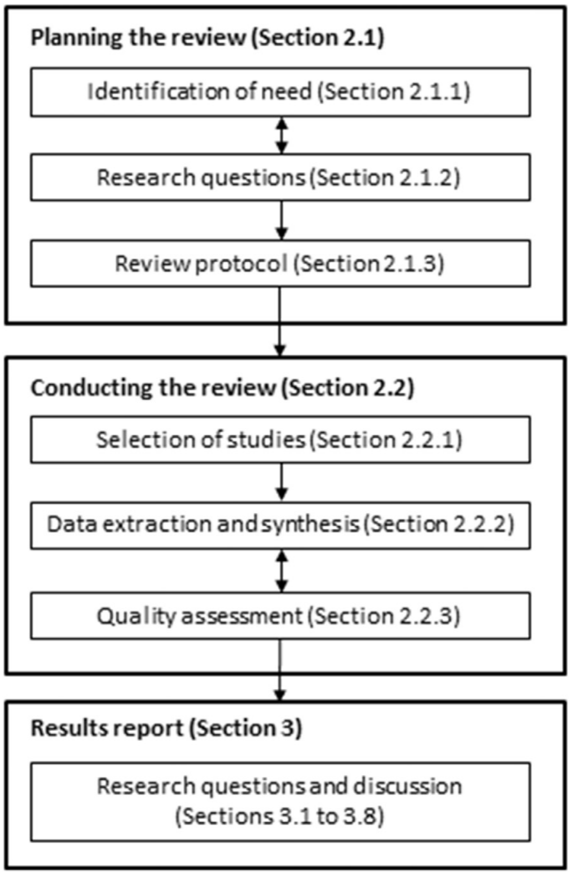
Planning the review (Section 2.1)
Identification of need (Section 2.1.1)
Research questions (Section 2.1.2)
Review protocol (Section 2.1.3)
Conducting the review (Section 2.2)
Selection of studies (Section 2.2.1)
Data extraction and synthesis (Section 2.2.2)
Quality assessment (Section 2.2.3)
Results report (Section 3)
Research questions and discussion (Sections 3.1 to 3.8)

Fig. 1. The systematic mapping study process.

2.1.1. Identification of need

Source code metrics represent an important field in Software Engineering. A large number of papers are published covering definitions, extraction mechanisms, formalization and empirical studies. The number of applications in which source code metrics are used is increasing, and has reached other computing areas such as Artificial Intelligence, Computer Security and Computer Programming, this last area being directly related to source code metrics. What is interesting about source code metrics is the fact that new metrics can be defined according to certain measurement needs, especially with the current interest in new programming paradigms. The new paradigms also increase the number of software quality attributes to be measured. With that many papers published every year, and the large number of metrics in the current literature, it is important to establish the state of the art of source code metrics by gathering evidences from current research. With those evidences, the current trends in source code metrics research can be obtained.

An automatic search was performed with the search string "*metrics literature review*" in order to find similar secondary studies about the topic. No studies were found with a similar scope and time period as the review proposed in this paper.

2.1.2. Research questions

The research questions defined for our study are presented in Table 1.

2.1.3. Review protocol

In this section the review protocol defined for our study is presented. We describe the search design and data extraction.