# Learning through analysis of coding practices in FLOSS projects

Jonas Gamalielsson, Alexander Grahn and Björn Lundell

University of Skövde, Skövde, Sweden,
{jonas.gamalielsson,alexander.grahn,bjorn.lundell}@his.se

**Abstract.** In recent years there has been increased interest in education related issues in FLOSS research. Through interaction with FLOSS projects that are deployed in a variety of usage contexts, students get unique opportunities for learning about development practices. In line with this, we propose an approach for analysing the relationship between coding policy and coding practice in FLOSS projects, intended to be part an assignment in a FLOSS development course. More specifically, the approach focuses on adherence to coding standards with respect to code commenting. The approach is demonstrated and applied on the PHP-based CMS tools Wordpress, Joomla, and Drupal, which are all provided as FLOSS projects.

## 1 Introduction

This paper draws from experience of offering FLOSS (Free/Libre Open Source Software, hereafter referred to as OSS (Open Source Software)) courses, and relates to project courses in OSS development aimed to be offered both at undergraduate as well as advanced level, and offered in a variety of different contexts including campus course, distance course, and contract teaching in professional organisations. In particular, it addresses an assignment with a focus on coding policy and practice in larger OSS projects.

In recent years there has been increased interest in education related issues in OSS research as evidenced by a wide variety of studies on various topics such as undergraduate research opportunities in OSS (Boldyreff et al., 2009), learning through practical involvement in OSS (Berdou, 2007; Kilamo et al. 2010; Lundell et al., 2007), teaching experiences in OSS courses (German, 2005), and learning through mining of OSS project metadata (Squire and Duvall, 2009). It has been claimed that literature aimed at teaching software engineering theory often use toy examples, and that “we need to find innovative ways of integrating project work in curricula” (Ghezzi and Mandrioli, 2005). Therefore, in courses on OSS development it would be important for students to increase their understanding of and skills regarding project work in large OSS projects. One aspect to consider in such projects is coding practices. By studying coding practices in large OSS projects, students can: 1) learn how to characterise coding practices, 2) learn about actual coding practices and how

practice relates to coding policies, and 3) learn how to best contribute to OSS projects in terms of coding practices.

As part of an assignment in OSS development courses we propose an approach for analysis of coding practices. In particular, the approach addresses practices in code commenting. Comments are part of the documentation of OSS projects, and are interesting to study since it has been noted that improved documentation can contribute to increased participation in an OSS project (Mockus et al., 2002). Further, the lack of documentation and updated documentation is a problem in many OSS projects, and one reason for this is that developers are often not required to provide projects with documentation (Levesque, 2004). Although many projects have policies in the form of coding standards, which contributors are expected to adhere to, there is limited research on the actual adherence to coding standards in large OSS projects.

Earlier studies have explored the growth of documentation and code over time in OSS projects (Fluri et al. 2007; Jiang and Hassan, 2006; Schreck et al., 2007). Further, the comment density of code in OSS projects has been explored (Arafat and Riehle, 2009; Elish and Offut, 2002). We note that there is currently a lack of research on how code commenting practices relate to coding standards. One exception to this is a limited study on how Java classes adhere to certain standard coding practices (Elish and Offut, 2002). Our proposed approach is more comprehensive in that it is based on the actual guidelines of the projects and that it provides detailed information about the occurrence of different kinds of coding errors over time.

## 2 Research approach

To detect the number of violations of a defined coding standard with respect to code commenting, PHP_CodeSniffer[1] was used in combination with a custom script, which collects the last revision each month from the SCM (Software Configuration Management system) of an OSS project. A custom standard was created to be used by PHP_CodeSniffer, and this was based on the current coding standards for a project.

To demonstrate the approach we decided to apply it on the three PHP-based CMS tools WordPress[2], Joomla[3], and Drupal[4], which are all provided as OSS projects. These tools were chosen since they are the three most used open CMS tools (Shreves, 2010), and that they are deployed in a variety of usage contexts for important systems in private and public sector organisations. The project data in the SCM repositories and current coding standards were collected on 1 June 2012 for the three tools from the locations stated in table 1. All files with a ".php" extension were analysed for each of the three tools. Further, in addition to quantitative processing of project data we recognise that students using the approach in a course context will scrutinise a variety of additional data sources (e.g. forums, mailing lists, documentation, blogs, and other sources related to the project being analysed), which promotes a more in-depth learning experience.

[1] http://pear.php.net/package/PHP_CodeSniffer
[2] http://wordpress.org/
[3] http://www.joomla.org/
[4] http://drupal.org/

**Table 1.** Location of SCM repositories and coding standards

| Tool | SCM | Coding standard |
|---|---|---|
| Wordpress | http://core.svn.wordpress.org/trunk/ | http://codex.wordpress.org/WordPress_Coding_Standards |
| Joomla | http://joomlacode.org/svn/joomla/development/trunk/ | http://docs.joomla.org/Coding_style_and_standards |
| Drupal | git://git.drupal.org/project/drupal | http://drupal.org/coding-standards |

## 3 Results

To demonstrate the proposed approach for analysis of coding practices, we here show results from the application to three OSS projects. Figure 1 provides an overview of the degree of adherence to the coding standards for the Wordpress, Joomla and Drupal projects by showing the average number of commenting errors per file for different revisions from the start of each project until end of May 2012. It can be observed that there is an increasing error rate for Wordpress (green trace in Figure 1) until the beginning of 2010, when error rate begins to drop. Further, it can be noted that the long-term trend in Joomla (red trace) is a decreasing error rate. Drupal (black trace) exhibits a more fluctuating error rate with a notable peak in early 2011. There may be various reasons for the variations in error rate such as external events affecting a project and changes in working practice within a project. As an example, we conjecture that the peaks in mid 2008 and early 2011 for Drupal may be related to the start of work on Drupal versions 7 and 8, respectively.

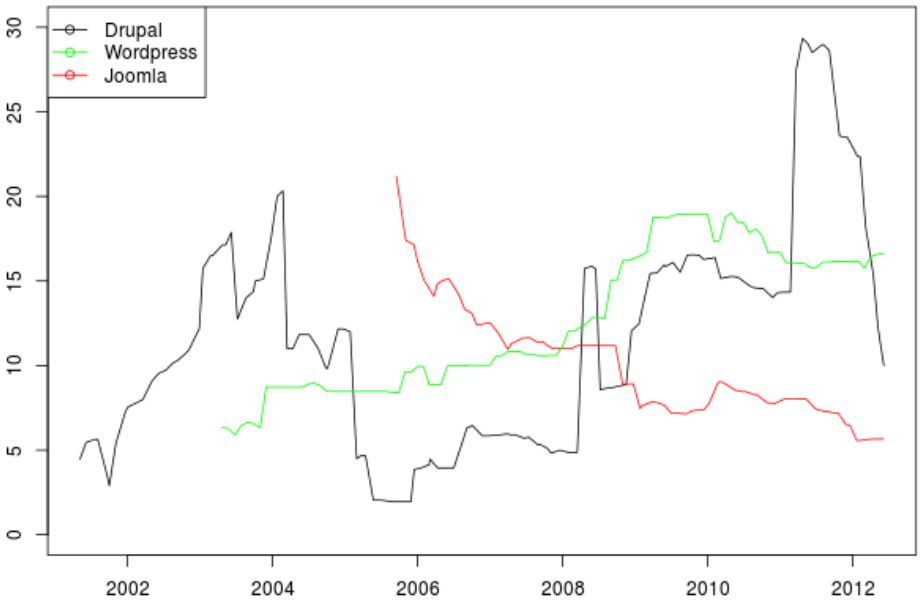


**Fig 1.** Average number of commenting errors per file over time

Table 2 shows the 12 most frequently occurring types of commenting errors in the Wordpress project over all revisions since the start of the project. The most common error is missing function comment, followed by missing tag in file comment, which together comprise 38.6% of all errors. For Joomla and Drupal the corresponding two most frequently occurring types of errors are related to incorrectly written function comments (representing 34,8% for Joomla and 38,0% for Drupal), rather than missing function comments or missing file comment tags as for Wordpress.

**Table 2.** Most frequently occurring types of commenting errors in Wordpress

| % | Type | Description |
|---|---|---|
| 21,4 | FunctionComment.Missing | The function definition is not documented |
| 17,2 | FileComment.MissingTag | The file doc block comment is missing a required tag |
| 13.3 | FunctionComment.WrongStyle | The function comment is written with the wrong comment-style (e.g. // instead of /**) |
| 11,6 | FunctionComment.MissingReturn | No @return tag in the function comment |
| 7,4 | Class.MissingTag | The class definition is not documented |
| 6,2 | Function.MissingParamComment | There is a empty comment for the parameter |
| 5,0 | Function.MissingParamTag | There is no comment for the parameter |
| 3,7 | Function.MissingVersion | Missing PHP version in the file comment |
| 2,0 | FileComment.Missing | There is no comment that documents the file |
| 2,0 | FunctionComment.ParamNameNoMatch | The name used in the param does not match the actual name of the parameter in the code |
| 1,0 | ClassComment.Missing | There is no comment that documents the class |
| 0,8 | FileComment.WrongStyle | Invalid type of file comment (e.g. // instead of /**) |
| 8,4 | *15 other types of errors* | |

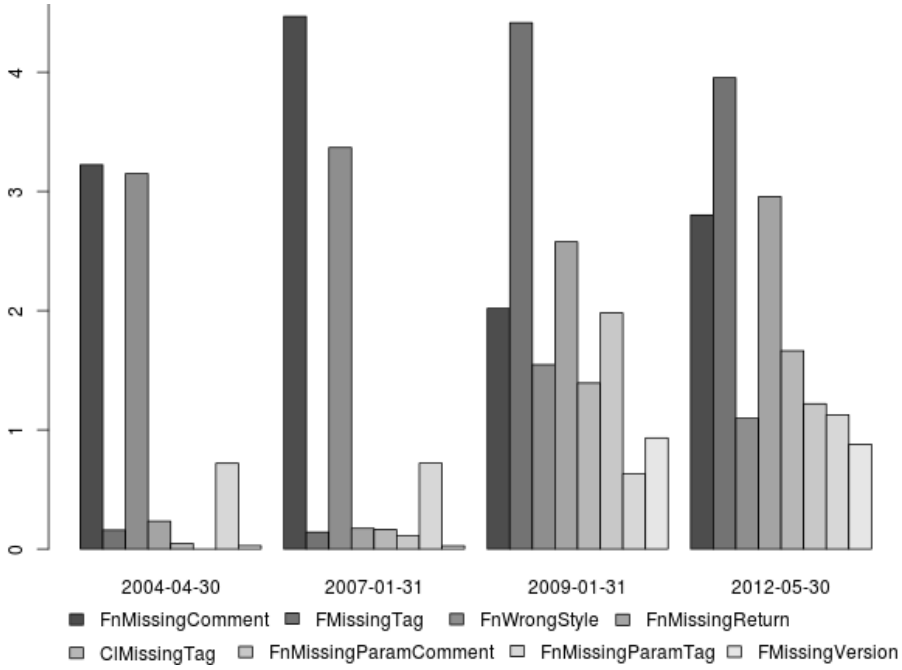


**Figure 2.** Error rate for different error types over different revisions of Wordpress

Figure 2 shows the error rate for the eight most common types of errors in table 2 for four specific revisions of Wordpress as a characterisation of changes in working practice with respect to code commenting. Missing function comments and wrong commenting style were the most common errors during the first two revisions in Figure 2, whereas missing tag in file comment and missing return tag in function comments are the dominating error types for the two later revisions. We note that a larger number of error types significantly contribute for the two later revisions compared to the two first revisions in Figure 2.

## 4 Conclusion & discussion

In this paper we have proposed an approach for analysis of coding practices in OSS projects as part of an assignment in an OSS development project course. Through use of such an approach students will be exposed to coding practices and can gain valuable insights from large and widely deployed OSS projects. The approach was demonstrated by applying it to the Wordpress, Joomla and Drupal projects. The focus was on practices regarding commenting, and a characterisation of the adherence to the coding standards of the three projects was presented. Although the focus was on commenting, the approach can easily be extended to cover all relevant aspects of coding standards, which would further promote learning about coding practices in the context of OSS development project courses.

There are different views on the need for code commenting in communities of different OSS projects. For example, some contributors in the Wordpress community advocate use of commenting practices whereas others find it unnecessary. It has for example been claimed that the “inline documentation effort is headed for failure unless all of the core developers understand that inline documentation is not only important, but required. Without it, you have a situation, where some of the code has inline documentation and most doesn’t.” (Santos, 2008), whereas others claim that comments “are a good way to help a new developer learn the internals” (Merrill, 2006). On the other hand, it has also been claimed that it is “good to have standards. It's not good to adhere to them too rigidly” (Wood, 2009). Further, Santos (2008) note that you “can’t force anyone to do anything in an open source community. Enough people do great things that I doubt inline documentation is a major boon or thorn to anyone. Depressing, but I’ll rather be coding myself thank you”. However, it should be noted that adherence to coding standards may be dependent on an individual’s motivation for participation. In addition to volunteer based code contributions, a substantial amount of contributions origin from professionals employed in commercial companies, which motivates analysis of different types of OSS projects.

The approach can be used in different types of course contexts, including campus courses, distance courses, and contract teaching scenarios. Especially for the two latter, the proposed approach may be particularly interesting from a life-long learning perspective. Results from previous research on professionals and their involvement in Open Source show that almost all participating in OSS development projects “cited skills development as an important outcome of participating” (Lundell et al., 2010), and that such skill development “happened through both detailed scrutiny of other

people's contributions and the rigours of writing and exposing their own contributions to scrutiny" (Lundell et al., 2010). With the availability of mature and widely deployed OSS projects, organisations and individuals involved in education obtain new opportunities on how to gain insights into development practices used in a variety of large and mission-critical systems. This in turn imposes new challenges for any organisation involved in offering courses in order to adapt to evolving needs for life-long learning as a long-term strategy for promoting increased innovation. With demands for increased flexibility in how courses are organised and conducted, we suggest that the proposed approach has an important role to play.

## References


Arafat, O., Riehle, D.: The Commenting Practice of Open Source. Companion to the Proc. of the 22nd Conference on Object Oriented Programming Systems, Languages, and Applications (OOPSLA Onward! 2009), ACM press, pp. 857-864 (2009)

Berdou, E.: Learning and the imperative of production in Free/Open Source development. In Feller et al. (Eds.), Open Source Development, Adoption and Innovation, Springer, Berlin, pp. 235-240 (2007)

Boldyreff, C., Capiluppi, A., Knowles, T., Munro, J.: Undergraduate Research Opportunities in OSS. In Boldyreff et al. (Eds.): OSS 2009, IFIP AICT 299, pp. 340-350 (2009)

Elish, M., Offutt, J.: The Adherence of Open Source Java Programmers to Standard Coding Practices. In Proc. of the 6th IASTED International Conference Software Engineering and Applications, pp. 193-198 (2002)

Fluri, B., Wursch, M., Harall, G.: Do Code and Comments Co-Evolve? On the Relation Between Source Code and Comment Changes. In Proc. of the 14th Working Conference on Reverse Engineering Software Quality Journal (WCRE 2007), pp. 70-79 (2007)

German, D.: Experiences teaching a graduate course in Open Source Software Engineering, In Scotto, M. and Succi, G. (Eds.) Proceedings of the First International Conference on Open Source Systems, Genova, Italy, 11-15 July 2005, pp. 326-328 (2005)

Ghezzi, C., Mandrioli, D.: The Challenges of Software Engineering Education. In Proc. of the 27th international conference on Software Engineering (ICSE 2005), 15-21 May 2005, St. Louis, Missouri, USA, pp. 637-638 (2005)

Jiang, Z. M., Hassan, A. E.: Examining the evolution of code comments in PostgreSQL. In Proc. of the 2006 International Workshop on Mining Software Repositories (MSR 2006), pp. 179–180 (2006)

Kilamo, T.: The Community Game: Learning Open Source Development Through Participatory Exercise. In Proc. of AMT 2010. Tampere, Finland, October 2010, ACM Press (2010)

Levesque, M.: Fundamental issues with open source software development. First Monday 9(4) (2004)

Lundell, B., Lings, B., Lindqvist, E.: Open source in Swedish companies: where are we? Information Systems Journal 20(6), 519-535 (2010)

Lundell, B., Persson, A., Lings, B.: Learning Through Practical Involvement in the OSS Ecosystem: Experiences from a Masters Assignment. In Feller et al. (Eds.), Open Source Development, Adoption and Innovation, Springer, Berlin, pp. 289-294 (2007)

Merrill, S.: [wp-hackers] Inline Documentation, http://lists.automattic.com/pipermail/wp-hackers/2006-February/004936.html, 15 February 2006, accessed 13 June 2012 (2006)

Mockus, A., Fielding, R.T., Herbsleb, J.D.: Two case studies of open source software development: Apache and Mozilla. ACM Transactions on Software Engineering and Methodology 11(3), 309-346 (2002)
Santos, J.: [wp-hackers] Inline Documentation Effort was a Failure, http://lists.automattic.com/pipermail/wp-hackers/2008-May/020214.html, 22 May 2008, accessed 13 June 2012 (2008)
Schreck, D., Dallmeier, V., Zimmermann, T.: How documentation evolves over time. In Proc. of the Ninth International Workshop on Principles of Software Evolution (IWPSE 2007), pp. 4-10 (2007)
Shreves, R.: Open Source Market share report, http://www.waterandstone.com/downloads/2011OSCMSMarketShareReport.pdf, accessed 13 June 2012 (2011)
Squire, M., Duvall, S.: Using FLOSS Project Metadata in the Undergraduate Classroom. In Boldyreff et al. (Eds.): OSS 2009, IFIP AICT 299, pp. 330-339 (2009)
Wood, S.: Wordpress Coding Standards, http://www.wptavern.com/forum/general-wordpress/1121-wordpress-coding-standards.html, 17 December 2009, accessed 13 June 2012 (2009)