Chapter XXI
Evaluating Open Source Software through Prototyping

Ralf Carbon
Fraunhofer Institute for Experimental Software Engineering (IESE), Germany
Software Engineering: Processes and Measurement Research Group, Germany

Marcus Ciolkowski
Fraunhofer Institute for Experimental Software Engineering (IESE), Germany
Software Engineering: Processes and Measurement Research Group, Germany

Jens Heidrich
Fraunhofer Institute for Experimental Software Engineering (IESE), Germany
Software Engineering: Processes and Measurement Research Group, Germany

Isabel John
Fraunhofer Institute for Experimental Software Engineering (IESE), Germany

Dirk Muthig
Fraunhofer Institute for Experimental Software Engineering (IESE), Germany

ABSTRACT

The increasing number of high quality open source software (OSS) components lets industrial organizations seriously consider integrating them into their software solutions for critical business cases. But thorough considerations have to be undertaken to choose the "right" OSS component for a specific business case. OSS components need to fulfill specific functional and non-functional requirements, must fit into a planned architecture, and must comply with context factors in a specific environment. This chapter introduces a prototyping approach to evaluate OSS components. The prototyping approach provides decision makers with context-specific evaluation results and a prototype for demonstration purposes. The approach can be used by industrial organizations to decide on the feasibility of OSS components in their concrete business cases. We present one of the industrial case studies we conducted in a practical course at the University of Kaiserslautern to demonstrate the application of our approach in practice. This case study shows that even inexperienced developers like students can produce valuable evaluation results for an industrial customer that wants to use open source components.

EVALUATING OPEN SOURCE SOFTWARE THROUGH PROTOTYPING

There is an increasing number of open source software (OSS) projects that release software components which provide almost complete sets of functionality required in particular domains. These components are often also of such high quality that in more and more cases industry is seriously considering to use them as part of their commercial products. In such scenarios, OSS components must certainly compete with any similar component on the market including other OSS projects and commercial solutions.

The model behind OSS is generally more attractive to companies than commercial business models, especially for small and medium-sized companies, due to the free distribution of OSS, the full access to sources and documentation, as well as quick responses and support by the community consisting of developers and other users. The implementation of this OSS model and the quality of the software, however, varies significantly from one OSS project to another. Hence, it is crucial for an organization to systematically investigate the implementation of the OSS model for the particular OSS projects whose software it considers to reuse.

Reusability of any type of software (including OSS, in particular) depends on the quality of the software itself as well as that of its documentation. Code quality is affected, for example, by code comments, structuring, coding style, and so forth. The quality of available documentation is defined by its readability, comprehensibility, or technical quality, and by its suitability for the intended reuse scenarios involving OSS. Besides documentation, the community supporting particular OSS projects is a crucial element, too. Response time and quality of community feedback depend on the overall size of the user group and the skill level of its members. All these aspects should be explicitly evaluated before an OSS is reused in real projects, let alone in critical projects. Note that all of these aspects may not only vary significantly from one OSS project to another, but also heavily depend on the concrete context and reuse scenarios of the OSS.

This chapter reports on a way to evaluate OSS in a holistic way. That is, OSS components are firstly evaluated like any other potential COTS (commercial off-the-shelf) component; and secondly they are used in a prototype project similar to, but smaller than the intended product developments, including an evaluation of the product in the context of the projected architecture to avoid architectural mismatches, as well as an evaluation of the support provided by the related community. To minimize the costs of such a pre-project evaluation, an evaluation team consisting of a group of graduate computer science students may be deployed. A prototyping approach can also be used to gather more detailed information on the adequacy of COTS components for a specific context. But especially for the selection of OSS components a prototyping approach pays off. The quality of the source code and the development documentation can be evaluated, for instance. This increases trust in the quality of the component. Furthermore, it is even possible to evaluate if the OSS component can be easily adapted by oneself to better fulfill the specific requirements.

The chapter presents experience from several OSS evaluation projects performed during the last few years in the context of a one-semester practical course on software engineering at the University of Kaiserslautern. The systematic and sound evaluation was supported by researchers of the Fraunhofer Institute for Experimental Software Engineering (IESE).

Each evaluation employed a temporary team of students to conduct a feasibility study, that is, realizing a prototypical solution based on OSS to be evaluated as specified by industrial stakeholders. The industrial stakeholder always provided a set of functional and quality requirements and a projected architecture for the envisioned prod-