Mining Instantiation Patterns for Java Framework Applications

Yunior Pacheco
Vrije Universiteit Brussel
Brussels, Belgium
ypacheco@vub.be

Ahmed Zerouali
Vrije Universiteit Brussel
Brussels, Belgium
azeroual@vub.be

Coen De Roover
Vrije Universiteit Brussel
Brussels, Belgium
cderoove@vub.be

***Abstract*—A major obstacle to framework reuse is the extent of the material that must be understood in order to adhere to and implement the necessary framework instantiation actions. Consequently, several techniques have been developed to automate the extraction of these actions from source code to better understand them and eventually implement them in appropriate ways. However, there is still a steep learning curve when extracting the interplay between them as it is often left implicit.**

We therefore propose an approach to mining framework *instantiation patterns* from existing client systems. Our approach leverages a graph-based representation to capture the common ways of implementing instantiation actions as well as their interplays, so called *instantiation associations*.

This presentation abstract reports on the results of a preliminary study in which we mined instantiation patterns on a set of Java projects and analyzed the prevalence of instantiation associations.

***Index Terms*—Framework Instantiation, Associations**

I. Introduction

The process in which a framework is tailored to application-specific requirements is called framework instantiation. Instantiation actions usually consist of subclassing, implementing interfaces, overriding methods, and calling framework super methods. It is important to learn how to correctly implement these instantiation actions to obtain productivity increases in software development processes. However, many frameworks contain hundreds or thousands of classes and are often incomplete, containing abstract classes and using design patterns to create flexibility. The larger and more sophisticated the library or framework, the more difficult this challenge will be, due to specific requirements and relationships between its components that are sometimes difficult to unravel. Existing approaches [1]–[3] address this problem by providing information about instantiation actions extracted from the source code of client applications. However, these approaches do not yet provide explicit information about the potential interplay between instantiation actions and the dependencies among them, nor concrete code examples that could help understand the full process of framework instantiation. This presentation abstract therefore presents the ongoing work on our approach for discovering framework instantiation patterns and their possible interplays.

II. Overview of the Approach

Our approach for mining instantiation patterns consists of two components: a source code importer and a pattern miner. The source code importer identifies instantiation actions from the source code of existing projects that instantiate the framework. Then, we create framework instantiation graphs which show how an instantiation action is implemented and how it relates to other actions. The pattern miner mines the *instantiation patterns* which are the frequent subgraph patterns in these graphs. The ultimate goal is to store the mined instantiation patterns and feed them to a recommender component, which is part of future work.

Preliminary results. We conducted a preliminary study in which we mined instantiation patterns from a corpus of projects that use tree popular Java frameworks: JavaFX, PLAY, GWT.

The results show that there is a frequent interplay between the mined patterns. The captured interplays constitute relevant information that could reduce the effort required by developers that go through a similar framework instantiation process. The proposed approach is able to find and identify instantiation patterns and the most important interplays between them. It constitutes the first automated technique for extracting these interplays that, due to their importance, should be studied in greater depth.

III. Conclusion

This paper presents a data-driven approach to assisting developers in the instantiation of frameworks. Our approach is based on the concepts of framework instantiation patterns and their interplays. We use data mining techniques to uncover them from existing clients of the framework. Our results will help developers understand the various instantiation actions that are required.

References

[1] Y. Pacheco, J. De Bleser, T. Molderez, D. Di Nucci, W. De Meuter, and C. De Roover, "Mining Scala Framework Extensions for Recommendation Patterns," in *SANER 2019 - Proceedings of the 2019 IEEE 26th International Conference on Software Analysis, Evolution, and Reengineering*, 2019.

[2] M. Asaduzzaman, C. K. Roy, K. A. Schneider, and D. Hou, "Recommending framework extension examples," in *ICSME*, 2017.

[3] M. Bruch, M. Mezini, and M. Monperrus, "Mining subclassing directives to improve framework reuse," in *MSR*, 2010.