Your Thesis title

Your Thesis subtitle

your name, your email

your University

Abstract. Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Keywords:

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

1 Introduction

Over time the software artifacts tends to change, to evolute, with time new requirements appear, the existing ones change, new bugs are found or some shiny new features are added. Because of the changes the artifact starts to drift apart from it's original design in order to incorporate all those modifications. Those modifications increases the complexity of the artifact and then make it less readable, more difficult to extend and harder to change, consequently making the quality lower and the maintenance costs higher.

In order to reverse or mitigate the decreasing quality of the software, programmers tend to improve the structure of the software by making it more readable and easy to understand, and consequently improving the software quality, in order words, programmers try to improve the quality of the software by refactoring the software.

Refactoring and restructuring are transformations that meant to improve the program structure without modifying the behavior. Maintaining the behavior is important because without that the modification would modify the meaning of the program and transform in another program. The difference between Refactoring and restructuring is that Refactoring is used in literature to define the transformations that improve the program preserving the behavior in Object Oriented paradigm [**?**] [**?**] whereas Restructuring is used for the rest. [**?**] [**?**] Because of refactoring and restructuring are tedious and error prone, it is preferable to use a tool that provides automated support to the refactoring operations that the programmer intends to do and therefore saving time and reducing the possibility of adding errors to a previous correct program.

There are refactoring tools for the different types of languages paradigms such as object oriented, functional, imperative, however the use of those tools were fully adopted by the object oriented and static programming languages with their IDE support, such as Java with eclipse, and InteliJ and com-

paring to the dynamic languages where there is a lack of refactoring tools. One difference of refactoring tools for the dynamic languages is the lack of information available in compile time about the program being refactored when compared to the static typified languages.

Despite of that, the dynamic languages are growing very fast within the scientific community and there are new dynamic languages everyday, they are good for creating fast prototypes because it is easy to see things moving and because of that they are easy to adopt and often used as a learning language for unexperienced users, such as Scheme/Racket and Python.

Because of the lack of restructuring operations for dynamic languages and for learning languages and environments the purpose of this paper is to propose restructuring operations that would be applied to an learning environment of a learning language.

In order to do that we propose a implementation of a restructuring tool for Racket and Python for DrRacket where there are a lack of restructuring operations.

Racket [1] programming language is a dialect of lisp and a descendant of Scheme and support objects, types and laziness evaluation, whereas Python is a very high-level programming language that supports the imperative, functional and object-oriented programming paradigms and is very popular in many areas.

The restructuring will be implemented in the DrRacket, DrRacket is an integrated development environment (IDE), that was formerly known as DrScheme. It is a simple IDE that was initially build for Racket programming language and it is aimed at inexperienced programmers. (**Fig. ??**). It was designed as a pedagogic environment [**?**] and it was used in many introductory programming courses in schools around the world. In addition to that DrRacket supports development and extension of other programming languages [**?**] and recently it have an implementation of python. [**?**]

The Section 2 addresses the objectives for this thesis work. Section 3 explore related work in refactoring and restructuring programs, some implemented restructuring tools and some implementations of language independent refactoring tools. Section 4 describes the architecture of the proposed solution. Section 5 explains how the tool will be evaluated and we conclude on section 6.

2 Objectives (1pg)

Clearly explain the project objectives.

3 Related Work

4 Architecture (2/3pgs)

Your proposed architecture. Can have lots of pictures and bullet points so it is easy to understand.

5 Evaluation (1/2pgs)

Explain how you are going to show your results (statistical data, cpu performance etc). Answer the following questions:

[1] http://racket-lang.org/

– Why is this solution going to be better than others.
– How am I going to defend that it is better.

6 Conclusions

Wrap up what you wrote.