Grading Ecosystem – Maycamp Arena Integration



# Introduction

The purpose of this document is to outline the integration of the grading ecosystem with the Maycamp Arena grader. Currently Maycamp Arena supports C/C++ as programming languages being judged. All problems are being graded in IOI style meaning that each individual test is being graded (accepted/wrong solution, runtime error during a test execution, time limit during a test execution). The idea of the project is to provide a grading system for high school students informatics competitions in IOI style. Initially it was a spoj0 fork but was completely rewritten in ruby – hence the architecture of the system follows a similar approach to the one used for the design of spoj0.

# Arena Maycamp Design

## Data Model

## High Level Design

The system contains two main components:

* web part – a rails application;
* grader - a ruby class that compiles the programs with g++ and runs them using a ruby runner, which observes the resources they use.

The project has the following directory structure:

* **app** -
* **config** – contains the configuration of the system including unicorn.conf.rb that contains the configuration for the Unicorn Rack HTTP server used to start the system
* **coverage** -
* **db** -
* **doc** -
* **ext** -
* **features** -
* **lib** – contains various useful utilities (e.g. compiler wrapper for use by the grader) along with the Rake tasks (defined in the tasks subfolder)
* **public** -
* **script** -
* **spec** – contains RSpec test specifications (tests provided in a BDD manner)
* **test** – contains various tests for the system (for the grader and the various features)
* **vendor** – contains the Ruby gems used by the system in the **cache** subfolder
* **config.ru** – the file is used by Rack servers to start the application
* **Gemfile** – groups the gems (Ruby dependencies) used by the project for different purposes (e.g. production, development);
* **Rakefile** – The Rake build file for the system (the Ruby variant of Makefile)
* **unicorn.ru** – sample configuration of the Unicorn Rack HTTP server over which the system can be deployed (not used)

.erb files (files that are rendered by the eRuby– Embedded Ruby template engine)

ActiveResource provides the necessary features to extend the application with REST-based web services.

# Extensions

# References

[1] Проектът Мейкамп Арена – използване на състезателни системи в обучението

по програмиране\*, Валентин Михов  
<http://www.math.bas.bg/smb/2011_PK/tom/pdf/438-443.pdf>