Developing a common web interface to various verification tools

Project description

Roland Meyer
rmy@student.ethz.ch

March 8, 2012

1 Motivation

Many research groups in computer science develop various tools such as code verifiers, compilers or program analysers. These tools are often not officially released i.e. they are not available in a stable major version since they are under constant development. There may be frequent or unforeseen changes to the APIs, and dependencies and build processes may not be thoroughly documented. This can make it difficult for people interested in using these tools as they have to keep track of the latest version and find out how to build and use it, or they might not even be allowed to install the tool because they don't have a licence for it.

2 Main task

The goal of this Bachelor's thesis is to develop a web service that allows an interested user to find a tool and use it without having to download and install it on his own computer. The user shall be able to simply upload input data through the web service, run the latest version of a tool and see the output. Input and output may consist of arbitrary textual content, e.g. source code. The focus shall be on providing a simple and easy-to-use interface to the user, such that finding and running a tool is just a matter of a few clicks.

Tool developers shall be able to easily include and update their tools in the web service to ensure that users have the latest version of the tools available. The tools are hosted independently of the central server, i.e. they are installed on remote servers and registered at the central server, which only communicates with these tools over a network connection using a predefined protocol. This solves both the installation and the licencing problem as the tools are only ever installed and run on the tool hoster's computer. Tools that are hosted should collect all input and output data and store it locally in the form of simple textfiles.

Security is not a key part of this project but should be considered when designing the network interfaces.

The issue of scalability shall be addressed, i.e. the system shouldn't break down immediately under increased workload.

The web service shall be implemented in the Scala programming language and the implementation shall be well documented and thoroughly tested to facilitate future maintenance.

3 Schedule

The following is a list of steps of the project and the estimated amount of effort needed to complete them:

- Learn and understand the Scala programming language (6%)
- Find and evaluate suitable web frameworks (6%)
- Architectural design for client and server (16%)
- Layout design of the user interface (16%)
- Implementation and testing (40%)
- (Extentions)
- Final report (16%)

4 Extentions

Depending on the progress of the project and personal preferences, these are some of the possible extentions of the project:

- To help developers refine their tools, input and output could be logged (on the central server).
- Users could have the possibility to write messages to the developers if they encounter bugs.
- Tool hosters can specify a set of options to their tools that is presented by the web interface in appropriate ways, e.g. using text fields, checkboxes or dropdown lists.
- Users could have the possibility to create permanent links to a certain tool for a certain input which they could store or send to others.