Distributed testing sessions for AutoTest

PROJECT PLAN

Master's project

Project period	10.03.2014 - 09.09.2014
Student name	Victorien Elvinger *(External Student)*
Status	6th semester Information Systems and Networks, ESSTIN, France 4th semester Services and Security of the Systems and Networks, University of Lorraine, France
Email address	victorien.elvinger5@etu.univ-lorraine.fr
Supervisor name	Dr. Chris Poskitt with Alexey Kolesnichenko and Yu Pei
Chair	Prof. Dr. Bertrand Meyer (Software Engineering, ETH)

PROJECT DESCRIPTION

Overview

AutoTest [1,2] is an automatic testing framework for the Eiffel programming language. It exploits the presence of *contracts* [3] -- executable preconditions, postconditions, invariants -- to automate the entire testing process, variously using them as built-in *filters* for test inputs and *oracles* for test execution. The framework has evolved over a number of years and PhD theses [4,5,6], has been integrated with a number of test data generation strategies (e.g. [7,8]), and its efficacy demonstrated by automatically unearthing previously undetected bugs in production-level libraries like EiffelBase.

Despite the merits of the system, a possible factor in limiting its effective use is the amount of time required to test an Eiffel program (AutoTest generates only approximately 20 test inputs per minute) -- a factor, which if addressed, will move the system towards realising a long-term aim of automatically providing appropriate feedback *while* the programmer is programming.

A first step towards addressing the performance limitation is to explore the large-scale distribution facilitated by cloud computing; in particular, the possibility of multiple testing sessions executing in parallel across multiple computational resources. For this we would learn from and build upon the related work of other researchers exploiting the cloud in software testing problems (see e.g. [9,10,11]).

Scope of the work

The goal of this project is to extend the AutoTest framework with support for distributed testing sessions across cloud-based resources.

Intended Results

The results include: a survey of related work; an analysis of performance bottlenecks in the AutoTest framework; an extension to the framework with support for distributed testing sessions; an evaluation of the extension with respect to the present version of AutoTest; and a Master's thesis (in English), to be made available on the Chair of Software Engineering's website.

References

[1] Bertrand Meyer et al.
Programs that test themselves
IEEE Computer, vol. 42, no. 9, pages 46-55, September 2009

[2] Chair of Software Engineering
AutoTest: Contract-based random testing tool
http://se.inf.ethz.ch/research/autotest/

[3] Bertrand Meyer
Object-Oriented Software Construction, 2nd edition
Prentice Hall, 1997.

[4] Andreas Leitner
Contract-based tests in the software process and environment
PhD thesis, ETH Zurich, 2008

[5] Ilinca Ciupa
Strategies for random contract-based testing
PhD thesis, ETH Zurich, 2008

[6] Yi Wei
Putting contracts to work for better automated testing and fixing
PhD thesis, ETH Zurich, 2012

[7] Ilinca Ciupa et al.
ARTOO: adaptive random testing for object-oriented software
Proc. International Conference on Software Engineering (ICSE 2008), pages 71-80, ACM, 2008

[8] Yi Wei et al.
Satisfying Test Preconditions through Guided Object Selection
Proc. International Conference on Software Testing, Verification, and Validation (ICST 2010), pages 303-312, IEEE, 2010

[9] Manuel Oriol, Faheem Ullah
YETI on the Cloud
Proc. ICST Workshops 2010

[10] Linda Di Geronimo et al.
A Parallel Genetic Algorithm Based on Hadoop MapReduce for the Automatic Generation of JUnit Test Suites
Proc. International Conference on Software Testing, Verification, and Validation (ICST 2012), pages 785-793, IEEE, 2012

[11] Louis Rose et al.
Towards a scalable cloud platform for search-based probabilistic testing
Proc. International Conference on Software Maintenance (ICSM 2013), pages 480-483, IEEE, 2013

PROJECT MANAGEMENT

Objectives and priorities

The goal of this project is to extend the AutoTest framework with support for distributed testing sessions across cloud-based resources. This will be achieved by:

- a period of familiarisation with the tool and its architecture;
- a study of the literature (with particular attention to papers about distributing aspects of program testing);
- an analysis of the tool to identify and categorise the existing performance bottlenecks;
- extending the framework to support multiple testing sessions across distributed resources;
- developing a mechanism to divide a set of input classes for distributed testing in a way that avoids redundancy; and;
- evaluating the extension against single-instance AutoTest.

The implementation work above should be done on the version of AutoTest in EVE (the research branch of Eiffel Studio).

Criteria for success

The project will be successful if it is able to exploit the cloud for the purpose of improving the throughput of AutoTest. The extension should be well-founded (with respect to related work and the analysis of the existing tool), well-engineered (e.g. submitted to a code review, excellent documentation), and properly evaluated. The project -- from inception to conclusion -- should be documented and evaluated to a high standard in a Master's thesis that adheres to ETH regulations.

Method of work

The student will be working for the majority of the six months in France, with the exception of three week-long visits to the Chair of Software Engineering at ETH Zurich. Apart from these three weeks, the student and supervisors will be in regular (at least weekly) contact via video conferencing, email, and shared documents.

Quality management

The project documentation will consist of the final report (as described above). The extension to AutoTest itself will be well-documented through the use of comments and the Eiffel Information System. It will be evaluated on libraries such as EiffelBase, for which we already have extensive testing data.

PLAN WITH MILESTONES

Project steps

1. Familiarisation with AutoTest and EVE (the research branch of Eiffel Studio)
2. Performance measurements; identifying the bottlenecks
3. Literature search on cloud-based testing; develop a plan for AutoTest accordingly
4. Selection of a cloud platform, and familiarisation
5. Implement the extension of AutoTest for parallel testing sessions
6. Evaluate the extension against production-level programs or libraries
7. Write the final report

Tentative schedule