Next Generation of Unit Testing

Proposal by Dayne Guerra Calle
Mentors: Julien Deplanque, Guillermo Polito

Introduction

Unit testing aims to ensure validity of software by checking that their individual parts work as expected by requirements. It provides confidence that software continue to answer its previous requirements when it has to evolve. Pharo uses a fork of the SUnit testing framework that was integrated in the system years ago, drifting apart in the meantime. Many parts of the Pharo ecosystem rely on Pharo's SUnit to enhance developers experience: the system browser, the command line test runner, the user-interface (UI) exposing testing capabilities, etc. However, as time goes, 2 problems arise:

1. Many projects using unit tests as input came to existence, each of them providing its own UI which is confusing for end-users; and
2. It happens that these tools implemented multiple times the same features inducing code duplication.

The purpose of this proposal is to solve the two problems above.

Motivation

The first problem with the current version of SUnit is that Test Runner is not extensible. Thus, it makes hard to add new features. This UI contains three tools related to test analysis: test runner, test coverage and test profiling. Pharo Community has created many additional tools, MuTalk implementing mutation testing in Pharo, parametrized tests, rotten green tests finder, Comments to test[1] (implemented by myself), etc.

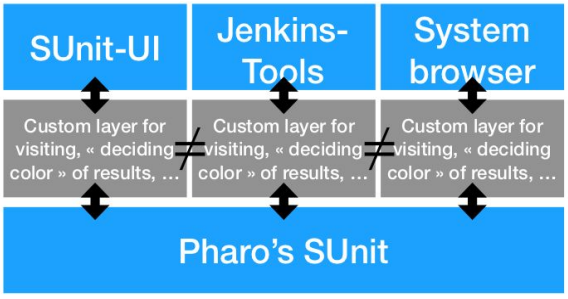
In the current state, these projects are hard to incorporate into the Test Runner. Each tool has its own user interface. Worst, some of them do not provide UI at all which makes their usage hard for newcomers. A common UI that allowing to plug these existing tools and tools of the future is needed.

The second problem is that some extensions of Pharo's version of SUnit created by tools using it were never integrated directly into SUnit project. This fact has two major drawbacks:

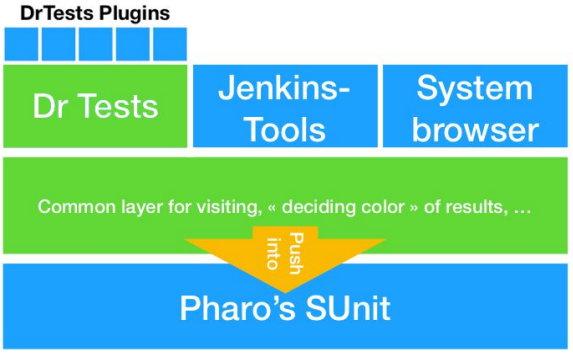
1. if multiple tools needed the same extension, they both implemented their own extension on their side which leads to code duplication (for example, the SUnit UI, Jenkins tools and the system browser define 3 different ways to collect tests defined in a package); and
2. the API to be used in order to manipulate SUnit is not clearly defined. This can lead to the same feature implemented multiple times but the resulting behaviour is slightly

[1] https://github.com/GDayne/DrTestsCommentToTest

different in each version (it is the case for the previous example concerning the collection of tests defined in a package).



The figure above summarize these two problems.



The figure above shows the proposed solution.

Project Goals

Goal: Push DrTests into Pharo and improve Sunit.

1. Enhance DrTests
2. Enhance SUnit first as part of DrTests
3. Merge the part of DrTests to Pharo's SUnit
4. Integrate DrTests in Pharo

Implementation

DrTests enhancements

DrTests[2] is a project started by Julien Deplanque intended to replace the existing TestRunner. We will start by redesigning the model of DrTests to support integration of various analysis tools. It is important to consider that each tool has different input and output. The goal of our model is to be robust enough and offer an easy way to incorporate new features into a single UI.

Before starting to modify the implementation, we need to design a new model capable of dealing with most of the actual analysis tools in Pharo. We want to incorporate these tools as plugins in the UI. DrTests is already capable of running tests and supports Comments to tests as a plugin(The one I did). Now we would like to add tests coverage, MuTalk, rotten

[2] https://github.com/juliendelplanque/DrTests/

tests, and parametrized tests. We will use Spec to build a powerful interface, which will be flexible, easy to understand, and allow us to add new features.

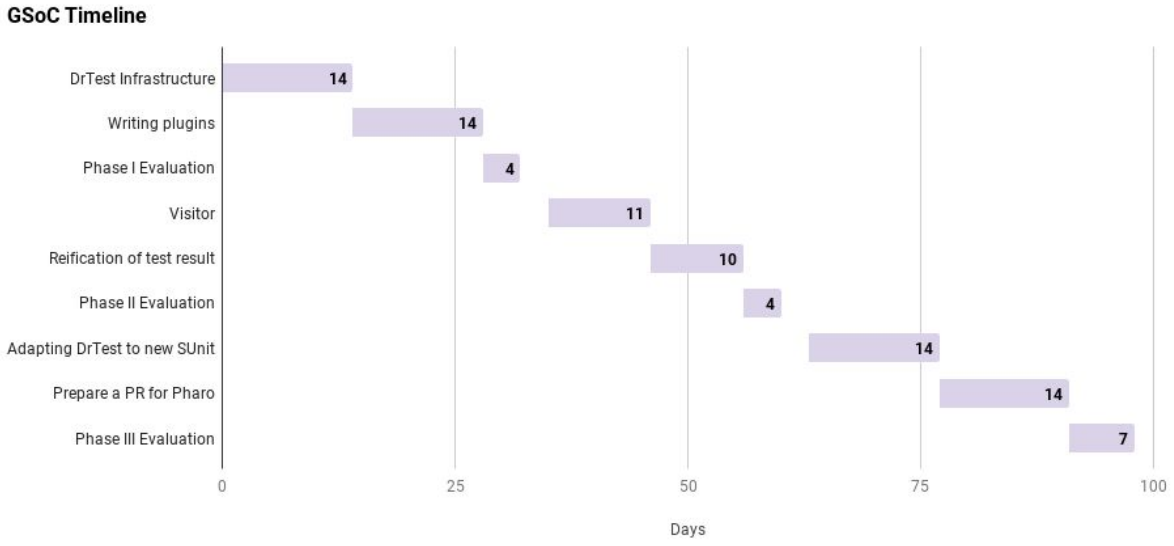
New SUnit Extension Points

Once we have this UI (DrTests) working with new features, we will need to improve Pharo's version of SUnit. We plan to identify the main problems of the infrastructure considering that SUnit is used by the system browser, the SUnit UI and Jenkins-tools, and each one of them has its own way to show the color of results, visit the tests in a package, and store the last execution of a test.

The objective is to have a common layer instead of distincts custom layers. To start, we will implement a visitor[3] for tests. This visitor will allow all users of SUnit to visit tests in the same way. We will also create objects to reify the kind of result a test is. This will allow us to remove duplicated code that decide the color of a test result.

Other similar problems occur because the API of SUnit is not easily extensible. For example, projects as ParametrizedTests and Comments to Test were complicated to plug into SUnit. Multiple methods needed to be rewritten since the hooks required were not available. This is why we need a more extensible SUnit.

Project Timeline



[3] https://en.wikipedia.org/wiki/Visitor_pattern

Phase I (May 27 - June 24) - DrTests Plugins

May 27 - June 10

- For DrTests infrastructure, study different projects:
  the currents in Tests Runner (test coverage and tests profiler)
  the ones that will be added (Mutation tests, rotten green tests finder and parametrized tests).
- Redesign the actual model and implement the new model.
- DrTests stable with tests runner and test coverage
- Write clear documentation

June 10 - June 24

- Understand how works Pharo's mutation testing library MuTalk[4].
- Implement MuTalk as a DrTests plugin.
- Understand how parametrized tests[5] work.
- Implement parametrized tests as DrTests plugin.
- Understand how rotten green tests finder[6] works.
- Implement rotten green tests finder as DrTest plugin.
- Write clear documentation.

Deliverables for the first evaluations

- DrTest working with 5 differents plugins.
- DrTest documentation.

Phase II (July 1 - July 22) - SUnit Extension Points

July 1 - July 12

- Understand SUnit implementation.
- Compare actual Pharo SUnit to latest SUnit in VisualWorks[7]
- Implement visitor pattern in SUnit project to manage the information.
- Write clear documentation.

July 12 - July 22

- Reification of TestResult
- Model the new common layer for SUnit
- Implement the new model first in DrTests
- Write clear documentation

---

[4] https://github.com/pavel-krivanek/mutalk
[5] https://github.com/tesonep/ParametrizedTests
[6] https://github.com/juliendelplanque/RottenTestsFinder
[7] http://sunit.sourceforge.net

Deliverables for the second evaluations

- New common layer for tools that works with SUnit (Jenkins Tools, Test Runner and the browser)
- Other extension points derived from parametrizable tests, MUTalk, rotten green tests
- SUnit documentation

Phase III (July 29 - August 26) - New SUnit Integration

July 29 - August 12

- Refactor DrTests to use the new SUnit layer.
- Refactor Calypso to use the new SUnit layer.

August 12 - August 26

- Prepare pull requests for Pharo.
- Write documentation for the new layer.

Deliverables for the third evaluations

- Upgraded DrTests version that works with the new SUnit layer.
- Pull requests for Pharo main project.

Communication

Blog for this Project

I will be writing one blog post per week documenting my progress and describing the contributions I make. I will continue using my blog on Medium: https://medium.com/@daynestorm.

I choose Medium because of its simplicity and consistent traffic it has and a simple design and easy access and tools for code. I have started with a post of what I did in Comment to test plugin, how was it implemented and a video demo with the plugin working.

For Gsoc I want to write about the progress during the process, maybe add new ideas while I am working on the project or request suggestions when I have questions. Receive feedback for people that is not involucatre in Pharo development but may have opinions about testing in differents languages.

Integration in the Community

for communication I use:
Discord when I have questions about some specific project and want to contact the person involved directly. And since the community is very active in the chats they usually answer very quickly.
Mailing list for ask open questions to everyone, and looking for some information I don't know who to ask. To read opinions from users and devs and receive feedback.

Why do I want to work on this?

Since I started doing contributions in DrTest and SUnit I feel very involved in them.
Because tests are used in almost every project in Pharo I consider having a good implementation and an easy interface essential to work. And to encourage others to use tests in their own projects and develop new features for tests analysis.

I want to continue working with pharo too, I planned for my master's final project work in testing, so my goal is to mix both things.
I choose Testing because I use it in every project and I am a TDD enthusiast.
This is my only proposal and I have the time to work on it. I really would enjoy working on this project, even after I would like to improve or create new things in Pharo.

I've made a plugin for DrTests that recognises executable comments in methods that are used usually to show an example of the method itself. These comments can be executed directly from the code.
The plugin converts these comments into test cases and then execute them. The name of the project is **Comment to test** and I worked on it making a copy of DrTest on a repository I created specially for this plugin:
(https://github.com/GDayne/DrTestsCommentToTest)
And working on my own in DrTest let me see that adding new features for test analysis in the same UI is very useful. And now I want to improve the project.
Besides while I was working on the plugin I started seeing some implementation in SUnit that can be refactored and that some things need to be upgraded.

Benefits to Community

In Pharo almost every project uses SUnit and because of that working in enhancing SUnit and provide a user interface is valuable for everyone.
At the end we will have:

- DrTests as the new UI for test.
- DrTests working with plugins.
- An extensible UI that allows everyone to add new features.
- Introduce new Sunit framework to be used.

Related Work

- **Pharo's SUnit** is the implementation of SUnit[8] but, the current version we have in Pharo is not up to date.
- **Test Runner** is the UI used in Pharo to execute and show the results of tests, it has incorporated tests coverage option and test profiling.
  But this is not extensible and the option are limited.
- **DrTests** project aims to provide a plugin-based UI to deal with tests in Pharo. It will provide the same features as the actual SUnit UI (i.e., running tests, profiling tests and computing code-coverage) but will allow to plug additional analysis on unit tests. The project is still on work.
- **Test Coverage** to measure what percentage of code has been exercised by a test suite. Test runner UI has this option.
- **MuTalk** implementing mutation testing in Pharo, offers an alternative to code coverage regarding test quality.
- **Rotten green tests finder** project finds rotten green tests in your source code. That is to say, tests having call to assert-like methods that is never executed at runtime.
- **Parameterized test** allows to generate different test cases for each of the possible configurations in a matrix.
- **Comments to test** added as a plugin, converts executable comments into test cases. It was implemented by me.

[8] http://sunit.sourceforge.net/

About me

Contact Information

Email	guerra.dayne@gmail.com
Facebook	https://www.facebook.com/Day.nelo
Github	https://github.com/gdayne

Skills

Languages	Pharo, Java, C#, JavaScript etc.

Experience

2019	**Summer School of languages (Smalltalk) |** Universidad Mayor de San Simón
2018	**TRUEXTEND |** Robotics Lab member
2014 - present	**San Simon Scientific Society |** Community member Duties: mentoring, responsible in logistic and event management.
2013 - present	**Women Techmakers Cochabamba|** Community member Duties: teaching, mentoring, meeting and code-lab co-organizer.
2013 - present	**Google Developers Group Cochabamba|** Community member Duties: mentoring and code-lab co-organizer.
2016 - present	**Twitch partner |** Streamer Duties: entertainment, gaming and programming.
2015 (2 semesters)	**Universidad Mayor de San Simón |** Teaching assistant Teaching assembly language

Education

2013 - present	Universidad Mayor de San Simón (Bolivia) Computer science

Achievements

2018	**Latinity Scholar |** Full scholarship Bogotá, Colombia.

2018	**Anita Borg Grace Hopper Scholar |** Full scholarship Houston, Texas.
2018	**ACM-ICPC BOLIVIA |** South regional finalist

Why am I good for this project?

- ✓ Experience with Pharo
- ✓ I have contributed to some projects, including DrTests and SUnit
- ✓ Knowledge of the current status of projects I want to work on
- ✓ Several years of OOP experience
- ✓ Active member in the community