# Towards Quality Assurance Tests for Mobile Robots

Luc Fabresse, Jannik Laval and Noury Bouraqadi
Mines-Telecom Institute, Mines Douai, France
e-mail: firstName.lastName@mines-douai.fr
Web: http://car.mines-douai.fr

**The Need for Quality Assurance.** Developing robots combines the challenge of building hardware and the corresponding software. Even in the case of an existing robotic hardware, the development of control software alone is more challenging than plain software development. Indeed, it introduces the hardware integration constraint and the physical problems. This makes building robots error prone. Besides, mobile robots are supposed to act nearby humans and even sometimes collaborate with them. Thus, Quality Assurance (QA) should be conducted with great care to ensure that the robot will function safely and comply with their specifications. In this paper, we discuss tests for robotics QA. We present our ongoing work as part of the CAIRE project that aims at improving robotic software development. It takes its inspiration from agile methodologies [1] well established in the software engineering community.

**First Tools for Robotics QA Tests.**

We are currently developing BoTest, a tool to support expressing and running robotics tests. It is implemented in the Pharo programming language [2], on top of the SUnit unit-test framework. So far, it provides support for expressing dependencies between tests. This is useful from the safety point of view as well as for quickly identifying the lowest-level failing test. It also guides testers for setting up the initial physical conditions of a test (e.g. robot pose, obstacles, light). Besides, it enables expressing verifications that involve testers such as measuring the distance travelled by a robot.

We are using BoTest to develop the software that controls two *identical* robots we bought. We started by writing tests based on specifications provided by the manufacturer and going up towards our application level. We provide here an example related to the laser rangefinder. We put a robot inside a 1 meter wide cube box (see Figure 1) and check that measured distances are within the expected range. The code below corresponds to the test where the robot is located in the bottom left corner of the box.

Figure 1: Testing Robot's Laser Rangefinder Inside a Box

```
1 testLaserWhenRobotAtBottomLeftOfTheBox
2   | laser notification |
3   self requestAction: 'Please, put the robot at the bottom left of the 1x1 box'.
4   laser := robot laserService.
5   laser enableNotificationsEvery: 10.
6   notification := self notificationOrNilFrom: laser.
7   allDistances := notification distances.
8   self assert: allDistances areLessThan: 0.75 andGreaterThan: 0.25
```

Test starts by requesting the operator to put the robot at the right position (line 3). Then, we trigger the laser service of the robot to publish notifications every 10 clock cycles (lines 4-5). Each notification gathers all measurements performed in a single laser scan. Last, we check that all measured distances are less than 0.75 meter and greater than 0.25 meter (line 8). Running this test allowed us to identify that the laser was partially covered with the robot structure. Indeed, some measured values were smaller than the minimum expected distance.

**Summary and Future Work.** We argue that robotics software development would really benefit from better tool support to improve the definition of QA tests. Such tools should help developers to express the robot acceptance tests. We introduced BoTest, our first step towards this end. It helps developers to write *repeatable* and *reusable* tests as described in [3]. It also helps to execute tests following an order that ensures *safety*.

As part of future work, we investigate the definition of tests during the prototyping stage of robots. This approach is inspired by Test-Driven Development (TDD) [4] agile methodology from the software engineering field. We investigate defining tests incrementally together with robotic software. The development process produces both a software to control a robot and tests that reflect requirements. These tests can be reused by the quality assurance (QA) team to verify that all robots produced by the same product line conform to the requirements.

**Acknowledgements.** This work is supported by Nord-Pas de Calais Regional Council through the CAIRE project (2012-2014).

## References

[1] Jim Highsmith and Martin Fowler. The agile manifesto. *Software Development Magazine*, 9(8):29–30, 2001.

[2] Oscar Nierstrasz, Stephane Ducasse, and Damien Pollet. *Pharo by Example*. Square Bracket Associates, July 2010.

[3] Luc Fabresse, Jannik Laval, and Noury Bouraqadi. Towards test-driven development for mobile robots. In D . Brugali, N . Hochgeschwender, and R . Philippsen, editors, *Proceedings of the ICRA 2013 Workshop on Software Development and Integration in Robotics (SDIR VIII)*, 2013.

[4] Kent Beck. *Extreme Programming Explained*. Addison-Wesley, 2001.