

Differences in automated testing on MeeGo and Android mobile platforms

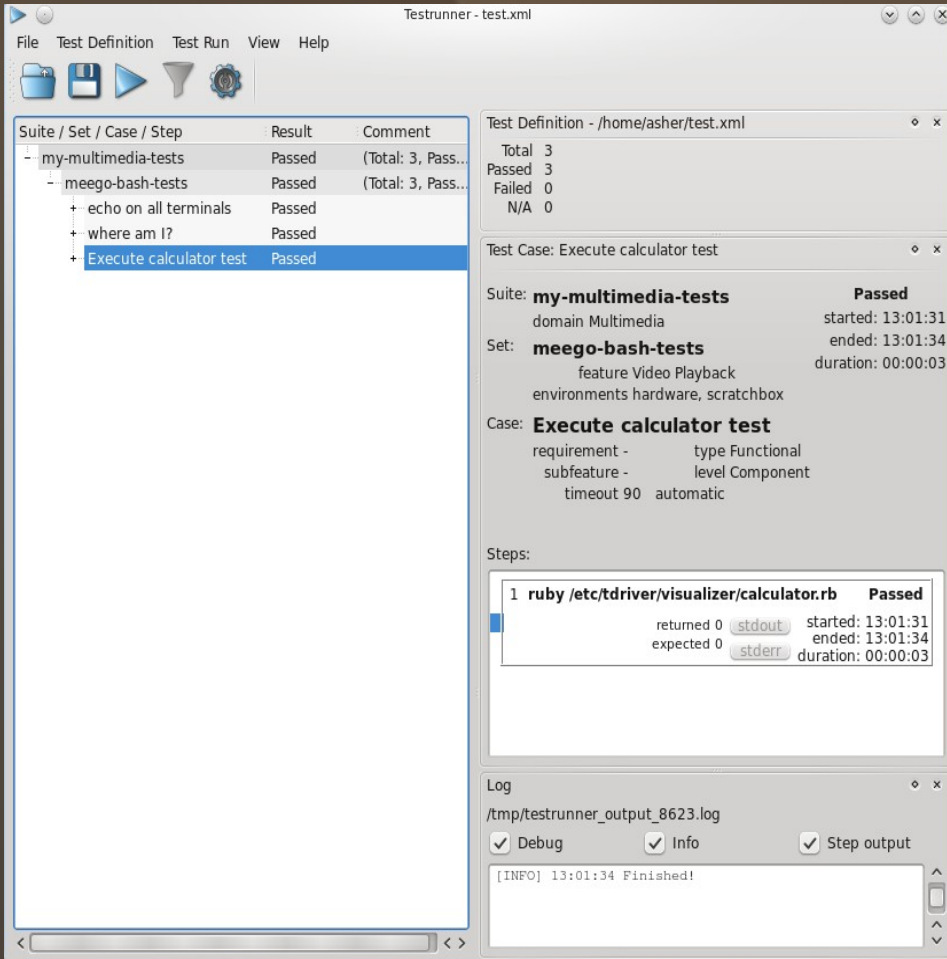
Oleksandr Kachur
Vladimir Sayenko



MeeGo tools

- Testrunner-lite – tool that executes tests on real device.
- Testrunner – GUI front end for testrunner-lite.
- Tdriver – automation test API for QT applications.
- Testplanner – tool for writing and executing test plans

Testrunner



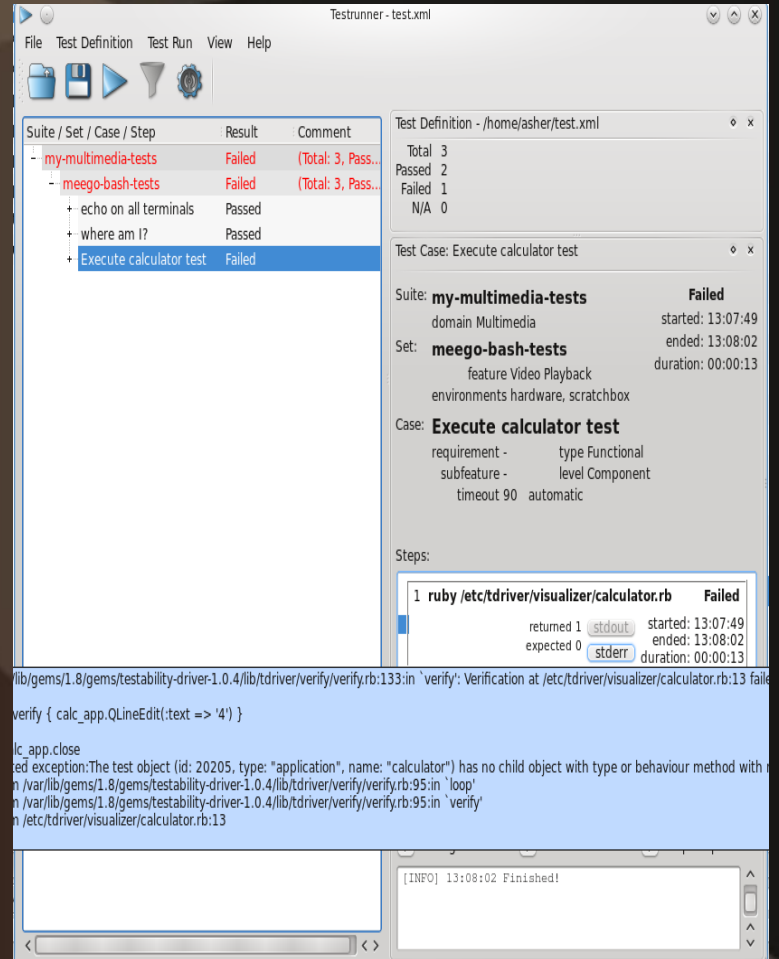
The screenshot shows the Testrunner application window titled "Testrunner - test.xml". The interface includes a menu bar (File, Test Definition, Test Run, View, Help) and a toolbar with icons for file operations and test execution. On the left, a tree view displays the test hierarchy: Suite / Set / Case / Step. The selected test case is "Execute calculator test" under the "meego-bash-tests" set, which is part of the "my-multimedia-tests" suite. The results table shows all tests passed.

Suite / Set / Case / Step	Result	Comment
my-multimedia-tests	Passed	(Total: 3, Pass...
meego-bash-tests	Passed	(Total: 3, Pass...
+ echo on all terminals	Passed	
+ where am I?	Passed	
+ Execute calculator test	Passed	

The main panel displays the test definition for the selected case: "Execute calculator test". It shows the suite name "my-multimedia-tests" (Passed), the set name "meego-bash-tests", and the case name "Execute calculator test". The steps section shows a single step: "1 ruby /etc/driver/visualizer/calculator.rb" which passed.

```
1 ruby /etc/driver/visualizer/calculator.rb Passed
   returned 0      stdout started: 13:01:31
   expected 0      stderr  ended: 13:01:34
                   duration: 00:00:03
```

The Log panel at the bottom shows the output: "[INFO] 13:01:34 Finished!".



The screenshot shows the Testrunner application window titled "Testrunner - test.xml". The interface is similar to the first screenshot, but the test results are different. The selected test case "Execute calculator test" is now marked as failed.

Suite / Set / Case / Step	Result	Comment
my-multimedia-tests	Failed	(Total: 3, Pass...
meego-bash-tests	Failed	(Total: 3, Pass...
+ echo on all terminals	Passed	
+ where am I?	Passed	
+ Execute calculator test	Failed	

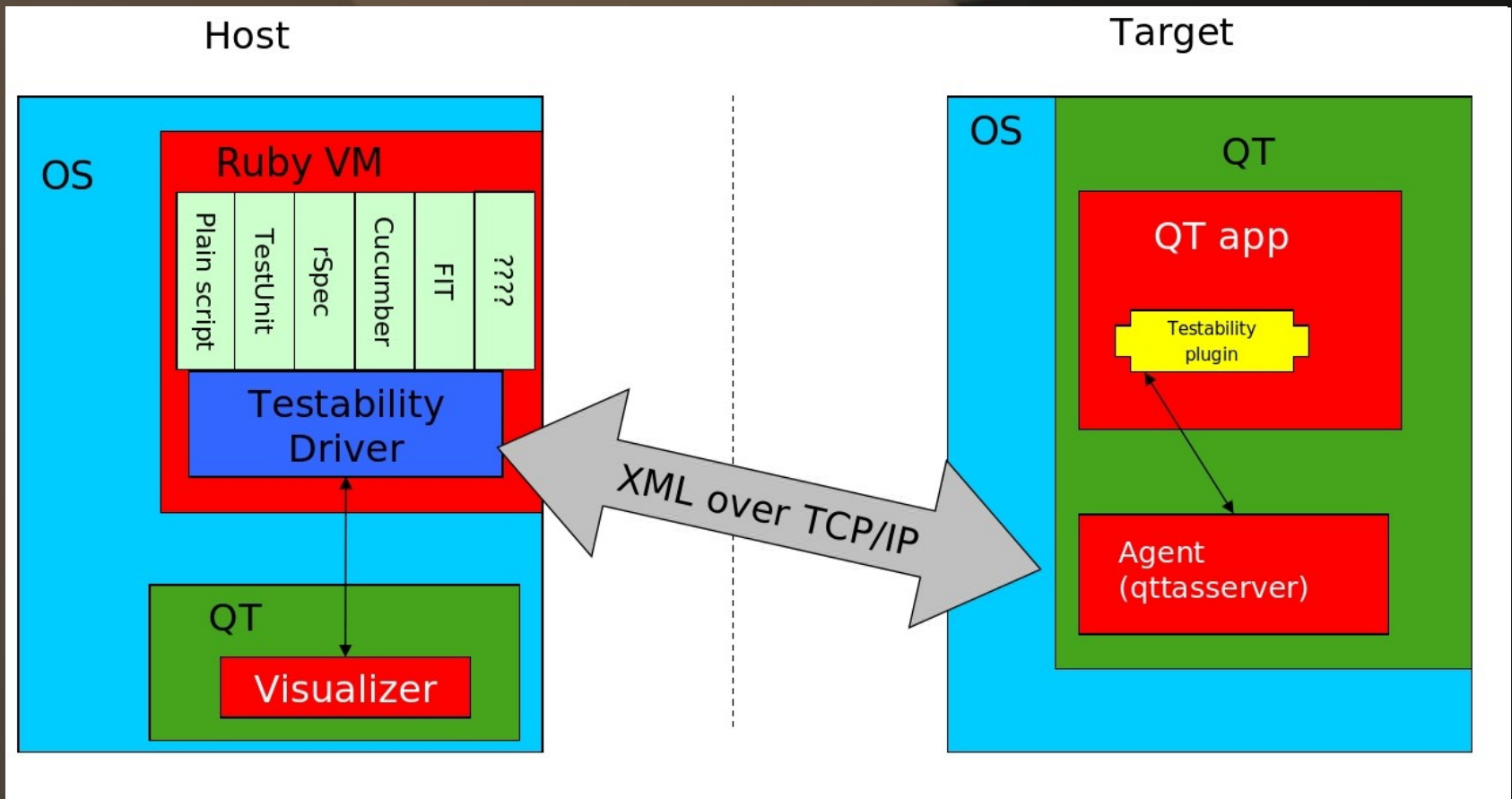
The main panel displays the test definition for the selected case: "Execute calculator test". It shows the suite name "my-multimedia-tests" (Failed), the set name "meego-bash-tests", and the case name "Execute calculator test". The steps section shows a single step: "1 ruby /etc/driver/visualizer/calculator.rb" which failed.

```
1 ruby /etc/driver/visualizer/calculator.rb Failed
   returned 1      stdout started: 13:07:49
   expected 0      stderr  ended: 13:08:02
                   duration: 00:00:13
```

The Log panel at the bottom shows the output: "[INFO] 13:08:02 Finished!".

TDriver

Testability Driver TDriver is a testing tool open sourced by Nokia. It will make test automation possible for Qt applications running on any platform that runs Qt



Android problem



112



111

I'd like to unit test my Android application but I found that test driven development in Android is far from trivial at the moment.

Any tips, tricks, war stories for building light weight and preferably fast running tests?

 [android](#) [unit-testing](#) [tdd](#)

[link](#) | [edit](#) | [flag](#)

edited [Mar 8 at 9:36](#)

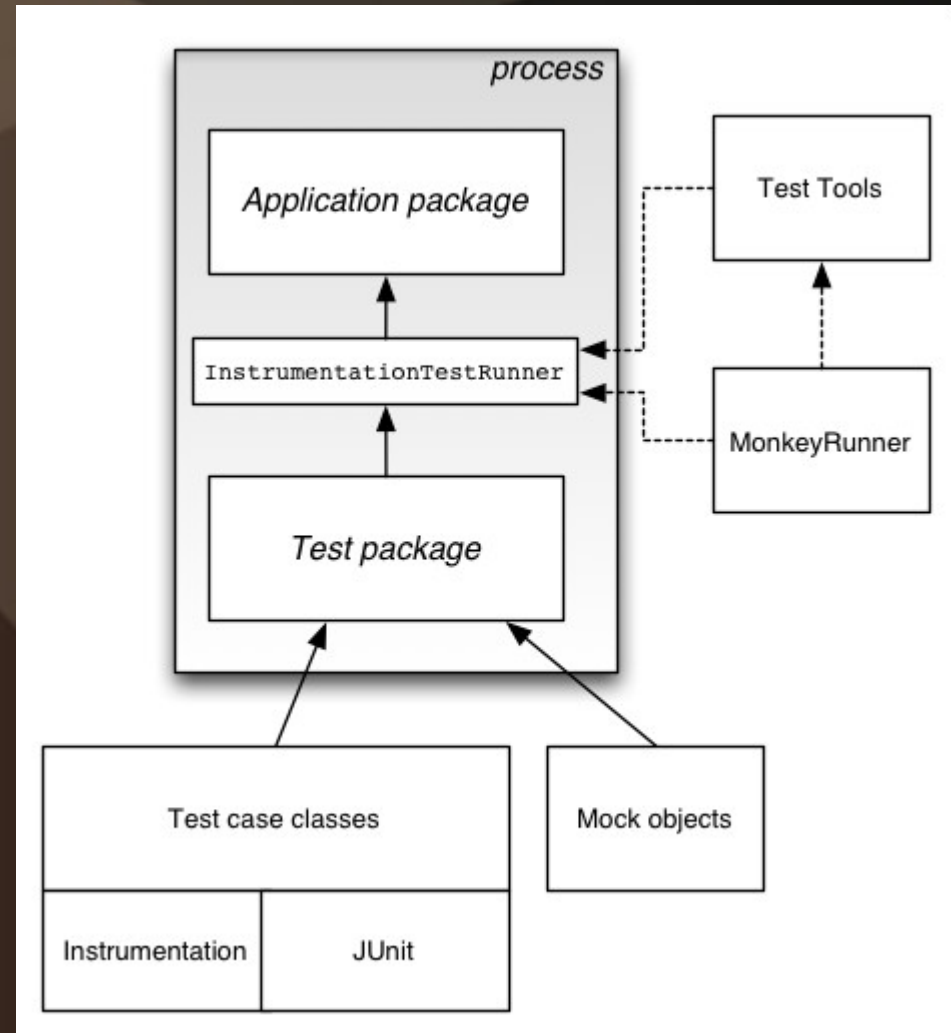
community wiki

[2 revs](#)

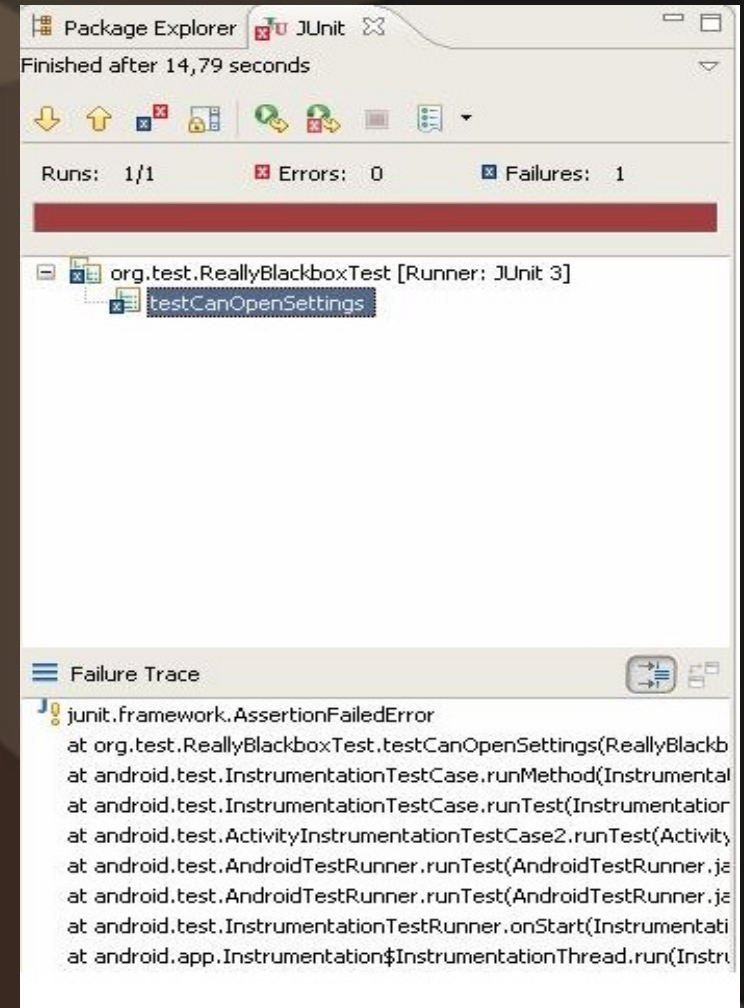
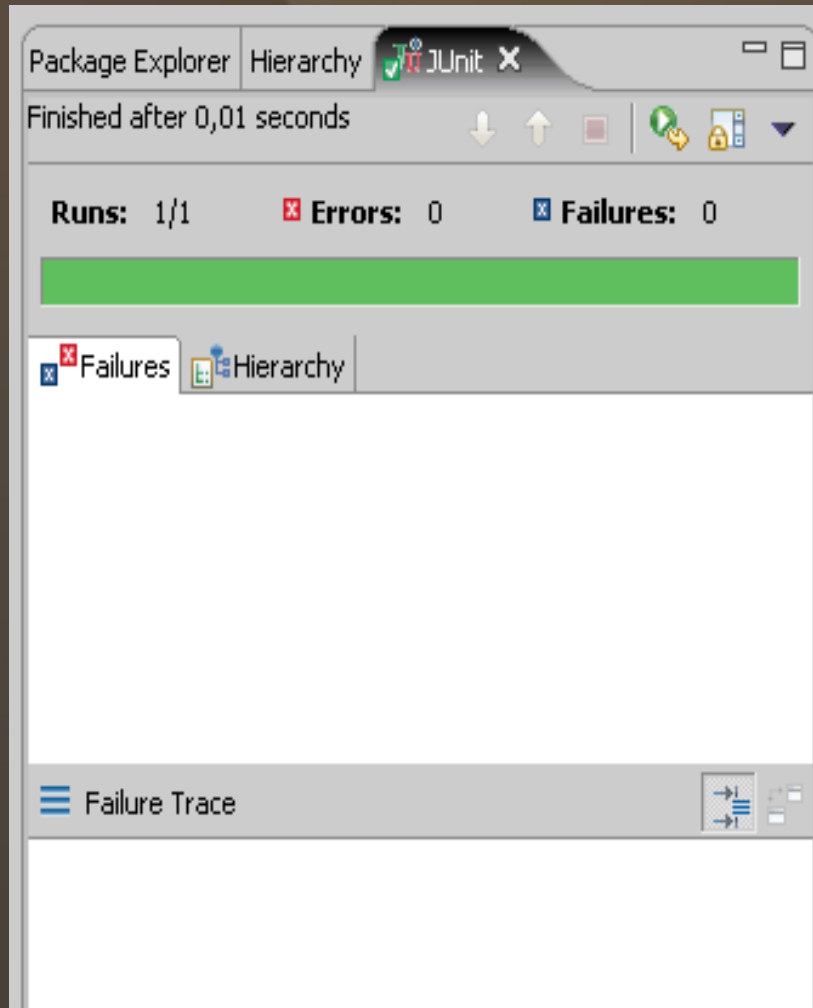
[nyenyec](#)

Android tools

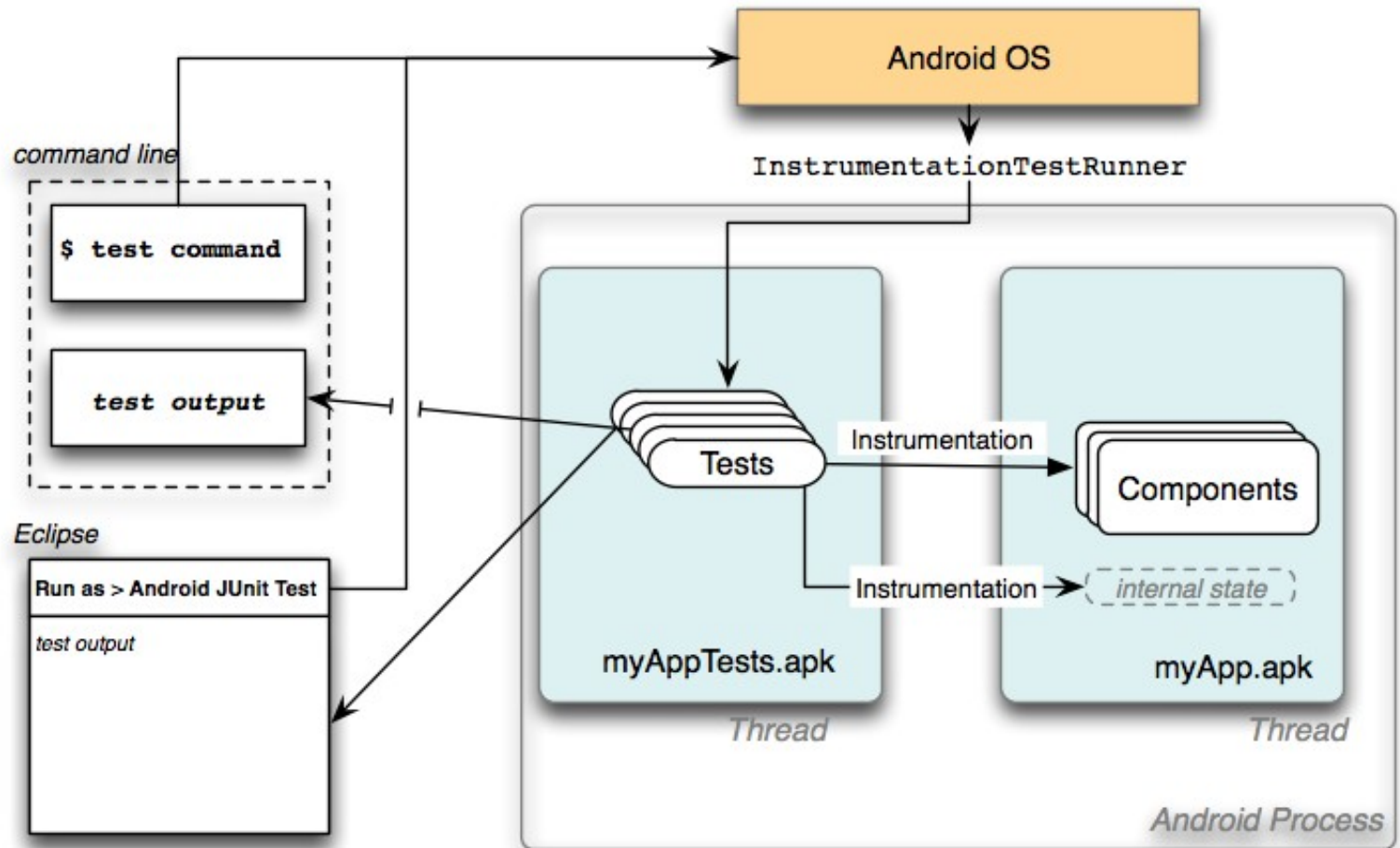
- ADB – android debug bridge. A versatile tool lets you manage the state of an emulator instance or Android-powered device.
- ddmlib – java lib from Dalvik Debug Monitor Service (need to recompile due to localhost hardcoded)
- Robotium - is a test framework created to make it easy to write automatic black-box test cases for Android applications. Open source. Released by Jayway
- Android Emulator - a QEMU-based device-emulation tool that you can use to design, debug, and test your applications in an actual Android runtime environment.
- Logcat - Lets you read system log messages that are output on an Android device or emulator.
- Junit/TestNG – unit testing lib



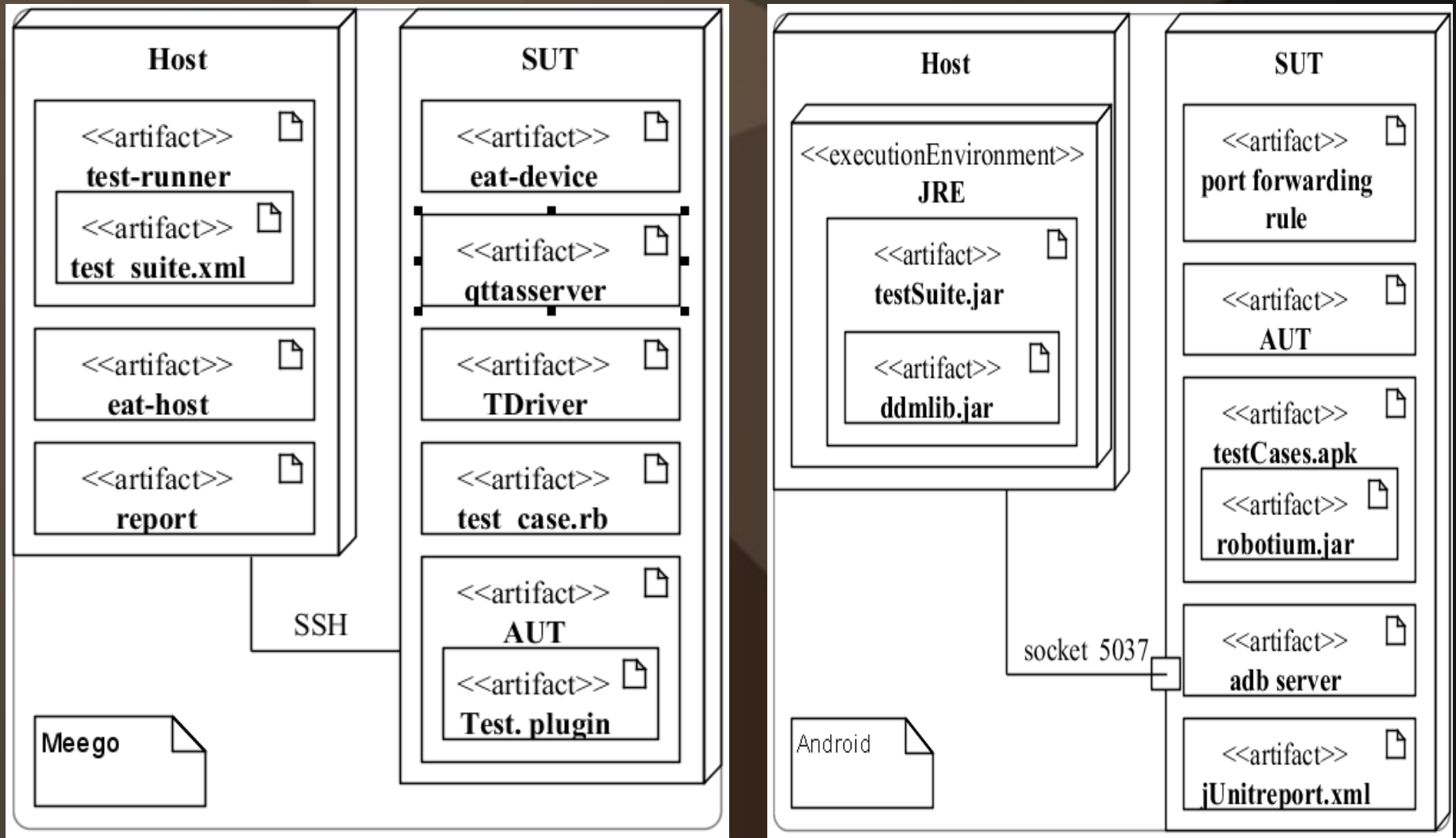
Eclipse JUnit view



Android instrumentation



MeeGo and Android host based testing architectures



MeeGo and Android testing features

	Android	MeeGo
Execution on real device	Impossible to interact without PC	+
System interaction	DDMS	bash
AUT interaction	Instrumentation	TDriver
Test plan execution	-	Testrunner
Semi-automated testing	-	+
Verify images	-	+
Unit testing	Junit/TestNG	Test::Unit, MiniTest
Monkey testing	+	+
Reporting	No out-of-box solution. Few open-source projects to generate junit report.	Testrunner (QA reports format) TDiver (several formats)

MeeGo testing fails when

- You write app without QT
- Your app doesn't have TDriver support
- You try to run app inside emulator on workstation with non-Intel hardware

Android testing fails when

- You don't have key for already packed app (but you have a chance to resign it)
- You are not familiar with Java
- You need to run tests remotely:
 - Hardcoded “localhost” in ddmlib
 - Hardcoded “localhost” in emulator (port forwarding rule is required)

MeeGo test flow

- Start test case (testrunner) – dev machine
- Call TDriver test (via SSH) – SUT
- Retrieve results (via SSH)
- Generate report

MeeGo demo test

Android test flow

- Start test case (IDE/Java) – dev machine
- Call Instrumentation test (via adb) – SUT
- Retrieve results (via adb)
- Generate report

Android demo test

Summary

- MeeGo testing fails if you don't have a real device or Intel hardware.
- Android doesn't provide out-of-box solution for host based automated testing but it can be easily implemented.
- With significant differences in tools and languages there is similar design in Android and MeeGo automation test frameworks.