Testing of Embedded Software Products

Vyacheslav Vanyulin E-mail: vyacheslav.vanyulin@auriga.com
Andrei Pronin E-mail: andrei.pronin@auriga.com

The Russian Software Engineering Expertise – Delivered Worldwide
Since 1990

Embedded Software Testing Challenges

- Very limited interaction interface capabilities
- Software update process constraints
- Relatively high dependency of the software on the details of the hardware platform
- Scarce and experimental hardware base
- Hard-to-reproduce defects
- Emphasis on race conditions

Specifics of Embedded Software Testing

- Time-consuming, routine and repetitive stress testing
- Electrical signal patterns as test inputs
- Need to seriously consider hardware defects
- Need to organize shared access to hardware
- Higher value of overnight and batch tests runs

Specifics of Embedded Software Testing, 2

- Higher importance of build & installation procedure organization
- Higher value of proper identification of tested configuration
- Relatively high number of parameters to be reproduced to reveal a defect
- High amount of defects which are hard to reproduce
- Relatively high importance of debug prints as the means for getting defect details

Automated vs. Manual Testing

- Wide use of manual testing for embedded projects is very difficult, if possible
- Automation is done for more than 95% of the final test cases
  - manual testing is usually used during investigation phase only
- Automation affects all aspects of the testing process not just test execution and documenting

Test Design and Tracing Requirements

- Great number of test cases does not allow to use a straightforward test design approach
  - get requirements →
  - design test cases →
  - run manually, optimize, fix, detail →
  - create script based on the manual case
- In embedded world the test cases are:
  - traced down to the embedded software requirements
  - traced up to the software requirements for the agents and framework

Validation of the Test Support Infrastructure

- Test support tools are designed in parallel with the target embedded software design
- The support and testing software shall be validated themselves
- Virtual Lab and Automated Test Framework also requires validation

Hardware Coverage Matrix

- The hardware coverage matrix is the set of all “test case - hardware unit” combinations to be tested
- The target software must be tested on the range of the possible target hardware
- Test cases implementation shall not be affected by the changing of the matrix

Software Builds

- The build number is obtained from running target software at the beginning of the test execution
- Fresh version is built on the predefined build environment. It assumes automatic ways of:
  - making the build
  - assigning it a unique number
  - tagging the source tree
  - archiving the binaries
  - transferring the binaries to the target board

Defect Tracking and Analysis

A defect may origin from:

- the target software
- the underlying hardware
- the test support infrastructure

- Hardware-caused defects requires a person with hardware engineer skills
- Both the test developer and hardware engineers are included in the triage committee for the project

Debug Support

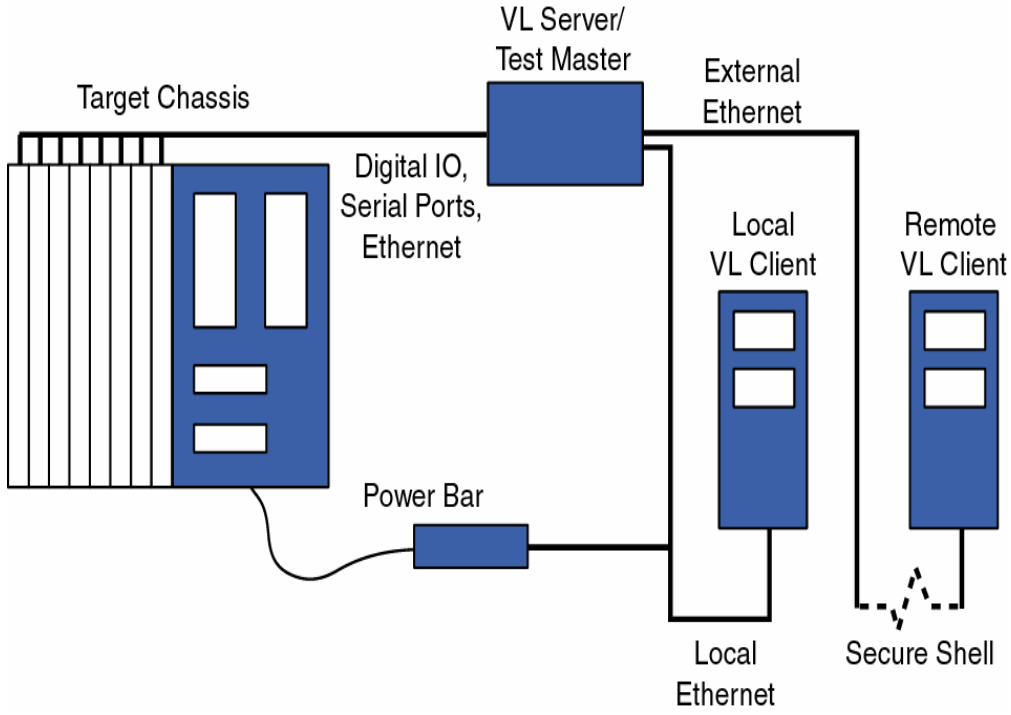
- Lack of debugging tools makes the debug prints the only available way of getting details
  - But it is the source of mystery: the timing and other characteristics of the debug and release versions of the target software may differ, and the defect seen on one version may never be seen for the other version
- Necessity to have built-in debug capabilities
- Test cases should know how to use extended debug capabilities

Two Test Run Modes

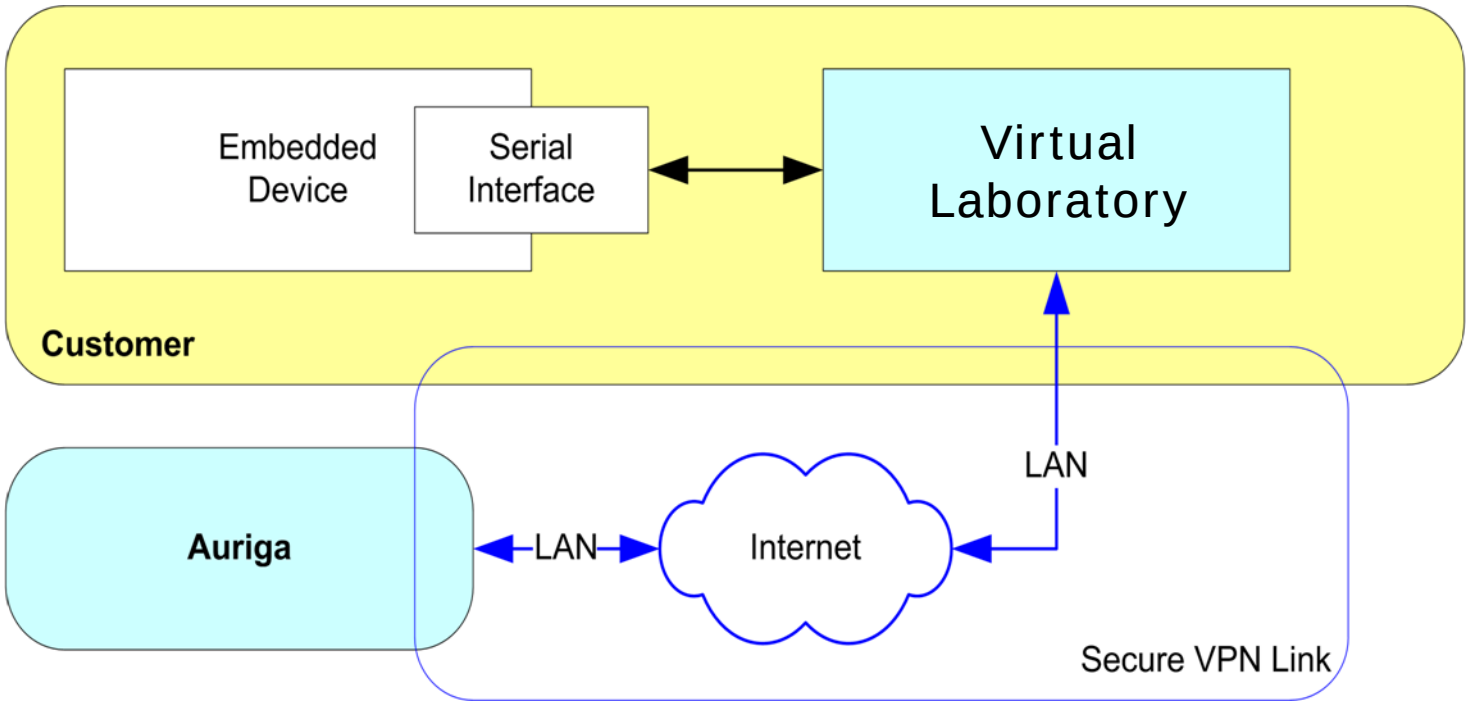
- Batch run – the goal is to detect as much issues per time as possible
  - If an error is detected, the test run doesn't stop, but rather captures all possible information about the system state at the time when the defect was discovered and continues with the next test case
- Until the first failure – the goal is to eliminate issues upon their appearance
  - And if a failure occurs, the test run is stopped, the system state is preserved, and a developer is notified and allowed to examine the system state in details to reveal the root cause of the failure.

Virtual Laboratory (VL)

- Allows Controlling of various hardware resources
  - Multiple serial ports
  - Multiple relay channels
  - Power supply for multiple boards
- Provides APIs that allow test suites using resources of the Virtual Laboratory



Virtual Laboratory (VL)



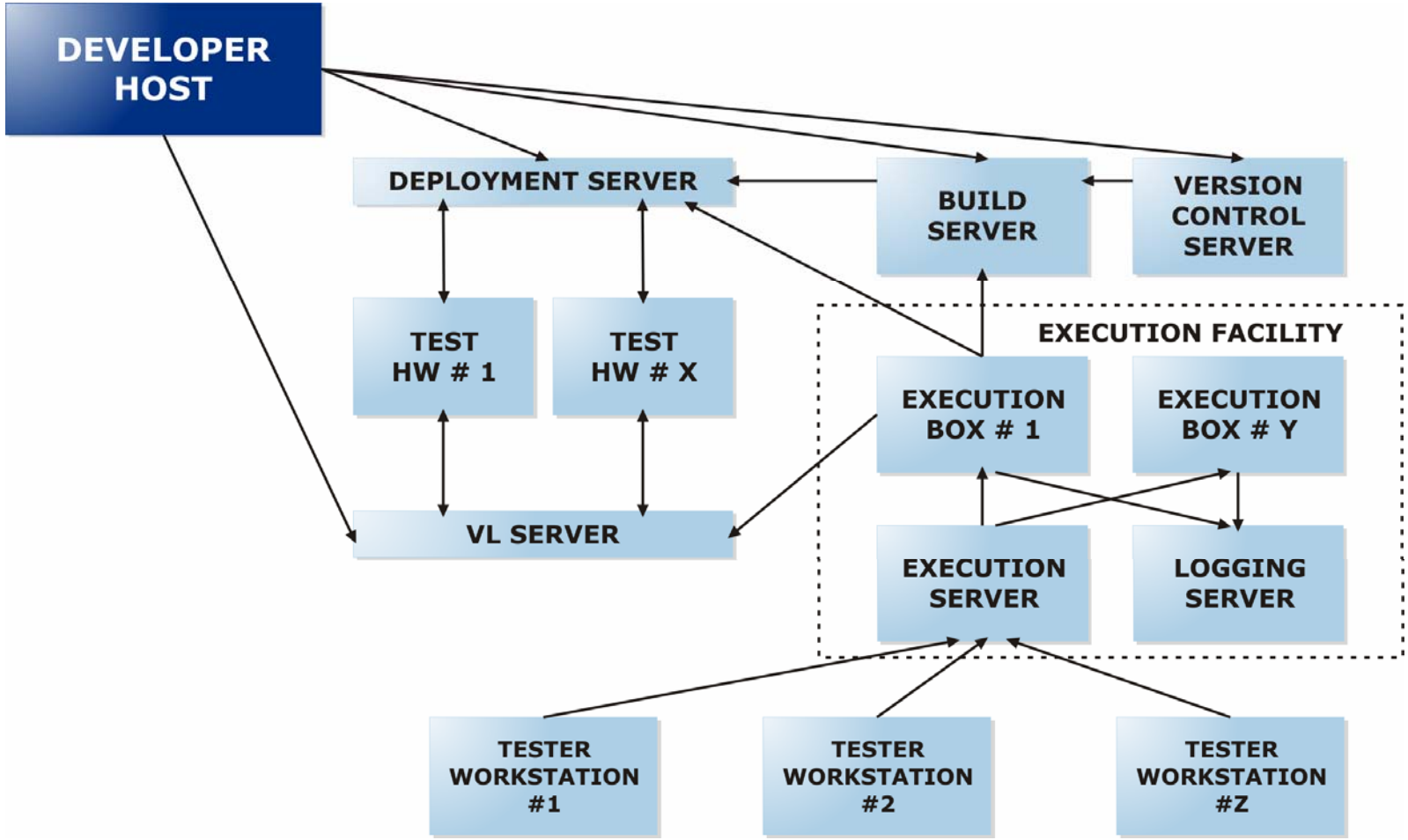
Automated Test Framework (ATF)

Automated Test Framework (ATF), 2

ATF provides API to automate:

- Build
- Archive
- Deployment
- Execution
- Logging
- Report generating
- Target control

Test Lab Infrastructure



Summary - Key Success Features

- Virtual Laboratory (VL) solution
- Re-usable Automated Test Framework (ATF) simplifying typical operations
- Special approach:
  - 95+% automated testing
  - Two modes of test executions
  - Using test agents
  - Using specially built-in debug capabilities

Questions?

Thank you!

Vyacheslav Vanyulin **E-mail: vyacheslav.vanyulin@auriga.com**
Andrei Pronin **E-mail: andrei.pronin@auriga.com**