

Shared Models for MBT for Software and Systems

A. Pietschker, A. Beimler
20-Oct-2011

 Giesecke & Devrient
Creating Confidence.

Giesecke & Devrient – From Printing Paper Securities to Providing High-Tech Solutions

Server software and services



Government solutions



Cards for payment and telecommunications



Banknote processing



Banknote and security paper




Banknote and security printing



1852

2010

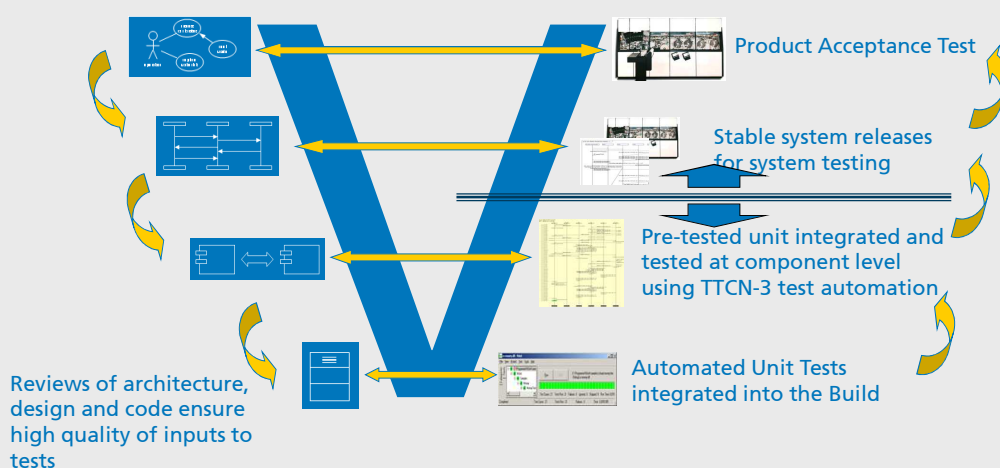
MBT UC 2011: Shared Models for MBT for Software and Systems
20.10.2011 Page 2

 Giesecke & Devrient

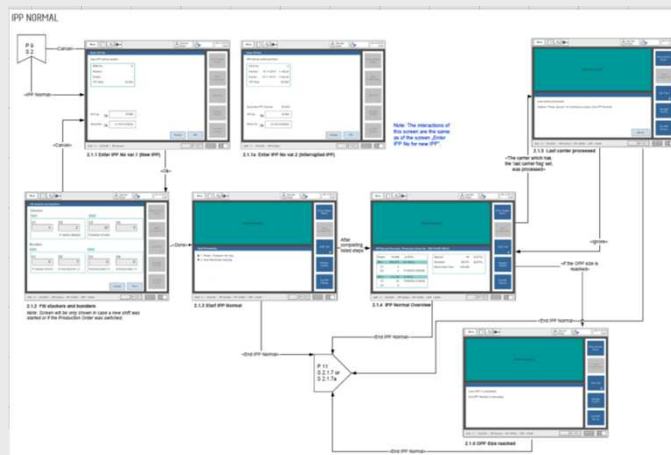
Why do I need an integrated test strategy?

- Modern testing employs several test phases
 - Review (static analysis)
 - Unit testing
 - Integration testing at several levels (Component, Subsystem, ...)
 - System testing
 - Acceptance testing
 - Field testing, Beta testing
 - Production testing
- An integrated test strategy aims at reducing the overall effort
 - Not everything is tested at every level
 - Tests from previous test phases and their results are taken into account when test at higher level are performed
 - Tests are run where it is easier or cheaper

Software Test Automation in the Development Cycle



Models from UI Specification

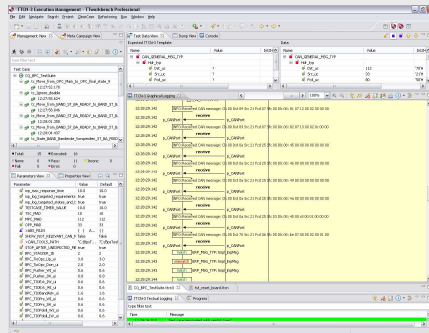


Test generation control with strategy

- Separate concerns
 - Model describes behavior
 - Strategy controls the test generation
 - Depth of state machine reached
 - Abstraction of information (TTCN-3 template vs. English in Excel)
- Adjust strategy when needed
 - Increase depth or coverage of model
 - Reassign tests from integration to system testing and vice versa

Targets for test generation

- Integration tests
 - TTCN-3 test automation



- System tests
 - Excel (HPQC)

The screenshot displays an Excel spreadsheet used for test case management. It includes a 'System Test Specification' section with fields for 'Test Case', 'Test Case ID', 'Test Case Description', and 'Test Case Status'. Below this is a table with columns for 'Test Step No.', 'Test Step Description', 'Test Step Category', and 'Duration'. The table lists various test steps such as 'Check that the system is running', 'Verify that the system is running', and 'Verify that the system is running', each with a corresponding duration.

Test Step No.	Test Step Description	Test Step Category	Duration
1	Check that the system is running	Pre-condition	<55.00
2	Verify that the system is running	Test Step	<55.00
3	Verify that the system is running	Test Step	<55.00
4	Verify that the system is running	Test Step	<55.00
5	Verify that the system is running	Test Step	<55.00
6	Verify that the system is running	Test Step	<55.00
7	Verify that the system is running	Test Step	<55.00
8	Verify that the system is running	Test Step	<55.00
9	Verify that the system is running	Test Step	<55.00
10	Verify that the system is running	Test Step	<55.00
11	Verify that the system is running	Test Step	<55.00
12	Verify that the system is running	Test Step	<55.00
13	Verify that the system is running	Test Step	<55.00

Summary

- Common models benefit from the right level of abstraction
- Common ownership reduces possibility of deviations
- Variable test generation enables a common understanding on implemented test strategy
- Buy-in from development and system test to work with models