Ideal Test Systems & Target Development Test Process

Tony Chang
June 9, 2010
Contents

- “Test” Fundamental Responsibilities & Activities
- Product Delivery Test Quality Indicators
- Product Test Coverage
- Ideal Test Systems

Target Development Test Process

- Pre-Testing Preparation
- Development Testing
- Continuous Integration
- DTTP – Design, Test, Tools Partnership
- Continuous Regression
“Test” Fundamental Responsibilities & Activities

- Build up the Test case Baseline & Coverage management system
- Develop new Feature Test Design & Plan
- Develop the Solution Integration Test Strategy
- Develop & Execute new Test cases – manual & automated
- Promote Test Automation technology
- Fault Identification, Trouble Report & Retest
- Test Environment Management

“Product Quality Assurance” is our #1 mission
Product Delivery Test Quality Indicators

- All test cases have been executed & regressed with 95%+ Pass rate
- Non-Critical/Gating remaining issues
- Manageable number of Minor un-resolved issues
- Provide remaining issues Patch Plan

BUT:

- Number of Test cases cannot directly reflect the Product’s Quality
- Number of bugs found during the Product Verification cycle cannot reflect the Product’s Quality as well

The key factor to evaluate the Product Test Quality is the “Product Test Coverage”
Product Test Coverage

- **Requirements-based Test Coverage**
  - Test the product/Software behavior against the every claim made in the Requirements & Design specifications

- **Functional Test Coverage**
  - Test the product/Software behavior against the Requirements, Design specifications, Customer Scenarios and examines what the program accomplishes from various areas:
    - Level 1 new feature basic function, PQRS Test Coverage
    - Level 2 new features’ interaction Test Coverage
    - Level 3 new & old functional interaction Test Coverage (N-1)

- **Regression Test Coverage**
  - Product Test case Baseline

- **Code Coverage**
  - Is a measure used in software testing to describe the degree to which the source code of a program has been tested and via the analysis process we’ll also be able to:
    - Find areas of a program not exercised by a set of test cases
    - Create additional test cases to increase coverage
    - Determine a quantitative measure of code coverage, which is an indirect measure of Quality
    - May indentify redundant test cases that do not increase coverage
Ideal Test Systems
Ideal Test Systems


✔ Fully Integrated R&D Management System: Requirements, Documentations, Test, Project, Production, Engineering, Beta/CAT

✔ Measureable Product Test Cases Coverage

☐ Measureable & Visible Products Quality Process: Requirements, Development, Test, Project, Production, Engineering, Beta/CAT

☐ Customized Test Case Execution Strategy

☐ Commercialized Test Environment & Tools: Product Solution Integration, Networks Solution Integration, Customer Solution Integration Center
Fully Automated Test Automation Environment

- Traffic Test Environment
- E2E Test Environment
- Nodal Feature Test Environment
- Designer Desktop Test Environment

* Single Framework for Nodal, E2E and Traffic
* Global Lab Sharing
* Test Case Sharing
* Designer Desktop Testing
Fully Integrated R&D Management System

Design Doc. Environment
(API to access Database for Feature list, FRS/FTS url)

Requirements Management System

Test case Management System

Defects Management System

Marketing, Design

Project Management System

Design, Test, Support

Testcases/Test Results;
Other customizable data such as CRs & enhancement

Defects & enhancement

Test Case Coverage/Management Objectives & Benefits

Objectives

- Ensure Product Test case Baseline Coverage
- Enforce Test case and Test Coverage ownership
- Improve Test Case Management (Quality, Efficiency, Coverage, Maintenances …)

Why do we need to implement TCM?

- Enforce the corresponding primes to improve Test case quality and integrity
- Vehicle to ensure & improve Product Test Coverage & Code Coverage
- Vehicle to proper upgrade the Regression Test cases to the next Release
- Vehicle to ensure the effectiveness of “Agile”, “Iteration”, “RBT” & “DFT”
- Vehicle to ensure the System Integrity & implementation of “VBS”
- Efficient Test cases Execution & Maintenance
- Efficient Quality management (Eg. CR/Patch Analysis)
- Vehicle to build up Product Test Knowledge base
Test Case Ownership -- Subsystems

- Organize Product related Test cases into manageable pieces called **Subsystems**
- Subsystems represent functional areas. Assigning **Owners** for each Subsystem
- Subsystem Owners/Primes responsible for documentation, execution, analysis, debug, creating and maintaining TP/TC
- Each subsystem is further broken down and composed by multiple **Funcnode**. Each Funcnode is composed by multiple **Funcleaf**, which is the smallest functionality unit in regression.
- Subsystem Owner assigned, Design Manager identified

Test Coverage Improvement

- The Subsystem Owners of the identified functionality will research the coverage improvement and write test cases to plug the holes in their subsystem coverage
- TCM test plan captures the coverage and functionality improvement
- Identify the design subsystems and modules for that test subsystem
- Update test strategy document with test coverage/Software module information and update document with code coverage results

Test Case Clean up Process

**Manageable Subsystems:**
- Test cases were grouped into manageable Subsystems
- Test cases executed on TICC per subsystem to understand the failures
- Failures analyzed and test cases corrected
- Test cases executed on TICC as frequently as possible, results analyzed and test cases cleaned up after every run to improve results

**Coding Standards:**
- Test case coding standards published and used to write and correct the test cases
- TEP coding standards published and adopted to improve test case execution results on TICC

Test Case Management -- by Subsystems

- System Upgrade
- Robustness & Capacity
- Bearer Service
- Teleservices
- Call Completion
- Call Offering
- Call Related
- Call Restriction
- Charging
- Supplementary Services
- GSM Compliance
- Intelligent Networks
- PSTN Signalling
- Mobility Management
- Regulatory
- Translations
- Location Based Services
- GSM Railways
- GSM/UMTS Functionality

Manageable Subsystems:
- System Upgrade
- Robustness & Capacity
- Bearer Service
- Teleservices
- Call Completion
- Call Offering
- Call Related
- Call Restriction
- Charging
- Supplementary Services
- GSM Compliance
- Intelligent Networks
- PSTN Signalling
- Mobility Management
- Regulatory
- Translations
- Location Based Services
- GSM Railways
- GSM/UMTS Functionality
Target Development Test Process
Target DT Process – Pre Testing Preparation

Pre Testing Preparations
1. Precise Customer Requirements
2. Precise Tools Requirements
3. Clearly defined “Design”, “Story” and “Module”
4. High Quality Feature Test Plan
5. High Quality Solution Test Strategy

Market Requirement

- Precise Feature & Tools Requirements
- Test Plan/Strategy
- Design Evaluation
- New Feature Analysis
- High Quality of Feature & Solution Test Plan & Strategy
The Importance of Feature Test Plan

- Tester’s most critical deliverable during the test cycle
- Can evaluate the tester’s feature knowledge
- Can evaluate the new feature’s test coverage
- Pre-define the test progress & evaluation point
- Ensure the Requirements Traceability’s coverage
- Provide the guideline for DTTP (Design, Test, Tools Partnership)
  - Define the “Capability Test cases” & schedule with designer
- Decide which test cases will be used for Regression
- Identify which test cases’ results can be shared with Data testing
- Lift up the testers test capability & product knowledge
Both the designers & testers are owning the features together – “DTTP, Design Test Tools Partnership”

- Clear metric to measure each Story’s progress & Status
- Story 1—4 needs to be completed Coding & Testing prior to TR4 (completed the Agile designer Testing/LLT)
- Remaining Test cases will need to be completed at TR4 (completed the Feature testing)
New Feature Devel. Testing
1. Integrate “White box” test tool
2. “Grey box TCs” generation & Execution based on Test Modeling & Record Playback
3. Auto-Results & Log Analysis
4. DTTP—Design, Test, Tools Partnership, early Feature Test Automation
5. Designer Desktop Testing for Capacity

Feature Development Testing

- Test & Automation Environment
- Auto-TC Generation for “Module/Grey box” testing
- MBT
- Fully Automated Test Automation Environment

- Weekly Build
- Automatic Smoke Testing
- MBT
- Test Automation Environment

New Feature design & Test

Daily

Test Design & TC Development

New Feature design & Test

Test & Automation Environment

每日

缺陷报告

缺陷分析及定位

产品需求分析

新需求/新特性

新版本开发

特性设计及 Story 分解

特性测试方案及验证计划

特性测试用例自动化

特性和 Story 开发

特性和 Story 自动验证

版本交付

产品持续构建

持续构建

持续

版本交付

• 全面自动回归验证
• 星期一“The Load is Ready for New Activities”
Fully Automated Designer Test Automation Environment

Designer Desktop Testing
- Coding MST TCs
  - Desktop Compile
    - Compile, Static Insp.
    - DART
    - MST Test
- New Load build
  - Via Interface Record & Playback capability to test existing function
- check-in
  - SVN Server
  - pressed
- 5 minute complete 70% code coverage
- 10 minutes complete all module verification

7*24 Test Automation Factory (CI)
- Module Level Compile
  - Compile, Static Insp.
  - DART
  - MST Test
  - Module Capacity
- VBS TC Selection
  - VBS TC & Code correlation mapping
- Daily Compile
  - Compile, Static Insp.
  - Sanity
  - VBS Test
- CI Testing
  - New Load build
  - SVN

7*24 Test Automation Factory (SDV)
- Feature SDV
- Regress
- Capacity
- Release Testing
- New Load build
- SVN
- TC & Code correlation mapping
- 10 hours 50 k test cases full recovery
DTTP (Design, Test Tools Partnership) Objectives:

- Pull forward the Test & Tools teams engagements at the early stage of the Requirements & Test Analysis cycle

- Ensure the Feature Quality during the TR3 & TR4 interval
  - Identify the Quality issues at the early stage of the release cycle
  - Shorten the Issues Resolution time during the development & test cycle

- Shorten the overall development & test cycle due to the parallel testing

- Enforce the Communication & Partnership among Design, Test & Tools members

- Enforce the Early Test Automation from the beginning of the Release Cycle

- Accommodate the implementations of Iteration, Agile & Early Test

- Lift up the Capability of the Test & Tools teams
The March to UT/IT/ST:

Timely contributions from each group (Design, PV, Tools) required to make capability completion a success!
DTTP Benefits

Ensure Product Early Quality, Shorten Test Cycle & TTM

- Pull forward the Test & Tools development activities
  - Early Test
  - Enable & Support the designer’s testing
  - Support the Agile Test Strategy – early Test Automation

- Lift up the Product Early Quality – design & test in-sync
  - Code Ready
  - Tools Ready
  - Test cases/Scripts Ready

- Doing the Test Automation from the beginning
- Shorten the Development Test Cycle & TTM
- Lift up the capabilities for Testers & Tools Developers
Target DT Process - Development Testing

- Code Modification Testing
  - 1. For every update, designers can Auto-Target select & Execute the TCs
  - 2. Automatic Results Analysis + related Failure data (Call Trace)
  - 3. Automatic Code Submission after the Passing of Target Testing

*CI -- Precise, Accurate, Fast*
Validation Before Submission

TMS

Test Cases Selection based on TCs & S/W Mapping
TCs Optimization

Compile & Pkg Test cases

1. Software Updates/Increments
   (New Features or CRs Updates)

2. Feature Updates
   - Design Prime compiles and creates modules/patches
   - Submit the Request to VBS process

3. CRs Updates
   - CRs designers compile and create modules/patches
   - Submit the VBS process

4. Test Execution
   - Retrieve Test cases
   - Execute Test cases
   - Monitor Status

5. Result Analysis
   - Validate Billing/Log Files
   - Generate Results
   - Collect Billing/Log
   - Match Test case
   - Expected Result
   - Result Files

6. Module Integrity Suite pass at 100%?
   - Yes
     • Submit Module to Weekly S/W Load build
     • Put Test Case ID’s in the update text
   - No
     • Correct issues
     • Build it into new modules/patches

HUAWEI TECHNOLOGIES CO., LTD. Huawei Confidential Page 23
VBS Applications – Continuous Integration & Patch Validation

**Table of Processes**

<table>
<thead>
<tr>
<th>Release Baseline</th>
<th>CI Application</th>
<th>Patch Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>ICP S/W Unified Platform</td>
<td>Daily Compile</td>
<td>Patch Validation</td>
</tr>
<tr>
<td>Baseline Insertions</td>
<td>Compile, Static Insp.</td>
<td>Issue Analysis</td>
</tr>
<tr>
<td>Insertion</td>
<td>VBS</td>
<td>Feature Interaction</td>
</tr>
<tr>
<td>Compile</td>
<td>Sanity</td>
<td>Target Regression</td>
</tr>
<tr>
<td>TC Execution</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TC &amp; Code Correlation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCID</td>
<td>Modules</td>
<td></td>
</tr>
<tr>
<td>VBS Database</td>
<td>Retrieve Impacted TCs</td>
<td></td>
</tr>
<tr>
<td>VBS Database</td>
<td>Select Impacted TC</td>
<td></td>
</tr>
<tr>
<td>Code Coverage Analysis</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Key Processes**

- **Modification Identification**
- **Improving Test Coverage**
- **Fast TC Selection** Complete the TC Selection Within 10 minutes
- **CI** Precise, Accurate & Fast for Code Modification Test
- **Patch Validation** Precise & Complete Test Coverage for Patches

**Components**

- **ICP S/W Unified Platform**
- **Baseline Insertions**
- **Baseline Insertion**
- **Compile & TC Execution**
- **TC & Code Correlation**
- **Code Coverage Analysis**
- **VBS Database**
- **TMSS Database**

**Integration Points**

- **Daily Compile**
- **Patch Validation**
- **Regression TC**
- **VBS TC**
- **Sanity TC**
- **CI TC**
Load Build & Auto-Regression
1. Friday 4:00 pm Auto-Load Build
2. Auto-Sanity Testing
3. Auto Bugs-Retest, Auto-Regression
4. Monday 8:00am “The Load is Ready for New Activities”

*Continuous Regression: Broad, Deep, Thorough*
Thank You

www.huawei.com