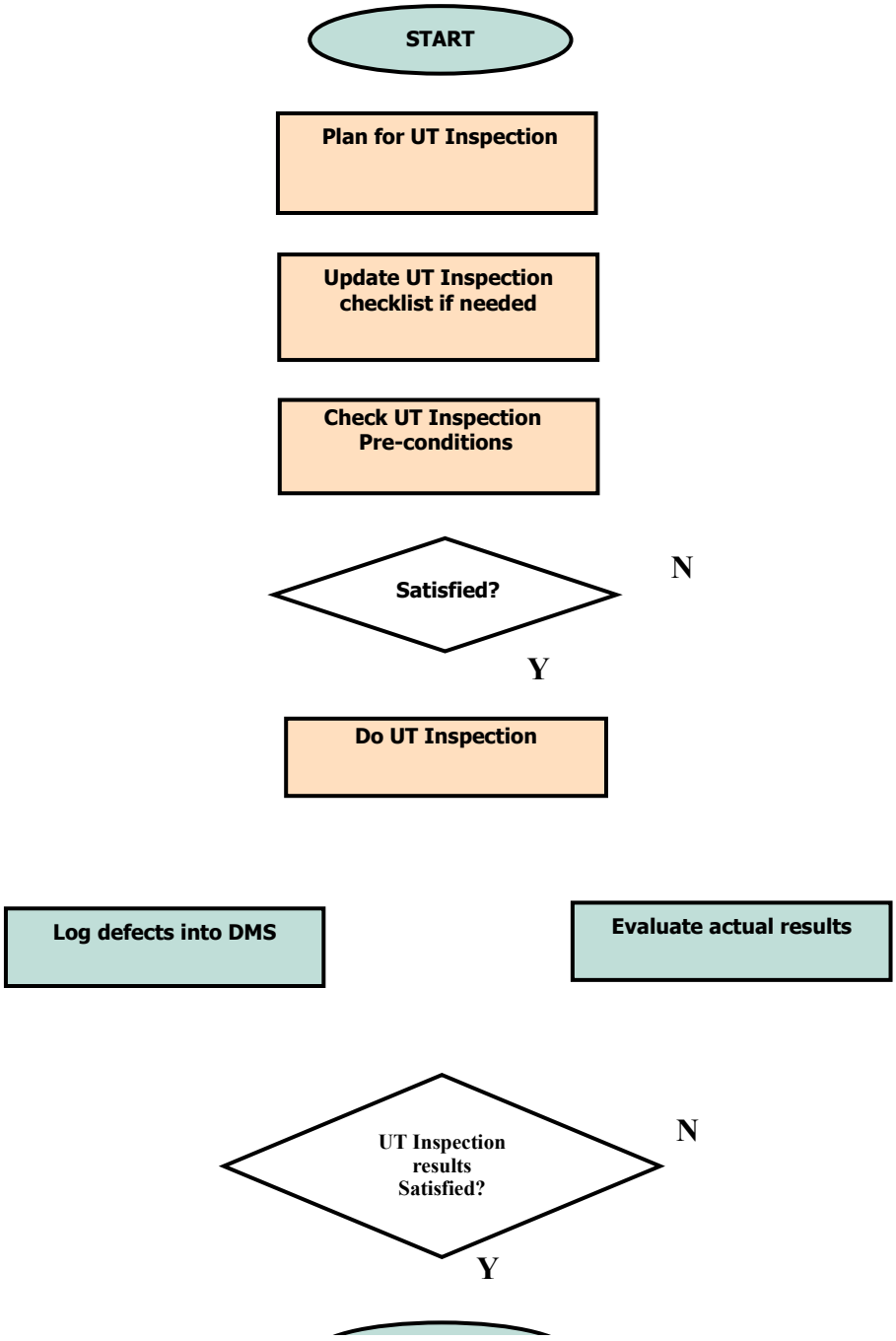
Introduction to Unit Test Inspection Checklist

I Objective

- To guarantee the quality and integrity of the components to be released
- This activity is applied for all releases by project team after UT to testers/QA

II How to use

Step 1. Quickly study the work flow of unit test inspection activities



Step 2. Study Guideline sheet to understand unit test inspection, then use the 'UT' sheet to do inspection

'N/A' in Assessment column

Step 4. Fill the comments you find in the 'Note' column if any

Step 5. Make your suggestion based on the comments in Note column and the summary. Refer to 'Guideline' sheet for pass criteria

Guideline for Unit Test Inspection

Purpose: - To ensure that software package after UT meets certain quality criteria and is sufficient enough for IT, ST or delivery to customer

When: - After Unit test is completed, before execution of Integration/System test
- Final inspection (testing only) for project with only UT (in combination with Final inspection checklist for document verification)

Exception:

The following project types may not apply UT inspection:
- Test projects
- Projects with many hotfixes released daily
- Projects with one staff
- Projects with limited schedule
- Projects with UT & IT/ST merged together (optional)

Who: - Test Leader or Person who is responsible for Integration/System test. Sometimes PM can do to verify completeness of UT.
(Projects with more than 1 tester must do UT inspection if exception criteria above are not satisfied)
- QA in case of final inspection for UT-only projects

How: Responsible person has to follow the steps below:
- PM & responsible person negotiated UT acceptance criteria, often done in kick-off meeting. Later the criteria can be changed as per project specifications but related staffs must be informed ahead. UT acceptance criteria may come in form of metrics and they can be defined in project plan or FI\Quality Strategy\Quality plan.
- Check Unit test Results & Report basing on the items in checklist. For explanation on each item, see the Notes column in UT sheet
- Execute sample test to validate the quality of Unit test (See Guideline for UT quick test below)
- Give decision "Passed" or "Inspect again" (Not Passed) basing on Pass Criteria,
- Fill the decision to the Checklist
- Inform PM, Development team Leader, Test team & QA about the decision.

Pass Criteria:
- All mandatory items in the checklist should be passed (marked **OK**)
- All mandatory items marked with **N/A** need to have explanation & evidence of approval
- In case UT is released with open issues (any mandatory items are marked **NOK**):
  + If UT acceptance criteria were defined, then the criteria must be met.
  (For example, UT can be released if 95% of Unit TCs are passed or 2/10 items in the checklist are NOK)
  + If a release with open issues is approved, reasons & evidences of approval are required

Note:
- 'OK' means that the item content is in the project work, mandatory requested for checking, and is satisfied
- 'NOK' means that the item content is in the project work, mandatory requested for checking, and is NOT satisfied
- 'N/A' means that the item content is out of the project work, therefore it does not need checking
- In FSOFT, the following UT tools are standardized:
  + UT tools: JUnit (Java), NUnit (.NET), CppUnit (C++), CUnit (C), EasyMock (JUnit), HtmlUnit (Java & HTML), Mock Objects (Java)
  (for more information about standard unit test tools and how to perform unit test, please see FSOFT guideline for unit test in Guideline_Software Unit Test)
  + Coverage tools: Maula (Hitachi Software Ltd. - Java & .NET), Cobertura (Java), NCover (.NET)
- Coverage metrics:
  + Statement Coverage: How many percentage of LOC that UT needs to cover
  + Branch Coverage: UT requires all conditional flows (if/else, switch, try/catch/finally, etc.) to be executed at least once
  + Path Coverage: UT requires all flows (from Start to End) to be executed at least once

Guideline for UT quick test

Preparation:
- Determine how much test to be sampled (for example: 10% of total UT test cases, 3/10 modules, etc.)
- Create your own test cases (if possible) or select UT test cases to be reused:
  + What to be chosen for verification should rely on analysis of priority/importance, current plan of development (which modules are available), frequency of usage, integration with other modules, boundary values, exception & error handling, difficulty, multi-threading, modules with many CRs, etc.
  + TCs for verification can be made as checklist, UT scripts or similar to IT test cases. Because UT mostly focus on modules, classes & functions, UT verification should not expect a system with full functionalities.
  + For modules, classes & functions, tester/QA should base on design & code documents (javadoc, code comments, ndoc, etc.) to create appropriate TCs (input, output, exception, etc.)

Execution:
- Run TCs one-by-one based on your checklist or prepared TCs. For special or difficult TCs , can request developer to execute them
- Ask developer to open UT test case/source code to verify whether input covers all/almost abnormal cases (if necessary)
- Record successful & failed TCs with evidences (how to ensure that the TCs are passed. Evidences are required for inspection but optional for testing)
- Make report & suggest decision based on UT inspection checklist

Unit Test Inspection Checklist

Project Code:
Inspector(s):
Work Product(s)/Module(s):
Inspection Date:
Inspection Effort (person-hour):

KLOC:
Number of Unit TCs:

- Select a value in the combo box
- Refer to Introduction sheet\ Step 3 for more guides

Check Item	Assessment	Priority	Notes
Have UT been executed with pre-defined test cases/test scripts?			Some projects/groups can execute UT without pre-defined UT test cases/test scripts
If Yes, have these UT test cases/test scripts been reviewed & approved?			Please refer to FSOFT_RADIO For Projects to know who reviews & approves UT test cases/test scripts
Has UT report been created?		Mandatory	- UT report must follow standard or customer supplied document templates, sometimes can be reports generated by UT tools - Information must be organized properly, with charts, tables and analysis if possible
Have UT evidences been captured and stored?			Any mean to verify the UT results, often they are screenshots, logs, output files, etc.
Has UT been executed in the correct environment?		Mandatory	Often as required by customer (hardware & software, OS languages, platforms, necessary applications/modules/packages/libraries) + Difference OSes: Windows, MacOS, Linux, Unix, etc. + Different OS languages: English, Japanese, French, etc. + Different platforms: x86, PPC, AMD, etc. + Different host applications: For example IE, Firefox, Netscape, Safari, etc. are different browers for web applications
If No, have the reasons been analyzed, reviewed & approved?			In some situations, software modules may need interfaces with external applications that are unavailable. Developers who perform UT must setup fake/simulated environment.
Do UT results remain the same among multiple runs?			UT results of one TC must be the same on: + Different environments + Different runs on the same environment
Have UT metrics met the target?		Mandatory	- Often in Objectives of plan documents (development plan, test plan) or FI\Quality Strategy\Quality plan. - In most cases, the purposes are defined as metrics, including both FSOFT & customer metrics.
Number of UT bugs per KLOC (i.e. 6.5 bugs/KLOC)			
Ratio of UT TC (Normal) (i.e. 60%-70%)			
Ratio of UT TC (Abnormal) (i.e. 20%-30%)			
Ratio of UT TC (Boundary) (i.e. 10%-20%)			
Statement Coverage (i.e. 100%)			
Branch Coverage (i.e. 80%)			
Path Coverage (i.e. 100%)			
<Add/remove metrics if needed>			
Have UT defects been logged for tracking?		Mandatory	Log into DMS, Excel or custom tools, any mean to make defects trackable
Are all remaining UT issues from the most recent UT inspection fixed?			- If the most recent UT inspection did not pass - If the most recent UT inspection passed with open issues
Have UT defects been closed before release?		Mandatory	
If Yes, have they been solved successfully and fully before release?			- Successfully: Bug is fixed - Fully: Related bugs & similar bugs are fixed, too
If No, have the UT acceptance criteria been met?			As defined in kick-off meeting, see Guideline for more information
If No and UT acceptance criteria are not met, is there any approval for release with open defects?			For some reasons, UT bugs may not be closed before release, resulting in open issues to be resolved in next release. In this case project team must present approval for release with open issues from customers or group leader or PM and promise for responsibility in case the open issues cause unexpected errors. The open defects can be changed to Accepted.
Are there any sample tests performed by inspector(s) on part of the package that meets UT acceptance criteria?		Mandatory	Inspector(s) may select/create TCs to verify results of UT randomly (see Guideline for sample of UT quick test).
If Yes, are all the test passed without errors?			Because inspector(s) will perform sample tests on part of the package that meets UT acceptance criteria, no defect should occur here.
<Add more rows if needed>			

Check Item	Assessment	Priority	Notes

*** Summary**

Number of 'OK' items	**0**
Number of 'NOK' items	**0**
Number of 'N/A' items	**0**

*** Comments**

*** Suggestion**

[ ] - Pass
[ ] - Not pass
[ ] - Other