Software Testing Methodologies

Defect Management

Test Execution & Defect Logging

Test Execution is a actual running activity of test specifications on developed system. This can be done via manual testing or automation testing tool.

Defect Logging is discovering, documenting and acknowledging a valid defect

Find Defect - Discover defects before they become major problems

Report Defect - Report defects to developers so that they can be resolved

Acknowledge Defect - Obtain development acknowledgement that the defect is valid and should be addressed

Defect Reporting

Recording the defects identified at each stage of the test process is an integral part of a successful life cycle testing approach. The purpose of this activity is to create a complete record of the discrepancies identified during testing

Defects are recorded for four major purposes:

- To ensure the defect is corrected
- To report status of the application
- To gather statistics used to develop defect expectations in future applications
- To improve the software development process

Defect Reporting

Most project teams use some type of tool to support the defect tracking process. This tool could be as simple as a white board or a table created and maintained in a word processor, or one of the more robust tools available today on the market.

Tools marketed for this purpose usually come with a number of customizable fields for tracking project specific data in addition to the basics. They also provide advanced features such as standard and ad-hoc reporting, e-mail notification to developers or testers when a problem is assigned to them, and graphing capabilities.

At a minimum, the tool selected should support the recording and communication of all significant information about a defect.

Defect Reporting

For example, a defect log could include:

Defect ID number

Descriptive defect name and type

Source of defect - test case or other source that found the defect

Method – SDLC phase of creation

Phases – SDLC phase of detection

Component or program that had the defect

Defect Reporting

- Defect severity
- Defect priority
- Defect status (e.g., open, fixed, closed, user error, design, and so on) - more robust tools provide a status history for the defect
- Date and time tracking for either the most recent status change, or for each change in the status history
- Detailed description, including the steps necessary to reproduce the defect
- Screen prints, logs, etc., that will aid the developer in resolution process
- Stage of origination
- Persons assigned to research and correct the defect

Severity versus Priority

Based on predefined severity descriptions, the test team should assign the severity of a defect objectively. For example a “severity one” defect may be defined as one that causes data corruption, a system crash, security violations, etc.
Severity levels should be defined at the start of the project so that they are consistently assigned and understood by the team. This foresight can help test teams avoid the common disagreements with development teams about the criticality of a defect. In large projects, it may also be necessary to assign a priority to the defect, which determines the order in which defects should be fixed.
The priority assigned to a defect is usually more subjective as it may be based on input from users regarding which defects are most important, resources available, risk, etc.

A Sample Defect Tracking Process

The steps below describe a sample defect tracking process. Depending on the size of the project or project team, this process may be substantially more complex

1. **Execute test and log any discrepancies-** The tester executes the test and compares the actual results to the documented expected results. If a discrepancy exists, the discrepancy is logged as a “defect” with a status of “open.” Supplementary documentation, such as screen prints or program traces, is attached if available

2. **Determine if discrepancy is a defect -** The Test Manager or tester reviews the defect log with an appropriate member of the development team to determine if the discrepancy is truly a defect, and is repeatable. If it is not a defect, or repeatable, the log should be closed with an explanatory comment.

A Sample Defect Tracking Process

3. **Assign defect to developer.-** If a defect exists it is assigned to a developer for correction. This may be handled automatically by the tool, or may be determined as a result of the discussion in step 2

4. **Defect resolution process -** When the developer has acknowledged the defect is valid, the resolution process begins. The four steps of the resolution process are:

A. **Prioritize the correction** - Three recommended prioritization levels are: **"critical"**, **"major"**, and **"minor"**. "Critical" means there is a serious impact on the organization's business operation or on further testing. "Major" causes an output of the software to be incorrect or stops or impedes further testing. "Minor" means something is wrong, but it does not directly affect the user of the system or further testing, such as a documentation error or cosmetic GUI error

A Sample Defect Tracking Process

B. **Schedule the correction-** Based on the priority of the defect, the correction should be scheduled

C. **Correct the defect -** The developer corrects the defect, and upon completion, updates the log with a description of the correction and changes the status to "Corrected" or "Retest". The tester then verifies that the defect has been removed from the system. Additional regression testing is performed as needed based on the severity and impact of the correction applied. In addition, test data, checklists, etc., should be reviewed and perhaps enhanced, so that in the future this defect will be caught earlier. If the retest results match the expected results, the tester updates the defect status to "closed." If the problem remains, the tester changes the status back to "Open" and this step is repeated until closure

A Sample Defect Tracking Process

D. **Report the resolution -** Once the defect has been corrected and the correction verified, appropriate developers, users, etc., need to be notified that the defect has been corrected, the nature of the correction, when the correction will be released, and how the correction will be released. As in many aspects of defect management, this is an area where an automated process would help. Most defect management tools capture information on who found and reported the problem and therefore provide an initial list of who needs to be notified. Computer forums and electronic mail can help notify users of widely distributed software. Test Reports are issued periodically throughout the testing process to communicate the test status to the rest of the team and management. These reports usually include a summary of the open defects by severity or priority. Additional graphs and metrics can also be provided to further describe the status of the application.

Thankyou!