BEST PRACTICE
Developing Testing Methodologies

The eight considerations listed below provide the framework for developing testing tactics. Each is described in the following sections.

- Acquire and study the test strategy
- Determine the type of development project
- Determine the type of software system
- Determine the project scope
- Identify the tactical risks
- Determine when testing should occur
- Build the tactical test plan
- Build the unit test plans

Acquire and Study the Test Strategy

A team familiar with the business risks associated with the software normally develops the test strategy, and the test team develops the tactics. Thus, the test team needs to acquire and study the test strategy, focusing on the following questions:

- What is the relationship of importance among the test factors?
- Which of the high-level risks are the most significant?
- Who has the best understanding of the impact of the identified business risks?
- What damage can be done to the business if the software fails to perform correctly?
- What damage can be done to the business if the software is not completed on time?

Determine the Type of Development Project

The type of project refers to the environment in which the software will be developed, and the methodology used. Changes to the environment also change the testing risk. For example, the risks associated with a traditional development effort are different from the risks associated with off-the shelf purchased software. Table 7-3 illustrates characteristics and testing tactics that can be used for different types of projects.

Table 7-3 Characteristics and Test Tactics for Different Project Types

Project Type	Characteristics	Test Tactics
Traditional system development (and most perfective maintenance)	Uses a system development methodology User knows requirements Development determines structure	Test at end of each task, step and phase Verify that specs match need Test function and structure
Iterative development, prototyping, CASE	Requirements unknown Structure predefined	Verify that CASE tools are used properly Test functionality
System maintenance	Modify structure	Test structure Works best with release methods Requires regression testing
Purchased or contracted software	Structure unknown May contain defects Functionality defined in user documentation Documentation may vary from software	Test functionality Verify functionality matches need Test fit into environment

Determine the Type of Software System

The type of software system refers to the processing that will be performed by that system. There are sixteen different software system types; however, a single software system may incorporate more than one of these types. Identifying the specific combinations of software making up the project can help analyze lessons learned on past projects with similar types of software.

Batch (General)	Can be run as a normal batch job and makes no unusual hardware or input-output actions (e.g., payroll program and wind tunnel data analysis program).
Event Control	Processes real-time data from external events, such as a computer program that processes telemetry data.
Process Control	Receives data from an external source and issues commands to that source to control its actions based on the received data.
Procedure Control	Controls other software; for example, an operating system that controls execution of time-shared and batch computer programs.
Advanced Mathematical Models	Resembles simulation and business strategy software, but has the additional complexity of heavy use of Mathematics.

Message Processing	Handles input and output messages, processing the text, or information contained therein.
Diagnostic Software	Detects and isolates hardware errors in the computer where it resides, or in other hardware that can communicate with that computer.
Sensor and Signal Processing	Similar to message processing, but it requires greater processing to analyze and transform the input into a usable data processing format.
Simulation	Simulates an environment, mission situation, or other hardware. Uses inputs from these to enable a more realistic evaluation of a computer program or a piece of hardware.
Database Management	Manages the storage and access of (typically large) groups of data. Such software can also prepare reports in user-defined formats based on the contents of the database.
Data Acquisition	Receives information in real-time and stores it in some form suitable for later processing; for example, software that receives data from a space probe and files it for later analysis.
Data Presentation	Formats and transforms data, as necessary, for convenient and understanding displays; typically, such displays would be for some screen presentation.
Decision and Planning Aids	Uses artificial intelligence techniques to provide an expert system to evaluate data and provide additional information and consideration for decision and policy makers.
Pattern and Image Processing	Generates and processes computer images; such software may analyze terrain data and generate images based on stored data.
Computer System Software	Provides services to operational computer programs (i.e., coordinates processing of components required to meet need).
Software Development Tools	Provides services to aid in the

	development of software (e.g., compilers, assemblers, static and dynamic analyzers).

Determine the Project Scope

The project scope refers to the totality of activities to be incorporated into the software system being tested – the range of system requirements and specifications to be understood. The scope of the testing effort is usually defined by the scope of the project. New system development has a much different scope from modifications to an existing system. When defining the scope, consider the following characteristics and then expand the list to encompass the requirements of the specific software system being tested.

New Systems Development

- Automating manual business process?
- Which business processes will or won't be affected?
- Which business areas will or won't be affected?
- Interfacing to existing systems?
- Existing systems will or won't be affected?

Changes to Existing Systems

- Corrective only?
- Maintenance re-engineering standards?
- Correction to known latent defects in addition to enhancements?
- Other systems affected?
- Risk of regression?

Identify the Tactical Risks

Strategic risks are the high-level business risks faced by the software system. They are decomposed into tactical risks to assist in creating the test scenarios that will address those risks. It is difficult to create test scenarios for high-level risks.

Tactical risks are divided into three categories:

- Structural Risks

These risks are associated with the application and the methods used to build it.

- Technical Risks
  These risks are associated with the technology used to build and operate the application.
- Size risks
  These risks are associated with the magnitude in all aspects of the software.

Determine When Testing Should Occur

The previous steps have identified the type of development project, the type of software system, the type of testing, the project scope, and the tactical risks. That information should be used to determine the point in the development process at which testing should occur.

For new development projects, testing can, and should, occur throughout the phases of a project. For modifications to existing systems, any or all of these may be applicable, depending on the scope. Examples of test activities to be performed during these phases are:

Requirements Phase Activities

- Determine test strategy
- Determine adequacy of requirements
- Generate functional test conditions
- Design Phase Activities
- Determine consistency of design with requirements
- Determine adequacy of design
- Determine adequacy of the test plans
- Generate structural and functional test conditions

Program (Build) Phase Activities

- Determine consistency with design
- Determine adequacy of implementation
- Generate structural and functional test conditions for modules and units

Test Phase Activities

- Test application system
- Installation Phase Activities
- Place tested system into production

Maintenance Phase Activities

- Modify and retest

Build the System Test Plan

Using information from the prior steps, develop a System Test Plan to describe the testing that will occur. This plan will provide background information on the system being tested, test objectives and risks, the business functions to be tested, and the specific tests to be performed.

The Test Plan is the road map that will be followed when conducting testing. The plan is then decomposed into specific tests and lower-level plans. After execution, the results from the specific tests are rolled up to produce a Test Report.

Build the Unit Test Plans

During internal design, the system is divided into the components or units that perform the detailed processing. Each of these units should have an individual Test Plan. The plans can be as simple or as complex as the organization requires based on its quality expectations.

The importance of a Unit Test Plan is to determine when unit testing is complete. It is not cost effective to submit units that contain defects to higher levels of testing. The extra effort spent in developing Unit Test Plans, testing units, and assuring that units are defect free prior to integration testing can have a significant payback in reducing overall test costs.

References

Guide – CSQA Common Body Of Knowledge, V6.2