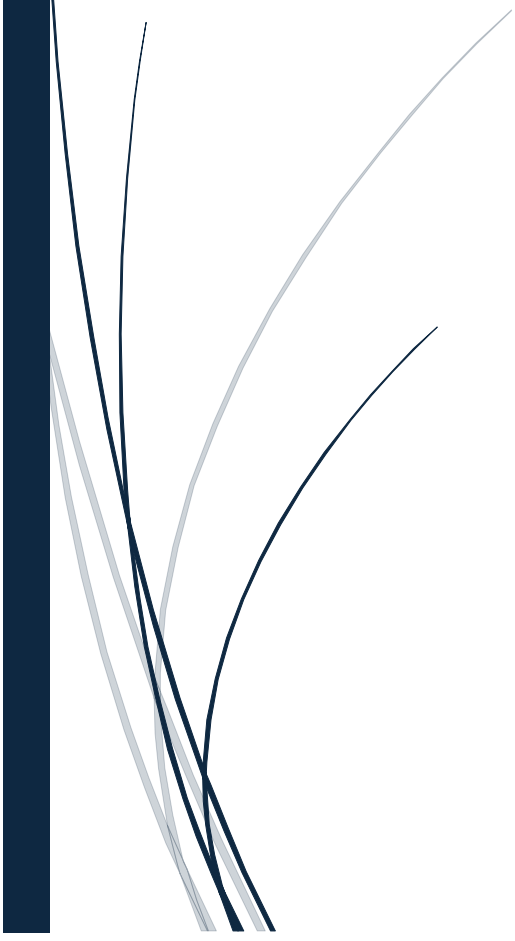
9th May, 2024

# Assignment 01

## Evolution Of Test Plan Template

**Abdul Aziz**
**FA21-BSE-058**



# Question 01:

Prepare and present in video form a report on the evolution of software test plans. The heart of every test plan is a set of effective test cases that determines its quality. During the evolution discuss the templates year wise and how its quality was improved. For every template you need to provide a proof of some test plan which was used in a company for a specific software/project.

# Answer:

## Test Plan:

A software test plan is a strategic document that dictates the entire testing process for a software development project.

## Evolution OF Test Plan:

### 1. Early Template (1980s – Early 1990s):

#### Structure:

This was very basic template of test plan (like a checklist we use).

#### Focus:

Simple verification that features were present and seemed to work on a basic level.

#### Quality Consideration:

These templates primarily addressed the most basic need to have a plan but lacked the detail to ensure comprehensive or thorough testing.

#### Template:

| SECTION | EXPLANATION |
|---|---|
| **INTODUCTION** | One or two sentences describing the project. |
| **FEATURE TO TEST** | A simple list of functionalities. |
| **TEST RESULT** | Columns for Pass/Fail next to each feature. |

#### Example:

Let's consider a company developing a basic word processing software in the late 1980s. Their test plan might have looked something like this:

- **Introduction:** Brief overview of the word processing software project.
- **Features to Test:** Basic functionalities like typing, saving, and printing documents.
- **Test Results:** Pass/Fail columns indicating whether each feature worked as expected.

## 2. Structured Template (Mid 1990s – 2000s):

### Structure:

Gained organization and formality, often driven by the rise of structured development methodologies.

### Focus:

Ensuring the plan aligned with requirements and had a clear purpose.

### Quality Considerations:

A more defined structure improved communication and set clearer expectations. Testers began thinking more deeply about the "why" of testing.

### Template:

| SECTION | EXPLANATION |
|---|---|
| **INTODUCTION** | Brief project overview. |
| **OBJECTIVES** | Concise statements of what testing aims to achieve. |
| **SCOPE** | Clear definition of what's included/not included. |
| **REFERENCES** | Linkage to formal requirements documents. |
| **TEST CASES** | Inputs, Steps, Expected Results for each test. |
| **ENVIRONMENT** | Basic description of hardware/software needed. |

### Example:

Imagine a software company in the late 1990s developing a customer relationship management (CRM) system. Their test plan might resemble this:

- **Introduction:** Overview of the CRM project, including its objectives.
- **Objectives:** Clear statements outlining what testing aims to achieve, such as validating customer data input and retrieval.
- **Scope:** Defined boundaries of testing, including modules like customer management, sales tracking, and reporting.
- **References:** Links to formal requirement documents detailing CRM functionality.
- **Test Cases:** Detailed steps and expected results for testing each CRM module.

## 3. Methodology-Driven Template (2000s – 2010s):

### Structure:

Reflects Agile and iterative development trends, with flexibility built in.

### Scope:

Adapting testing to the pace of development, outlining the types of tests used at different stages.

### Quality Considerations:

Explicitly considering testing methodology ensures a more targeted and efficient approach aligned with the project's overall development strategy.

### Template:

| SECTION | EXPLANANTION |
|---|---|
| **INTRODUCTION** | Brief project overview. |
| **OBJECTIVES** | Concise statements of what testing aims to achieve. |
| **SCOPE** | Clear definition of what's included/not included. |
| **TEST APPROACH** | Unit Testing, Integration Testing, System Testing, etc. |
| **TEST CASES** | May include prioritization (High, Medium, Low). |
| **ENVIRONMENT** | Basic description of hardware/software needed. |
| **ROLES & RESPONSIBILITIES** | Who performs what types of tests. |
| **RISKS** | Potential issues & contingency plans. |

### Example:

Suppose a company is developing a web-based e-commerce platform in the mid-2000s, following Agile methodology. Their test plan might reflect this:

- **Introduction:** Brief overview of the e-commerce project and its Agile development approach.
- **Objectives:** Statements aligning testing goals with Agile principles, emphasizing iterative feedback.
- **Scope:** Definition of testing phases (e.g., sprint testing), specifying which functionalities will be tested in each iteration.
- **Test Approach:** Description of unit testing, integration testing, and user acceptance testing procedures.
- **Roles & Responsibilities:** Assignment of testing tasks to development team members and QA testers.

## 4. Comprehensive & Metrics-Focused Template (2010s – Till Now):

### Structure:

Emphasize automation, continuous testing, and data-driven insights. Testers and developers collaborate closely.

### Scope:

Maximizing feedback loops, measuring effectiveness, and treating testing as an integral part of software quality assurance.

### Quality Considerations:

Comprehensive templates drive a proactive approach to quality, measure progress, and enable informed decision-making about test strategy adjustments over time.

### Template:

| SECTION | EXPLANATION |
|---|---|
| **INTRODUCTION** | Brief project overview. |
| **OBJECTIVES** | Concise statements of what testing aims to achieve. |
| **SCOPE** | Clear definition of what's included/not included. |
| **TEST APPROACH** | May include automation, continuous testing, security testing, performance testing. |
| **TEST CASES** | Highly detailed, may link to code repositories or test scripts. |
| **ENVIRONMENT** | Cloud simulators, different device types, etc. |
| **ROLES & RESPONSIBILITIES** | May include developers in some testing activities. |
| **RISK & MITIGATION** | Potential issues & contingency plans. |
| **TEST DATA MANAGEMENT** | How data is generated, secured, used. |
| **TOOL INTEGRATIONS** | Bug trackers, test management tools. |
| **METRICS & REPORTING** | How progress and results will be communicated. |

## Example:

Consider a mobile app development company in the current decade, focusing on a ride-sharing application. Their test plan might encompass:

- **Introduction:** Overview of the ride-sharing app project, emphasizing its continuous delivery model.
- **Objectives:** Statements outlining quality goals, such as ensuring app stability and security.
- **Scope:** Description of testing environments (e.g., iOS, Android) and supported device types.
- **Test Approach:** Incorporation of automated regression testing, performance testing, and security testing.
- **Test Cases:** Detailed scenarios covering various user interactions and edge cases.
- **Tool Integrations:** Usage of bug trackers, test management tools, and analytics platforms.
- **Metrics & Reporting:** Defined metrics for tracking test coverage, defect density, and user satisfaction.

## Key Differences in the Evolution of Software Test Plan Templates:

| Template Evolution Phase | Key Changes | Benefits |
|---|---|---|
| **Early Templates (1980s - Early 1990s) to Structured Templates (Mid 1990s - 2000s)** | • Introduction of formal sections (Objectives, Scope, References). | • Better alignment with requirements.<br>• Improved clarity and organization. |
| **Structured Templates to Methodology-Driven Templates (2000s-2010s)** | • Explicit Test Approach section.<br>• Inclusion of test prioritization.<br>• Focus on roles and responsibilities.<br>• Consideration of risks. | • Targets testing to match project methodology (Agile, Waterfall, etc.).<br>• Improves the efficiency of test efforts.<br>• Encourages risk management. |
| **Methodology-Driven Templates to Comprehensive & Metrics-Focused Templates (2010s - Present)** | • Emphasis on test automation and continuous testing.<br>• Inclusion of non-functional testing (performance, security).<br>• Detailed test environment specifications.<br>• Focus on test data management.<br>• Integration with development and reporting tools.<br>• Defined test metrics. | • Faster feedback loops.<br>• Addresses holistic quality considerations.<br>• Promotes data-driven decision-making.<br>• Treats testing as part of the software quality lifecycle |

## Summary:

In the early days, testing was basic, like using checklists to ensure basic functionality. Then, it became more structured with formal documents outlining objectives and scope. As projects evolved, testing adapted, specifying types of testing for efficiency. Today, testing is comprehensive, using automation and examining all aspects of software, including performance and safety. Detailed test plans track progress and aid decision-making.