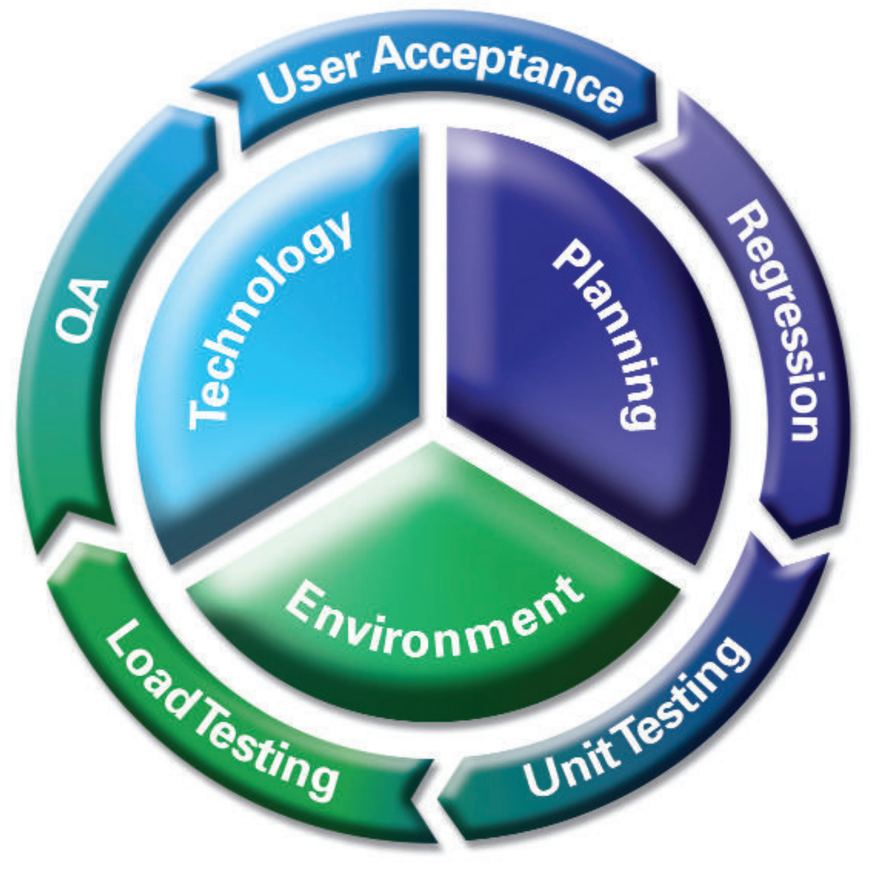
Software Testing - Best Practice

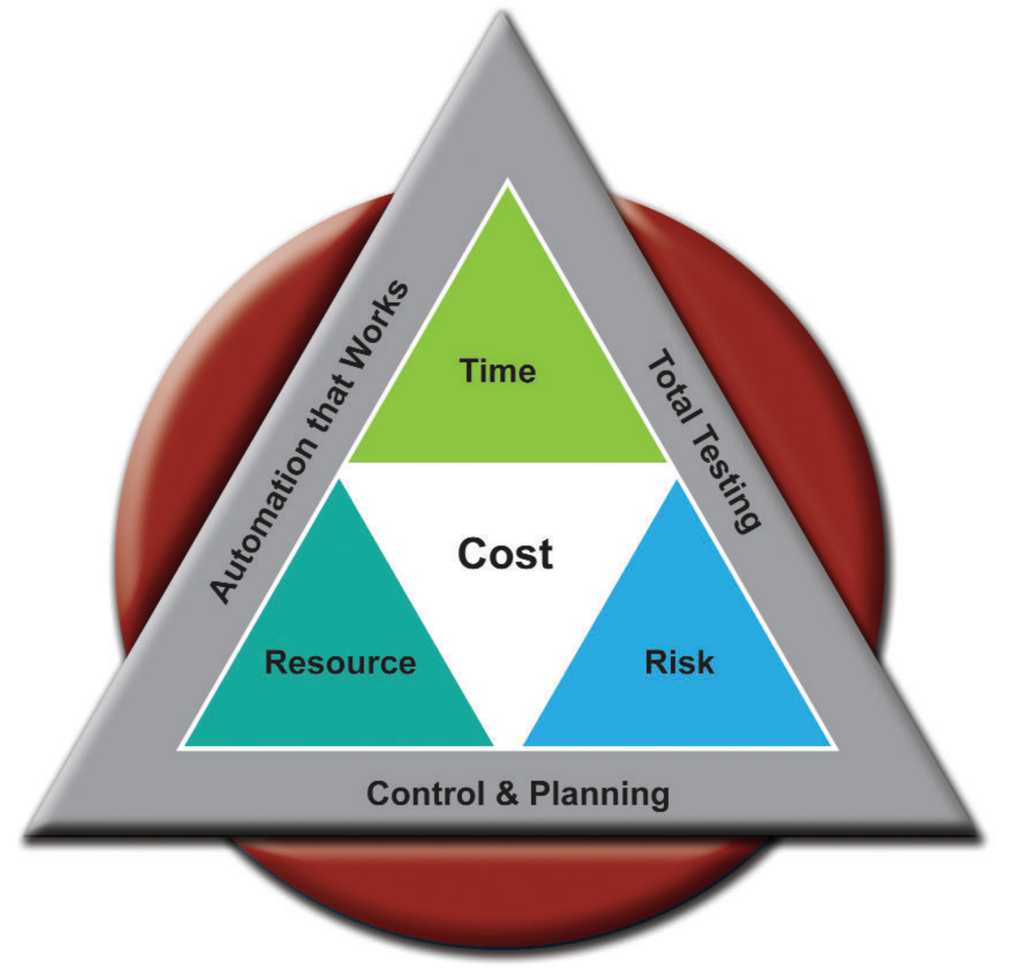
Original Software

The Testing Lifecycle



Correctly deployed software test automation solutions can empower all players in the development process by intuitively providing the information they need. The benefits to development time scales, user confidence and ongoing maintenance will be immense.

The Quality Conundrum



Resources, risk and application time-to-market are often in conflict as IS team strive to deliver quality applications with their budgetary constraints. This is the quality conundrum.

Plan

A test plan should define the optimum process of validating that the software is fit for purpose and meets the requirements. To help meet these objectives consider the following……

Project
Business objectives
Timescales - is this a constraint?
Market forces - competition and time to market?
Benefits - what will the business gain
User needs - the real user requirement
Resources
Scope

Functional Requirements
Use cases - how will the software be used
Features - an excellent building block for a test case
Specifications
Performance - in realistic business terms

Non Functional Requirements
Usability
Environment - can you build the same in test?
Data - underlying, source of, input
Compatibility - other systems and interfaces.
Designed with testing in mind
Risk - which are key system functions

Test Cases Design
Data - background and scenario
Positive and negative cases - error handling
Scenarios - testing each requirement
Process steps - depending on time & tester knowledge
- Detailed instructions?
- User flexibility?

Expectation - the correct results
Verification items - how to confirm the correct result
- Visible
- System - eg database

Automation - can & should this be automated?
Risk - how important is this test

Apply to
- Unit testing
- System Testing
- User Acceptance testing

Hints. Engage as early as possible in the requirements and design process to capture key elements of test scenarios at the source. Look beyond the obvious, include all outputs not just the user-visible.

Data

Data drives testing. Think about the following when defining data needs and your approach. Poor data will result in poor testing, wasted time and lower quality.

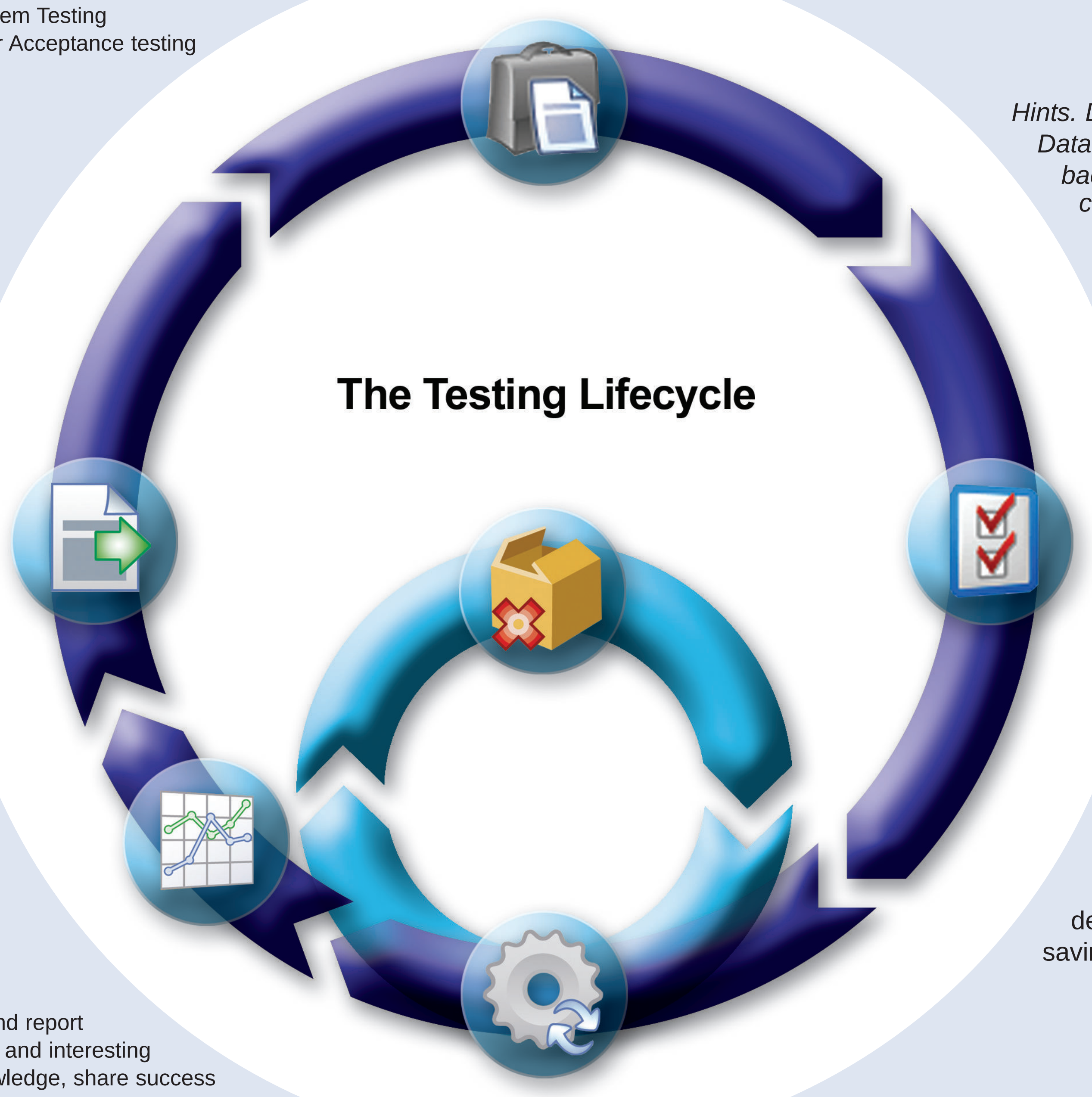
Data Needs
Scenarios - common ground

Data Sources
Existing production data?
Conversion from prior system
Extraction of subset from production
Generation - enter from scratch
Pair-wise/all pairs - testing combinations

Analyse current data - what is in use now?
Coverage - getting a good mix
Common scenarios and rare values
Consistency - especially is subset, must be intact
Positive and negative - separate good & invalid cases
Interface simulation - especially inbound interfaces
Data confidentiality and sensitivity - obscure or 'scramble'
sensitive ex-live data
sensitive ex-live data

Data Management
Environments
Back up strategy
Checkpoint and roll-back approach
Business rule checking
Matching to scenarios
Security
Sharing - across teams, inside the team

*Hints. Data is an asset. Where possible develop an approach for Data Protection that enables data to be kept and reused, matching background and input data with scenario results - this will save considerable time and effort.
Watch out for legislative factors - confidentially, security, improper use.*



Report

Data from testing is the raw material for improvement. Get as much detail as possible in issue diagnosis, stand back from the numbers to find ways to improve your processes.

Diagnostics - more the better to aid reproduction
Screen capture
Process
Input
Data
Deviance from requirement
Database effects
Environment

Compliance - keep proof and evidence
Steps
Pass and fail
User/tester
Environment - under what conditions
Traceability - who, what, when
Format of storage - method and access

Metrics
Test counts
Issue counts
Defect counts
Rework
Defect types
Percentage completion
Time and effort
Failure frequency

The Team
Set targets
Measure and report
Make it fun and interesting
Share knowledge, share success
Share the problems

Management informed
Agree 3-5 key facts
Report the key facts at least one a week
With any problem, present a solution

The key to successful test management is like navigation - you need to know where you are now. Distribute task and issue updating into the hands of the testing workforce so management does not become a data entry exercise.

Execute

Good testing is about knowing two things - what you are trying to achieve and what actions you actually performed. It is important to capture all the possible data to enable development teams to understand and reproduce errors easily, saving everyone's time.

Steps:
Actions and expectations - proving the requirement
Pass/Fail - track as progressing

Verification - where to look for the answer
User interface
Component structure
Tiers
Database
Interfaces
Logs

Capture - track what actually happened
Process
Input
Actual

Results - for issue diagnosis package and audit
Visible
Database
Interfaces
Performance

Regression - re-use passed tests?
Add to regression pack
Automation - keep it simple, minimise maintenance
Data pack - build and keep this in parallel
Sequence and dependencies

Issues
Track issues detected
Causes - categories for analysis
Types of issue
Severity - impact on testing and on the system
Affected objects
Track re-tests

Hints. Use tools if possible to capture what testers actually do, it will save time in diagnosing and for audit. Manage data to match the scenario. Review tests for regression suitability.

Testing Checklist

Plan:
- Define **scope** of testing
- Understand **Business Objectives**
- Define **success criteria**
- Assess **resource** requirements
- Agree **timescales**

Prepare:
- **Input data**
- **Processes**
- **Expectations**

Define:
- **Environment**
- Test **Data** (data extraction)
- **Applications**

Perform Tests:
- **Unit** Testing
- **QA** Testing
- **Load** Testing
- **User Acceptance** Testing
- **Regression** Testing

Check/Verify Results:
- **Database**
- **Visual Layer**
- **Reports**
- **Performance**

Revise & Improve:
- **Build Knowledge**
- **Increase Coverage**
- **Feedback** to Developers

Report:
- **Error – Diagnostics**
- **Management & Auditing**
- **Metrics**
- **Compliance**

Store:
- For **Re-use**
  Audit ready

more information can be found at **www.origsoft.com**