



## Testing Deliverables Hierarchy

The methodology uses a consistent set of deliverables for each element tested. The deliverables hierarchy provides a means to describe the relationship between each of these testing deliverables.

**Test Approach:** Defines the methods and techniques used to test the element at that particular testing level. The approach is defined well in advance of execution of the actual test.

**Test Model:** A composite deliverable used to manage the Test Plan and Test Scripts. There is a test model for each level of testing (Component, Assembly, and Product).

**Test Plan:** A composite deliverable used to manage Test Conditions and Expected Results and the Test Cycle Control Sheet. The Test Plan exists for each level of testing (Component, Assembly, and Product).

**Test Scripts:** The Assembly Test Scripts define the steps, input data, and expected output for a number of test conditions. Scripts exist for each level of testing (Component, Assembly, and Product).



## Testing Approach Template Description

- [1] Product** - The name of the product for which this deliverable was produced.
- [2] Release** - The specific product release number for which this deliverable was produced.
- [3] Platform** - The type of platform (operating system and hardware) on which the product was designed to run.
- [4] Configuration** - A description of how the product is configured; for example, a specific application setting, or a tool that was used to manage a group of application settings.
- [5] Prepared by/Date** - The initials of the person who originally prepared this deliverable, and the date on which the deliverable was prepared.
- [6] Approved by/Date** - The initials of the person who approved the deliverable (if applicable), and the date on which it was approved.
- [7] Version** - The version number of the deliverable, indicating how many times it has been revised.
- [8] Status** - An indicator of whether the deliverable has been approved by a project manager or not.
- [9] Overview** - A brief description of the testing stage.
- [10] Test Objectives and Scope** - A statement of purpose and goals for the testing stage---This section expands on and provides more detail than the test strategy.
- [11] Risks** - The risks and management approach.
- [12] Regression Testing Approach** - A description of how regression testing within this stage should be planned and performed - This section expands on and provides more detail than the test strategy.
- [13] Test Environment Requirements** - The hardware, software, application configurations and physical arrangements that enable test preparation and execution - The environment description encompasses the approach to creating, maintaining and sharing the test data together with the conditions and constraints that testing tools must satisfy to meet the test requirements stated above. This section expands on and provides more detail than the test strategy.
- [14] Metrics** - The measurements to be gathered to judge effectiveness, facilitate continuous improvement, track progress etc.---This section expands on and provides more detail than the test strategy.



## Department of Education Student Financial Assistance

**[15] Name** - The name of the person signing off on the completed test stage.

**[16] Date** - The date on which the test stage was signed off.

**[17] Cell Leader** - The leader of the work cell responsible for the test stage.

**[18] Test Activity** - A subset of the test stage (for example, "Execute Assembly Test").

**[19] Entry/Exit Criteria** - The entry and exit criteria applicable to the specified test activity.

**[20] Signed off by** - The name of the person signing off on the specified entry or exit criterion.

**[21] Date** - The date on which the specified entry or exit criterion was signed off.

**[22] Test Resources and Work Plan** - A detailed work plan showing how all planning, preparation, environment, execution, and management tasks, assigned resources, and budgets should be developed.



## Testing Approach Template Sample

<b>IPT Name:</b>		
<b>Deliverable Name:</b> Testing Approach Template		<b>Date Completed:</b>
<b>Contact Information</b>		
	Name	Channel Unit
IPT Sponsor		
Channel Task Manager		
CIO Task Manager		
Contractor Task Manager		
<b>Task Order Number:</b>		

### Test Approach

[1] **Product:** Library Management System (LMS)  
 [2] **Release:** 1.0  
 [3] **Platform:** Windows NT/SQL Server  
 [4] **Configuration:** Standard

[5] **Prepared by:** PMW                      **Date:** 8/15/1998  
 [6] **Approved by:** CMC                      **Date:** 8/15/1998                      [8] **Status:** Approved

### [9] Overview

This document contains the following sections:

- Test Objectives and Scope
- Risks
- Regression Testing Approach
- Test Environment Requirements
- Metrics
- Entry/Exit Criteria
- Test Resources and Work Plan



## [10] Test Objectives and Scope

### Objectives

Successful completion of the assembly test ensures that the assemblies meet the Application Design. The objective is to find and correct problems in the:

- Interactions between the partitioned business components of LMS
- Interactions between the partitioned business components of LMS and the technical infrastructure

### Scope

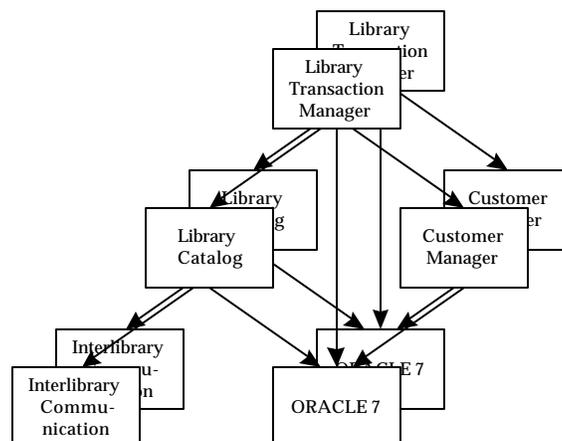
The partitioned business components considered are:

- LibraryTransactionManager
- LibraryCatalog
- CustomerManager

The technical infrastructure components considered are:

- ORACLE7 Database Management System
- Custom system interface for interlibrary communication

The diagram below summarizes the component interaction diagram. (Multiple interactions between components are shown as a single arrow.)





ID	Contents	Remarks
1	LibraryCatalog InterlibraryCommunication	ORACLE7 stubbed out
2	LibraryCatalog InterlibraryCommunication ORACLE7	
3	CustomerManager ORACLE7	
4	LibraryTransactionManager LibraryCatalog CustomerManager InterlibraryCommunication ORACLE 7	

The test conditions will be defined as follows:

- Reuse the component test conditions.
- Add test conditions as necessary to obtain 100% message path coverage.

The test cycles will be organized as follows, for each assembly:

- Cycle 1: test conditions that exercise the most frequent paths
- Cycle 2: test conditions that exercise all other legal paths
- Cycle 3: test conditions that exercise the error and exception handling logic

All cycles will be independent to minimize the overall calendar time required to test. In addition, each cycle will be run three times (i.e., three passes):

- The objective of pass 1 is to get through the test as quickly as possible, finding as many defects as possible and implementing workarounds where needed.
- The objective of pass 2 is to regression test the defects fixed from pass 1, and determine if the pass 1 workarounds caused any more defects.
- The objective of pass 3 is to regression test defects fixed from pass 2; no defects should be found.

By planning three passes, you are able to build in regression test runs to ensure that defects are completely fixed and that the fixes did not break anything else.

### [11] Risks

The following list must be included in the risk watch list during assembly test.



<b>Name</b>	<b>Inadequate component test</b>
Context	The scope of the assembly test should be limited to integration problems. Each partitioned business component (and its modules) should have been sufficiently tested before.
Symptoms	Problems are traceable to individual partitioned business components, rather than the interactions between them.
Consequences	Significant schedule slippage occurs as defects must be corrected before the related assembly test can proceed.
Detection	The testing metrics count more than five defects/workday imputable to component test.
Correction	Component testing is performed again with more stringent test criteria (e.g., escalate from statement coverage to decision coverage) before assembly test proceeds.
Avoidance	The yield from component test is monitored to ensure that it is within the bounds found on similar projects.

<b>Name</b>	<b>Inadequate configuration management</b>
Context	Assembly test brings together components from various sources. In addition, bug fixes make it necessary to introduce new versions. These manipulations should not perturb the test.
Symptoms	Problems are traceable to incorrect configurations, rather than content problems.
Consequences	Test productivity decreases as resources spend their time on activities that add no value.
Detection	Issue Log contains more than one problem/day related to configuration management.
Correction	Authority to move components into the assembly test environment is centralized.
Avoidance	Before they are implemented, all proposed bug fixes and design modifications are jointly reviewed by the application architect, the work cell leaders, and the test manager.

## **[12] Regression Testing Approach**

The three pass approach for assembly test will facilitate regression testing of defects found in assembly test. In addition, the entire assembly test model will be documented, repeatable and automated to be easily re-executed for each pass.

The component test environment must be maintained to regression test defects found in assembly test. In addition, the component test models must be documented and repeatable.

For each code fix, a complete component test will be re-executed. Any new conditions created as result of implementing a fix will be added to the existing set of test conditions.

The details of the fix-it process are described in the Issue Control Process in the LMS Test Strategy Deliverable.

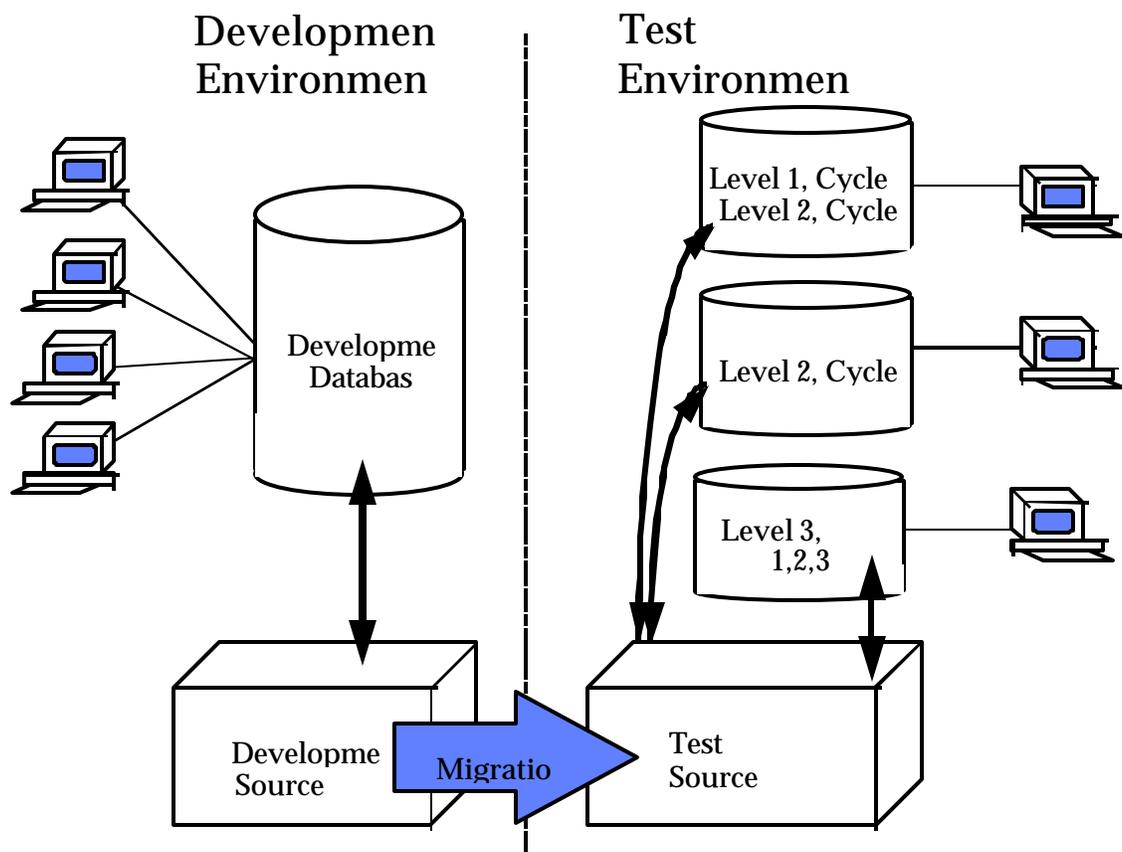


## [13] Test Environment Requirements

### Technical Configuration

The assembly test environment will resemble the actual production environment as closely as possible. All hardware and packaged software used in the creation of the assembly test environment will duplicate those chosen for the production environment. This includes hardware and software configuration for workstations, LAN setups, and SQL DB setups.

The assembly test environment will be logically separated from the component test environment. The development code will be migrated to the assembly test environment.



### External Interfaces

The Interlibrary Communication system interface will be imported from the Technical Infrastructure test environment. The Database Management System will be installed and configured by the Environment Support team.



## Test Data Management

The common test data will be copied to the assembly test environment and modified as needed by the development cell to satisfy all the assembly test conditions.

After each successful execution of a cycle, a database backup will be made by the test executor. This backup will serve two purposes. It can be used as evidence of a successful execution of the cycle. It can also be used to restart a test execution after a certain sequence of upstream cycles. The details of backup procedures in using the SQL utility will be provided by the technical support cell prior to the start of the assembly test execution.

## Source Environment

The code for assembly test will be migrated from the component test environment as defined in the migration procedures. Initially, the assembly test environment will contain only a subset of the components for LMS, since component test of all components will not be completed at this point. As components are component tested, the assembly test team will pull them into the assembly test environment.

## Automation

- Configuration Management - **PVCS** will be used to manage the source code and the test models.
- Test Planning - **Test Plan Management System (TPMS)** will be used to define and maintain the relationships between components of a test plan (i.e., test schedule, test execution tracking, test cycles, test scripts, test conditions, test condition generation, input data, and expected results).
- Test Execution - **WinRunner** will be used to support and automate the conduct of tests (i.e., extract input data and expected results from the repository, load this data into appropriate test execution tools, automate the execution of the test).
- Test Coverage Measurement - **LDRA Toolset** will be used to document the parts of each assembly that have been executed during the test. This test management and quality management tool ensures that the paths intended to be tested have actually been executed.
- Source Code Debugger - **Sentinel** from AIB Software Corp will be used as a dynamic analyzer. It provides information about the activity of a program. It assists in code tracing and edition; it allows the entering of break points to step through a program, to track the progress of execution, and to identify errors interactively.
- Emulation Tools - **Cantata** from Quality Checked Software will be used as a test driver. It provides target platform architecture emulation (including both custom infrastructure and system software product components) and stubs that simulate modules and components in a minimal fashion. It also provides harnesses and drivers that emulate the context in which a module or a component will be called in the production environment.



- Problem Management - **PTS** (Problem Tracking System) will be used to track all issues during assembly test. It records relevant information from detection and documentation to resolution (e.g. Problem Tracking, Impact Analysis, Statistical Analysis).
- Data Management - SQL server database utilities will be used to make backups as needed, and to quickly restart the testing process.

### **Modification of System Parameters**

There will be one experienced analyst responsible for making changes to system tables and the system date. Any necessary changes will be documented through the Issue Control Process and prioritized.

### **Environment Cleanup**

At the end of each pass of execution, a backup of the database will be made. These backups will be stored on the network. There will be a directory for each pass named PASS1, PASS2, PASS3. Within each pass directory there will be a backup at the end of each cycle, named after the cycle number.

### **System Software Upgrades**

There are no software upgrades planned during the testing of the Library Management System.

### **Security**

Each test executor will require a unique user ID. Generic IDs (i.e., ATE1, ATE2, etc.) will be established by the technical support cell so that execution is not stopped because a test executor is sick, on vacation, etc.

## **[14] Metrics**

The following metrics will be collected and evaluated throughout assembly test:

### **Stage Containment Repair Effort Percentage**

- Definition:  $(\text{Days spent fixing defects from a particular stage} / \text{Original days spent in the stage}) \times 100$
- Target: Less than 10%
- Frequency of collection and evaluation: Weekly



### Repair Effectiveness Percentage

- Definition:  $(\text{Total number of defects fixed correctly the first time within a specific stage} / \text{Total number of defect fixes attempted within the stage}) \times 100$
- Target: 95%
- Frequency of collection and evaluation: Weekly

### Test Planning Rate

- Definition:  $\text{Total number of conditions documented and signed off for this stage} / \text{Total number of workdays spent to date on test planning for this stage}$
- Target: To meet or exceed the planned productivity
- Frequency of collection and evaluation: Weekly

### Test Preparation Rate

- Definition:  $\text{Total number of cycles that have been scripted and signed off} / \text{Total number of workdays utilized to date for scripting this test stage}$
- Target: To meet or exceed the planned productivity
- Frequency of collection and evaluation: Weekly

### Test Execution Rate

- Definition:  $\text{Total number of cycles executed in the stage} / \text{Total number of workdays spent executing to date in this test stage}$
- Target: To meet or exceed the planned productivity
- Frequency of collection and evaluation: Daily

### Problem Rate

- Definition:  $\text{Total number of problems reported} / \text{Days of execution}$
- Target: Below or at the planned problem rate
- Frequency of collection and evaluation: Daily

### Problem Fix Rate

- Definition:  $\text{Total number of problems fixed} / \text{Days of execution}$
- Target: Above or at the planned fix rate
- Frequency of collection and evaluation: Daily

Refer to the Testing Metrics Job Aids for more information.

### ***Entry and Exit Criteria***

**Stage:** Assembly Test

**Exit Sign-Off**  
**[15] Name:**



[16] Date:  
[17] Cell Leader:

[18] Test Activity	[19] Entry/Exit Criteria	[20] Signed Off By	[21] Date
	<b>Entry Criteria:</b>		
	Assembly testing for release 3 of the Book Purchasing application must have finished. No assembly test configurations can co-exist.		
	The Interlibrary Communication system interface must have been upgraded to provide the protocol assumed by LMS.		
	<b>Exit Criteria:</b>		
	A configuration audit must have been completed on the source code and assembly test information after all test cycles run successfully.		

For entry and exit criteria that apply to all projects, refer to the Entry and Exit Criteria: Assembly Test Job Aid.

**[22] Test Resources and Work plan**

**Resources**

The assembly test conditions will be developed by two developers concurrently with the design of the application. One of them is named Assembly Test Lead. All test conditions and expected results will be approved by the Test Manager.

The assembly test scripts will be developed and executed by a dedicated team of four drawn from the development cells. The Assembly Test Lead heads the team. The Test Manager signs off on the deliverables. The Project Manager approves the final report.

**Work Plan**

See the activities Plan Assembly Test and Prepare and Execute Assembly Test in the project plan.