Arcade Game Maker Product Line – Unit Test Plan Template

ArcadeGame Team

July 2003

Table of Contents

1 Overview 1
- 1.1 Identification 1
- 1.2 Document Map 1
- 1.3 Concepts 2
- 1.4 Readership 2

2 Template Plan 3

3 Analyses and Standards 5
- 3.1 Coverage Standards 5
  - 3.1.1 Functional 5
  - 3.1.2 Structural 5
- 3.2 Analyses 5
  - 3.2.1 Test suite construction 5
  - 3.2.2 Incremental test analyses 5
    - 3.2.2.1 Change Impact Analysis 6
    - 3.2.2.2 Diff 6

4 Modifying the Unit Test Plan 7
- 4.1 Test effectiveness 8

5 References and Further Reading 9

Revision Control Table				
Version Number	**Date Revised**	**Revision Type A-Add, D-Delete, M-Modify**	**Description of Change**	**Person Responsible**

V2.0	6/1/04	M	Responding to review comments	JDMcGregor

1 Overview

1.1 Identification

The Arcade Game Maker Product Line organization will produce a series of arcade games ranging from low obstacle count to high with a range of interaction effects. More detailed information can be found in the Arcade Game Maker scope document. This document describes how individual code assets are tested in the AGM product line.

1.2 Document Map

The Arcade Game Maker Product Line is described in a series of documents. These documents are related to each other as shown in Figure 1. This map shows the order in which the documents should be read for the first time. Once the reader is familiar with the documents, the reader can go directly to the information needed.

This is the unit test plan template. Product Line organizations use this document to capture how each code unit is tested. This document follows the outline provided in [McGregor 01b].

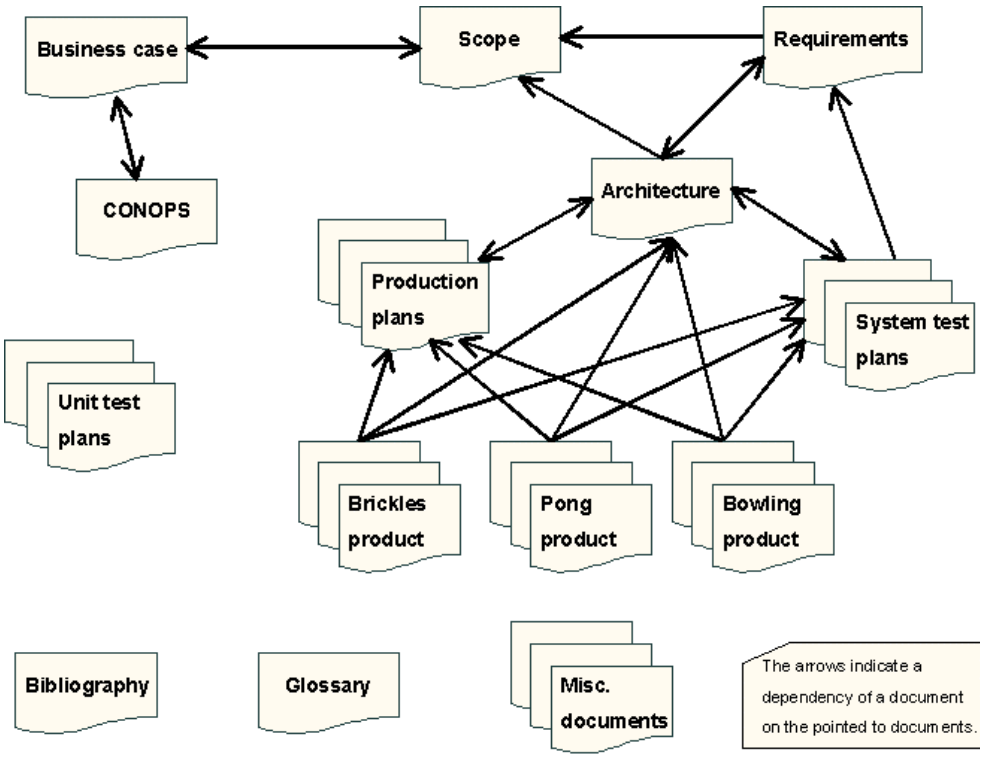

Figure 1- Document Map

1.3 Concepts

For definitions of basic concepts see the Glossary document.

1.4 Readership

This document is intended to provide some level of information to all of the stakeholders in the Arcade Game Maker framework but is primarily intended for core asset development teams. The unit test plan describes for a manager the resources that are needed to test an asset. Technical members of the organization can use the unit test plan template as the guide for developing a complete unit-specific test plan.

2 Template Plan

The AGM product line organization has decided to follow the IEEE 829 standard for test plans. The following template is filled in for each unit. See [McGregor 01b] for definitions of each section.

1. Introduction
2. Test Items
3. Tested Features

Increment	Functionality	Schedule
1		
2		
3		

4. Features Not Tested (per cycle)
5. Testing Strategy and Approach

    5.1 Syntax

    5.2 Description of Functionality

    5.3 Arguments for tests

    5.4 Expected Output

    5.5 Specific Exclusions

    5.6 Dependencies

    5.7 Test Case Success/Failure Criteria

6. Pass/Fail Criteria for the Complete Test Cycle
7. Entrance Criteria/Exit Criteria

The unit test phase runs concurrently with the unit development phase. The construction of test cases begins while the specification of the unit is being developed. Feedback from test case development allows the developer to see inconsistencies and ambiguities in the specification and to correct these.

The unit test phase is exited when the developer has implemented the required unit and it has passed all of the tests required by the standards defined in the test plan template.

8. Test Suspension Criteria and Resumption Requirements

Unit tests are suspended for one of two reasons:

1. the available functionality has been adequately tested and passed those tests but additional functionality remains to be developed, or
2. the available functionality has not passed the tests and the developer has sufficient information to make another development pass.

Tests are resumed when the developer has constructed additional functionality of revised the existing functionality.

9. Test Deliverables/Status Communications Vehicles
10. Testing Tasks
11. Hardware and Software Requirements

No special hardware is required for the tests of this cluster.

The DotUnit executable and the accompanying framework classes are required for building test classes.

12. Problem Determination and Correction Responsibilities

The developer has responsibility for corrections. The users of the cluster are responsible for reporting any problems to the listed owner of the cluster.

13. Staffing and Training Needs/Assignments
14. Test Schedules

Increment	Functionality	Schedule

15. Risks and Contingencies

3 Analyses and Standards

In this section we describe the techniques and the agreed-upon standards used to test each code asset in the AGM product line. These are justified and put into context in [McGregor 01b].

3.1 Coverage Standards

3.1.1 Functional

For each service on a component, a test case is constructed for every clause of the post-condition.

Test the unit invariant before and after each service invocation. This can be done in conjunction with the service test cases.

3.1.2 Structural

A test case is constructed for each sequence of statements. This effort can be optimized by first executing the functional tests with a code coverage tool running. Then only construct test cases to cover sequences of statements that were not covered by the functional test cases.

Be certain that this includes all exceptional sequences.

3.2 Analyses

3.2.1 Test suite construction

The AGM product line organization has decided to use the test case selection techniques described in [McGregor 01b]. Read that information before constructing test suites.

3.2.2 Incremental test analyses

After the initial test suites have been created, different techniques are used to maintain the test suites. Every time the unit is changed, apply these techniques.

3.2.2.1 Change Impact Analysis

Use the results from the Change Impact Analysis that was conducted by the developers. That analysis will have identified those portions of the asset that will be modified when the change is implemented.

3.2.2.2 Diff

A tool such as diff can be used to show exactly what the difference is between two versions of the unit. Then tests can be modified to address just those differences.

4 Modifying the Unit Test Plan

A specific unit test plan is modified every time the unit being tested is changed using techniques discussed in section 3.2. The generic unit test plan is modified when it is shown that the techniques are not producing effective test cases and the standards are not producing satisfactory results. That is the focus of this attached process.

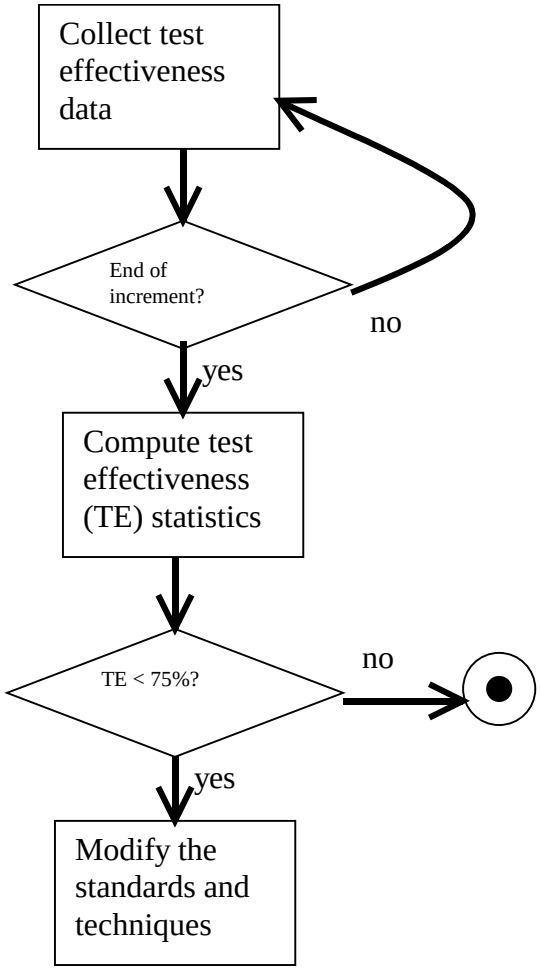

Figure 2-Attached process

4.1 Test effectiveness

Test effectiveness is measured by how many defects escape detection by the tests that are run. For the unit test phase, any defect found in a component after unit test counts against the test effectiveness of the unit test. It is computed as:

$$TE = \frac{TotalDefects - DefectsFoundAfterTest}{TotalDefects}$$

Defects are cataloged as they are identified and analyzed to determine their origin. At the end of a product line increment the test effectiveness is computed on a component by component basis. When the average TE goes below 75% the test coverage standards are made more comprehensive.

5 References and Further Reading

For references see the Bibliography document.