



Australian Government

Department of Education, Employment and Workplace Relations

ICAPRG404A Test applications

Release: 1

ICAPRG404A Test applications

Modification History

Release	Comments
Release 1	This Unit first released with <i>ICA11 Information and Communications Technology Training Package version 1.0</i>

Unit Descriptor

This unit describes the performance outcomes, skills and knowledge required to prepare test plans, write test procedures or scripts according to test plans, and maintain test plans and scripts.

Application of the Unit

This unit is relevant to those responsible for test plan preparation, execution, maintenance, and reporting of tests; as well as defect management in the software development life cycle.

The positions in this role include programmers, developers, testers and test analysts.

Licensing/Regulatory Information

No licensing, legislative, regulatory or certification requirements apply to this unit at the time of endorsement but users should confirm requirements with the relevant federal, state or territory authority.

Pre-Requisites

Not applicable.

Employability Skills Information

This unit contains employability skills.

Elements and Performance Criteria Pre-Content

Element	Performance Criteria
<i>Elements describe the essential outcomes of a unit of competency.</i>	<i>Performance criteria describe the performance needed to demonstrate achievement of the element. Where bold italicised text is used, further information is detailed in the required skills and knowledge section and the range statement. Assessment of performance is to be consistent with the evidence guide.</i>

Elements and Performance Criteria

1. Determine testing need in development	1.1 Identify testing role across software development life cycle 1.2 Identify testing types and testing tools 1.3 Recognise testing benefits, standard and terms
2. Prepare test plan document	2.1 Gather requirements to develop test plan 2.2 Analyse and identify test data using various test-case design techniques 2.3 Define and design test cases
3. Write and execute test procedures	3.1 Choose and adopt a unit test framework 3.2 Design and implement algorithm in test procedures 3.3 Perform test executions
4. Review test results	4.1 Record test results 4.2 Analyse test results 4.3 Produce test progress reports 4.4 Manage defects

Required Skills and Knowledge

This section describes the skills and knowledge required for this unit.

Required skills

- communication skills to:
 - liaise with developers on programming matters
 - liaise with managers or leaders on report and result matters
 - seek requirements and information from business and technical experts
- literacy skills to:
 - develop reports and documentation in relation to presentation of information
 - read and interpret program specifications developed by business and technical experts
- problem-solving skills to apply basic debugging techniques in the context of software and application development
- research skills to:
 - locate and interrogate complex and varied sources of information
 - source information from available sources
- technical skills to:
 - develop a small scale application
 - execute an application.

Required knowledge

- characteristics of programming language
- detailed knowledge of input and output requirements
- overview knowledge of software development life cycle
- processes and techniques related to small-size application development.

Evidence Guide

The evidence guide provides advice on assessment and must be read in conjunction with the performance criteria, required skills and knowledge, range statement and the Assessment Guidelines for the Training Package.

Overview of assessment	
Critical aspects for assessment and evidence required to demonstrate competency in this unit	Evidence of the ability to: • apply test case design techniques to develop test plan and test cases that adequately cover specific test requirements • design, implement and execute test procedures or scripts by using a unit test framework and an integrated development environment (IDE) • create test progress reports based on recorded test results.
Context of and specific resources for assessment	Assessment must ensure access to: • software development environment • specific test requirements, such as test-plan document • suitable code to be tested • word-processing software • appropriate learning and assessment support when required • modified equipment for people with special needs.
Method of assessment	A range of assessment methods should be used to assess practical skills and knowledge. The following examples are appropriate for this unit: • verbal or written questioning to key issues in testing software applications related to a specific project • evaluation of candidate's ability to create testing project and execute test cases in a specific context • observation of test cases development and documentation based on specific requirements.
Guidance information for assessment	Holistic assessment with other units relevant to the industry sector, workplace and job role is recommended, where appropriate. Assessment processes and techniques must be culturally appropriate, and suitable to the communication skill level, language, literacy and numeracy capacity of the candidate and the work being performed. Indigenous people and other people from a non-English speaking background may need additional support. In cases where practical assessment is used it should be combined with targeted questioning to assess required

	knowledge.
--	------------

Range Statement

The range statement relates to the unit of competency as a whole. It allows for different work environments and situations that may affect performance. Bold italicised wording, if used in the performance criteria, is detailed below. Essential operating conditions that may be present with training and assessment (depending on the work situation, needs of the candidate, accessibility of the item, and local industry and regional contexts) may also be included.

Testing types may include:	• acceptance testing • accessibility testing • alpha testing • beta testing • component testing • integration testing • performance testing • security testing • system testing • usability testing.
Testing tools may include:	• graphical user interface (GUI) test tools: • AutoHotkey • AutoIt • Selenium • White • load and performance tools: • BrowserMob • QTest • SiteBlaster • Xceptance LoadTest • static analysis tools: • Coverity • FindBugs • FxCop • PMD • test coverage tools: • JCover • NCover • test design tools: • ADL • AllPairs.java • DGL • Jenny

	• Multi • TSL • unit test tools: • csUnit • MbUnit • NUnit • OCUnit • xUnit.
Terms must include:	• defect • failure • quality • verification • validation.
Test-case design techniques may include:	• black box design techniques: • boundary value analysis • decision table test • equivalence partitioning • error guessing • desk checking • white box design techniques: • basic path testing • branch coverage • condition coverage • loop testing • statement coverage.
Unit test framework may include:	• cppUnit • jUnit • mbUnit • NUnit • ocUnit • pyUnit • xUnit.
Test procedures may include:	• fake • mock • stub • test class • test fixture • test method.

Unit Sector(s)

Programming and software development