Australian Government

Department of Education, Employment and Workplace Relations

# ICAPRG520A Validate an application design against specifications

**Release: 1**

# ICAPRG520A Validate an application design against specifications

## Modification History

| Release | Comments |
|---|---|
| Release 1 | This Unit first released with *ICA11 Information and Communications Technology Training Package version 1.0* |

## Unit Descriptor

This unit describes the performance outcomes, skills and knowledge required to check the software application design against specifications and apply validation techniques across the system life cycle.

## Application of the Unit

This unit applies to individuals employed in the area of software development responsible for verifying and validating software-design specifications.

They may be individuals who work as software project managers, testers, software engineers, system analysts and software developers.

## 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 |
|---|---|
| *Elements describe the essential outcomes of a unit of competency.* | *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.* |

# Elements and Performance Criteria

| | |
|---|---|
| 1. Evaluate software-requirement specification | 1.1 Review ***software requirement specifications document*** to ensure that requirements are completely specified and correct before software design begins<br>1.2 Validate the software requirement specifications document |
| 2. Create a proof-of-concept prototype | 2.1 Use rapid application development tools to create prototype system<br>2.2 Present prototype system for demonstration to appropriate person<br>2.3 Validate proof of concept |
| 3. Evaluate software design | 3.1 Determine if the design is complete, accurate, consistent and feasible<br>3.2 Validate ***software design document***<br>3.3 Validate ***database structure and elements***<br>3.4 Validate ***user interface*** (UI)<br>3.5 Review software risk analysis |
| 4. Evaluate source code | 4.1 Validate consistency between code and software design document<br>4.2 Validate logical structure and syntax using ***static analysis tools*** |
| 5. Evaluate testing requirements | 5.1 Review and validate test plans<br>5.2 Review and validate test cases |
| 6. Document validation | 6.1 Document results of validation exercise<br>6.2 Recommend current software design or itemise required alterations<br>6.3 Submit report to appropriate person for action |

# Required Skills and Knowledge

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

### Required skills

- analytical skills to determine audience needs
- communication skills to:
  - interact with developer to ensure proper implementation
  - provide leadership and motivation
- literacy skills to:
  - create technical documentation related to software design
  - read and interpret complex technical and non-technical information from a range of sources
- technical skills to use:
  - software design tools, such as unified modelling language (UML) tools
  - word-processing software
  - UI design.

### Required knowledge

- basic knowledge of database design and implementation
- business and technical modelling using UML tools at intermediate level
- current software development methodologies
- detailed knowledge of system development life cycle (SDLC)
- object-oriented programming
- open-source development tools
- organisational procedures for documenting technical specifications
- software-testing techniques.

# 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:<br>• verify and validate software application design, including:<br>  • creating proof of concept (PoC) prototype<br>  • evaluating the technical specification and comparing against the PoC<br>  • interpreting and evaluating software design documentation requirements and confirming details with the client<br>  • analysing and validating user interface and database requirements<br>  • effectively using static analysis tools<br>  • walkthrough documents, test plans and test cases. |
| **Context of and specific resources for assessment** | Assessment must ensure access to:<br>• tools to create prototype systems<br>• static analysis tools<br>• test plan and test cases<br>• appropriate learning and assessment support when required<br>• 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:<br>• review of candidate's validation documents<br>• evaluation of validation documents. |
| **Guidance information for assessment** | Holistic assessment with other units relevant to the industry sector, workplace and job role is recommended, where appropriate.<br>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.<br>Indigenous people and other people from a non-English speaking background may need additional support.<br>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.*

| | |
|---|---|
| ***Software requirement specifications document*** may include: | • reports<br>• risks and resourcing<br>• software functions<br>• software system inputs and outputs<br>• timeframe. |
| ***Software design document*** may include: | • module test plan generation<br>• test-design generation<br>• traceability analysis<br>• update software risk analysis. |
| ***Database structure and elements*** may include: | • database consistency<br>• database integrity<br>• database structure. |
| ***User interface*** may include: | • appearance<br>• consistency<br>• design<br>• functions<br>• navigation. |
| ***Static analysis tools*** may include: | • Blast (c language)<br>• Checkstyle (for Java)<br>• JSLint (JavaScript checker)<br>• StyleCop (for .net languages). |

# Unit Sector(s)

Programming and software development