Higher National Unit specification

General information

Unit title: Software Development: Project (SCQF level 7)

Unit code: HA4J 34

Superclass: CB

Publication date: January 2016

Source: Scottish Qualifications Authority

Version: 02

Unit purpose

The purpose of this Unit is to apply knowledge and skills of software analysis, design, implementation and testing to produce a software product composed of multiple sub-programs. The Unit is aimed at learners wishing to begin preparing for a role as an entry-level software developer.

Learners will analyse a problem, design the operation and interaction of program components, produce working code to meet requirements, and test a completed solution to prove functional operation.

Learners should have previous knowledge and experience of the software development process including analysis, design, implementation and testing in at least one contemporary programming language and software development environment.

The Unit is suitable for learners who have completed the Units *Software Development: Analysis and Design* (SCQF level 7) and *Software Development: Implementation and Testing* (SCQF level 7). It involves the practical application of knowledge and skills, and will give learners exposure to the whole software development life-cycle.

On completion of the Unit, learners will be competent in the production of components of software products and may progress to the Unit *Software Development: Project* at SCQF level 8.

Higher National Unit Specification: General information (cont)

Unit title: Software Development: Project (SCQF level 7)

Outcomes

On successful completion of the Unit the learner will be able to:

1 Plan the development of a non-complex software product.
2 Design the structure of a non-complex software product.
3 Develop a non-complex software product.
4 Test the operation of a non-complex software product.

Credit points and level

2 Higher National Unit credits at SCQF level 7: 16 SCQF credit points at SCQF level 7)

Recommended entry to the Unit

The Unit is suitable for learners who have completed the Units *Software Development: Analysis and Design* (SCQF level 7) and *Software Development: Implementation and Testing* (level 7), or have prior equivalent knowledge. It involves the practical application of knowledge and skills, and will give learners exposure to the entire software development life-cycle.

Core Skills

Achievement of this Unit gives automatic certification of the following full Core skill and Core Skills component:

Complete Core Skill	Problem Solving at SCQF level 6
Core Skill component	Providing/Creating Information at SCQF level 5

Context for delivery

If this Unit is delivered as part of a Group Award, it is recommended that it should be taught and assessed within the subject area of the Group Award to which it contributes.

The Assessment Support Pack (ASP) for this Unit provides assessment and marking guidelines that exemplify the national standard for achievement. It is a valid, reliable and practicable assessment. Centres wishing to develop their own assessments should refer to the ASP to ensure a comparable standard. A list of existing ASPs is available to download from SQA's website **(http://www.sqa.org.uk/sqa/46233.2769.html)**.

Equality and inclusion

This Unit specification has been designed to ensure that there are no unnecessary barriers to learning or assessment. The individual needs of learners should be taken into account when planning learning experiences, selecting assessment methods or considering alternative evidence.

Further advice can be found on our website **www.sqa.org.uk/assessmentarrangements**.

Higher National Unit specification: Statement of standards

Unit title: Software Development: Project (SCQF level 7)

Acceptable performance in this Unit will be the satisfactory achievement of the standards set out in this part of the Unit specification. All sections of the statement of standards are mandatory and cannot be altered without reference to SQA.

Where evidence for Outcomes is assessed on a sample basis, the whole of the content listed in the Knowledge and/or Skills section must be taught and available for assessment. Learners should not know in advance the items on which they will be assessed and different items should be sampled on each assessment occasion.

Outcome 1

Plan the development of a non-complex software product.

Knowledge and/or Skills

- Describe the stages in the software development life cycle
- Describe the need for security
- Explain the roles in the software development life cycle
- Describe planning methods
- Use planning tools
- Create development plan
- Create test plan

Outcome 2

Design the structure of a non-complex software product.

Knowledge and/or Skills

- Carry out requirements analysis
- Produce wireframe designs
- Produce system interaction diagrams
- Select data structures
- Select algorithms
- Write pseudocode

Outcome 3

Develop a non-complex software product.

Knowledge and/or Skills

- Use software development tools including editing, compilation and debugging tools
- Implement design
- Create working components
- Assemble components into working system
- Write internal and external documentation

Unit title: Software Development: Project (SCQF level 7)

Outcome 4

Test the operation of a non-complex software product.

Knowledge and/or Skills

- Use testing tools
- Test components
- Test system
- Identify errors
- Diagnose causes of errors
- Correct identified errors

Evidence Requirements for this Unit

Learners will need to provide evidence to demonstrate their Knowledge and/or Skills across all Outcomes. All evidence will be in the form of practical competence.

Evidence of practical competence will take the form of **at least one complete software product**. There is no requirement for separate evidence for cognitive and practical competence.

The product may be **non-complex** but sufficient to demonstrate the Knowledge and Skills in each Outcome. Candidates must develop at least one **complete software product**. Where the product is non-complex (more complex solutions are acceptable), it must be **error free** and meet the functional specification in entirety. It is permissible for more complex solutions to contain non-critical errors so long as the product demonstrates the required competencies.

The following evidence is the **minimum** required:

1 Software development plan
2 Test plan
3 System design
4 Component designs
5 Source code
6 Object code
7 Test logs

This evidence may be supplied on paper or digitally or a combination of these.

Higher National Unit specification: Statement of standards (cont)

Unit title: Software Development: Project (SCQF level 7)

The standard of the evidence should be consistent with the SCQF level of this Unit. At this level, the evidence should collectively or individually demonstrate:

- clear appreciation of the overall software development process.
- computational thinking.
- understanding of the principles of programming.
- correct application of theories, concepts and principles.
- application of basic and routine professional skills, techniques, practices and/or materials associated with the software development industry.
- convey program design in well-structured and coherent form.
- implement plans in accordance with timescales.
- demonstrate professional conduct.

There are **no time limitations** on the production of evidence. The evidence may be produced at any time during the life of the Unit. Learners **may use reference materials** when undertaking assessment.

Evidence may be wholly or partly produced under controlled conditions. When evidence is produced in uncontrolled or loosely controlled conditions it must be authenticated. The Guide to Assessment provides further advice on methods of authentication.

The Guidelines on Approaches to Assessment (see the Support Notes section of this specification) provides specific examples of instruments of assessment.

Higher National Unit Support Notes

Unit title: Software Development: Project (SCQF level 7)

Unit Support Notes are offered as guidance and are not mandatory.

While the exact time allocated to this Unit is at the discretion of the centre, the notional design length is 80 hours.

Guidance on the content and context for this Unit

This Unit may be delivered as a stand-alone Unit or following completion of the two other Units making up the Professional Development Award (PDA) in Software Development at SCQF level 7.

Learners should be encouraged to actively explore various resources including the Internet in order to familiarise themselves with a wide range of methodologies and tools as appropriate to each Learning Outcome. In addition, they must be aware of the internet safety, security, confidentiality and health and safety procedures of the organisation. It is also important to maintain the security and confidentiality of data and information and therefore leaners should be encouraged to back up and check for viruses on a regular basis.

The overall aim of this Unit is for learners to apply skills and knowledge of software analysis, design, implementation and testing to produce a simple software product composed of multiple sub-programs. or a component of a moderately-complex software product. The focus of the Unit is on practical competencies and learners will analyse a problem, design the operation and interaction of program components, produce working code to meet requirements, and test a completed solution to prove functional operation.

It is anticipated that substantial amount of time will be spent on practical tasks, and although learners may use tools that they have sourced themselves; appropriate tools to complete all of the Outcomes should be made available through their centre.

Learners will be working on real-life software production, and will be expected to add new features and/or enhance functionality in existing code-bases, or to produce component parts of new products.

Outcome 1

This Outcome allows learners to apply the principles of an appropriate software development process, such as Agile or another contemporary methodology, to produce a project plan for the structure and operation of a simple software product or a component of a moderately-complex software product.

The overall intention is that their planning could be taken into design in Outcome 2. However, at this level, learners may be provided with starting points for each Outcome, so that they can achieve each Outcome independently of the others.

Outcome 2

This Outcome allows learners to demonstrate the design of the operation of their product and structure of their code through visualisation tools such as wireframes, diagrams, illustrations and pseudocode.

The overall intention is that their planning could be taken into implementation in Outcome 3. However, at this level, learners may be provided with starting points for each Outcome, so that they can achieve each Outcome independently of the others.

Outcome 3

This Outcome allows learners to develop a simple software product or a component of a moderately-complex software product.

Learners will turn a design into a working software product. This should interact with users to perform its operation and reply to the users with its results. Pseudocode from the design phase should be turned working code using constructs such as sequence and selection and sub-programs and functions to meet design requirements.

Although there is no prescription on which languages should be used, learners should endeavour to use idioms appropriate to the chosen language.

The overall intention is that the product implementation in Outcome 3 can be used as a starting point for Outcome 4. However, at this level, learners may be provided with starting points for each Outcome, so that they can achieve each Outcome independently of the others.

Outcome 4

This Outcome allows learners to apply their knowledge and skills to test a simple software product or a component of a moderately-complex software product.

Learners will demonstrate their ability to test the operation of code to an acceptable level/criteria. This may be manual testing to a use-case plan, or automated testing using ‘Unit’ or ‘integration’ or other approaches to testing code. Learners need to be able to decipher error messages and identify what file/function/line is the source of the error, then suggest alternative solutions and correct the code accordingly.

Learners could write their own test plan from Outcome 1 (if they also implemented it in Outcome 3), or they could be provided with a product and use-cases to test to. If learners cannot identify errors in their own code, eg in the product created to meet previous Outcomes, then they will need to be provided with broken code of equal complexity to test against.

Unit title: Software Development: Project (SCQF level 7)

Guidance on approaches to delivery of this Unit

The Outcomes in this Unit should be delivered once learners have completed learning equivalent to the *Analysis and Design* and *Implementation and Testing* Units at level 7.
It would be expected that the delivery of this Unit's Outcomes progresses sequentially and that assessment would be undertaken at the end of each Outcome, eg learners should be confident in their analysis and design skills before completing implementation and then testing. However, if learners are being provided with starting points for each Outcome, then delivery need not be sequential.

It is recommended that learners be guided through the full life-cycle of a number of software development projects as part of learning and teaching prior to assessment. This will allow learners to gain confidence in the application of their skills to increasingly complex products with more independence.

Throughout the Unit, the focus of projects could be on software products that have a relevance to the learner and to their further learning. Products could be anything from a command-line calculator through to a simple web/mobile-app game.

Where learners are creating a component to work with ‘a moderately-complex software product’, this is defined by a product that has multiple functionality and interfaces to APIs and data persistence.

Guidance on approaches to assessment of this Unit

Evidence Requirements for this Unit will be for practical competence. Evidence can be generated using different types of assessment. The following are suggestions only. There may be other methods that would be more suitable to learners.

The assessment for all Outcomes could comprise a series of practical tasks completed over a period of time. Learners may have access to online resources, and the assessment need not be done under supervision, but all work should be authenticated as the learner's own. This could be done by oral questioning of the learner and/or observation of some aspects of the learner’s work.

For each Outcome, learners should identify or be given a product definition of appropriate complexity. This can be for the design, implementation and testing of a simple software product, or for a component of a moderately-complex software product. Learners may use the same product for all Outcomes, or different ones for individual Outcomes.

Centres are reminded that prior verification of centre-devised assessments would help to ensure that the national standard is being met. Where learners experience a range of assessment methods, this helps them to develop different skills that should be transferable to work or further and higher education.

Unit title: Software Development: Project (SCQF level 7)

Outcome 1: Plan the development of a simple software product.

Learners may be presented with or devise their own, simple software product definition, or component of a moderately-complex product scope.

- **Analysis of requirements**

  Learners should create appropriate artefacts demonstrating their analysis of the product scope, specification and requirements.

  Assessment tools for this might take the form of white-boarding in a discussion, or sampling from a set of planning materials.

- **Software development process**

  Learners should demonstrate use of a contemporary development methodology to produce a project plan for the structure and operation of a simple software product or a component of a moderately-complex software product.

- **Test plan**

  Learners should prepare an outline test plan to meet agreed criteria using appropriate software testing methodologies. This could include plans for manual and/or automated testing.

Assessment for the last two Knowledge and Skills might take the form of observed view and discussion or screenshots of project management software tools outlining main project requirements/deliverables/resources/timeframes.

Outcome 2: Design the structure of a simple software product.

Learners may design the product scoped in Outcome 1, or design a product of equal complexity.

- **Produce wireframe designs**

  Wireframe designs should indicate both the expected user interface and user interaction, output and feedback. There should be appropriate wireframes for the major cases of functionality and for edge-case scenarios.

  Assessment could take the form of an 'elevator-pitch' style presentation, a written report format or be presented in a diagrammatic style.

- **Produce system interaction diagrams**

  Learners should be able to demonstrate the complete operation or flow through the input/outputs of their product. Assessors should select two interactions between code and user input or output and ensure learners have produced detailed structure charts or system interaction diagrams for them.

  Assessment tools for this might take the form of white-boarding in a discussion, or sampling from a complete set of system interaction diagrams.

Unit title: Software Development: Project (SCQF level 7)

- **Write pseudocode and select algorithms**

  Learners should describe a minimum of two sub-routines of their design by writing pseudocode outlining how the algorithms they have selected would function in the operational program. Learners' pseudocode should also demonstrate how it would be incorporated into a larger product if necessary.

Assessment might take the form of observed peer discussion of the learner's pseudocode.

Outcome 3: Develop a simple software product.

Learners may develop the product scoped in Outcome 2, or develop a product of equal complexity.

- **Build working components of software**

  This is the core activity of this Outcome, so the primary measure of success to meet this Outcome is to produce working code. This means that the product runs from start to finish without exceptional errors. A minimum of 50% of this Outcome should be based on achieving working code/ working components of software.

- Code should be well-structured and readable with its intention clearly documented with comments or idioms.

- Learner's product should interact with the user and process their input according to design requirements. Users should receive feedback from the product and be able to continue interaction.

Assessment might take the form of observation by Assessor of working product, with access to source code to check the structure for coverage of relevant knowledge and skills.

For Outcome 4: Test the operation of a simple software product.

Learners may test the operation of the product they have developed for Outcome 3, or a product of equal complexity.

- Check operation of code.

- Diagnose causes of errors.

- Correct identified errors.

- Learners should provide evidence that their code works as expected by the design.

- They should be able to identify the source of any errors by deciphering error messages received in their product code, or in code provided to them. They should be able to rectify the error to return to working code.

Unit title: Software Development: Project (SCQF level 7)

- This type of assessment may take the form of a professional discussion by Assessor with learner demonstrating the steps undertaken to diagnose and correct one or more errors in their own product.

- Alternatively, learners could be asked to write a ‘bug report’ on program code provided to them and asked to document the steps required to duplicate a bug in their product. This could be followed by learner making appropriate corrections. This type of assessment might take the form of a blog post, report or a personal demonstration to the Assessor of what steps the learner took to achieve results.

Opportunities for e-assessment

E-assessment may be appropriate for some assessments in this Unit. By e-assessment we mean assessment which is supported by Information and Communication Technology (ICT), such as e-testing or the use of e-portfolios or social software. Centres which wish to use e-assessment must ensure that the national standard is applied to all learner evidence and that conditions of assessment as specified in the Evidence Requirements are met, regardless of the mode of gathering evidence. The most up-to-date guidance on the use of e-assessment to support SQA’s qualifications is available at **www.sqa.org.uk/e-assessment**.

Opportunities for developing Core and other essential skills

The Unit will provide opportunities for the development of Computational Thinking skills.

This Unit has the Providing/Creating Information component of Information and Communication Technology embedded in it. This means that when learners achieve the Unit, their Core Skills profile will also be updated to show they have achieved Providing/Creating Information at SCQF level 5.

This Unit has the Core Skill of Problem Solving embedded in it, so when learners achieve this Unit their Core Skills profile will be updated to show that they have achieved Problem Solving at SCQF level 6.

History of changes to Unit

Version	Description of change	Date
02	Core Skills Component Providing/Creating/Information at SCQF level 5 embedded and Core Skill Problem Solving at SCQF level 6 embedded.	20/05/16

Additional copies of this Unit specification can be purchased from the Scottish Qualifications Authority. Please contact the Business Development and Customer Support team, telephone 0303 333 0330.

General information for learners

Unit title: Software Development: Project (SCQF level 7)

This section will help you decide whether this is the Unit for you by explaining what the Unit is about, what you should know or be able to do before you start, what you will need to do during the Unit and opportunities for further learning and employment.

The purpose of this Unit is to apply knowledge and skills of software analysis, design, implementation and testing to produce the components of a software product composed of multiple sub-programs. The Unit is aimed at learners wishing to begin preparing for a role as an entry-level software developer.

You will analyse a problem, design the operation and interaction of program components, produce working code to meet requirements, and test a completed solution to prove functional operation.

The Unit is suitable for learners who have completed the Units *Software Development: Analysis and Design* (SCQF level 7) and *Software Development: Implementation and Testing* (SCQF level 7). It involves the practical application of knowledge and skills, and will give learners exposure to the whole software development life-cycle.

You will need to provide evidence to demonstrate their Knowledge and/or Skills across all Outcomes. Evidence of practical competence will take the form of **at least one complete software product**. There is no requirement for separate evidence for cognitive and practical competence.

On completion of the Unit, you will be competent in the production of components of software products and may progress to the Unit *Software Development: Project* at SCQF level 8.

This Unit has the Providing/Creating Information component of Information and Communication Technology embedded in it. This means that when you achieve the Unit, your Core Skills profile will also be updated to show you have achieved Providing/Creating Information at SCQF level 5.

This Unit has the Core Skill of Problem Solving embedded in it, so when you achieve this Unit your Core Skills profile will be updated to show that you have achieved Problem Solving at SCQF level 6.