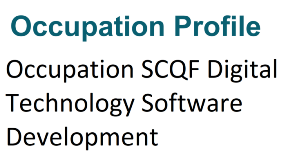
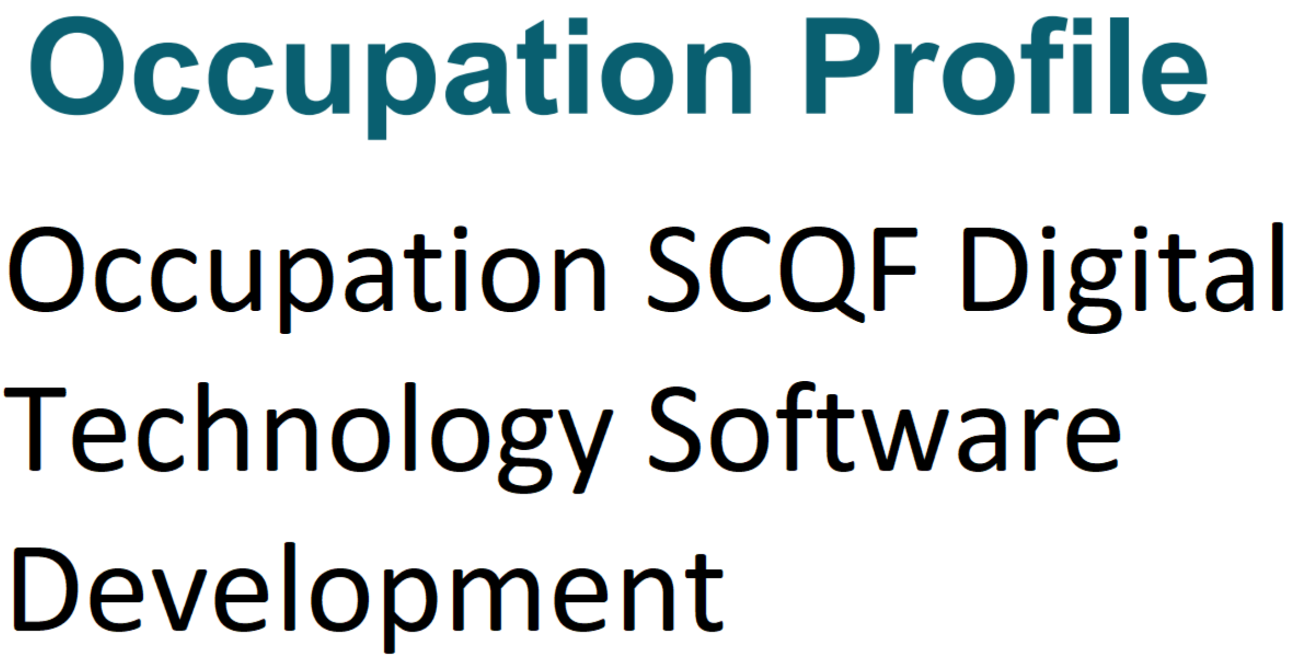


Occupation Profile

Occupation SCQF Digital Technology Software Development

Occupation SCQF Digital Technology Software Development Installation Guide

Occupation SCQF Digital Technology Software Development



Occupation Profile

Occupation SCQF Digital Technology Software Development

Contents

Purpose

This occupation profile consists of 8 work situations routinely carried out in Software Development roles at this level. Collectively these describe all the performance requirements and knowledge and understanding requirements apprentices need to demonstrate competence in the occupation. Each work situation has a unique reference number and is set out as follows:

- Work situation title, goal, brief outline, performance requirements and knowledge and understanding requirements

Work Situation

Applying methods and principles in project management

URN: SDS 007

Goal of work situation:
This work situation involves using project management tools to plan, organise and monitor the progress of activities to achieve production quality performance indicators.
Brief outline:
This is about applying methods and principles of project management in line with organisational requirements. This includes ensuring activities are delivered in accordance with the business case and safe systems of work, and involves liaising with and reporting progress to stakeholders, ensuring activities contribute to key milestones and deliverables.

Performance requirements

1. Providing support to prepare business cases for approval of activities
2. Identifying roles, responsibilities and skill sets needed for project activities and resources
3. Planning and scheduling projects in line with agreed objectives, timescales, and organisational requirements
4. Managing activities in line with plans and to achieve milestones
5. Managing change in line with organisational procedures
6. Escalating to relevant personnel where there are deviations from plans
7. Identifying, agreeing, and implementing contingencies to mitigate problems
8. Communicating plan progress in formats to meet the needs of all relevant stakeholders
9. Reporting on progress in line with organisational reporting procedures
10. Collating and evaluating lessons learned to contribute to the continuous improvement of activities

Knowledge and understanding requirements

1. Relevant legislation and codes of practice, safe systems of work, risk and impact assessments for activities
2. The principles and approaches to developing good business cases
3. Different methodologies to plan and deliver activities and how to apply these
4. The tools and processes for identifying and analysing risks and opportunities and how to use them
5. Techniques and tools for monitoring and reviewing risks including when and how to escalate to management
6. Quantitative and qualitative measures of risk analysis and how to apply these
7. The importance of monitoring and controlling project performance including accountability
8. Industry specific tools and software for monitoring performance
9. The importance of establishing an agreed change control process, and the impact and consequences that changes can have on schedule, resources, and budget
10. The type of changes that may affect key performance criteria including time, cost, quality, and business case
11. The importance of contingency plans
12. The importance of evaluating and monitoring the benefits and challenges of activities and how to do this
13. Different ways, formats and frequency of reporting and presenting information on progress to internal and external stakeholders
14. The importance of liaising with internal and external stakeholders and how to do this

Supporting digital business transformation

URN: SDS 187

Goal of work situation:
To identify, evaluate and prioritise the opportunities to apply digital technology to improve operations by transforming business processes.
Brief outline:
This involves evaluating the organisational processes to propose digital technology solutions within businesses to reduce costs, enhance performance and deliver improved services as a result of digital transformation.

Performance requirements

1. Identifying and documenting organisational processes which require digital technology improvement
2. Establishing information requirements of the organisational processes requiring digital technology improvement
3. Evaluating the potential for digital technology solutions to transform the organisational processes that deliver organisational competitiveness
4. Analysing organisational processes to propose potential digital technology solutions
5. Conducting relevant research to inform decision making for digital transformation
6. Conducting health and safety risk assessments of digital transformation scenarios
7. Developing and delivering well-structured digital technology proposals in the form of business reports and presentations which resonate with stakeholders

Knowledge and understanding requirements

1. The meaning and significance of the ‘digital economy’ and ‘digital transformation’

2. How to model business processes
3. How organisations manage and implement technology driven change
4. How to formulate proposals for new digital technology solutions, including estimation of both costs and benefits
5. How digital technologies can be integrated within business processes
6. How digital transformation of business processes is implemented to provide improved productivity and service benefits
7. The legislation, regulations and organisational policies that relate to digital technology and safe use of IT in the workplace
8. The range of professional and unprofessional behaviour in digital technology contexts
9. The principles of business change and how organisations develop in the context of technological change
10. The organisational business objectives and how business strategy is used to achieve these
11. The range of metrics which might be used to evaluate the success of business operations
12. Current issues and ethical aspects in digital transformation implementation
13. The safe use of digital technology equipment in business operations

Developing meta-skills and personal professionalism

URN: SDS 012

Goal of work situation:
To develop meta-skills and personal professionalism through reflective practice, goal setting and active learning to improve own performance in line with organisational requirements.
Brief outline:
This is about taking responsibility for the development of own meta-skills and personal professionalism. This involves reflecting on and learning from practice; seeking and acting on feedback; agreeing and working towards own goals for continuous professional development (CPD); and managing own wellbeing.

Performance requirements

1. Self-evaluating meta-skills regularly to identify own strengths and improvement needs for development
2. Identifying own strengths and improvement needs for professional development
3. Setting and agreeing SMART objectives for personal development and to achieve business objectives
4. Planning development activities to improve own performance and to achieve business objectives
5. Completing formal and informal activities to support and progress own development
6. Seeking and acting on feedback to improve own performance
7. Critically reflecting on own performance and involvement in activities to support own development and achievement
8. Critically evaluating the development and application of meta-skills in own work to identify future development needs
9. Completing and maintaining records and documents in line with organisational policy and procedures.

Knowledge and understanding requirements

1. The purpose and importance of meta-skills including their definitions and how they relate to own work
2. The importance and impact of personal professionalism within the organisation and own role
3. How to use critical reflection and reflective practice to identify gaps in role specific knowledge, skills and meta-

skills and the purpose and importance of this

4. How to participate effectively in performance reviews
5. How to set and agree SMART goals – Specific, Measurable, Achievable, Realistic, Time-bound
6. How to prepare development plans, including their content and duration
7. The importance of career and personal goals, including collective organisational learning, when planning own development
8. Sources of up-to-date and appropriate information to support own CPD activities
9. The impact and benefits of CPD including the organisation's key performance indicators (KPIs) and how they are measured and recorded
10. The importance of managing well-being for success in own role and where to get support
11. Appropriate ways to seek and act on feedback to develop own skills and knowledge including the process of 360-degree feedback
12. Different learning models and styles and how to use these for own development

Designing software

URN: SDS 018

Goal of work situation:
To specify and design software to meet defined requirements by following agreed design standards and principles.
Brief outline:
This is about individuals designing software components and modules and specifying user interfaces using appropriate modelling techniques following agreed software design standards, patterns, and methodology. This will involve assisting as part of a team in the design of components of larger software systems and creating and communicating software design options considering both functional and non-functional requirements, security requirements and existing systems.

Performance requirements

1. Specifying the design of software to meet defined requirements, following agreed design standards and principles
2. Using whiteboarding, wireframes and flow diagrams to show workflows and deliver prototypes of software designs to stakeholders to support evaluation of alternative solutions and trade-offs
3. Identifying software features and implementing design patterns to translate the requirements into a design which provides a basis for software construction and testing
4. Communicating software designs to software developers to guide them in how the software will deliver the features and functionality specified in the software requirements specification
5. Separating back-end and front-end software design processes:
    - back-end – create data models/Entity Relationship Diagrams (ERD) and data flow diagrams
    - front-end – create user interface designs
6. Contributing to reviews of design work with others as appropriate

Knowledge and understanding requirements

1. How to interpret and follow software requirements and functional/technical specifications
2. How to create and document detailed designs for software applications or components

3. The modelling techniques, standards, patterns, and tools used in software design and how to apply them
4. Different software design approaches and how to select these for a given set of software requirements
5. How to produce ERDs and data flow diagrams for back-end design
6. How to specify and design user and system interfaces
7. The need to create multiple design views/prototypes to address the concerns of the different stakeholders
8. How to model, simulate or prototype the behaviour of proposed software designs to enable approval by stakeholders
9. How to design functional and non-functional requirements
10. The need to consider the target environment, performance, and security requirements
11. How new requirements fit alongside existing software and legacy systems
12. How to review, verify and improve own designs and the designs of others against specifications
13. The importance and characteristics of the full stack approach

Developing software

URN: SDS 019

Goal of work situation:
To plan, create and document new and amended software components to deliver agreed requirements to stakeholders.
Brief outline:
This is about individuals coding, verifying, testing documenting, amending, and refactoring software using agreed standards and tools, to achieve a well engineered result. Individuals will be involved in collaborating with others to review work as appropriate.

Performance requirements

1. Selecting and applying software development paradigms relevant to the end user context
2. Developing effective user interfaces for the platform being developed
3. Writing good quality code (logic) with sound syntax
4. Linking code to databases and data sets to provide access to data stores
5. Applying agreed standards and tools to achieve well-engineered software including code commenting, naming and layout
6. Refactoring software to improve its structure, legibility, efficiency and reusability
7. Reviewing own software development activities to find and eliminate problems and identify productivity improvements
8. Collaborating with others in work reviews to support improvements to the software development processes adopted
9. Using third party integration tools (screen scraping data and integrating to other systems) to capture, reformat and display data more conveniently
10. Staging and deploying validated code into live enterprise environments
11. Debugging code and applying structured techniques to problem solving

Knowledge and understanding requirements

1. How to review software requirements and design specifications

2. How to read and write technical software documentation
3. The software development lifecycle
4. How to operate at all stages of the software development lifecycle
5. Good practice approaches for the relevant software development paradigm, including object oriented, event driven or procedural
6. How to develop software for web, mobile and fixed platforms
7. The range of development tools available and how to use them
8. How to produce software code directly using a command line editor
9. Industry standard software and web design and accessibility frameworks and guidelines and how to apply them, including those from W3C (world wide web consortium)
10. How to implement unit testing at each stage of software development
11. How to develop software using industry standard software languages, and development environments and tools
12. How to resolve software development problems through online research to find solutions
13. How to design, build and interface with databases to provide data creation, updating and deletion functions
14. How teams work effectively to produce software
15. The importance of considering different approaches and tools including cost and efficiency

Implementing software methodology

URN: SDS 021

Goal of work situation:
To adopt and adapt software development lifecycle models based on the context of the work and selecting appropriately from plan-driven approaches or adaptive iterative/agile approaches.
Brief outline:
This is about individuals adopting and adapting appropriate software design and development methods, and tools and techniques. This will involve selecting appropriately from plan-driven approaches or adaptive iterative/agile approaches, and ensuring they are applied effectively by the organisation.

Performance requirements

1. Implementing appropriate software development methodologies including predictive (plan-driven) approaches or adaptive (iterative/agile) approaches
2. Identifying user requirements and translating these into user stories
3. Working as part of a software development team to develop and deliver software solutions
4. Meeting with stakeholders/project owners to agree on deliverables
5. Employing version control of source code and related artefacts to manage the development and configuration of source code, tested software, and documents
6. Working within a Continuous Integration (CI) capability
7. Agreeing development tasks, responsibilities, and priorities with the development team
8. Operating within a sprint or development phase to deliver within timescales
9. Agreeing software completion end and sign-off arrangements

Knowledge and understanding requirements

1. Modern techniques used in the design and development of software systems
2. The different approaches to organising software development using plan-driven or adaptive iterative/agile methodologies
3. How teams work effectively to develop software solutions embracing agile and other development approaches.
4. The importance of version control and how to apply industry standard solutions
5. How to implement software development using Continuous Integration
6. The importance of continual improvement within a dynamic information driven environment
7. The benefits of frequent merging of code, ensuring all tests pass
8. The implementation of automation within software development
9. The variety of documents that are required to support software development
10. How to provide task estimates to inform software project planning
11. How to implement collaborative software development, team working and organising software development

Providing user and software documentation

URN: SDS 030

Goal of work situation:
To plan and create design, development, and user documentation for new and amended software components.
Brief outline:
This is about individuals drafting and maintaining documentation for software applications design and development. This involves the drafting of documentation and user guides and supporting end users through training. Individuals will use document management software and tools to manage documents.

Performance requirements

1. Planning, designing, creating, amending, verifying, testing, and deploying developer and end-user documentation for new and amended software
2. Producing user story documentation
3. Developing documentation of new and amended software designs to deliver agreed requirements
4. Creating and updating the documentation of software development and testing methods and tools
5. Developing coding documentation to describe in the software code modules, functions, including code comments, variable naming, and layout
6. Producing embedded software documentation to provide user guidance including tool tips and help guides
7. Producing user documentation in the form of manuals and guides
8. Developing training materials and guides as required
9. Following organisational styles and templates for documentation
10. Testing documentation
11. Producing video and blog support media

Knowledge and understanding requirements

1. The software development life cycle (SDLC)
2. The importance of documentation in software development
3. The difference between product (user) and process (developer) documentation and how to implement them
4. The different types of documents that are created throughout the software development lifecycle, including

software process documentation, software product documentation and user documentation

5. How documentation varies depending on the software development methodology adopted
6. How to develop user stories that document the user view of a software product and describe user actions and results
7. How to create user interface documentation, guides, and tool tips
8. How to produce data model diagrams that support back end design documentation
9. How to interpret and review developed code through the embedded code documentation
10. How to document software products
11. The coding industry standards (naming, indentation, and code comments) and how to apply them
12. What documents to update and when
13. The types of documentation required to support users, including tool tips, help guides, manuals and training materials

Providing software testing and assurance

URN: SDS 031

Goal of work situation:
To plan, design, execute and report on software tests, using appropriate testing tools and techniques and conforming to agreed organisational standards.
Brief outline:
This is about reviewing requirements and specifications and defining test conditions. This includes designing and running test cases and test scripts and preparing test data. This also includes reporting on test activities and results and identifying and reporting on bug and defect tracking.

Performance requirements

1. Analysing requirements to determine appropriate testing strategies
2. Designing and implementing test plans with instrumented code
3. Computing test coverage and yield according to specified criteria to ensure test cases are covering all functionality of the software
4. Identifying how much code is exercised when test cases are run
5. Writing tests in various contexts (including unit testing, system testing, integration testing and load testing)
6. Applying industry standard testing techniques
7. Producing suitable test data to cover the range of possible and impossible inputs, each designed to prove software works or to highlight any flaws
8. Performing usability testing to ensure that software will operate correctly under live user situations
9. Identifying, tracking, and fixing bugs identified in software, recreating the bug, and escalating if necessary
10. Supporting software quality assurance and acceptance testing of the final solution
11. Contributing to test reports and communicate test results

Knowledge and understanding requirements

1. The purpose of testing and ensuring new and amended systems, configurations, packages, or services, together with any interfaces, perform as specified
2. The testing life cycle and how it is applied to software development projects

3. The role of testing within software development, as a tool for design improvement as well as a validation process
4. What is meant by Test Driven Development (TDD) and the test pyramid
5. How development is underpinned by testing
6. How to select and apply a testing strategy and testing techniques that are appropriate to a particular software system or component
7. How to create a test plan
8. How to design test cases and create test scripts and test data
9. The different test techniques for performing unit testing, system testing integration testing and load testing
10. The importance of identifying opportunities for automated testing and implementing these
11. The role of usability testing and how to implement this
12. The importance of representative and accurate test data and data volumes as well as deliberately wrong data to support software testing
13. The use of bug tracking tools to record and manage resolution
14. The difference between bugs and enhancements, and the need to agree to replan software development to accommodate these
15. The relevant legislation and internal/external standards related to software testing and how to apply these
16. How to assess test results against expected results and acceptance criteria and verify these through traceability to software requirements

The Relationship Between Meta-skills And Work Situations

Meta skills alignment												
Work situation	Adapting	Collaborating	Communicating	Creativity	Critical thinking	Curiosity	Feeling	Focusing	Initiative	Integrity	Leading	Sense making
Applying methods and principles of project management	✓	✓	✓		✓			✓		✓		
Supporting digital business transformation	✓		✓		✓			✓				
Developing meta-skills and personal professionalism	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Designing software	✓	✓	✓			✓						✓
Developing software	✓			✓				✓	✓			
Implementing software methodology	✓	✓	✓		✓						✓	
Providing software testing and assurance	✓	✓	✓	✓								✓
Providing user and software documentation	✓		✓	✓				✓				

The table above indicates where there are opportunities to develop and evidence meta-skills in each work situation within the occupation profile. Please note, this information is for guidance, and indicates where meta-skills are be opportunities for individuals to develop and evidence other meta-skills when carrying explicit rather than an exhaustive list. There may out their role.

Relationship Between National Occupational Standards

The relationship between National Occupational Standards and work situations

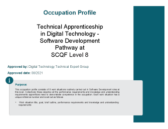
The table below indicates where there are links between National Occupational Standards and each work situation within the occupation profile.

Applying methods and principles of project management	• Project management suite • Engineering and Manufacturing suite 4 • Engineering Leadership and Manufacture suite 4 • Industrial Design Suite	• Maintain IT project-based documentation TECIS30131 • Initiate an IT project TECIS 30141 • Monitor and control the delivery of an IT project TECIS 30143	
Supporting digital business transformation	• Carry out business process design and improvement assignments ESKITP2024.03	• Assist in the design, implementation and maintenance of change management plans and assignments ESKITP2034.03	• Use safe and secure practices when working with digital systems ESKITU040
Developing meta-skills and personal professionalism	• Business and Administration suite • Management and Leadership suite		
Designing software	• Develop data models of proposed solutions ESKITP2085.04 • Data Design Level 3 role ESKITP4053 • Data Design Level 4 role ESKITP4054	• Human Computer Interaction/ Interface (HCI) Design Level 3 role ESKITP4063 • Human Computer Interaction/ Interface (HCI) Design Level 4 role ESKITP4064	• Systems design Level 3 role ESKITP4073 • Systems design Level 4 role ESKITP4074

Developing software	• Software Development Level 3 role ESKITP5023 • Software Development Level 4 role ESKITP5024	• IT Technology Systems Installation Level 3 role ESKITP5053 • IT Technology Systems Installation Level 4 role ESKITP5054	• Software Development Process Improvement Level 3 role ESKITP5063 • Software Development Process Improvement Level 4 role ESKITP5064
Implementing software methodology	• Software Development Level 3 role ESKITP5023 • Software Development Level 4 role ESKITP5024		
Providing software testing and assurance	• Design tests for software products TECIS503301 • Develop detailed test plans TECIS503402	• Develop and conduct user acceptance tests TECIS503303 • Analyse and interpret the results of software testing activities TECIS503304	
Providing user and software documentation	• Software Development Level 3 role ESKITP5023 • Software Development Level 4 role ESKITP5024	• IT Technology Systems Installation Level 3 role ESKITP5053 • IT Technology Systems Installation Level 4 role ESKITP5054	

Occupation Profile

Documents / Resources

 Occupation Profile Technical Apprenticeship in Digital Technology - Software Development Pathway at SCQF Level 8 **Approved by:** Digital Technology Technical Expert Group **Approved date:** 08/2021 **Purpose:** This occupation profile consists of 8 work situations routinely carried out in Software Development roles at this level. Collectively these describe all the performance requirements and knowledge and understanding requirements apprentices need to demonstrate competence in the occupation. Each work situation has a unique reference number and is set out as follows: • Work situation title, goal, brief outline, performance requirements and knowledge and understanding requirements	Occupation SCQF Digital Technology Software Development [pdf] Installation Guide SCQF Digital Technology Software Development, SCQF, Digital Technology Software Development, Technology Software Development, Software Development, Development

References

- User Manual