This is a United Kingdom government regulated qualification which is administered and approved by one or more of the following: Ofqual, Qualifications Wales, CCEA Regulation or SQA.
CONTENTS

3. Introduction
4. Qualification Suitability and Overview
4. SFIA Levels
6. Learning Outcomes
7. Syllabus
11. Examination Format
11. Question Weighting
12. Recommended Reading
14. Using BCS Books
15. Document Change History
Introduction

The final stage within the BCS three-stage Higher Education Qualification programme, the Level 6 Professional Graduate Diploma (PGD) enables candidates who have already achieved the Level 5 Diploma in IT to gain depth of knowledge and expertise in their field.

Our modules have been created in-line with the SFIAPlus framework and latest developments in the industry, giving you a competitive edge in the IT job market and showing your dedication to the industry. You will have the opportunity to learn about topics such as advanced database management, network information systems, web engineering and programming paradigms, as well as to build upon knowledge and skills developed during the Level 5 Diploma.

To successfully achieve the qualification, candidates need to complete:

- One core module (Professional Project in IT)
- Four optional modules

Depending on entrance conditions, completing the Level 6 PGD in IT may support entry onto a Master’s degree course at selected global universities.

Software Engineering 2 optional module

The Software Engineering 2 module is an optional module that forms part of the Level 6 PGD in IT – the final stage within the BCS three-stage Higher Education Qualification programme.

Candidates will develop an understanding of software development and its evolution, as well as enhancing their existing knowledge of developing and maintaining high-quality software systems. Their understanding will be based on the discipline’s theoretical foundations and demonstrated by critically applying software engineering theory to real-world problems in practical applications. Throughout the module, candidates will gain a thorough understanding of the processes used in software systems engineering, software products, and theory, laws and models, as well as the relationships between these areas.
Qualification Suitability and Overview

Candidates must have achieved the Diploma in IT or have an appropriate exemption in order to be entered for the Professional Graduate Diploma (PGD). Candidates can study for this PGD by attending a training course provided by a BCS accredited Training Provider or through self-study, although it is strongly recommended that all candidates register with an approved centre. Studying with an approved centre will deliver significant benefits.

Candidates are required to become a member of BCS, The Chartered Institute for IT, to sit and be awarded the qualifications. Candidates may apply for a four-year student membership that will support them throughout their studies.

The Level 6 PGD is suitable for professionals wishing to gain an advanced formal IT qualification, and this module may be particularly relevant for candidates who are interested in career opportunities such as software, web or app development, application support analysis, or programming.

<table>
<thead>
<tr>
<th>Total Qualification Time (Certificate)</th>
<th>Guided Learning Hours (Module)</th>
<th>Assessment Time (Exam)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1414 hours</td>
<td>250 hours</td>
<td>Three hours</td>
</tr>
</tbody>
</table>

SFIA Levels

This award provides candidates with the level of knowledge highlighted within the table, enabling candidates to develop the skills to operate successfully at the levels of responsibility indicated.

<table>
<thead>
<tr>
<th>Level</th>
<th>Levels of Knowledge</th>
<th>Levels of Skill and Responsibility (SFIA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>K7</td>
<td></td>
<td>Set strategy, inspire and mobilise</td>
</tr>
<tr>
<td>K6</td>
<td>Evaluate</td>
<td>Initiate and influence</td>
</tr>
<tr>
<td>K5</td>
<td>Synthesise</td>
<td>Ensure and advise</td>
</tr>
<tr>
<td>K4</td>
<td>Analyse</td>
<td>Enable</td>
</tr>
<tr>
<td>K3</td>
<td>Apply</td>
<td>Apply</td>
</tr>
<tr>
<td>K2</td>
<td>Understand</td>
<td>Assist</td>
</tr>
<tr>
<td>K1</td>
<td>Remember</td>
<td>Follow</td>
</tr>
</tbody>
</table>
**SFIA Plus**

This syllabus has been linked to the SFIA knowledge skills and behaviours required at Level 6.

**ASUP4**

Maintains application support processes, and checks that all requests for support are dealt with according to agreed procedures. Uses application management software and tools to investigate issues, collect performance statistics and create reports.

**PROG4**

Designs, codes, verifies, tests, documents, amends and refactors complex programs/scripts and integration software services. Contributes to selection of the software development approach for projects, selecting appropriately from predictive (plan-driven) approaches or adaptive (iterative/agile) approaches. Applies agreed standards and tools, to achieve well-engineered outcomes. Participates in reviews of own work and leads reviews of colleagues’ work.

**DESN4**

Designs components using appropriate modelling techniques following agreed architectures, design standards, patterns and methodology. Identifies and evaluates alternative design options and trade-offs. Creates multiple design views to address the concerns of the different stakeholders of the architecture and to handle both functional and non-functional requirements. Models, simulates or prototypes the behaviour of proposed systems components to enable approval by stakeholders. Produces detailed design specification to form the basis for construction of systems. Reviews, verifies and improves own designs against specifications.

**DLMG5**

Defines systems development projects which support the organisation’s objectives and plans. Selects, adopts and adapts appropriate systems development methods, tools and techniques selecting appropriately from predictive (plan-driven) approaches or adaptive (iterative/agile) approaches. Ensures that senior management is both aware of and able to provide the required resources. Facilitates availability and optimum utilisation of resources. Monitors and reports on the progress of development projects, ensuring that projects are carried out in accordance with agreed architectures, standards, methods and procedures (including secure software development). Develops road maps to communicate future development activity.

**TEST4**

Accepts responsibility for creation of test cases using own in-depth technical analysis of both functional and non-functional specifications (such as reliability, efficiency, usability, maintainability and portability). Creates traceability records, from test cases back to requirements. Produces test scripts, materials and regression test packs to test new and amended software or services. Specifies requirements for environment, data, resources and tools. Interprets, executes and documents complex test scripts using agreed methods and standards. Records and analyses actions and results, and maintains a defect register. Reviews test results and modifies tests if necessary. Provides reports on progress, anomalies, risks and issues associated with the overall project. Reports on system quality and collects metrics on test cases. Provides specialist advice to support others.
Learning Outcomes

Upon completion of this module, candidates will be able to:

• Demonstrate a critical understanding of software systems engineering theory in the form of laws and models, and of associated methods, tools, and techniques.
• Critically apply the above to practical situations found throughout the Software Life Cycle from software requirements engineering, system specification, design, implementation, validation and verification, through to maintenance and evolution; and to recognize the potential for software reuse throughout the life cycle.
• Appreciate the importance of software project management, software economics, estimation and planning as well as the management of software project teams and their productivity.
• Discuss critically recent advances in software engineering: component-based software engineering, model driven software development, the Agile paradigm including Extreme programming, software product lines engineering and community based software development such as Free and Open Source Software development.
• Appraise advanced software concepts and their applicability in practice: Design Patterns and Frameworks, Software Refactoring techniques, Software Architectural analysis, Software as a Service.
Syllabus

1. Analysis and improvement of software processes

Learners will be able to:

1.1 Explain and apply software process improvement.

Indicative content

- Analyse and show understanding of software life cycle models.
  
  a. Waterfall
  b. V-model
  c. Prototyping
  d. Spiral model
  e. Incremental development
  f. Evolutionary development
  g. Agile models, including extreme programming

Guidance

Candidates are expected to understand what software process improvement is. They should be familiar with one or more of the methods or frameworks identified here, and should be able to apply or recommend one of these frameworks to a given scenario.

1.2 Analyse and show understanding of software life cycle models.

Indicative content

- ISO and IEEE Software Engineering standards, e.g.:
  
  a. CMM
  b. CMM-I
  c. SPICE (ISO/IEC 330XX)
  d. ISO 9001
  e. ISO/IEC 12207
  f. ISO/IEC 90003
  g. IEEE 1012

Guidance

Candidates should be able to show knowledge of various software development lifecycle models, consider process lifecycles, and recommend a particular approach for a given scenario. They need to have an appreciation for the traditional as well as modern approaches, of which agile and extreme are examples.

1.3 Demonstrate knowledge and awareness of software requirements engineering.

Indicative content

- Requirements management
- Requirements elicitation

Guidance

Candidates should show awareness of the different stages of requirements engineering, and be able to outline and recommend different strategies of management for a given case study scenario. They should also demonstrate knowledge of the requirements elicitation process, and be able to evaluate and recommend appropriate tools and techniques for eliciting user requirements.
1.4 Demonstrate knowledge and awareness of software management.

**Indicative content**
- a. Project management
- b. Cost estimation
- c. Planning
- d. Personnel management
- e. Team building

**Guidance**
Candidates should be able to demonstrate their knowledge of project management methods and techniques.

---

1.5 Demonstrate knowledge and awareness of the evolution of software.

**Indicative content**
- a. Lehman’s Laws of Software Evolution
- b. Related models and studies

**Guidance**
Candidates are required to demonstrate their awareness of and possibilities for software measurement. This includes the theory, methods and techniques, and their strengths and weaknesses.

---

2. Analysis and improvement of software products

**Learners will be able to:**

2.1 Demonstrate awareness of and ability to apply software maintenance methods.

**Indicative content**
- a. Impact Analysis
- b. Reverse and re-engineering of software

**Guidance**
Candidates should show awareness of software maintenance and methods available and applicable to the engineer for a given case study scenario. Candidates are expected to apply and recommend appropriate methods for the scenario presented.

---

2.2 Analyse and explain software architecture and software refactoring.

**Indicative content**
- a. Architectural styles, examples, and applications
- b. Architectural models
- c. Model-driven development

**Guidance**
Candidates are expected to demonstrate knowledge of various software architectures and evidence the strengths and weaknesses of different models in software development.
2.3 Evidence knowledge and awareness of software metrics.

**Indicative content**

a. Software complexity measures
b. Measures of software coupling and cohesion
c. Models and associated measures of software quality

**Guidance**

Candidates should evidence their knowledge of and their ability to use methods and techniques for measuring aspects of the software process and product.

3. Advanced topics in software engineering

**Learners will be able to:**

3.1 Evidence knowledge and awareness of the methods and techniques for software reuse.

**Indicative content**

a. Component based software engineering
b. Software product lines
c. Design patterns

**Guidance**

Candidates are expected to evidence their knowledge and awareness of the practice of software reuse, in particular the methods and the techniques available.

3.2 Analyse and explain software as a service.

**Indicative content**

a. Platform as a service
b. Infrastructure as a service, including:
   - Web services
   - Cloud computing
   - Dynamic reconfiguration of software systems

**Guidance**

Candidates are expected to evidence their knowledge and awareness of software as a service, in particular the methods and the techniques currently in use today.
3.3 Demonstrate knowledge and awareness of open-source software engineering practice.

**Indicative content**

- a. Open-source software
- b. Open-source development as a community
- c. Strengths and weaknesses in comparison with proprietary-based development
- d. Open source as good engineering practice

**Guidance**

Candidates are expected to demonstrate their knowledge and awareness of the practice of open-source development and efforts made to formulate as an engineering discipline.

3.4 Demonstrate knowledge and awareness of UML and its use.

**Indicative content**

- a. Object-constraint language
- b. Use of assertions
- c. Pre- and post-condition
- d. Invariants

**Guidance**

Candidates are expected to demonstrate knowledge of the object-constraint language and to be able to apply various constructs of assertion, pre- and post-conditions, and invariants, to a given software construction scenario.
Examination Format

This module is assessed through completion of an invigilated written exam.

<table>
<thead>
<tr>
<th>Type</th>
<th>Three written questions from a choice of five, each with equal marks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duration</td>
<td>Three hours</td>
</tr>
<tr>
<td>Supervised</td>
<td>Yes</td>
</tr>
<tr>
<td>Open Book</td>
<td>No (no materials can be taken into the examination room)</td>
</tr>
<tr>
<td>Passmark</td>
<td>10/25 (40%)</td>
</tr>
<tr>
<td>Delivery</td>
<td>Paper format only</td>
</tr>
</tbody>
</table>

Adjustments and/or additional time can be requested in line with the BCS reasonable adjustments policy for candidates with a disability, or other special considerations including English as a second language.

Question Weighting

Candidates will choose three questions from a choice of five. All questions are equally weighted and worth 25 marks.
Recommended Reading

Primary texts

Title: Software Engineering (tenth edition)
Author: I. Somerville
Publisher: Pearson
Date: 2015
ISBN: 978-1292096131

Title: Software Engineering: A Practitioner’s Approach (eighth edition)
Author: R. S. Pressman and B. Maxim
Publisher: McGraw-Hill Education
Date: 2014
ISBN: 978-1259253157

Additional Texts

Title: The Mythical Man-Month
Author: F. P. Brooks
Publisher: Addison-Wesley
Date: 1995
ISBN: 978-0201835953

Title: Software metrics: a rigorous and practical approach (third edition)
Author: N. Fenton and S. Pfleeger
Publisher: CRC Press
Date: 2015
ISBN: 978-1439838228
<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Design Patterns: Elements of Reusable Object-Oriented Software</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author</strong></td>
<td>E. Gamma, R. Helm, R. Johnson and J. Vlissides</td>
</tr>
<tr>
<td><strong>Publisher</strong></td>
<td>Addison-Wesley</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>1994</td>
</tr>
<tr>
<td><strong>ISBN</strong></td>
<td>978-0201633610</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Managing the Software Process (SEI)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author</strong></td>
<td>H. Watts</td>
</tr>
<tr>
<td><strong>Publisher</strong></td>
<td>Addison-Wesley</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>1989</td>
</tr>
<tr>
<td><strong>ISBN</strong></td>
<td>978-0201180954</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Code Quality: the Open Source Perspective</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author</strong></td>
<td>D. Spinellis</td>
</tr>
<tr>
<td><strong>Publisher</strong></td>
<td>Addison-Wesley</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>2006</td>
</tr>
<tr>
<td><strong>ISBN</strong></td>
<td>978-0321166074</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Extreme Programming Explained: Embrace Change (second edition)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author</strong></td>
<td>K. Beck</td>
</tr>
<tr>
<td><strong>Publisher</strong></td>
<td>Addison-Wesley</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>2004</td>
</tr>
<tr>
<td><strong>ISBN</strong></td>
<td>978-0321166074</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Title</strong></th>
<th>Agile Software Development (second edition)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Author</strong></td>
<td>A. Cockburn</td>
</tr>
<tr>
<td><strong>Publisher</strong></td>
<td>Addison-Wesley</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>2006</td>
</tr>
<tr>
<td><strong>ISBN</strong></td>
<td>978-0321482754</td>
</tr>
<tr>
<td>Title</td>
<td>Author</td>
</tr>
<tr>
<td>--------------------------------------------</td>
<td>---------------------------------------</td>
</tr>
<tr>
<td>Software Architecture in Practice (SEI)</td>
<td>L. Bass, P. Clements and R. Kazman</td>
</tr>
<tr>
<td>Requirements Engineering</td>
<td>E. Hull, K. Jackson and J. Dick</td>
</tr>
<tr>
<td>Software Evolution</td>
<td>T. Mens and S. Demeyer</td>
</tr>
</tbody>
</table>

Using BCS Books

Accredited Training Organisations may include excerpts from BCS books in the course materials. If you wish to use excerpts from the books you will need a license from BCS. To request a license, please contact the Head of Publishing at BCS outlining the material you wish to copy and its intended use.
Document Change History

Any changes made to the syllabus shall be clearly documented with a change history log. This shall include the latest version number, date of the amendment and changes made. The purpose is to identify quickly what changes have been made.

<table>
<thead>
<tr>
<th>Version Number</th>
<th>Changes Made</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version 1.0</td>
<td>Document Creation</td>
</tr>
<tr>
<td>July 2021</td>
<td></td>
</tr>
</tbody>
</table>
CONTACT

For further information please contact:

BCS
The Chartered Institute for IT
3 Newbridge Square
Swindon
SN1 1BY

T +44 (0)1793 417 445

www.bcs.org

© 2021 Reserved. BCS, The Chartered Institute for IT

All rights reserved. No part of this material protected by this copyright may be reproduced or utilised in any form, or by any means, electronic or mechanical, including photocopying, recording, or by any information storage and retrieval system without prior authorisation and credit to BCS, The Chartered Institute for IT.

Although BCS, The Chartered Institute for IT has used reasonable endeavours in compiling the document it does not guarantee nor shall it be responsible for reliance upon the contents of the document and shall not be liable for any false, inaccurate or incomplete information. Any reliance placed upon the contents by the reader is at the reader’s sole risk and BCS, The Chartered Institute for IT shall not be liable for any consequences of such reliance.