Unit 9: Software Development Lifecycles

Unit code	**J/615/1631**
Unit level	**4**
Credit value	**15**

Introduction

The software development lifecycle is an integrated process that promotes building good quality, secure software throughout the entire development process. The aim of this unit is to provide students with the knowledge and skills needed to understand software development lifecycles and to demonstrate their knowledge by implementing a software development lifecycle with a suitable methodology.

This unit introduces students to lifecycle decision-making at different stages of the software development process. Students will examine various lifecycle models and appreciate their particular characteristics to understand which project environments they are most appropriate for. Theoretical understanding will be translated into practical skills through an actual software development lifecycle project and students will become confident in the use of particular tools and techniques relevant to a chosen methodology.

Among the topics included in this unit are iterative and sequential models of software development lifecycles and reference frameworks for initially capturing conceptual data and information through a feasibility study and requirement gathering techniques through to analysis, design and software implementation activities.

As a result students will develop skills such as communication literacy, critical thinking, analysis, reasoning and interpretation, which are crucial for gaining employment and developing academic competence.

Learning Outcomes

By the end of this unit students will be able to:

LO1 Describe different software development lifecycles.

LO2 Explain the importance of a feasibility study.

LO3 Undertake a software development lifecycle.

LO4 Discuss the suitability of software behavioural design techniques.

Essential Content

LO1 **Describe different software development lifecycles**

Software development lifecycles:

Lifecycle models: understanding and use of predictive (Waterfall, Prototyping, RAD) and adaptive (Spiral, Agile, DSDM) software development models.

Lifecycle stage and connectivity: feasibility study, analysis, design, implementation, testing, review or analysis, design, implementation, maintenance, planning; requirements traceability.

Test and integration: building test environments; developing test harnesses; black box/white box testing; incremental testing; acceptance test and integration approaches; changeover strategies, trials and Go-Live prerequisites.

LO2 **Explain the importance of a feasibility study**

Importance of feasibility study:

Requirement gathering techniques: e.g., interviews, observation, investigation

Key drivers: performance and efficiency; legacy systems upgrade; automation; elimination of human error.

Feasibility criteria: issues e.g. legal, social, economic, technical, timescales; organisational constraints.

Components: purpose; structure; intended audience; outcomes.

Requirements: MosCow; Functional; non-functional; user; constraints.

LO3 **Undertake a software development lifecycle**

Carry out software development lifecycle:

Identify requirements: stakeholders; requirements identification; requirements specification e.g. scope, inputs, outputs, processes and process descriptors; consideration of alternate solutions and security considerations; quality assurance required.

Constraints: specific to activity e.g. costs, organisational policies, legacy systems, hardware requirements.

Report documentation: structure e.g. background information, problem statements, data collection process and summary, recommendations, appendices.

Systems analysis terminology and tools: data stores and entities; data flows; process representation techniques relationships – 1:1, 1:Many (1:M) and Many:Many (M:M).

Investigation: e.g. upgrading computer systems, designing new systems.

Techniques: examples relevant to methodology chosen e.g. Context Diagrams, Data Flow Diagrams (DFDs), Entity Relationship Diagrams (ERDs); Business Systems Options (BSOs); Technical Systems Options (TSOs); quality considerations e.g. Total Quality Management (TQM).

LO4 **Discuss the suitability of software behavioural design techniques**

Evaluate suitability of software behavioural design techniques:

Techniques: Flowcharts; Pseudocode; Formal specification Methods; Event/State/Data Driven; Finite State Machines (extended-FSM)/FSP; problem of e-FSM state explosion; reachability analysis, safety, liveness properties; Automatic analysis and animation tools.

Learning Outcomes and Assessment Criteria

Pass	Merit	Distinction
LO1 Describe different software development lifecycles		**D1** Assess the merits of applying the Waterfall lifecycle model to a large software development project.
P1 Describe two iterative and two sequential software lifecycle models. **P2** Explain how risk is managed in these models.	**M1** Discuss, with an example, why a particular lifecycle model is selected for a development environment.	
LO2 Explain the importance of a feasibility study		**D2** Assess the impact of different feasibility criteria on a software investigation.
P3 Explain the purpose of a feasibility report. **P4** Describe how technical solutions can be compared.	**M2** Discuss the components of a feasibility report.	
LO3 Undertake a software development lifecycle		**M3** Evaluate the process of undertaking a systems investigation with regards to its effectiveness in improving a software quality.
P5 Undertake a software investigation to meet a business need. **P6** Use appropriate software analysis tools/techniques to carry out a software investigation and create supporting documentation.	**M3** Analyse how software requirements can be traced throughout the software lifecycle. **M4** Discuss two approaches to improving software quality.	
LO4 Discuss the suitability of software behavioural design techniques		**D4** Present justifications of how data driven software can improve the reliability and effectiveness of software.
P7 Discuss using examples the suitability of software behavioural design techniques.	**M5** Analyse a range of software behavioural tools and techniques. **M6** Differentiate between a finite state machine (FSM) and an extended-FSM, providing an application for both.	

Recommended Resources

Textbooks

Ferguson, J. (2014) *BDD in Action: Behavior-driven development for the whole software lifecycle*. Manning.

Dennis, A. and Haley, W. (2009) *Systems Analysis and Design*. John Wiley & Sons Ltd.

Lejk, M. and Deeks, D. (2002) *An Introduction to System Analysis Techniques*. 2nd Ed. Addison-Wesley.

Murch, R. (2012) *The Software Development Lifecycle: A Complete Guide*. Kindle.

Websites

www.freetutes.com	FreeTutes "Systems Analysis and Design – Complete Introductory Tutorial for Software Engineering" (Tutorial)
www.ijcsi.org	IJCSI International Journal of Computer Science Vol. 7, Issue 5, September 2010 "A Comparison Between Five Models Of Software Engineering" (Research)
www.ijcsi.org	IJCSI International Journal of Computer Science Vol. 6, Issue 1, 2015 "Software Development Life Cycle Models – Comparison, Consequences" (Research)

Links

This unit links to the following related units:

Unit 6: Managing a Successful Computing Project

Unit 13: Computing Research Project

Unit 28: Prototyping

Unit 30: Application Development

Unit 32: Game Design Theory

Unit 34: Systems Analysis & Design

Unit 47: Games Development