

SE 2AA4 / CS 2ME3 Software Design I

Alan Wassyng and Spencer Smith

January 4, 2017

1 What the students should know and be able to do

1. Students should know and understand
 - (a) the software engineering life cycle and the role of software design
 - (b) the need for rigour
 - (c) software design processes that enable us to cope with complex projects
 - (d) the importance of modularity
 - (e) the role of specification – requirements and module interfaces
 - (f) Mill’s black-box model and the finite state machine model
 - (g) the principle of information hiding
 - (h) verification – especially testing
 - (i) the difference between black-box and white-box testing
2. Students should be able to
 - (a) understand & create tabular expressions
 - (b) understand & create pre- and post-conditions
 - (c) decompose a requirements spec into a modular design
 - (d) evaluate the quality of a design
 - (e) create an information hiding design
 - (f) create a module interface specification
 - (g) create a module internal design
 - (h) implement the design in code
 - (i) create black-box tests for a class
 - (j) create white-box tests for a class

2 Mapping to Attributes with their Indicators

Knowledge Base for Engineering

- | | |
|--|--------------|
| 1. Competence in Mathematics | 1e, 2a–2b |
| 3. Competence in Engineering Fundamentals | 1b–1d, 2c–2d |
| 4. Competence in Specialized Engineering Knowledge | 1a–1i, 2e–2j |

Problem Analysis

- | | |
|---|----------------------|
| 2. Ability to identify reasonable assumptions | 1c–1g, 2a–2c |
| 4. Ability to decompose and organize a problem into manageable sub-problems | 1a, 1c–1d, 2c, 2e–2g |
| 6. The ability to use modern/state of the art tools | 2h |

Investigation

- | | |
|--|-----------|
| 1. Able to recognize and discuss applicable theory knowledge base | 2a–2b |
| 2. Capable of selecting appropriate model and methods and identify assumptions and constraints | 2f, 2i–2j |
| 5. Assess the accuracy and precision of results and recognize limitations of the approach | 2d, 2i–2j |
| 6. Properly documents and communicates processes and outcomes | 2c–2j |

Design

- | | |
|---|------------------|
| 1. Recognizes and follows an engineering design process | 1a–1i, 2a–2j |
| 2. Recognizes and follows engineering design principles | 1b–1d, 1g, 2c–2g |
| 3. Obtains experience with open-ended problems | 2c–2j |
| 7. Properly documents and communicates processes and outcomes | 2c–2j |

Use of Engineering Tools

- | | |
|---|----|
| 2. The ability to use modern/state of the art tools | 2h |
|---|----|

3 Rubrics

Rubric 1: Student work used: assignments.

Topic	Below	Marginal	Meets	Exceeds
Decompose into modules decompose requirements into well-structured modules 2c, 2d, 2e	cannot account for all requirements in the decomposed system %	can include all requirements, but design has too few or too many modules %	can include all requirements, and decomposition is reasonable but information hiding not achieved %	can decompose into well-structured modules complying with information hiding, secrets traced %
Document module behaviour document MIS and MID for each module 2f, 2g, 2e	cannot create adequate MIS and MID for each module %	can create MIS but not MID for each module %	can create both MIS and MID for each module %	can create both MIS and MID for each module, and MIS does not violate information hiding %
Implement design create executable code from design 2h	cannot create implementation - not complete or cannot execute adequately %	can create code, but code does not match design %	can create code that matches design, but code is not efficient %	can create efficient code that matches design %
Testing testing of the implementation 2i, 2j	cannot create adequate tests %	can create black-box tests but not white-box tests %	can create white-box tests but not black-box tests %	can create both black-box and white-box tests %

Rubric 2: Student work used: midterm exam

Topic	Below	Marginal	Meets	Exceeds
Modular design quality of modular design 1c-1d, 1g 2d-2e	cannot understand what makes a quality decomposition %	can understand basic decomposition quality, but not finer points %	can understand basic decomposition quality, including finer structural points %	can understand advanced decomposition quality, and information hiding %
MIS and MID understand MIS and MID 1e, 2f-2g	cannot understand MIS or MID %	can understand MIS but not MID %	can understand basic concepts of both MIS and MID %	can understand advanced concepts of both MIS and MID %
Tabular expressions create and understand tabular expressions 2a	cannot understand or create tabular expressions %	can understand tabular expressions but cannot create them %	can understand and create tabular expressions, but don't understand completeness and disjointness %	can understand and create tabular expressions including completeness and disjointness %
FSM work with models, especially FSMs 1f	cannot understand or create FSMs %	can understand FSMs but cannot create them %	can understand and create FSMs %	can understand and create FSMs and use them for analysis %

Rubric 3: Student work used: final exam

Topic	Below	Marginal	Meets	Exceeds
Software life-cycle general software engineering life cycle and role of software design 1a, 1c	cannot explain the phases in the life-cycle or the role of software design %	can understand the basic life cycle phases, but not relationships between them %	can understand relationships between life cycle phases, but not the role of people %	can understand the software life cycle and the role of people %
Modular design quality of modular design 1c-1d, 1g 2d-2e	cannot understand what makes a quality decomposition %	can understand basic decomposition quality, but not finer points %	can understand basic decomposition quality, including finer structural points %	can understand advanced decomposition quality, and information hiding %
MIS and MID understand MIS and MID 1e, 2f-2g	cannot understand MIS or MID %	can understand MIS but not MID %	can understand basic concepts of both MIS and MID %	can understand advanced concepts of both MIS and MID %
FSM work with models, especially FSMs 1f	cannot understand or create FSMs %	can understand FSMs but cannot create them %	can understand and create FSMs %	can understand and create FSMs and use them for analysis %
Tabular expressions create and understand tabular expressions 2a	cannot understand or create tabular expressions %	can understand tabular expressions but cannot create them %	can understand and create tabular expressions, but don't understand completeness and disjointness %	can understand and create tabular expressions including completeness and disjointness %
Pre- and post-conditions create and understand pre- and post-conditions 2a	cannot understand or create pre- and post-conditions %	can understand pre- and post-conditions but cannot create them %	can understand and (almost) create pre- and post-conditions, but don't understand logic expressions well enough %	can understand and create pre- and post-conditions, including well-formed logic expressions %
Testing black-box and white-box tests 2i, 2j	cannot create adequate tests %	can create black-box tests but not white-box tests %	can create white-box tests but not black-box tests %	can create both black-box and white-box tests %