On Assuring Learning about Code Quality

Diana Kirk, Tyne Crow, Andrew Luxton-Reilly & Ewan Tempero
The University of Auckland
ACE 2020

Motivation

- Development costs of a software product are incurred mostly after the first delivery i.e. during the *maintenance* phase.
  - enhancements
  - correcting defects
  - modifying for different environments

Motivation

- Code quality significantly affects these costs
- Employers need evidence that developers are knowledgeable about code quality

- *Code Quality should be a key competency of graduates of computing qualifications*

- *Would like to provide evidence that code quality is taught from the first course in programming*

Motivation

- Research question:

  To what extent is code quality taught and assessed in first courses in programming?

Approach

- Code quality has many aspects. Our interest is in supporting maintainability.
- We focus on comprehensibility, one of the key aspects of maintainability.

Comprehensibility: How easy or difficult is it to understand a piece of code?

Approach

- Our focus relates to code style. We exclude
  - syntax (does the code adhere to the rules of the language - form)
  - semantics (does the program work as expected - meaning).

Approach

- Base analysis on Stegeman et al. rubric for code quality
  - Documentation: names, headers, comments
  - Presentation: layout, formatting
  - Algorithms: control flow, idiom, expressions
  - Structure: decomposition, modularization
- By ‘code quality’, we mean the elements represented by this framework.

Stegeman, M., Barendsen, E. & Smetsers, S. (2016). ***Designing a Rubric for Feedback on Code Quality in Programming Courses.*** *In Proc. Koli Calling ‘16.*

Approach

- Compare three systems for computing education:
  - Process based on ACM Guidelines for Computer Science (ACM CS)
    - *de-facto standard for universities*
  - New Zealand (NZ) high school curriculum for technology
    - *basis of education in NZ schools*
  - Cambridge Assessment International Education for Computer Science (CAIE CS)
    - *internationally recognised education system*

Universities - Foundations

- ACM/IEEE Guidelines for CS and SE curricula
  - created by joint task force on computing curricula
  - Body of Knowledge (hierarchical knowledge structure)

Software construction	CS	Coding techniques, mechanisms for quality	S
		Coding standards	S
Software evolution	CS	Characteristics of maintainable software	S
Computing essentials	SE	Comments , structure, readability	M
		Code reuse	M
Software design	SE	Information hiding, coupling, cohesion	M

S = Should ; M = Must

Universities - Operationalisation

- Learning Outcomes (LOs) for 141 introductory CS courses
  - LOs represent what lecturers believe is important
  - high alignment between LO and ACM Guideline concepts

USA	63
England	26
Australia	8
Ireland	7
Scotland	6
New Zealand	4
Canada, Netherlands, Sweden	3
Lebanon, South Africa, Turkey, Wales	2
China, Czeck Republic, Denmark, Hungary, Jordan, Kenya, Phillipines, Singapore, Switzerland, United Arab Emirates	1

Becker, B., & Fitzpatrick, T. (2019). ***What Do CSI Syllabi Reveal About Our Expectations of Introductory Programming Students.*** *In Proc. SIGCSE 2019.*

Universities - Operationalisation

- 141 courses
  - 41 (29 percent) have LOs that mention code quality
  - mostly not specific

Focus	#	Example
General	12	Write … good programming style
Rationale	11	Document … for future maintenance
Specific	9	… employ functions and modularity
Document	8	Generating clear documentation
Embedded	5	Write well-structured … code

Universities - Operationalisation

- Syllabus and LOs set by lecturers

Criterion	Universities
Quality content	Individual
Variability	High

For most first year courses, we cannot be confident that students are exposed to concepts relating to code quality and its importance.

Universities - Assessment

- Typically
  - several components assessed individually, aggregated and total compared with thresholds
  - often part-marks for questions
  - marking rubrics created and marked internally

Criterion	Universities
Assessment	University
Assessors	University
Marking	Aggregated

Even if the course covers code quality aspects, a student might succeed overall while having failed to achieve in components relating to code quality.

NZ Schools - Foundations

- New Zealand Curriculum (NZC) for technology
  - initiative of the NZ Ministry of Education
  - consulted with teachers, subject experts, Māori leaders
  - vision is creative, resilient, confident, lifelong learners

NZ Schools - Operationalisation

- Students work towards National Certificate of Education (NCEA) in Digital Technologies
  - main topic for programing is
    - *Develop a computer program*
  - based on curriculum plus supporting materials
  - quality terms found in *Progress Outcomes* plus *exemplars*

Apply key concepts to design and develop
Annotate code with … implementation decisions
Clear variable names using conventions

NZ Schools - Operationalisation

- Effectively a `community of practice'
- BUT large variation in classroom context
  - class level and mix; teaching and programming experience; environments used

Criterion	NZ Schools
Quality content	Guidelines
Variability	Medium

We can be confident that students are exposed to quality concepts.

NZ Schools - Assessment

- Assessed by set of Achievement Standards
  - 3 levels (ages 14-18)
- Achievement levels
  - ***D****evelop,* ***I****nformed,* ***R****efined*
- Expectations that student satisfies all the requirements for a given level. However, some concepts are *optional* i.e. are selected from a given list.

NZ Schools - Assessment

Standard: Develop a computer program

1	91883	D	Set out code clearly
			Document with comments
			User defined methods, functions or procedures
		I	Variable names and comments that describe behaviour
			Follow language conventions
			Effective use of methods, functions, procedures
			Use constants, variables and derived values in place of literals
2	91896	D	Clear code and documentation
			Methods, functions, procedures using parameters and/or return values
		I	Document with names and comments that describe code function and behaviour
			Follow programming language conventions
			Effective use of methods, functions, procedures, …
			Use constants, variables and derived values in place of literals

NZ Schools - Assessment

- Marking rubrics created by NZQA
  - standards explicit about expectations for quality
  - all chosen elements must be demonstrated to achieve grade
- BUT
  - not all elements chosen
  - marking by teachers who vary in understanding
  - standards updated more frequently than curriculum

Criterion	NZ Schools
Assessment	NZQA
Assessors	School
Marking	Compulsory

We can be reasonably confident about what a student with a specific grade knows about quality.

Cambridge - Foundations

- Cambridge International (CI)
  - syndicate formed by University of Cambridge in UK
  - over 150 years old
  - leading provider of international qualifications for 14-19 year olds
- Believe best results when curriculum, teaching and assessment closely aligned.

Cambridge - Operationalisation

- Computer Science syllabus
  - "encourage the use of computational thinking … by the use of abstraction and decomposition"

- Relevant topics for programming are:
  - *Algorithm design and problem solving (2.1)*
  - *Programming (2.3)*
  - *Software development (2.4)*
  - *Software engineering (4.4)*

Cambridge - Operationalisation

- Quality-related terms in LOs

Topic	#	LO terms
Algorithm design	2.1.1	Suitable identifier names
		Decompose problem … modules
Structured programming	2.3.6	Use / explain procedure / function
Abstraction	4.1.1	Functions with suitable parameters

Cambridge - Operationalisation

- Approach is holistic and quality concepts supported
- BUT no ‘why’?

Criterion	Cambridge
Quality content	CAIE
Variability	Low

We can be confident that students are exposed to some quality techniques but cannot be sure they understand e.g. ‘maintainability’

Cambridge - Assessment

- Question papers and marking schemes “set by appropriately qualified personnel”
  - all questions must be attempted
  - grade based on aggregated marks achieved

Criterion	Cambridge
Assessment	CAIE
Assessors	CAIE
Marking	Aggregated

Even if the course covers code quality aspects, a student might succeed with no knowledge of code quality.

Cambridge - Assessment

- In marking schemes available on line (5 examples), we found no evidence of marks being awarded specifically for quality techniques.
- We infer that ‘quality’ is not seen to be of primary importance, at least in recent years.

Limitations

- Examined LOs from only 141 CS1 courses
  - representative?
- Examined first programming courses only
  - some believe quality should be taught later
- Only two high school curricula
  - differed significantly and so fruitful comparison
- Focus on code quality
  - clearly not the only key competency required

Summary

- Our perspective is that
  - stakeholders care about reducing maintainability costs
  - it's our job to provide clear evidence that students have been exposed to issues relating to code quality
  - we believe that quality-related habits must be ingrained early

Summary

- Our main finding is the **significant variation** in the commitment to software quality
  - consistency with which quality expectations are stated
  - focus on quality during assessment

We believe that this represents a significant problem that requires attention.

Future work

- Questions
  - Are ACM and CAIE coverage of quality deeply considered, or a result of having roots in ‘Knowledge based’ systems?
  - Does a focus on quality **techniques** without understanding **why** impact a student’s motivation
  - Is there a difference in appreciation of code quality between students taught in a knowledge-based system and those taught in a student-centred system?

Thank You