A critical analysis of Computer Science A-level within a college curriculum, using design and evaluation theory to determine the efficacy of self-assessment and quality improvement

Candidate number: 26725452
Word count: 3,982

Introduction

There is no clearly set definition of curriculum. A naïve description might be that it is the content taught for a subject studied: a syllabus. It is tempting for teachers to only think about the subjects that they teach. However, it is important to consider the wider experience of the students, which ranges from the other subjects they study, to the institution's tutorial and pastoral programmes, to the sort of food that is sold in the canteen. All these things contribute to the overall curriculum. It is also crucial to recognise the difference between the 'planned' and 'received' curriculum (Kelly, 2004). This addition includes the student experience as well as the teacher's duty. A teacher might interpret what is written by an awarding body in their course specification, then deliver it in a certain way. Some detail or bias may be added or removed in this transformation, until the student eventually receives some interpretation of the original syllabus. By including the received curriculum in the definition, the way in which institutions and teachers choose to deliver a course can also be evaluated.

This essay follows the curriculum delivered by a sixth form college in the south of England, hereafter referred to as 'the college'. The precise aspect that is reviewed, analysed and evaluated is Computer Science A-level. Of course, it is still important to consider the wider curriculum that this programme sits within. The most popular companion courses are Mathematics and Physics, with 68% and 40%, respectively, of students enrolled for Computer Science also studying them at A-level.

Curriculum Design

The planned curriculum at the college follows the AQA Computer Science A-level specification. This contains a comprehensive list of subject content to be taught, as well as a clear description of how the course is to be assessed (AQA, 2019). The specification is not a completely original creation by AQA; its synthesis is part of a chain of documents produced for the A-level course, which undergo regular review and revision. The basic expected outcomes of assessment are “set by Ofqual and are the same across all AS and A-level Computer Science specifications and all exam boards” (AQA, 2019, p.112). These assessment objectives (AOs) are used by awarding bodies like AQA when designing assessment criteria. Together, they cover Bloom’s taxonomy of learning: AO1 covers ‘knowledge’ and ‘comprehension’; AO2 covers ‘application’ and ‘analysis’; and AO3 covers ‘synthesis’ and ‘evaluation’ (Bloom, 1956; Ofqual, 2014). Ofqual restrict the weightings for each of the AOs to 30-40%. They also enforce that A-level courses must not be assessed more than 20% by coursework, with the remaining proportion taken up by examinations (Ofqual, 2014). Course specifications must follow the aims, subject knowledge and skills set out by the Department for Education (2014). However, this guide is brief and the full AQA specification goes into much more detail.

One of the most prominent theoretical models of curriculum development from the last century is the ‘product’ model. Its main innovators were: Ralph Tyler, who believed that the successes of education could most effectively be measured when the expected outcomes were initially defined; and Bloom (1956), who scientifically categorised types of aims and objectives (Kelly, 2004). With this model, it is first decided what behavioural objectives constitute ‘success’, followed by formulation of a bank of content, skills and activities to achieve these objectives. Then, there is emphasis on learners being assessed against them to determine their level of success (Smith, 2000).

At the highest level, UK state education unambiguously follows a product model, and this is no different for Computer Science A-level. The explicit “Aims and Objectives” outlined by government show that behavioural objectives are at the root of the curriculum development process (Ofqual, 2014; DfE, 2014). At the course specification level, there is a large set of facts which students are expected to know and understand. If students can recall the facts correctly, they will be successful (AQA, 2019). There is also a significant skills element in the course, which includes “a minimum of 10% mathematics” (DfE, 2014, p.3) and an element of computer programming. Students are expected to have a pre-defined level of competence in these skills in order to be successful.

A key advantage of the objectives-based approach is that the students can be objectively assessed. Crucially, this makes the outcomes of students comparable, which is favourable to recruiting employers and institutions of further study. For example, FE institutions should be able to assume prerequisite knowledge or competence based on a threshold of attainment at GCSE. Similarly, universities should be able to assume what a student with an A* grade in Mathematics A-level should know and be able to do. This should allow these institutions to be fairer in their admissions processes, and to design their own curricula based on a common starting point in order to waste less time with students learning content twice. Of course, the situation is never so perfectly streamlined. Although students may be fairly assessed, there is no guarantee that relevant knowledge and skills are captured. Also, the recent UK A-level reforms have led to fewer terminal examinations, which cannot realistically assess the entire body of content in the course syllabus.

Another identified limitation is that it can inhibit the freedom and creativity of both teachers and students. Students get little say in what they learn about, because the set of expected outcomes is already defined. Also, “it turns educators into technicians” (Smith, 2000), whose role is to deliver a course that has been largely already planned for them. In Computer Science, there is so much content for teachers to cover that very little time is left for personalising the curriculum based on their own or their students’ interests.

The largest criticism of the product model is that it does not acknowledge learning for learning’s sake. It exists only for utilitarian purposes, to give students the knowledge and skills required to be effective workers who contribute positively to the economy. The view of Kelly (2004, p.68) is that this is “leading to a lowering of educational standards whatever it seems to be achieving in terms of its own spurious statistics”. This is to say that league tables and measures used to evaluate student, teacher, institution and subject success are fabricated by the product curriculum and serve no true ‘educational’ purpose. This sort of view led to a different way of approaching curriculum design, called the ‘process’ model. This model disregards prescriptive behavioural objectives and focuses more on the means rather than the ends of education. It lends less importance to the exact content that should be delivered, and instead recognises intrinsic value in the processes by which content is experienced by students (Smith, 2000).

There are aspects of the planned curriculum which appear to follow the process model. One of these is the skill of computer programming as a theme throughout the course. Although students are examined and assessed in their competence at programming, it is seen as something which students learn to appreciate as a tool for solving problems. Another idea that is central to the Computing curriculum at all levels is ‘computational thinking’: “an approach to solving problems, designing systems and understanding human behaviour that draws on concepts fundamental to computing” (Wing, 2008, p.3717). This is a “key process” and “something that a pupil of Computer Science should be able to do” (CAS, 2012, p.9). Although computational thinking is prescribed as a behavioural objective of the A-level curriculum, the idea is certainly more in line with the process model. This is because it ignores the exact problems which are to be solved and focuses instead on practising a broad process by which to solve any problem. Flexibility is left to both teachers and students to decide on appropriate examples of problems to tackle using the process. This focus is very good because today’s A-level needs to prepare young people for a technological future with jobs, systems and problems which do not exist yet.

Perhaps the most relevant limitation of the process model is that it is not designed for a ‘product’ desire to be bound by a cage of assessments. In the current UK education system, examinations are unfortunately necessary. This is an issue because the process model “can never be directed towards an examination as an objective without loss of quality” (Stenhouse, 1975, p.95). Despite the best intentions of curriculum planning to encourage computational thinking, there undoubtedly is a loss of accuracy due to primarily gearing students towards terminal A-level examinations and minimal coursework. This is because “grades are attainable without understanding” (Stenhouse, 1975, p.96).

It is important to recognise how A-level Computer Science sits between lower and higher levels of study. Recently, there has been significant government reform to the Computing curriculum below FE, with ‘Computer Science’ being brought into the suite of GCSEs and the National Curriculum. With fundamentals now being taught from Key Stage 1, it is the government’s hope that “a high-quality computing education equips pupils to use computational thinking and creativity to understand and change the world” (DfE, 2013, p.1). The aims of the GCSE and A-level are virtually identical, and the apparent level of difficulty in content is higher for the A-level. (DfE, 2013; 2015; 2014).

The recent reforms aim to transform Computing into a ‘spiral curriculum’. At A-level, it is even made explicit that “all specifications in computer science must build on the knowledge, understanding and skills established at key stage 4” (AQA, 2019, p.111). A spiral curriculum cyclically revisits a subject at increasing levels, each time building on the previous level (McKimm, 2007). This is a good thing since, as with the more traditional sciences and mathematics, Computer Science cannot simply be understood after a 2-year A-level course. However, the current state of low take-up at GCSE and A-level means that the transitions into and out of A-level are disjointed. FE institutions cannot set a prerequisite entry requirement of GCSE Computer Science because not enough schools offer it. Setting such expectations would disadvantage students unable to study the subject well at school. An identical issue exists for HE admissions onto Computer Science degrees. The conflict between expectations for learners in the planned curriculum and the reality of the learner profiles in the A-level classroom must cause problems. If expected outcomes assume the existence of a functional spiral curriculum but some learners are taught from scratch at each level of the ‘spiral’, there undoubtedly must be an impact on the attainment of those outcomes. At the college, and nationally, significantly lower than average grades are achieved in A-level Computer Science qualifications (JCQ, 2018). This is not necessarily an indication of underperformance because of the way grade boundaries are moderated by Ofqual. However, the problem still exists for those learners who are not lucky enough to progress as planned through the spiral curriculum.

Curriculum Evaluation

The course specification is just one part of Computer Science A-level. At the college, decisions are made about how to deliver the formal specification, from structuring topics into a scheme of work down to planning individual lessons and activities. Since this part of the curriculum is very much under the control of the college, there is scope to self-evaluate the effectiveness of these decisions. In doing so, plans can be drawn up for improving the quality of the 'received' curriculum. At the college, Computer Science is not a standalone course, but part of a much larger 'Maths and Computer Science' department, which itself is one of eleven departments. Perhaps non-intuitively, it makes more sense for Computer Science to be grouped with Mathematics than with IT. This is because "Computer Science is a quintessential STEM discipline" (CAS, 2012, p.4). The self-assessment policy of the college involves every department producing a Self-Assessment Report (SAR) and resulting Quality Improvement Plan (QIP) (College, 2018b). The SAR for Maths & Computer Science makes reflective comments on the state of the subjects within the department (Maths A-level, Further Maths A-level, Maths Studies Cert., Maths GCSE and Computer Science), highlighting key strengths as well as areas for improvement. These comments are based on various quantitative and qualitative evaluative tools. The efficacy of this process at the college is to be analysed against Kirkpatrick's four-level theory of evaluation. Although this model was initially developed for application in adult training contexts, its flexibility means that it can be applied to educational curricula too (Naugle et al., 2000). It is worth recognising that this evaluative model is designed to suit a product-based curriculum; "What topics should be presented to meet the needs and accomplish the objectives? The answers to this question establish the topics to be covered" (Kirkpatrick & Kirkpatrick, 2006, p.9). Since the method of evaluation can often shape the curriculum itself (Kelly, 2004), there would be a danger if using Kirkpatrick's model for the more process-based elements of the curriculum, such as computational thinking or computer programming. It should not necessarily be a set of hard rules to aim towards, but more of a guide.

The first level looks at the personal reactions of participants (Kirkpatrick & Kirkpatrick, 2006). One important evaluative tool used by the department is an annual multifaceted review called the '3D Review'. In 2018/19, it aimed to analyse the efficacy of part of the department's QIP at the time, which focused on improving results of low prior achievers. The review included teacher peer-observation and student voice. One conclusion from the student questionnaires was that there needed to be a revision of the way weekly assignments worked. The use of student voice here was in line with Level 1: 'Reaction'. Although it did not evaluate the students' learning, it did give an indication of whether the structure of the programme worked for them.

After ascertaining the reaction of students to the course, the second level involves evaluating the "extent to which participants change attitudes, improve knowledge, and/or increase skill" (Kirkpatrick & Kirkpatrick, 2006). In an educational context, this is usually judged through assessment. At the weekly level at the college, students are given brief 'Key Assignment Reviews', which aim to gauge students' knowledge and skills studied in the previous week. The length and effectiveness of these were discussed in one of the 3D Review meetings. Students are also set weekly homework for them to assess their own knowledge. However, most homework is self-marked, as students are given answers along with the questions. A danger here is that some students use the answers as a crutch when completing homework and are not getting the 'desirable difficulty' of forced retrieval that has been shown to contribute towards better learning. Bjork & Bjork (2011, p.61) describe "the generation effect, which refers to the long-term benefit of generating an answer, solution, or procedure versus being presented that answer". A related danger is that teachers do not receive a consistent, reliable measure of student learning, which hinders fine-grained evaluation at the 'Learning' level.

The ultimate assessment of learning at the college uses final A-level grades. Apart from the 3D review, the other main evaluative tool is aggregate data on course entries, retention and achievement. In 2017/18 there were "too few A*s in [Computer Science], too many D/E grades" (College, 2018a). The proportion of students achieving A*-B was 37%, compared with a departmental A-level average of 57%. On the surface this looked like poor performance and the self-assessment seemed to conclude that the teachers were responsible. However, the national average equivalent figure was 39% (JCQ, 2018). It is therefore important to compare college figures to local and national benchmarks.

Another way to produce comparable aggregate data is to look at the 'value added' by the curriculum using the Alps system. Courses receive Alps grades between 1 and 9, which indicate where that course places in the national rankings for adding value to students. The attraction to Alps is firstly due to not being penalised for taking on low-achieving students. However, perhaps the biggest selling point for Alps is that it gives an extremely simple grade that can be compared year-on-year to see the progression of a course. At the college, Alps indicated that there were no significant overall concerns that the high-grades rate had implied; Computer Science had "largest ever intake, Alps still at 5" (College, 2018a). An Alps grade of 5 meant the course was roughly at the 50th percentile of the National Benchmark: it was on par with the average (Alps, 2019).

All departmental SARs are collated, and overall strengths and weaknesses are identified in the college SAR. For example, the 2017/18 value-added data showed that boys made significantly worse progress compared with girls (College, 2018b). However, this was not the case for Computer Science, where the boys achieved Alps 5 compared with 6 for the girls (College, 2018a). This shows that there is value in self-assessment at both the department and college level. The department is focusing on gender success rates as a means of evaluating Equality & Diversity aspects of course provision. "Boys and white students outperform their counterparts. This is something we are working on changing" (College, 2018a). There is nothing in the formal Computer Science specification which explicitly challenges gender stereotypes (AQA, 2019), so such challenges must come as teacher inputs into the curriculum. However, what should be of larger concern is that the number of girls on the course is pitifully low, with only 13% of new students in 2018/19 being female. Perhaps the curriculum is failing to recruit enough young women. Regardless, more needs to be done to encourage female recruitment to the programme. Although this is a systemic issue which may appear from a young age, FE still has responsibility for encouraging more girls to choose Computer Science as an A-level.

There is no doubt that discussing the value added by a college's curriculum is much more meaningful than simply comparing attainment metrics such as high-grades. However, with such a large emphasis on Alps at the college, there is a significant danger that it overlooks some of the oversimplifications that Alps can make. The main underlying assumption of Alps is that students with higher entry profiles are expected to achieve higher A-level grades. Minimum Expected Grades (MEGs) are generated for each student based on their GCSEs. Alps grades are calculated by taking the average difference between actual A-level results and MEGs for students in the dataset, then comparing this average to national benchmarks (Alps, 2019). The crucial oversimplification here is that a student's MEG is based on all their GCSEs and is the same regardless of which courses they take at college. Analysis of the college's Computer Science results from 2014-2018 shows that the Product Moment Correlation Coefficient between average GCSE grade and actual grade is 0.55. This score would generally be accepted as a moderate correlation for human data, but certainly not a strong one. It implies that Alps MEGs and, consequently, Alps course grades should not be relied upon wholly for Computer Science. Another big issue with Alps is that, if all institutions were to make equal improvements to their student progress, there would be no change in the Alps grades. For a course to achieve a better Alps grade next year, similar courses at other institutions must make relatively slower improvement. The inherent competition in the system could be discouraging institutions from working collaboratively to improve the country's overall education system.

Kirkpatrick's third level is 'Behaviour'. In the training context, this involves the transfer of learning to the workplace (Kirkpatrick & Kirkpatrick, 2006). In education, this could be the application of transferable skills to future learning. The SAR talks of the department's "focus on encouraging student independent study skills and ways to develop these" (College, 2018a). Termly assessments offer an indication of the longer-term progress of attitudes to study like this. However, these assessments do not directly measure the intended behaviour. Instead, the department tracks the weekly submission of homework from each student and chases up any non-submission using an agreed procedure. The 2018 SAR reported that teacher implementation of this strategy was not sufficiently consistent but aimed to improve on this in the future.

The fourth stage is 'Results': whether the curriculum has 'worked' in the end, both for students and teachers. Alps and achievement rate measures are nationally standard tools for evaluating the effectiveness of a curriculum. However, what should be of more importance is evaluating how well students are prepared for the next stage after college. The college keep track of university destinations and alumni graduate results (College, 2018b) but there is no apparent evaluation of the success of employability programmes. The department even admits that "we do not have a strong policy on employability" (College, 2018a). Unless the college believe that HE, which only 68.9% of students attend, is the only successful outcome, they could be doing more to evaluate how well they prepare students for work.

Without measuring the employability 'results' of leavers, there is no way to meet Kirkpatrick's fourth level for evaluating the efficacy of this area. However, even if the curriculum is preparing students for work in Computer Science, there may be signs to suggest that the curriculum itself is influencing the technology industry for the worse. Leading technology professionals are under increasing global scrutiny for the ways in which their business decisions could have a negative impact on society. The government is pushing for regulation of products like social media, artificial intelligence, driverless cars and targeted political advertising. Chair of the DCMS committee, Damian Collins, was quoted talking about

> “issues that the major tech companies are well aware of, yet continually fail to address. The guiding principle of the 'move fast and break things' culture seems to be that it is better to apologise than ask permission.”
>
> (Wakefield, 2019)

These issues could be perpetuated by the utilitarian development of the Computer Science (and wider STEM) curriculum. The heavy ‘product’ nature of content and right-or-wrong nature of assessment could be encouraging a black-and-white hidden curriculum of ‘innovate before thinking’. There is mention of the ethical and social consequences of technology in the curriculum content, although it seems like an afterthought rather than a change of “guiding principle” (DfE, 2014). At the undergraduate level, these issues are also drowned out by other content (ACM, 2013; QAA, 2016). Given the government’s recent concerns with the technology industry, it should be making curriculum change at all levels one of its priorities to address this issue.

Conclusions

By placing Computer Science A-level alongside two of the biggest theories of curriculum design, it seems that the ‘planned’ curriculum primarily follows an objectives-based model, with some process-based aspects that struggle to fit inside strict national assessment frameworks. Although it is intended to be part of a spiral curriculum, the current teething problems with provision of the subject mean that over-ambitious incremental objectives cannot be met by all learners.

The evaluation framework at the college is strong, although there needs to be significantly more attention given to self-assessment reporting and quality improvement of the Computer Science 'received' curriculum, especially given its current trend of growth (45% growth in 2018/19 intake). The 2017/18 3D review only looked at Maths and Further Maths A-levels. This was probably because A-level Mathematics was, and still is, by far the largest course in the department. Nonetheless, there is no other framework for documenting evaluation in Computer Science. The department should also decide how to take part in a national drive to increase female numbers, and the department and college should develop a deeper evaluation of the success of their employability duties, perhaps including tracking employment destinations of leavers.

The government has been increasingly noticing ethical and social issues overlooked by technology companies and Ofsted has been building its new inspection framework for 2019 (Ofsted, 2019). Since "inspection procedures control and determine the nature of the curriculum offered" (Kelly, 2004, p.128), it will be interesting to see where the government and its regulatory powers take the Computer Science curriculum in the coming years.

References

Alps. (2019). *How Alps works | Easy to use data analysis tool for pupil tracking*. Retrieved February 18, 2019, from https://alps.education/how-alps-works/

Assessment and Qualifications Alliance. (2019). *AS and A-level Computer Science Specification: Specifications for first teaching in 2015*. Retrieved February 11, 2019, from https://filestore.aqa.org.uk/resources/computing/specifications/AQA-7516-7517-SP-2015.PDF

Association for Computing Machinery. (2013). *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. Retrieved February 21, 2019, from https://www.acm.org/binaries/content/assets/education/cs2013_web_final.pdf

Bjork, E. L., & Bjork, R. A. (2011). 'Making things hard on yourself, but in a good way: Creating desirable difficulties to enhance learning'. *Psychology and the real world: Essays illustrating fundamental contributions to society*, 2(1), 59-68.

Bloom, B. S. (1956). *Taxonomy of Educational Objectives, Handbook I: The Cognitive Domain*. New York: David McKay Co Inc.

The College. (2018a). *Departmental Self-assessment report: Maths & Computer Science*.

The College. (2018b). *Whole college Self-assessment report*.

Computing at School. (2012). *Computer Science: A curriculum for schools*. Retrieved February 15, 2019, from https://www.computingatschool.org.uk/data/uploads/ComputingCurric.pdf

Department for Education. (2013). *National curriculum in England: computing programmes of study*.

Department for Education. (2014). *GCE AS and A level subject content for computer science*.

Department for Education. (2015). *Computer Science: GCSE Subject Content*.

Joint Council for Qualifications. (2018). *GCE A Level & GCE AS Level Results Summer 2018*. Retrieved February 17, 2019, from https://www.jcq.org.uk/examination-results/a-levels/2018/main-results-tables/a-level-and-as-results-summer-2018

Kelly, A. V. (2004). *The Curriculum: Theory and practice*. Sage.

Kirkpatrick, D. L., & Kirkpatrick, J. D. (2006). *Evaluating Training Programs: The Four Levels Third Edition*. San Francisco: Berret-Koehler Publishers Inc.

McKimm, J. (2007). *Curriculum design and development*. London: School of Medicine, Imperial College Centre for Educational Development. Retrieved February 17, 2019, from https://faculty.londondeanery.ac.uk/e-learning/setting-learning-objectives/Curriculum_design_and_development.pdf

Naugle, K. A., Naugle, L. B., & Naugle, R. J. (2000). 'Kirkpatrick's evaluation model as a means of evaluating teacher performance'. *Education*, 121(1), 135-144. Retrieved February 20, 2019, from https://link.galegroup.com/apps/doc/A66960815/AONE?u=googlescholar&sid=AONE&xid=90196ac9

Office for Standards in Education. (2019). *Draft Education Inspection Framework for 2019*.

Office of Qualifications and Examinations Regulation. (2014). *GCE Subject Level Guidance for Computer Science*.

Quality Assurance Agency for Higher Education. (2016). *Subject Benchmark Statement: Computing*. Retrieved February 21, 2019, from https://www.qaa.ac.uk/docs/qaa/subject-benchmark-statements/sbs-computing-16.pdf?sfvrsn=26e1f781_12

Smith, M. K. (2000). 'Curriculum theory and practice', *The encyclopaedia of informal education*. Retrieved February 4, 2019, from http://infed.org/mobi/curriculum-theory-and-practice/

Stenhouse, L. (1975). *An introduction to curriculum research and development*. Heinemann.

Wakefield, J. (2019). 'Facebook needs regulation as Zuckerberg 'fails' - UK MPs'. *BBC News*. Retrieved February 18, 2019, from https://www.bbc.co.uk/news/technology-47255380

Wing, J. M. (2008). 'Computational thinking and thinking about computing'. *Philosophical transactions of the royal society of London A: mathematical, physical and engineering sciences*, 366(1881), 3717-3725.