Implementing the computing national curriculum in England

Introduction

In September 2014, England introduced a new curriculum subject, computing. This included programming and other elements of computer science in the curriculum for pupils from age 5 onwards, replacing Information and Communication Technology (ICT). Three years on, we can look back at how successful the implementation of the new curriculum has been, and perhaps draw out some lessons for those seeking to implement a similar change.

The English coalition government that introduced the computing curriculum alongside reform of the other subjects also deliberately pulled back from the details of implementation. The British government's stated policy was that 'government should only do what only government can do'.

Oates (2011a) argued that a coherent curriculum should have content arranged in an order associated with age-related progression, and that elements of content, assessment, pedagogy, teacher training, resource materials and incentives should all line up and act in a concerted way. Below, I examine each of these elements in turn, to explore the extent to which England's implementation of its computing curriculum has been *coherent*.

Content

The computing programmes of study were developed through an open consultation led by the British Computer Society and the Royal Academy of Engineering. The curriculum encompasses computer science, information technology and digital literacy, perhaps best thought of as the foundations, applications and implications of digital technology.

The curriculum places computational thinking and creativity at the heart of computing, as the twin, golden threads which run throughout the programmes of study:

> "A high-quality computing education equips pupils to use computational thinking and creativity to understand and change the world." (Department for Education, 2013)

The programmes of study are remarkably concise, running to just two and bit pages to cover computing from age five to sixteen. The Department for Education also developed content requirements for the elective GCSE (Department for Education, 2015) and A Level (Department for Education, 2014) computer science qualifications, and it is these specifications, rather than the programmes of study, which typically drive the content of computer science lessons for 14-18 year olds.

A particular challenge for those teaching computing was the 'all at once' way in which the computing curriculum and new qualifications were put in place: the curriculum for lower secondary pupils assumes that they have studied the primary curriculum, and the GCSE and A Level specifications similarly assume that students have studied the lower secondary and GCSE courses respectively.

Assessment

Very little detail was provided to teachers on how to assess the new national curriculum. The national curriculum commission (Oates, 2011b) were clear in their advice that the old system of levels of attainment should be removed and not replaced. Many schools, however, seemed reluctant to move from a system that had been used to assess progress and attainment for the previous 24 years. Frameworks such as Dorling & Walker (2014)'s 'Progression Pathways' were developed, and initially received positively by teachers. More recently, the McIntosh Commission reiterated the importance of removing and not replacing levels (McIntosh, 2015).

Pragmatically, assessing pupils' capability in computing must use a combination of questions, to assess knowledge, and projects, or perhaps problems, to assess the application of that knowledge (qv Grover, Cooper, & Pea, 2014). The McIntosh Commission recommended the development of item banks to facilitate assessment (McIntosh, 2015), and CAS has made some progress here, through Project Quantum (Oates et al., 2016), which has crowdsourced over 7,500 multiple choice computing questions.

At GCSE, assessment remains problematic. The exam boards included a non-exam assessment (NEA) within the specifications, counting for 20% of the final exam grade. This is a programming project undertaken by pupils individually in response to a detailed problem specification. Regrettably, there appears to have been widespread misconduct, with pupils, and some teachers, sharing the task specification, and even complete solutions in various online fora, in contravention of the exam regulations (Ofqual, 2017). After consultation, it was decided that scores from the non-exam assessment would no longer count towards final grades.

A-level coursework is much more open-ended, with pupils developing a computing project of their own choice, in which their skills can be effectively demonstrated. Take up of this qualification remains very low (in 2017, there were just 7,607 entries), with a worryingly low proportion of entries from female students (under 10% in 2017).

Pedagogy

Compared to longer established subjects, there seems little concrete knowledge about how best to teach computing. English teachers have taken a largely pragmatic line in deciding how best to teach computing, figuring out for themselves what works for their pupils in their schools. This is perhaps unsurprising, given the conclusions of Fossati & Guzdial (2011) that:

> Computer Science instructors rely mostly on intuition and anecdotal evidence to make decisions about changes in their daily teaching practice.

English teachers have made use of social media, CAS hub meetings, and professional development run by CAS appointed 'Master Teachers' to share their classroom practice with one another. Sentance & Csizmadia (2015) present results of their survey into the strategies adopted by some 357 teachers. There seems little consistency in these teachers' free text responses.

The English curriculum places computational thinking at the centre of computing education, and this emphasis perhaps explains some teachers' preference for 'unplugged' pedagogies - 13% of the above sample, despite rather mixed evidence for the efficacy of this approach (Taub, Ben-Ari, & Armoni, 2009). There is some confusion over what is meant by computational thinking, with many seeing it as some generalised approach to problem solving (e.g Csizmadia et al., 2015). However, there is little evidence that computational thinking can be applied effectively to problems outside of those involving computation: perhaps computational thinking without computation is merely thinking. Tedre & Denning (2016) argue that we should be honest about the scope of computational thinking, as 'very powerful mental tools for people who design computations'

It seems mistaken to assume that the computational thinking habits of mind automatically acquired *automatically* through learning programming. Rather, it appears that this must be accompanied by more or less explicit instruction in logical reasoning, algorithms, abstraction and generalisation. In Straw, Bamford, & Styles (2017)'s evaluation of Code Club, whilst there was significant progress in coding skills amongst those who attended for a year, there was no significant difference in improvement on the test for computational thinking.

Teacher training

Excellent computing education requires excellent computing teachers. There remains a shortfall in the numbers of teachers qualified to teach computing in secondary education (Royal Society, 2017) and very few primary school teachers have an academic or professional background in computer science. To address this shortage, more computing teachers can be recruited and trained, or existing teachers can undertake the professional development necessary to be able to teach computing with confidence and competence.

The Teaching Agency developed subject knowledge requirements for entry into computer science teacher training (Teaching Agency, 2012). These encompass the CS knowledge deemed necessary for teaching primary and secondary levels. These have provided some benchmark for teacher training and professional development, as well as setting a standard for entry into secondary initial teacher training courses.

Despite generous, tax-free bursaries and scholarships for those with good honours degrees or postgraduate qualifications who choose to train to become secondary computing teachers, only 66% of the allocated training places have been taken up for the current academic year (Department for Education & National College for Teaching & Leadership, 2017).

Rather than waiting for a new generation of computing teachers to join the profession, efforts have been made to address the professional development of those already in the profession. Most professional development thus far has focussed on the need to equip teachers' with the content knowledge, and perhaps the technological knowledge, needed for teaching computing to their classes; it is only recently that attention has been paid to pedagogical knowledge.

A variety of professional development programmes have been offered. Notable amongst these is Computing At School's Network of Excellence in Computer Science Teaching (see, e.g. Sentence et al. (2012) and Boylan & Willis (2015)). This network, funded by grant from central government, includes a number of lead schools, more than 400 'master teachers' supporting their colleagues locally, and ten, university-based regional centres. In primary education, the Barefoot Computing project has reached more than 45,000 teachers through workshops, conceptual guides and exemplar lesson plans (British Telecom & Accenture Strategy, 2017).

The Royal Society (2017) called for a massive increase in professional development for computing teachers, focussing particularly on the need to retrain those teaching computing to GCSE who lack any post A-level qualification in computing. The government's 2017 budget included £84m of public funding to establish a National Centre of Computing Education, and at least 40 hours of professional development for presently unqualified secondary computing teachers.

Resource materials

Oates (2014) made a persuasive case for the importance of text books and other teaching resources in embodying learning theory, providing a clear delineation of content, ensuring coherent progression, offering stimulation and support for reflection and encouraging 'expansive application'. However, government has, perhaps out of fear for being seen as anti-competitive, not led, or funded, the development of text books for computing, but has rather left this to the publishing industry, not-for-profit organisations, and individual teachers.

The British Educational Suppliers' Association and the Publishers' Association jointly developed guidelines for computing textbook publishers (BESA & The Publishers Association, 2015), but there is little evidence that those developing materials have been influenced by these.

A range of commercial materials have been produced, with some translated into foreign editions for countries following England's lead. Alongside these commercial offerings high quality, free and liberally licensed materials are available from, amongst others, the BBC, Barefoot Computing, the Raspberry Pi Foundation and Code Club. Computing At School provides an online resource sharing portal, by members, for members (and others): at the time of writing this hosts some 4,417 teaching resources.

There is interest in 'physical computing' amongst English computing teachers, even though this is not required by the curriculum or qualifications. The BBC's micro:bit platform provides a readily accessible introduction to programming a simple micro-controller using block- or text-based languages, and the Raspberry Pi has become the default platform for any developers wishing to use programming for control and monitoring.

Incentives

Whilst a top down approach to curriculum change can control content and assessment, and, with funding, ensure suitable training and resources are in place, the actual implementation

of any such change is in the hands of head teachers and teachers. Whilst data on take-up is hard to come by - as yet no reliable evaluation of the curriculum change has been undertaken at scale - anecdotal evidence suggests most primary schools are now teaching computing, and the majority of state funded schools now enter at least some students for the qualification at GCSE, albeit for relatively small groups of students on the whole.

There has been a broadly consistent message on the need to prepare students for life and work in a world where digital technology will play an important role. Computer science has been recognised as a foundational discipline, alongside physics and music, as something *all* students should learn. This moral argument is a persuasive one for some school leaders and many parents.

Whilst all schools funded through local authorities are legally required to teach the national curriculum, privately funded independent schools, and academies and free schools funded directly by central government are not. The majority of secondary schools, and a significant minority of primary schools, are now directly funded academies, and thus exempt from the requirement to teach computing. Despite this freedom, many have chosen to teach computing, at least thus far (Kemp, Wong, & Berry, 2016).The head of England's education inspectorate, Ofsted, indicated that there will be greater focus on the breadth and balance of schools' curricula in future inspections (Spielman, 2017), which may see wider implementation of computing in academies.

Schools are incentivised by published league tables of pupils' performance in examinations. Computer Science at GCSE can be included alongside the older sciences in the crucial Achievement 8 and Performance 8 accountability measures, and in the 'English baccalaureate' (EBacc) set of academic subjects (BCS, 2012). Analysis suggests that Computer Science at GCSE is harder than most other subjects (Thomson, 2016), and this may disincline school leaders from entering all but the brightest and best of their students for the subject. It is hoped that once the computing curriculum between ages 5 and 14 is fully bedded down, GCSE performance in computer science will be comparable to other EBacc subjects.

The Russell Group of leading universities recognises that A level computer science is a useful qualification for a number of science, engineering, medical and social science disciplines (Russell Group, 2016). However, computer science is not listed as a 'facilitating subject' - those which are regarded as opening doors 'to more degrees and more professions than others'. This may be because few computer science degrees currently require the A level as an entry requirement; indeed Cambridge University's Computer Laboratory actually appears to discourage applicants from offering this (Computer Laboratory, 2015).

Concluding remarks

It is hard not to admire the courage of ministers and others who took the decision to make computer science part of the English national curriculum from age five to sixteen. Despite the relative haste with which the curriculum content was developed, and the reluctance of government to lead its implementation, schools, teachers and pupils have responded

positively to this challenge, and the comparison with what pupils learnt in the former ICT curriculum is very favourable. The Royal Society (2017) describe the current state of computing education as 'patchy and fragile'. They argue that

> future development and sustainability depend on swift and coordinated action by governments, industry, and non-profit organisations. Neglecting the opportunities to act would risk damaging both the education of future generations and our economic prosperity as a nation.

For other jurisdictions considering or about to embark on a similar change, a more coherent, joined-up approach to the *implementation* of a computer science curriculum, considering teaching, assessment, resources, training and incentives alongside the design of curriculum content, would result in a more widely adopted, and more robust computer science education for all.

References

BCS: The case for Computer Science as an option in the English Baccalaureate. Swindon: BCS, 2012.

BESA, The Publishers Association: Guidance for the Publishing of Computing Teaching Resources. London: BESA / The Publishers Association, 2015.

Boylan M., Willis B.: Independent study of computing at School Master Teacher programme. Sheffield: Sheffield Hallam University, 2015.

British Telecom, Accenture Strategy: Tech know-how. The new way to get ahead for the next generation. London: BT plc, 2017.

Computer Laboratory: Frequently Asked Questions. Cambridge: Computer Laboratory, 2015. https://www.cl.cam.ac.uk/admissions/undergraduate/faq/ [last checked: 06.02.2018]

Csizmadia A., Curzon P., Humphreys S., Ng T., Selby C., Woollard J.: Computational thinking: A guide for teachers. Cambridge: Computing At School, 2015.

Department for Education: The national curriculum in England. Lodnon: DfE; DfE, 2013.

Department for Education: GCE AS and A level subject content for computer science. London: DfE, 2014.

Department for Education: Computer science GCSE subject content. London: DfE, 2015.

Department for Education, National College for Teaching & Leadership: Initial teacher training census for the academic year 2015 to 2016, England. London: DfE, 2017.

Dorling M., Walker M.: Computing Progression Pathways. Cambridge: Computing At School, 2014.

Fossati D., Guzdial M.: The Use of Evidence in the Change Making Process of Computer Science Educators. Special Interest Group on Computer Science Education (SIGCSE) Conference 2011: 685–690.

Grover S., Cooper S., Pea R.: Assessing computational learning in K-12. Proceedings of the 2014 conference on Innovation & technology in computer science education - ITiCSE '14 2014: 57–62.

Kemp P.E., Wong B., Berry M.G.: The Roehampton Annual Computing Education Report. London: University of Roehampton, 2016.

McIntosh J.: Final report of the Commission on Assessment without Levels. London: DfE, 2015.

Oates T.: Could do better: Using international comparisons to refine the National Curriculum in England. Cambridge: Cambridge Assessment, 2011A.

Oates T.: The Framework for the National Curriculum A report the Expert Panel for the National Curriculum review. Deparment for Education 2011B: 1–77.

Oates T.: Why textbooks count. Cambridge Assessment 2014: 1–23.

Oates T., Coe R., Jones S.P., Scratcherd T., Woodhead S.: Quantum: tests worth teaching to. Cambridge: Computing At School, 2016.

Ofqual: Consultation on revised assessment arrangements for GCSE computer science. Coventry: Ofqual, 2017.

Royal Society, The: After the reboot : computing education in UK schools. London: The Royal Society, 2017.

Russell Group: Informed Choices. London: Russell Group, 2016.

Sentance S., Csizmadia A.: Teachers ' perspectives on successful strategies for teaching Computing in school. IFIP TC3 2015: 1–11.

Sentence S., McNicol A., Dorling M., Crick T.: Grand challenges for the UK: Upskilling teachers to teach computer science within the secondary curriculum. ACM International Conference Proceeding Series 2012: 82–85.

Spielman A.: Recent primary and secondary curriculum research. London: Ofsted, 2017. https://www.gov.uk/government/speeches/hmcis-commentary-october-2017 [last checked 06.02.18]

Straw S., Bamford S., Styles B.: Randomised Controlled Trial of Code Clubs. Slough: NFER, 2017.

Taub R., Ben-Ari M., Armoni M.: The effect of CS unplugged on middle-school students' views of CS. ACM SIGCSE Bulletin 2009: 41: 99.

Teaching Agency: Subject knowledge requirements for entry into computer science teacher training. London: DfE, 2012.

Tedre M., Denning P.J.: The long quest for computational thinking. Proceedings of the 16th Koli Calling International Conference on Computing Education Research - Koli Calling '16 2016: 120–129.

Thomson D.: Which are the most difficult subjects at GCSE? London: Education Datalab, 2016. http://educationdatalab.org.uk/2016/02/which-are-the-most-difficult-subjects-at-gcse/ [last checked 06.02.18]

Author

Miles Berry
University of Roehampton, London, UK.
m.berry@roehampton.ac.uk