Unplugged Computational Thinking for Fun

Paul Curzon
Queen Mary University of London
Mile End
London E1 4NS. UK.
p.curzon@qmul.ac.uk

Abstract: Computational thinking is a fundamental skill set that is learned by studying Informatics and ICT. We argue that its core ideas can be introduced in an inspiring and integrated way to both teachers and students using fun and contextually rich cs4fn 'Computer Science for Fun' stories combined with 'unplugged' activities including games and magic tricks. We also argue that understanding people is an important part of computational thinking. Computational thinking can be fun for everyone when taught in kinaesthetic ways away from technology.

Keywords: Computational thinking, cs4fn, 'unplugged' computing, kinaesthetic teaching, fun

1 Introduction

Computational thinking was popularised by Wing (2006) as a unique set of key competencies that students learn from studying Computer Science/Informatics. Rather than being a single skill, it is an integrated set of approaches to problem solving including algorithmic thinking, logical thinking, abstraction, generalisation, pattern matching, and evaluation. We argue that understanding people is also an important part of computational thinking based problem solving: computer scientists ultimately solve problems for people and those solutions have to, therefore, work for people.

Whilst computational thinking skills may be gained in a traditional way, just by studying for an Informatics degree, Wing raised the question of how it can be taught explicitly and especially at primary and high school level (Wing 2008). This has become a timely question as a variety of countries are increasing the proportion of computing in their school curriculum. In the UK, for example, from September 2014 a new computing syllabus replaces the past

ICT syllabus that focused on the use of technology (Department for Education, 2013). Computational thinking is a core aspect of the skills students will be expected to learn. Similar initiatives are being introduced in other countries too also with computational thinking playing a central role (e.g., Denmark, (Caspersen, Nowack, 2013), New Zealand (Bell, Andreae, Lambert, 2010).

However, many students even those self-selected at university level struggle to learn to program in depth, so teaching everyone to program well based on a deep understanding of the linked skill set, at pre-university level is a challenging endeavour. If programming is a precursor to learning computational thinking because it is by learning to program that one gains computational thinking skills, then following this approach seems unlikely to be successful as a way to learn those skills except for the best students. It is hard to gain deep computational thinking skills or even understand what they involve, from writing small simple programs alone. Therefore, by following this approach the skills will be learnt in any depth only late in the educational system.

Furthermore, while programming is a core part of computing, the subject is much more than just programming, as is computational thinking. Computing is a naturally interdisciplinary subject that if taught that way can be of interest to a much wider range of students (including girls), not just those interested in programming for its own sake. A variety of projects have explored ways to introduce computing without the focus being just on programming (Bell, Curzon, Cutts, Dagiene, Haberman, 2011). Unplugged approaches (Bell, Fellows, Witten, 1998; Bell, 2000; Curzon, McOwan, Cutts, Bell, 2009) – essentially constructivist, kinaesthetic activities – are one such approach. They have proved an extremely popular way to introduce basic computing concepts and enthuse and motivate young students about computing. Whilst the original focus of unplugged techniques was on primary school students, they have been shown to be a successful way of inspiring students of all ages (Curzon, McOwan, Cutts, Bell, 2009).

Given computational thinking is a fundamental skill set, we are exploring ways to introduce it in a way that is fun and relevant to children of all ages with a wide range of interests and educational backgrounds, not just those initially interested in programming. It also potentially provides an alternative more general focus than programming around which to structure the early teaching of computing. We are therefore interested in how one can introduce the overall idea of computational thinking as a coherent skill set, for younger age groups who still have limited computing experience. We are also concerned with how to inspire students about computational thinking in its own right, so that they want to learn the skills independent of learning to program. We believe unplugged activities, linked to contextually rich storytelling, is potentially a powerful way to achieve this.

2 Discussion

We have been using two interlinked approaches. The first is the idea of telling offbeat 'Computer Science for Fun' stories and the second is the idea of using 'unplugged' activities. We discuss the background of each briefly below, before describing two specific examples. We then outline recent evaluation both of our overall project and of workshops with teachers specifically about computational thinking.

2.1 Telling Offbeat Stories with Unplugged Activities

Our work telling fun, computing stories arose out of the cs4fn project (Curzon, 2007; Curzon, McOwan, 2008; Curzon, Black, Meagher, McOwan, 2009; Myketiak, Curzon, Black, McOwan, Meagher, 2012; Meagher, Curzon, McOwan, Black, Brodie, 2013). It is a public engagement project aiming to inspire school students about interdisciplinary computing. It consists of magazines (Curzon, McOwan, Black, 2005–2014), booklets such as on computer science magic tricks (McOwan, Curzon, 2008; McOwan, Curzon, Black, 2009), women in computing (Black, Curzon, Myketiak, McOwan, 2011) and computational thinking (Curzon, 2014). There is also a linked website (cs4fn, 2005) containing further articles.

Across all these outputs we have focussed on telling playful, contextually rich and offbeat stories about interdisciplinary research. By using research stories we focus on leading edge ideas and technology showing the future of the subject. The playful nature using quirky links along with the narrative story structure helps make them engaging. A focus on interdisciplinary problems widens the potential interest beyond just those already interested in computing. By making stories contextually rich we ensure they lead to deeper understanding and can be open ended with more potential avenues to explore beyond the core problem. It also grounds the stories and linked problems much more in the real world, rather than them having the feel of puzzles. Where possible we also make sure the stories are not reliant on existing knowledge meaning they work across a wide agerange and ability. We do not write 'for children' but in a highly accessible style, even about technical topics, again allowing them to work for a wide audience.

The original focus of cs4fn was on written stories, however this led to a strand of linked, fun interactive talks and shows on topics such as Artificial Intelligence, the Magic of Computing (Curzon, McOwan, 2008; Curzon, McOwan, 2013) and computational thinking. These talks and shows, given to student groups in schools and at science festivals, adopted a variation of the 'unplugged' style of teaching (Bell, et al., 1998; Bell, 2000). Kinaesthetic activities are used to explain computing concepts without technology. An important aspect of the way we have presented the activities is to build them into contextually rich stories in a similar way to the written articles. Initially most of these talks focussed directly on computing concepts with computational thinking ideas implicit. More recently, we are using them to more explicitly introduce computational thinking itself. We are also writing a series of linked booklets that tell the full stories of the talks in a way that makes computational thinking explicit (e.g., Curzon, 2014).

We are now taking this approach a step further, providing resources such as props and activity sheets for teachers to use to teach computational thinking. This is being done through our Teaching London Computing project (2014). It is a computing education project aiming to support teachers in London preparing to switch from the past curriculum that focussed on basic digital literacy and the use of computers to a new computer science based curriculum. Despite the project being London focussed, the resources are being made freely available to others through the Teaching London Computing website. To date we have written up a range of activities and given linked workshops focussed on secondary schools (age 11–18) around four themes:

- Introducing computational thinking
- Algorithmic thinking
- Unplugged programming
- Computational thinking: understanding people

We have also given workshops on how these unplugged activities can be used to introduce computational thinking at primary school level (up to age 11). An evaluation of these workshops is discussed below.

2.2 Sample Stories with Linked Unplugged Activities

To illustrate the approach we describe two of our stories and linked activities: one around helping a person who is totally paralysed to communicate, the other around magic tricks and human error.

As a way of introducing computational thinking in general we have used the story of Jean-Dominique Bauby. He had locked-in syndrome, a condition resulting from a stroke that leaves the person totally paralyzed. Despite this, Bauby wrote an autobiographical book about life with the condition (Bauby, 1997). We explore the problem of how he wrote the book and more generally how computational thinking might lead to people with locked-in syndrome being able to communicate more easily. We overview that story below; a full version is given in Curzon (2014). We use it as a way of introducing a range of computational thinking ideas in a real-world problem solving setting but in the absence of technology.

Bauby could see, hear and think but not talk. The only movement he could make was to blink one eye. He had a human helper to aid writing the book, though had no technological help at all. In this story, we explore with the class ways of communicating with locked-in syndrome including issues such as the need for some kind of agreed code. We describe how Bauby had the helper read out the letters A, B, C,... until Bauby blinked. They would write that letter down and then start again. We get the audience to try this in pairs as an unplugged activity asking them to think about problems and improvements. They usually come up with issues like the need to undo a letter and the need for a way to deal with punctuation and digits, for example. Improvements often suggested include starting with the most common letters (which Bauby actually did), and predicting the word before it is finished (taken from predictive texting). We then discuss how long it would take to write the book and look at best, worst and average case (for the basic case 13 questions per letter) in terms of number of questions asked (i.e., letters spoken).

In the final section we point out that we can do much better – at worst only 5 questions are needed to determine any letter of the alphabet. We note that even though they may not realise it everyone knows the right kind of question. We just need to switch problems to the game of 20 questions to show this (a further unplugged activity). We play a game where the speaker thinks of a famous person and the audience work out who it is by asking yes/no questions. From the start the audience ask questions like "Are they female?" not ones like "Is it Ghandi?". They can also normally say why that kind of question is better. Looking at how efficient this is we see it that always asking such halving questions only takes 20 questions to get from a million possibilities to 1. We can now transfer that solution to letters of the alphabet – the 5 questions used each just halve the remaining portion of the alphabet. The audience normally agree we have come up with a better way to communicate with locked-in syndrome. We finish however with a twist, pointing out that Bauby may have found blinking hard and if so we have made it 5 times harder as he must blink 5 times per letter with our solution rather than once with the algorithm he used. It is important to understand the problem from the perspective of the people involved before coming up with solutions.

This story introduces various aspects of computational thinking that we point out as we tell the story. It involves *algorithmic thinking* in coming up with an algorithm to communicate but with a clear focus on solutions that work for people, *evaluation* about the functionality, efficiency and usability of the solution, *pattern matching* and *generalisation* in translating solutions for other problems, *abstraction* for example in thinking about work to communicate one letter, rather than time to communicate the whole book, and so on. All the components of computational thinking are integrated, being used together to solve the problem – a problem that does not involve technology and that is contextually very rich.

A second group of stories we tell are based around magic tricks as the unplugged activity. The core idea is that magic tricks are essentially algorithms, though followed by a magician, rather than a computer. A trick is a series of steps that must be followed precisely and in the right order if the trick is to work. However the link is deeper than this. For a trick to work it needs more than the algorithm. It needs a good presentation based on an understanding of cognitive psychology. Similarly a program combines an algorithm with interaction design based on a similar understanding. Magic tricks can thus be used to teach the importance of taking people into account in computational thinking problem solving. They can also be used in teaching human-computer interaction topics (Myketiak, Curzon, McOwan and Black, 2012). A trick can be used as a demonstration before a discussion or the activity taken further. For example, the class might be set the task of writing down crib sheets for them to do the trick (i.e., write down the algorithm), or of creating new variations such as their own presentation based on the core algorithm of a trick demonstrated.

We have developed one such trick into a story about design to avoid human error in collaboration with CHI+MED (2009). It is a research project on how to design safer medical devices that help clinicians avoid making mistakes in their use. Magicians show how one can engineer a system – a magic trick – so that everyone makes the same mistake at the same time. Software engineers have to engineer systems in a way that ensures no one makes mistakes. One example we use is to present a trick called 'The Four Aces' which uses simple misdirection. In the trick, a set of Aces mysteriously jump from one pile to another without anyone noticing. We point out how this shows that we do not see everything that is in our field of view especially if our attention is drawn elsewhere.

We then consider an interface to a medical device such as an infusion pump which is set up by a nurse to deliver a given dose of a drug (say 15.5mg/hour over 2 hours). If the nurse's attention is drawn away from the screen when setting it up, for example because the start button is not close to the screen, he may not notice that the decimal point did not register, say, especially in a stressful and busy hospital ward. That could lead to the patient being given a massive overdose. Magicians use misdirection to pull our attention to the wrong place. A good interface designer will use similar tricks to make sure the attention is drawn to the right place – the screen of the medical device in this example.

This story also introduces a variety of aspects of computational thinking that we point out as we tell the story. It involves *algorithmic thinking* in creating the algorithms behind magic tricks, again with a focus on solutions that work for people. It also involves *evaluation* in checking that a trick really works in practice. The link between magic and programs is another example of *pattern matching* and *generalization* and this can be made explicit with tricks that are based directly on computer algorithms: search algorithms and error-correcting codes, for example. Students can be set the task of writing their own solutions and so have to decide on a suitable level of *abstraction* for the description. *Abstraction* can also be introduced in exploring arguments about whether a trick always works using an appropriately abstracted model. The separate aspects of computational thinking come together in an integrated way within the story, which is again contextually very rich and so open-ended.

2.3 Evaluation

The cs4fn approach of telling such rich, offbeat research stories has been extremely popular, in general. For example, demand for physical copies of the magazine is strong and has increased steadily. Over 1700 schools across the UK subscribe to copies, in many cases receiving class sets of 30–200 copies to distribute. The total number of magazines sent to subscribers amounts to around 18.000 copies. They are used in various ways according to the local need: with 'gifted and talented' groups, in normal classes and with 'problem' classes, to use directly in class and for students to read in their own time, and as a different kind of reading material placed in literacy boxes.

We have in the past sent physical copies to subscribers in over 80 countries (though unfortunately no longer have funding to continue to do this). In a survey of teachers conducted in 2012, 98 % rated the cs4fn magazine as either "excellent" or "good". 77 % agreed that they could use articles or ideas from cs4fn in their lessons.

The website has been similarly successful. Between 2008 and 2013 it has received over a million visits and PDFs of our magazines and other resources have been downloaded over 890.000 times. Web users have been positive in surveys with over two thirds of respondents saying cs4fn helped them see more ways computer science is used in the real world. After visiting the cs4fn website, they reported thinking of computer science as more interesting and thinking of a variety of careers that would use computer science.

In terms of live performances, we have given school shows on computing topics to nearly 20.000 school students in schools, reaching over 10.000 more at science festivals. Feedback from teachers has been highly positive about these shows: for example, in follow-up surveys 100 % of those surveyed said that they met their needs and that they would recommend them to others. The surveys also suggest teachers believe that students are more likely to take computing courses as a result. Individual surveys with students at the end of shows have also always been highly positive.

An external evaluation concluded that the cs4fn project as a whole has had impact in a variety of ways, including conceptual impacts (e.g., a more positive perception of computer science by students), instrumental impacts (e.g., in the form of long lasting resources), capacity building impacts (e.g., in teachers picking up both the content and style of teaching), attitude/cultural change impacts (e.g., at our home institution in elevating the importance of public engagement) and enduring connectivity impacts (e.g., through ongoing use of resources by a wide range of groups of people). A detailed description of the evaluation is given in Meagher, Curzon, McOwan, Black, Brodie (2013b) with summary in Meagher, Curzon, McOwan, Black, Brodie (2013a). The project has shown that the approach of using accessible writing in offbeat ways about interdisciplinary computing is a highly popular approach that does appear to inspire students and teachers. It remains further work to explicitly evaluate the magazines, booklets and shows in terms of their success specifically in introducing computational thinking ideas, however.

For our 'Teaching London Computing' workshops supporting teachers we have conducted an evaluation directly related to computational thinking. We gave out a short postevent feedback survey to teachers attending. Survey questions were on a 5-point Likert scale with a ranking ranging from 'strongly disagree' (1) to 'strongly agree' (5). In total 117 completed the survey. The feedback was extremely positive. Across all five workshops answers to the question: “The workshop was Useful” gave average response 4.57 (n=116). Similarly a question as to whether “The workshop was Confidence Building” gave average response 4.41 (n=113).

The surveys for the first and last workshop about introducing computational thinking and how understanding people mattered to computational thinking included explicit questions about computational thinking itself. The responses show that the workshops certainly helped participants understand what computational thinking was and give them useful ideas about how to teach it. Average responses (on the same 5-point Likert scale) were:

- “As a result of the workshop I now have *a better understanding of* computational thinking”: **4.28 (n=72)**
- “As a result of the workshop I now have *a better understanding of* computational thinking, about people” **4.63 (n=24)**
- “As a result of the workshop I have *new ideas about how to teach* computational thinking” **4.25 (n=68)**
- “As a result of the workshop I have *new ideas about how to teach* computational thinking, about people” **4.75 (n=24)**

Thus there is evidence that this approach of unplugged activities embedded in contextually rich stories is an effective way to explicitly introduce computational thinking ideas at least to teachers themselves. More detailed analysis of this data is in progress, however. It also remains further work to evaluate whether the approach leads to better understanding of students taught by those teachers.

3 Conclusion

Kinaesthetic “unplugged” activities embedded in contextually rich stories provides a potentially powerful way to introduce the ideas and skills of computational thinking in an integrated way that is accessible to all. The focus on understanding people as part of computational thinking helps add to that richness and power of the stories. We have argued that this approach provides a fun way to engage a wide range of students with computational thinking, not just those directly interested in coding. Evidence to date suggests the approach certainly does help teachers themselves understand the concepts and gives them confidence and ideas of how to teach it.

In future work we hope to investigate the approach’s effectiveness for teaching students as part of the curriculum. Selby, Dorling and Woollard (2013) are developing a framework for teaching and assessing computational thinking, within the UK curriculum. This is based around a set of core computational thinking concepts including abstraction, decomposition, algorithmic design, evaluation, and generalisation (Selby, Woollard, 2013). We intend to also integrate our activities and stories with that framework.

4 Acknowledgements

‘Teaching London Computing’ was created by Paul Curzon and William Marsh and is supported by the Mayor of London and Department of Education. cs4fn was created by Paul Curzon and Peter McOwan and funded by EPSRC (EP/F032641/1) (2008–2013). Both projects have been supported by Google’s CS4HS programme. The CHI+MED project is funded by EPSRC (EP/G059063/1). Nicola Plant, Jonathan Black, Jo Brodie, Chrystie Myketiak, Laura Meagher and many others have contributed to this work. We have also had helpful conversations about computational thinking with Mark Dorling, Michael Caspersen and Henning Agesen.

References

Bauby, J.-D. (1997): *The Diving-Bell and the Buttery*. Fourth Estate.

Bell, T., Fellows, M., Witten, I. (1998): *Computer Science Unplugged: Offline Activities and Games for All Ages*. Available at: www.lulu.com

Bell, T. (2000): A low-cost high-impact computer science show for family audiences. In *Austral-asian Computer Science Conference (ACSC 2000)*, pp 10–16.

Bell, T., Andreae, P., Lambert, L. (2010): Computer Science in New Zealand high schools. In *Proceedings of the Twelfth Australasian Conference on Computing Education* – Volume 103. Brisbane, Australia: Australian Computer Society, Inc.

Bell, T., Curzon, P., Cutts, Q., Dagiene, V., Haberman, B. (2011): Introducing students to computer science with programmes that don't emphasise programming. In *Proceedings of ITiCSE 2011, The 16th Annual Conference on Innovation and Technology in Computer Science Education ACM SIGCSE*, p. 391, June 2011. Darmstadt, Germany.

Black, J., Curzon, P., Myketiak, C., McOwan, P. W. (2011): A study in engaging female students in computer science using role models. In *Proceedings of ITiCSE 2011, The 16th Annual Conference on Innovation and Technology in Computer Science Education ACM SIGCSE*, pp. 63–67, June 2011. Darmstadt, Germany.

Caspersen, M. E., Nowack, P. (2013): Computational Thinking and Practice – A Generic Approach to Computing in Danish High School, In A. Carbone, J. Whalley (Eds.), *15th Australasian Computer Education Conference (ACE 2013)*, Adelaide, South Australia,. Conferences in Research and Practice in Information Technology (CRPIT), Vol. 136. pp. 137–143.

CHI+MED (2009), CHI+MED: *Making Medical Devices Safer*, Retrieved from http://www.chi-med.ac.uk/cs4fn (2005), Computer Science for Fun, Retrieved from http://www.cs4fn.org/

Curzon, P. (2007): Serious fun in computer science. In *Proceedings of the 12th Annual Conference on Innovation and Technology in Computer Science Education ACM SIGCSE* (Invited Keynote), p. 1. ACM. Dundee, Scotland.

Curzon, P. (2014): *Computational Thinking Searching to Speak*. Queen Mary, University of London. Available from http://teachinglondoncomputing.org/resources/inspiring-computing-stories/

Curzon, P., McOwan, P. W. (2008): Engaging with computer science through magic shows. In *Proceedings of ITiCSE 2008, The 13th Annual Conference on Innovation and Technology in Computer Science Education ACM SIGCSE*, pp. 179–183. ACM. Madrid, Spain.

Curzon, P., Black, J., Meagher, L. R., McOwan P. W. (2009): cs4fn.org: Enthusing students about computer science. In T. O. C. Hermann, T. Lauer, M. Welte Eds), *Proceedings of Informatics Education Europe IV*, pp. 73–80. Freiburg, Germany.

Curzon P., McOwan P. W. (2013): Teaching formal methods using magic tricks. In *Fun with Formal Methods: Workshop at the 25th International Conference on Computer Aided Verification*, July.

Curzon, P., McOwan, P. W., Black, J. (2005–2014) cs4fn: *Computer Science for Fun Magazine*. Queen Mary University of London.

Curzon, P., McOwan, P. W., Cutts, Q., Bell, T. (2009): Enthusing and inspiring with reusable kinaesthetic activities. In *Proceedings of ITiCSE 2009, The 14th Annual Conference on Innovation and Technology in Computer Science Education ACM SIGCSE*, pp. 94–98. ACM, July. Paris, France.

Department for Education (2013): *The National Curriculum in England*, Framework Document. Available from https://www.gov.uk/government/uploads/system/uploads/attachment_data/file/210969/NC_framework_document_-_FINAL.pdf

McOwan, P. W. &. Curzon, P. (2008): *The Magic of Computer Science*. Queen Mary, University of London. Available from http://www.cs4fn.org/magic/magicdownload.php

McOwan, P. W., Curzon, P., Black, J. (2009): *The Magic of Computer Science II: Now we have your attention*. Queen Mary, University of London. Available from http://www.cs4fn.org/magic/magicdownload.php

Meagher, L. R., Curzon, P., McOwan, P. W., Black, J., Brodie, J., (2013a): *cs4fn Final Evaluation Executive Summary*. Queen Mary University of London. Available from http://www.cs4fn.org/project/evaluation/

Meagher, L. R., Curzon, P., McOwan, P. W., Black, J., Brodie, J., (2013b): *cs4fn Final Evaluation Report*. Queen Mary University of London. Available from http://www.cs4fn.org/project/evaluation/

Myketiak, C., Curzon, P., Black, J., McOwan, P. W., Meagher L. R. (2012): cs4fn: A flexible model for computer science outreach. In *Proceedings of the 17th ACM Conference on Innovation and Technology in Computer Science Education (ITiCSE)*, pp. 297–302.

Myketiak, C., Curzon, P., McOwan, P. W., Black, J. (2012): Teaching HCI through magic. In *The Contextualised Curriculum: A CHI 2012 workshop*, May.

Selby, C., Woollard, J. (2013): *Computational Thinking: The Developing Definition* Available: http://eprints.soton.ac.uk/356481/.

Selby, C., Dorling, M., Woollard, J., (2014): *Evidence of Assessing Computational Thinking*, Submitted for publication.

Teaching London Computing (2014): *Teaching London Computing*, Retrieved from http://www.teachinglondoncomputing.org.

Wing, J. M. (2006): Computational thinking. *Communications of the ACM*, 49, pp. 33–35.

Wing, J. M. (2008): Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 366(1881), pp. 3717–3725.

Biography

Paul Curzon is a Professor of Computer Science at Queen Mary University of London, interested in inspirational ways of teaching computing/computational thinking. He runs cs4fn enthusing students about computer science worldwide and ‘Teaching London Computing’ developing playful resources for computing teachers. He was made a UK National Teaching Fellow in 2010 in recognition of his excellence in teaching.