Adoption of Computer Science in NZ schools

Tim Bell, Heidi Newton, Caitlin Duncan, Sam Jarman
University of Canterbury
tim.bell@canterbury.ac.nz, heidi@csfieldguide.org.nz, caitlin.duncan@pg.canterbury.ac.nz, saj77@uclive.ac.nz

ABSTRACT

In 2011, Programming and Computer Science standards were made available as part of NCEA in New Zealand high schools. Because little guidance and professional development was available initially, teachers have found it challenging to present the content effectively to their students. In response to this, several resources and professional development opportunities have been made available, including the widely used Computer Science Field Guide for Computer Science, and several programming resources specific to the new standards. In this paper we outline the deployment of the new standards and supporting material, and look at the uptake of the new standards over the first three years that they were phased in. This reveals increasing participation at schools, and higher enrolments at university as a flow-on effect. The introduction of Computer Science has also helped to address perception and stereotypes about the industry, with high achievement by female students, although participation rates are not ideal.

Keywords: computer science education, schools, NCEA

1. INTRODUCTION

Much of the problem with the shortage of students taking up computing as a career can be traced back to a lack of awareness of the possibilities, and not having the opportunity to discover their passion for it (Margolis & Fisher, 2003). Several countries, including New Zealand, are addressing the shortage of suitable staff in the computing industry by introducing more rigorous computing curricula in schools. New Zealand introduced new achievement standards to NCEA starting in 2011, with the last set of standards being phased in for 2013 (Bell *et al.*, 2014a).

The new computing standards offered in New Zealand are available for the students sitting NCEA, typically in the last three years of high school. The standards are grouped together under "Digital Technologies", which is currently a part of the technology learning area. Because the phasing in of standards was completed only in 2013, the system is still in an "early adopter" phase, although much information has been gleaned through the process, and already thousands of students have been exposed to programming and computer science through these standards.

Having computing in schools raises many issues that we will consider in this paper. A key issue that has emerged during adoption is teacher preparedness – there are hundreds of digital technologies teachers in New Zealand who have been given a lot of new material to teach, and they have had very little time to adapt (Thompson & Bell, 2013). There is also the question of what topics should be taught at school, and especially how early students should have the opportunity to learn programming. In this paper we review the effect of introducing the new standards now that they have been completely phased in, examining the initial uptake, and considering the impact on teachers and schools. We also reflect on future developments that could be considered, contrasting the situation in New Zealand with new curricula being adopted in the UK (in September 2014) and Australia (in 2015), and discuss how teachers can be supported through this time of extensive changes.

This paper appeared at ITX 2014, incorporating the 5th annual conference of Computing and Information Technology Research and Education New Zealand (CITRENZ2014) and the 27th Annual Conference of the National Advisory Committee on Computing Qualifications and CSANZ2014, Auckland, New Zealand, October 8-10, 2014. Mike Lopez and Michael Verhaart, (Eds).

2. THE NEED FOR COMPUTER SCIENCE IN SCHOOLS

A key element in the 2011 changes has been to move from a focus on learning to *use* digital systems, to including more on being able to *develop* them, particularly using tools and techniques from computer programming and computer science. Ideally students will be exposed to many of the ideas that are used in industry, and through this discover if they have a passion for the subject. Already there is much anecdotal evidence of students switching to a career in computer science or software engineering after being exposed to it in school through the new standards; previously it was rarely mentioned, and terms like "computer science" may even have meant something different in a school.

A secondary purpose of the new material is to provide students with background skills as preparation for further study. This will include having them gain an appreciation that careers in computing involve skills beyond programming, including communication skills and maths. Through this they can be encouraged to keep up the supporting subjects rather than letting them slip because of the common assumption that people in computing only work alone on a computer writing programs, and don't need to communicate with others or solve complex problems. There is an element of increasing the skill level of school leavers, primarily in programming, as the level 3 (year 13) programming standard requires students to program in an OO language and develop applications with a GUI. To put this in perspective, the coverage of programming done by a student who has completed the level 1 to 3 programming standards is roughly equivalent to one course out of the eight taken in a typical first-year university degree, that is, about 4% of a degree – it's a useful headstart, but the most significant change is in the improved awareness of students arriving for further study rather than significantly reducing the time taken to get qualifications. Nevertheless, universities are responding to the new skills by providing more challenging options for students who have the extra preparation. Because NCEA standards are optional, and many schools don't offer the new standards, pathways must still be maintained for students who haven't taken the new courses at school. However, as these become more widespread, the accelerated pathway is likely to become the norm, and the existing pathways would become remedial.

Many approaches to introducing computing to students focus on programming. While this is a key activity of software developers, if it is taught as an end in itself then it can filter out the people who are more interested in using programming to achieve a more interesting goal. For this reason the New Zealand standards in computer science touch on much of the bigger picture, including topics such as artificial intelligence, computer vision, working with large amounts of data, designing usable interfaces, and other outcomes that can be accomplished if the student learns to program. This is particularly important for attracting more female students (Margolis & Fisher, 2003).

Another goal of many computing curricula around the world is to introduce students to *computational thinking* (CT)[1] an idea first suggested by Papert (1996), and popularised by Wing (2006). CT is a generalisation of programming and many related disciplines, where students work with problems and draw on computational ideas to develop solutions that are based around a process. The solution need not be a computer program at all, but could be something that relies on approaches such as decomposition, abstraction, or an implicit algorithm. Examples could include working out the most efficient way to get passengers seated on a plane, or how someone with locked-in syndrome (unable to communicate except by blinking) would be able to spell out sentences, or even write a book (Curzon, 2013). Learning to program helps with general Computational Thinking skills, and vice versa, which provides a motivation for teaching programming to all school students that goes well beyond preparing them for a career in the software industry.

3. ENGAGEMENT IN COMPUTER SCIENCE BY NZ STUDENTS

The new standards in Digital Technologies in NCEA have 5 "strands": Digital Information, Digital Infrastructure, Digital media, Electronics, and Programming and Computer Science. These are offered over three levels (1, 2 and 3) that generally correspond to the last three years of school (years 11, 12 and 13) respectively. At each level there are typically about three standards for each strand, and a course in a school would be made by putting together several standards to suit the class. For example, at level 1 in Computer Science and Programming there are three standards:

- AS91074 (1.44): Demonstrate understanding of basic concepts from computer science
- AS91075 (1.45): Construct a plan for a basic computer program for a specified task
- AS91076 (1.46): Construct a basic computer program for a specified task

All three of these standards combined would be enough material for about half of a full-year course, so a whole course would be made up by combining them with related standards, such as web design, electronics, or "generic" technology standards that could cover general principles such as design, or the effect of technology on society.

The AS91074 code gives the NZQA registered number for the standard. Each has the short-hand alternative number within the technology standards; for example, in "1.44", the 1 indicates that it is level 1, and the 44 indicates that it is Computer Science. There are corresponding level 2 and 3 standards (2.44 and 3.44) that extend the coverage of Computer Science. The full matrix of technology standards is available from the national Ministry of Education website, Te Kete Ipurangi (TKI)[2]. More background on how the standards work, and the details of the Computer Science and Programming standards, is available elsewhere (Bell *et al.,* 2014a).

The decision for a particular standard to be offered in a school is made by the school when putting together courses, and then students can then elect which courses they take. Figure 1 shows the number of Level 1 (Year 11) students who have taken the new standards over the first three years they were offered (2011 to 2013). Each strand has multiple standards available at Level 1, so there are multiple lines for each strand. Figure 1 shows some increase in the number of students involved in the Programming and Computer Science strand over the three years, but some other areas have decreased, particularly the Digital Information strand. Digital Information is closest to some of the traditional material on learning to use standard productivity tools, but the new standards go into more depth. There have been problems with low pass rates, and this may have deterred some schools from offering it in subsequent years.

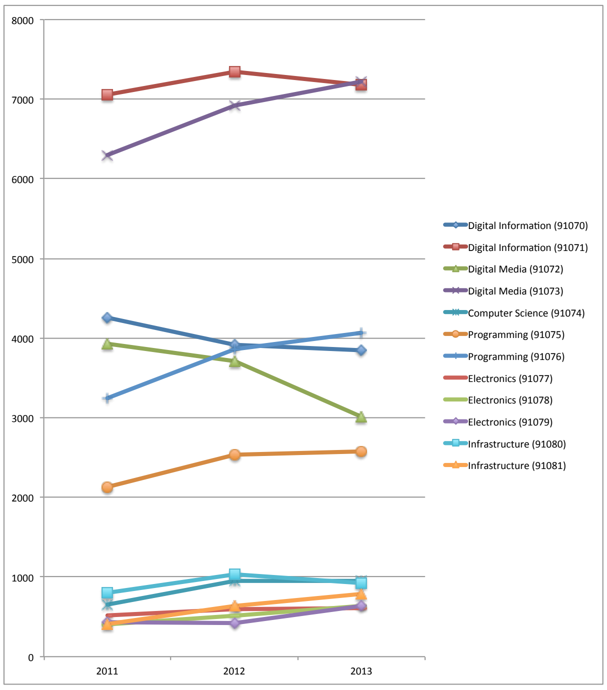


Figure 1: Number of Year 11 NZ students taking each strand of the new NCEA standards in the first three years of their introduction

Table 1 shows the number of students passing the main computer science (x.44) and programming (x.46) standards over the first three years they were offered. At level 1 (year 11) and level 2 (year 12) the numbers are increasing each year as more schools offer the new standards. This reflects an increase in the number of schools offering the standards, which in turn is likely to be influenced by the availability of resources to support them. For the computer science standards the "Computer Science Field Guide" was released in 2012 (Bell *et al.,* 2014b) and new tools specifically designed for teaching programming have become available during this time, including the CodeAvengers[3] JavaScript online tutor system, and the "Programming and Problem Solving" text for

[1] http://www.google.com/edu/computational-thinking/index.html

[2] http://ncea.tki.org.nz/Resources-for-Internally-Assessed-Achievement-Standards/Technology

[3] http://www.codeavengers.com/

Java and Python from the University of Otago[4]. The availability of such tailored resources, along with strong peer support (Thompson & Bell, 2013) are likely to have boosted teacher confidence for delivering these standards.

Table 1: Number of students passing the new computer science and programming standards from 2011 to 2013

Level	Standard	2011	2012	2013
1	CS (1.44)	440	646	694
2	CS (2.44)	-	485	589
3	CS (3.44)	-	-	423
1	Programming (1.46)	2,213	2,795	3,025
2	Programming (2.46)	-	1,008	975
3	Programming (3.46)	-	-	630

For the Computer Science topics (1.44, 2.44 and 3.44), Table 1 shows that the number of students passing level 2 is higher than the number passing level 1 from 2011 to 2012; the increase in the pipeline is possible because there isn't a strict pre-requisite structure in place, and the general increases are likely to be attributable to better teaching resources and professional development available for teachers each year. From year 12 (level 2) students in 2012 to year 13 (level 3) students in 2013 the pipeline decreased by 13%. This shows good continuing interest considering that in those years in NZ there were 55,969 year 12 students and 48,905 year 13 students, which is also a 13% decrease.

Having computer science in high school has the potential to increase the rate of female participation because it provides the opportunity to prevent prejudices that arise from lack of information about what the industry is really like. Figure 2 shows the percentage of female students enrolled in year 11 courses over the first three years the standards were offered.

The proportion of female students participating each year has been decreasing except for the Electronics strand. The *number* of female students has also decreased; for example, the 1.46 (AS91076) standard had 1,164 female participants in 2011, and 1,064 in 2013, even though the total number of students has been increasing. The increase in the proportion in the electronics standards reflects a substantial increase in numbers from an initially small uptake: in 2011 the standards had between 20 and 22 female students in total participating, and in 2013 there were up to 104 students taking the standards.

Table 2 shows the total number of female students attempting the programming and computer science standards.

The total number of students (male and female) passing level 3 programming in 2013 is only 28% of the number of students passing level 1 programming 2 years earlier (see Table 1), but the number of female students attempting level 3 is less than 5% of the number that attempted level 1; that is, female students are retained at a much lower rate than male students. A similar effect can be observed in the computer science standards; while the pipeline doesn't shrink so much overall (in fact, the number of students passing level 3 is 96% of the number passing level 1), the number of female students particpating at level 3 is only 38.5% of level 1. We can only speculate on the reason for these differences, and research is needed to find out what is happening to cause this. One way to address the problem will be to introduce computing earlier in the school curriculum, as it becomes increasingly difficult to influence career paths after around year 8 (12 years old) – this is discussed in a later section.

[4] http://www.cs.otago.ac.nz/year12dt/

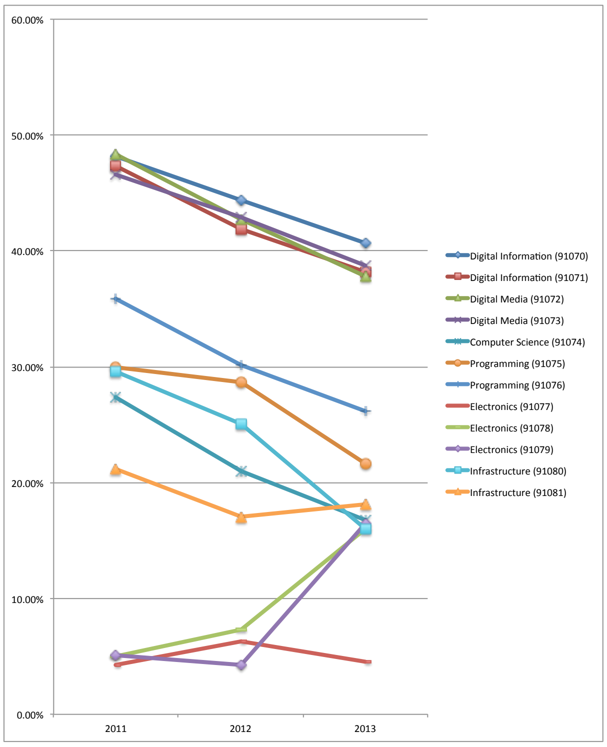
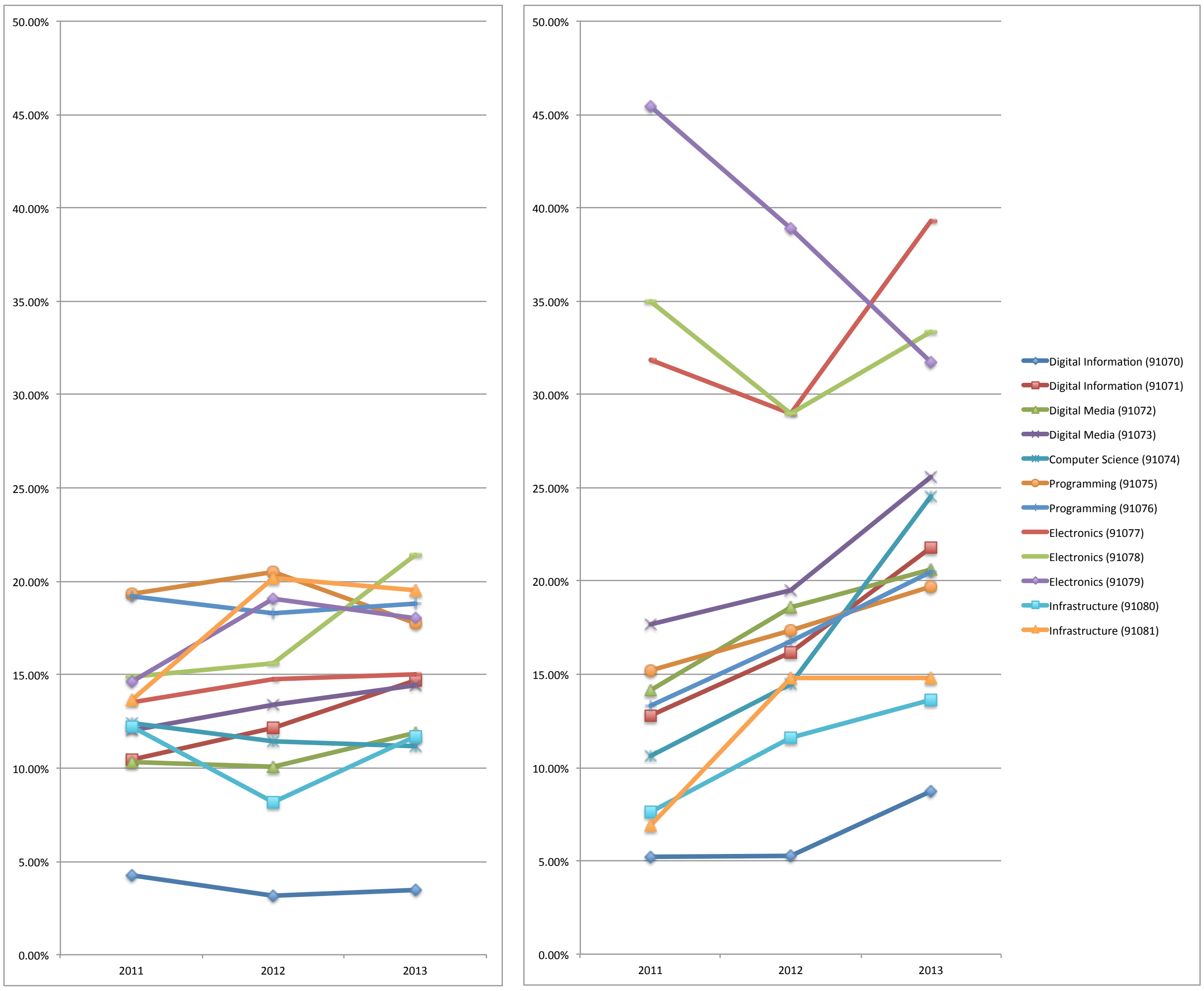


Figure 2: Percentage of female students in each strand of Digital Technologies

Table 2: Number of female students attempting the new computer science (1.44/2.44/3.44) and programming (1.46/2.46/3.46) standards from 2011 to 2013

Level	Standard	2011	2012	2013
1	CS (1.44)	179	200	159
2	CS (2.44)	-	89	139
3	CS (3.44)	-	-	69
1	Programming (1.46)	1,164	1,165	1,064
2	Programming (2.46)	-	274	252
3	Programming (3.46)	-	-	55

In contrast to the decreasing proportion of female students participating, the achievement levels of females has been increasing. Figure 3 shows the percentage of students who received an Excellence result for their work. Typically about 14% of students receive excellence across all NCEA subjects, although this doesn't operate as a quota or "grading to the curve" system, so students taking a particular standard or year might get a higher or lower proportion. In Figure 3 there are a variety of proportions, although in general the proportion of female students attaining Excellence is increasing, while the proportion of male students remains fairly static. The particularly high results for electronics reflects a small but



Male students Female students

Figure 3: Percentage of students getting excellence in Digital Technologies achievement standards

disporportionate number (about 34) of female students who received Excellence for their work.

Table 3 shows the average grades of students separated by gender. The weighted grades use 1 for Not achieved, 2 for achieved, 3 for Merit, and 4 for Excellence. Using this measure, the decreasing number of female students is also paralleled by increasing achievement relative to the male students. In fact, for level 1 computer science, female students are 0.4 points above the male students. Again, there could be multiple reasons for this, but at least the lack of quantity of students is somewhat countered by the quality of students.

The number of schools in which male students achieved the Year 11 Computer Science standard (1.4, AS91074) increased from 25 in 2011 to 39 in 2013, while for females it increased from 16 to 31. While the number of schools is relatively small, the increase has been steady, with the number for females almost doubling. This reflects increasing confidence among teachers as more professional development and resources are available, but also shows that there is a long way to go.

Table 3: Average grade of students (weighted from Not achieved = 1 to Excellence = 4) for male:female students

Level	Standard	2011	2012	2013
1	CS (1.44)	2.1 : 2.1	2.1 : 2.2	2.1 : 2.5
2	CS (2.44)	-	2.0 : 2.2	2.2 : 2.5
3	CS (3.44)	-	-	2.0 : 1.9
1	Programming (1.46)	2.3 : 2.1	2.3 : 2.3	2.3 : 2.4
2	Programming (2.46)	-	2.3 : 2.2	2.3 : 2.3
3	Programming (3.46)	-	-	2.6 : 2.5

The first cohort of students who had access to the new standards arrived at universities at the beginning of 2014. While it is too early to draw conclusions about the impact on student numbers, at Canterbury University a sudden increase of 23% was observed in the number of students taking the key class required to major in Computer Science or Software Engineering (COSC122). Furthermore, the number of female students in that class had fluctuated between 7.5% and 7.9%

in the previous three years, but in 2014 it jumped to 13.0%. Even the latter proportion is very small, but the higher gender balance occurring in schools may help in the long term.

4. FUTURE DIRECTIONS FOR COMPUTING IN SCHOOLS

Up until 2014 New Zealand has been ahead of the English-speaking world by having nationally adopted standards in Computer Science and Programming in high schools. However, in September 2014 the UK is adopting a new computing curriculum that includes a significant amount of programming and computer science from the *very first year* of primary school. Australia will be doing the same at the beginning of 2015, and some states in the US are also adopting broader computing curricula. Although CS and programming are not part of the New Zealand curriculum for pre-NCEA years, some schools in New Zealand have been introducing these topics in years 9 and 10 or earlier. Anectodal evidence suggests this is to encourage students to take these as NCEA subjects, and to better prepare them.

Several non-English-speaking countries are active in the area of pre-tertiary computer science and programming curricula, so general principles can be learned from them – and have indeed been shared (Ragonis *et al*., 2010). But resources and curricula in these countries are based on a different culture, school system and language, so cannot be used directly. Israel has had computer science (including programming) in high school since 1995. South Korea places an emphasis on the use of computers at all school levels and from middle school up introductory computer science is taught as well. Denmark had a process parallel to the New Zealand changes from 2011 to 2013 (Caspersen & Nowack, 2013). Estonia is introducing programming from the first year of primary school using a locally developed system called "ProgeTiiger", and there are indications that Finland is considering doing something similar. More details on what is happening in other countries have been collected in a report prepared to help the UK transition to a new computing curriculum[5].

Having Computer Science and Programming in the primary and intermediate school curriculum raises several issues, but also has potential benefits. Often the focus is on "Computational Thinking", which is a general skill that includes programming and other concepts from computer science (such as "problem decomposition, pattern recognition, pattern generalization to define abstractions or models, algorithm design, and data analysis and visualization"[6].

Although countries teaching computing from year 1 of school are often reported in the media as teaching "programming", all countries introducing programming from year one of primary school are using cut-down systems that match the cognitive level of typical students. Typically these are systems such as "Beebots" (a small robotic device pre-programmed with forward/back/left/right buttons) or "ScratchJnr" (a simplified version of Scratch language that also focuses on "turtle" style motion), which are very physically based, and the "programs" consist mainly of sequence of directional instructions. These systems teach ideas such as sequence, testing and debugging, but don't use more abstract concepts such as variables, and only touch lightly on ideas that require more reasoning, such as iteration and selection. General computer science concepts are often taught using "unplugged" activities; these were developed in New Zealand, currently are more widely used in the new primary school curricula in the UK and Australia (Bell *et al*., 2012).

Exposing students to programming and computer science concepts before the age of 12 seems likely to be more effective in building their confidence and reducing the gender gap in the area. There is little research on this yet, since few school systems offer it at present, but there is some evidence mounting. For example, Riegle-Crumb *et al.* (2011) note that students begin to form career aspirations before entering high school; after puberty they are more influenced by peers, and female students often experience a drop in self-esteem and confidence in their computing ability around the age of 12 years, unless they have had previous experience to boost their confidence (Margolis & Fisher, 2003).

Looking at it from a different angle, Neil Fraser surveyed colleagues at Google, asking them at which age they could first program, and obtained a simple rating of their programming skills[7]. He found that the staff most likely to become good programmers had reported starting programming between grade 3 and 7 (about 8 to 13 years old). This doesn't imply that the reverse is true, but it adds weight to the value of exposure to programming before 12 years old. In natural languages the age of 12 has also proved to be the point at which learning new languages becomes significantly more difficult (Johnson & Newport, 1989).

While positive experiences early in a school career can help shape a student's career plans, negative experiences can also have a lasting impact (Margolis & Fisher, 2003), and so it is important that if computing is to be taught in primary schools then teachers need to be confident and well prepared, otherwise the whole approach can backfire. Introducing such material at primary school will also raise other concerns; parents and educators may worry about overloading the curriculum or replacing important fundamentals with something that they don't see as important.

In July 2014 the New Zealand government released a report titled "A nation of curious minds – He Whenua Hihiri i te Mahara"[8] that recommends reviewing the positioning and content of digital technology within the New Zealand Curriculum, and also providing more professional development support for teachers. This opens the possibility of addressing the concerns with the high school offerings by preparing students prior to high school, which is more likely to influence attitudes and address gender balance issues. Australia and the UK are currently deploying such curricula, so New Zealand will be able to draw on international experience.

At present, while few New Zealand primary schools offer CS and programming education, there is still a large interest in this learning area for younger age groups among parents, often driven by a passion for computing expressed their children. This demand is being partially filled with coding clubs, which have generally been created and run by individuals in different areas of New Zealand rather than a central organisation. However this is changing, as can be seen from the recent creation and rapid expansion of CodeClub Aotearoa[9].

[5] http://www.computingatschool.org.uk/data/uploads/internationalcomparisons-v5.pdf

[6] https://www.google.com/edu/computational-thinking/what-is-ct.html

[7] https://neil.fraser.name/news/2012/07/01/

[8] http://www.msi.govt.nz/update-me/major-projects/science-and-society-project

[9] http://www.codeclub.org.nz/

5. TEACHER SUPPORT

Support for teachers in the transition to teaching computer science is critical. Very few existing Digital Technologies teachers have a computing qualification – in Thompson & Bell (2013), only 11% of DT teachers surveyed had a computer science degree, and a further 10% had a full degree in a related area. As noted earlier, the student experience is heavily influenced by teacher confidence, and the lower participation in Level 3 standards is very likely linked to this.

Soon after Computer Science became a senior subject that New Zealand high schools could teach, it became apparent that teachers desperately needed material developed specifically for the new standards, as no other English-speaking country had a curriculum as comprehensive as ours, and teachers could not work out how to deliver the new topics effectively. The "NZ Computer Science Field Guide" (NZ CSFG, available at http://csfieldguide.org.nz) was developed to address this problem (Bell *et al.,* 2014b). It is an open-source, interactive, online "textbook" that introduces a wide range of topics in computer science, without necessarily expecting students to be competent programmers before tackling the range of topics covered. It is a pilot for a wider range of computer science field guides intended for international use in a variety of contexts.

Key elements of the guide are that it uses a constructivist approach (students learn by interacting with the guide, and working out principles for themselves). It is designed to be engaging and humorous, and it tackles topics that many people would consider "too hard" for high school students by presenting them in a creative form that allow real engagement. Each chapter has several key elements. First there is a short, usually humorous, video designed to engage students with the content, and to introduce them to the big ideas in the topic; for example, the chapter on Network Protocols begins with a commentary made in a light plane, discussing the importance of protocols for the accurate delivery of messages (in this case, for pilots) to avoid misunderstandings (see Figure 4). Next in the chapter is text that takes students through sections that introduce "key concepts" in the topic, provide in-browser interactive activies and games to experience the concepts, and a "project" that students can use for their assessment. Chapters conclude with a section called "the whole story" which discusses details that would be more complex than needed for the standard, but will be of interest to some students. Also important for students are the use of information boxes named "Jargon Buster" and "Curiosity" to allow for greater clarity and further exploration respectively. For teachers, there is a special teacher-only version that has extensive notes and pedagogical advice, which is to support teaching of relevant CS standards in NCEA.

Figure 4: Image from the introductory video for the CS Field Guide chapter on Network Protocols.

Since the Field Guide was released in January 2013 it has had a total of 128,000 pageviews from 50,400 users. In mid 2014 it typically had 150 to 300 users each weekday, with about a quarter of that on weekends. Over 100 New Zealand teachers have registered on a mailing list to be informed about updates, with teachers from overseas joining now as well.

Providing resources for teachers that are designed specifically for the curriculum requirements make a big difference to teacher confidence, and therefore the extent to which the new material is likely to be adopted (Thompson *et al*., 2013).

6. CONCLUSION

Because the decision to deliver a particular NCEA standard is made by each school, the uptake of the new standards rests very much on the confidence and training of individual teachers who will either champion the new area or avoid it. Already thousands of students have had some experience of programming and computer science through the new standards introduced in 2011, and this appears to be having a positive effect on the supply pipeline. However, there are still issues, including a low proportion of female students, and the standards not being available in all schools. These can be addressed by providing better support for teachers, and introducing topics relating to programming and computer science earlier in the curriculum. These will provide more opportunities for school students to find out if topics like programming, computer science and software engineering are for them.

7. ACKNOWLEDGMENTS

We are grateful to NZQA for supplying the raw statistics on which much of this paper is based, and Jack Morgan for assistance with our CS education research.

8. REFERENCES

Bell, T., Andreae, P., & Robins, A. (2014a). A case study of the Introduction of Computer Science in NZ schools. *ACM Transactions on Computing Education (TOCE)*, 14(10):10:1–10:31.

Bell, T., Duncan, C., Jarman, S., & Newton, H. (2014b). Presenting Computer Science Concepts to High School Students. *Olympiads in Informatics*, 8:3–19.

Bell, T., Rosamond, F., & Casey, N. (2012). Computer Science Unplugged & related projects in math & computer science popularization. In Bodlaender, H. L., Downey, R., Fomin, F. V., & Marx, D., editors, *The Multivariate Algorithmic Revolution & Beyond: Essays Dedicated to Michael R. Fellows on the Occasion of His 60th Birthday*, volume LNCS 7370, pages 398–456, Heidelberg. Springer-Verlag, Berlin, Heidelberg.

Caspersen, M. & Nowack, P. (2013). Computational Thinking & Practice—A Generic Approach to Computing in Danish High Schools. In *Proceedings of the 15th Australasian Computing Education Conference, ACE 2013*, pages 137–143, Adelaide, South Australia, Australia.

Curzon, P. (2013). Cs4Fn & Computational Thinking Unplugged. In *Proceedings of the 8th Workshop in Primary & Secondary Computing Education*, WiPSE '13, pages 47–50, New York, NY, USA. ACM.

Johnson, J. S. & Newport, E. L. (1989). Critical period effects in second language learning: The influence of maturational state on the acquisition of English as a second language. *Cognitive Psychology*, 21(1):60–99.

Margolis, J. & Fisher, A. (2003). *Unlocking the clubhouse: Women in computing*. MIT press.

Papert, S. (1996). An exploration in the space of mathematics educations. *International Journal of Computers for Mathematical Learning*, 1(1):95–123.

Ragonis, N., Hazzan, O., & Gal-Ezer, J. (2010). A survey of computer science teacher preparation programs in Israel tells us: computer science deserves a designated high

school teacher preparation! In *Proceedings of the 41st ACM technical symposium on Computer science education*, SIGCSE '10, pages 401–405, New York, NY, USA. ACM.

Riegle-Crumb, C., Moore, C., & Ramos-Wada, A. (2011). Who wants to have a career in science or math? exploring adolescents' future aspirations by gender & race/ethnicity. *Science Education*, 95(3):458–476.

Thompson, D. & Bell, T. (2013). Adoption of new Computer Science high school standards by New Zealand teachers. In Knobelsdorf, M., Romeike, R., & Caspersen, M. E., editors, *The 8th Workshop in Primary & Secondary Computing Education (WiPSCE 2013)*, Aarhus, Denmark. ACM.

Thompson, D., Bell, T., Andreae, P., & Robins, A. (2013). The role of teachers in implementing curriculum changes. In *Proceeding of the 44th ACM technical symposium on Computer science education (SIGCSE '13)*, pages 245–250, Denver, CO. ACM.

Wing, J. M. (2006). Computational thinking. *Commun. ACM*, 49(3):33–35.