

Dear Professors,

Firstly thank you for taking so much note of my brief piece in THES. I had expected it to sink without trace. I would like to reply to some comments made in the THES.

I would point out that the views expressed in the THES article, the BCS article, the Sydney Morning Herald and here are purely my own opinions and are not necessarily the views of the School of Computing at De Montfort University or De Montfort University itself.

The role of an academic as I understand it is to challenge existing views, to seek new ideas and seek to move fields forward. We should not be held back by the demands of industry nor institutional views within existing academic communities.

The evolving role of IT Professionals: My Background

I first came into contact with programming using an IBM1130 at my Dad's college, when I learnt FORTRAN and developed a dating program on punch cards. Although I did a degree in Microbiology, I worked at Hursley Park during my summer vacations, looking at database structures and doing assembler diagnostic programming for the IBM8100, one of the first attempts to put a microprocessor (the M6800) into the back of a 3270 and give it some processing power. I've spent much time in commercial programming in the utilities and in the late 80s developing resource management systems for acute hospitals. This involved a wide range of programming languages including IBM's REXX as well as experience in software audit and software quality practices. I have taught programming languages, database design and theory and systems analysis and design.

My teaching work and research within the information systems has moved towards IT service management, teaching the application of ITIL and ISO20000 and investigating the service aspects of IT provisions, particularly looking at the issues involved in service improvement plans (Cater-Steel & McBride, 2007). I have investigated the management of software quality and I am currently writing a paper using a new ethnographical approach to understanding software quality management.

I have continued to explore the core discipline, recognising the severe limitations of computing and the problems hit in dealing with systems that have difficulty adapting and are easily broken. Hence I was the only UK participant in a Biological Framing of Problems in Computing workshop held at the Sante Fe Institute in August 2002 (McBride, 2002a), where I worked with leading US computer scientists including Richard Gabriel, Ward Cunningham (inventor of XP) and Dave Thomas. The inspiration from this workshop is still driving my thinking. I gave two talks at Symposia of the EPSRC Network on Evolvability in Biological and Software Systems in 2002 in which I was looking at alternative approaches to software development and the development of individuality in software systems (McBride, 2002b, 2002c). I was the only UK delegate at the First International Conference on Design Science Research in Information Systems and Technology which aimed to bring the focus back in information systems to the IT artefact and its development. Here I presented a poster concerning exploring quantum theory as a metaphor for new

approaches to database design (McBride 2006). I am also looking a creativity support systems and how the contribution or non-contribution of computer systems to creativity might be evaluated (McBride and Brown, 2007). As part of the BSc ICT I co-designed modules in Open Source Software Practice and Service-Oriented Computing and supported them through the appropriate management boards to assure that they are delivered starting from September 2007 to final year students. In my definition for the BSc ICT I wrote a syllabus for a final year module in Biologically Inspired Computing.

I consider myself in interdisciplinary, drawing on whatever ideas are appropriate to shed light on problems in the implementation and use of computer systems. This has lead me to write on jazz improvisation (McBride, 2003) and to apply structuration theory (McBride, 1995) and actor-network theory to problems in information systems (Cater-Steel and McBride, 2007). My interest in interdisciplinary work led me, for example, to write a paper for a geography academic journal (McBride, 2003).

While my research is very speculative and incomplete, and my discipline, if I have one, is information systems, my small scholarly efforts in no way suggest a lack of engagement with the core discipline as your article suggests.

The Evolving Role of Computing in Industry

As I've taught and researched at De Montfort I've recognised a change in focus in industry. a move from constructing systems to procuring them, from programming to services.

There is a growing realisation that IT in organisations, and as practiced by suppliers and software houses, is a service. The focus of the role of IT in organisations is on the delivery of technology, education and day-to-day support in order to meet the information needs of an organisation or group of organisation – whether that is the business needs for a SAP system, or the needs of CERN for high performance computing. IT within organisations is about services, procurement, contract management and integration.

This realisation has lead to the recognition of the importance of service sciences. Service sciences is a growing area (See the CACM issue on it). The BCS has an IT Service Management web page and since 2004 IBM has been heavily promoting service sciences. IT departments are pursuing standards for managing their services. There has been a massive growth in interest in ITIL and ISO20000, which has spread to the US and Australia from Europe. Additionally, the demands of Sarbanes-Oxley legislation in the states have driven the growth of concern in IT service management.

The IT Service Management Forum, which drives the development of ITIL (now in version 3) and the use of ISO2000 and grown tremendously and worldwide. Articles in the British Computer Society ITNOW magazine (McBride, 2001, ITNow, 2005) have pointed out the educational challenge of such a shift in focus.

Such a service focus arises, I believe, from the maturing of the industry. Application products, component usage, software and hardware platforms all indicate a mature

industry which is moving into a different phase of development. A majority of our graduates will work in service environments, seeking to harness people, software and hardware resources to meet the business needs of their organisation.

Motivation for Death of Computing Articles

The THES and Death of Computing came out of two strands. Firstly, as part of my job I must attract more students to computing. Numbers at DMU in computing have fallen. I have to look at where kids are in school with ICT, A levels and GCSEs and where they need to be in industry and provide a course that crosses that gap. I'm looking for a strategic direction for computing. The result of a lot of thinking, together with input from my colleagues, and examining the e-skills agenda, and talking to staff at the cabinet office, and IBM, was the BSc ICT, the Guide to the degree which is available on the web, see this page: http://www.dmu.ac.uk/faculties/cse/courses/computing/info_com_tech.jsp.

Having had this validated it is my job to provide PR for it. I started off by contacting ICT teachers locally. This included attending a careers fair and is part of a wider effort at DMU to engage schools ICT in the wider reality of IT practice in industry. As part of that publicity plan, the wider media was to be engaged through articles in THES, TES, Education Guardian.

The second thing that fired me off was reading an IT supplement in THES, which was supposed to encourage applications to ICT degrees. I found it disappointing. You know the image IT has, as portrayed on programs like "The IT Crowd". The computer geeks, you wouldn't want to be in the same room with. They can't talk to you, they won't look at you, they have difficulty empathizing. The pictures in this supplement including the front cover had 'computer geek' written all over them (I'm making no criticism of the researcher and his work I'm addressing the image that came across). The piece by Professor Mander seemed to be saying we need more maths, more computation, we need to be more academic in our computing (to whom will that appeal?) and clashed with a piece about industry skills by an industrialist which identified a wide range of non-academic skills and interdisciplinary skills. To cap it all, the supplement ended by suggesting that using a pen computer was a paradigm shift which it clearly isn't. This supplement offered nothing to promote computer science or spread the coverage of ICT. Its appeal was for a limited set of programming-minded students who would go for computer science anyway.

Both these aspects concerned me and led to the construction of the BCS article. I needed to write it in a way that would gain people's attention. That meant using metaphors and simplification which may result in hyperboles but would get the point across. I've read a book on writing newspaper articles, studied writing and am currently trying to develop skills in poetry in order to connect with people. I also listened to the opinion editor's guidance about structuring the article and about the audience.

If we are to connect with people we must write in their own language. We need people like Simon Sharma, Steve Jones, or Richard Dawkins (whose skills are unsurpassed even if you disagree with his message). Who of us has got a computer

science book onto Tesco's shelves? That means engaging with the ethics, as DMU's Professor Rogerson does, engaging with the social aspects, identifying that computer effects, and computer failures are as much social and managerial as technical. It means engaging with ICT across society – hospitals, schools, media. When Professor Sloman of Imperial College came to talk about Ubiquitous Computing at DMU, most of the questions, from a technical audience, concerned ethical, social and moral issues.

When THES required that my draft be rewritten, I sent my first draft to the BCS to appear in ITNow. Within less than two weeks, with almost indecent haste, it was put in Computing and on the BCS site. Subsequently other pieces have appeared in Computing commenting similarly which suggest a lack of satisfaction within industry of what computing departments are providing in terms of graduates. I have spoken with industrial IT staff in the NHS, banking, universities and manufacturing. I have talked with IBM and the UK Cabinet Office's IT professions team. Such discussion shaped my thinking for the BSc ICT. Nobody demanded more programming, more formal methods or new armies of trained programmers.

The Replies

If I had written about the death of philosophy I would have been ignored. It would be water off a ducks back. But because I wrote about computing people responded. You are worried. That's because there is a major problem in computing. People I talk to in industry are in agreement with my analysis. I have had a significant amount of positive response by email, particularly from IT professionals and people who interface with computing, for example in humanities. It is clear that there are real issues here that we can't hide away from.

The THES response

Your response in THES does not really work. You firstly did not consider what a newspaper article needs to look like and the need for a punchy attention-grabbing start. You've let Steve Farrar add an introductory paragraph which gives the debate and me more legitimacy than you would like and I would expect. Next you link Computer Science with Chemistry, Physics and Biology, which are constantly in the news because departments are closing. Biology almost doesn't exist as a standalone discipline now. You have immediately aligned Computer Science with other 'failing' disciplines.

You then suggest that my complaints are a description of what happens in departments that don't do computer science research. My focus was on computer science education whether in research-led departments or not. I did not suggest that Microsoft doesn't hire computer science PhDs.

Significantly, you do not comment in the dearth of UK Computer science PhD candidates reported in the ICT International Perceptions EPSRC report. Indeed, that report does not get mentioned. This is not surprising, since it supports my main argument. It talks about the eroding pipeline of UK students. It suggests that too little visibility is given to mixed ICT careers. It comments on the lack of 'talented

resources spanning the core ICT technologies and the areas of application'. It says that the emphasis has been on the 'engine room'. It recommends communicating the true breadth of career potential beyond 'university research laboratories and fields of cubicles with displays and keyboards'. This report needs your careful attention.

You do not tackle the economic arguments. The key issue is globalisation, and the cheapness of Indian and Chinese graduates. It's not that we can't pursue such goals, but the Indians are providing equal quality for one tenth of the cost. Your response cannot be to train more programmers, but to become different and incredibly innovative. The focus must be on value-adding. Our focus must be on design. I think we must significantly expand the design element of computer science courses and use components and approaches such a MVC to enable students to focus on the design issues of clients.

My worry is not that lots of the basic programming work is done in India -. for example, NHS system for NPfIT are at least partly developed in India – it's that Indian IT companies and universities are giving large amounts of investment and staff time to think up new ideas. I don't think any of my pessimistic views or anybody else's are going to shift IT jobs to India. That's plainly ridiculous. The arguments are to do with markets and globalisation. Globalisation creates a need for increased innovation and adaptability. The UK computer science (and information systems) community needs to respond with new designs and innovative ideas.

There is a possibility of computing going the way of the textile industry and possibly even faster because the start-up costs are lower for software development than textiles. Hence our focus must be on innovation and creativity. Our graduates must be the most creative people around. Leading edge work like Professor Johnson's in Bath on Creativity must be progressed. There must be a massive renewed focus on design and on coming up with new ideas, new directions, new applications

It concerns me that you next reveal your colours and box yourself clearly in the 'geeks camp' by criticising IT/ ICT degrees for having little rigorous programming. It is clear that CS should not be equated just with programming. Your article simply strengthens that connection. My impression from industry is that more programmers are not needed in larger numbers that is not the skills gap. Is computer science just about rigorous programming and students hiding in cubicles churning out Java? Do all the applications of computers need an army of computer science programmers to do the programming? Probably not. A lot may be componentised. The products are stable, a lot of the programming been done. E-skill degrees may be unproven, but just see what SFIA contains – a skills definition approved by a massive range of leading companies. Programming and development are a small part of a range of skills needed for today's IT Professional.

Not only do you need to investigate e-skills instead of looking down your nose at it. You need to investigate the real need of industry and then find ways of communicating those needs to students in schools and careers officers.

The Mander Response

In contrast to the lacklustre reply of Wilkes et al, the reply by Keith Mander on the BCS website was excellent. He recognised that there is a problem, that you can't underestimate it. He emphasises the complex nature of the employment market. He recognises that the production of foundational components will require fewer graduates than the broader industry (my concern is more with the broader industry). He faces up to the criticism in the International Review of ICT. And he is encouraging about the future prospects. Note that he comments on the need for more high level design skills – the need for skills in innovating new products, creative designs (where has the creativity in software development gone?), not programmers, not lots of writers of interrupt handlers. He recognises the sea-change that's occurring and calls for 'an alliance not seen since the early days of computing.' I would agree with him. Indeed, the key issue is about improving design skills, teaching creativity and innovation and not the ability to copy programs.

Teaching Computing in a Changing Environment

At De Montfort, my mission is to put well rounded, thinking students into industry. I guess we're differentiated from places like Oxford. Our focus is on employability and providing students with skills that will get them jobs (I'm not saying people at Oxford won't get jobs! I'm saying that our focus is much more on employability). It would be wrong to expect Harrods to be the same as Asda. The customer profile is different, the products are different and the customer expectations are different. However, that does not mean there is any difference in quality.

I try to get the students to think theoretically as well. On my module, we'll think about Chaos Theory, Actor-Network theory and Scripting as well as the practical aspects of implementing ITIL. But at the end of the day people don't come to DMU for a grounding in Theoretical Physics. Their parents want them to get jobs and it's my duty to align what I teach with industry.

The situation in computer science education is well recognised. Nothing I have said is new. Neither are the main concepts contentious. Furst, M and DeMillo, R.A (2003) . at the Georgia Tech recognise that the globalisation of computing will have a significant effect on the need for and particularly the skill set required by graduates. They consider an approach to computer science education which develops expertise in 'multiple high-value areas of computing' and creates graduates who are boundary crossers. They consider that computer science graduates learning traditional computer science curricula will find it difficult to differentiate themselves from others in a global market. Symphonic thinking is about the ability to 'synthesise rather than analyse, to see relationships in seemingly unrelated fields'. This involves a holistic approach to computer systems and understanding how computer systems fit together. I think our thinking approaches often tend to be too reductionist. This may arise from the mathematical training of many computer scientists. But surely even advances in mathematical sciences require the sort of holistic thinking that combines different strands or disciplines in mathematics in the way Andrew Wiles did?

Patterson (2006) clearly identifies the need for new computer science curriculum. He identifies a drastic need to change the curriculum. He recognises the huge disconnect between professors who have never worked as professional programmers and the tools, libraries and component driven ICT world of 2007. He suggests a need for

modules that build software using tools and components. He further calls for much more attention to be paid to courses to parallelism and concurrency. He suggests courses which use open source and involve student in actual open source development. And he suggests that students build their own supercomputer. He notes in ending that our curriculum must catch up with 21st technology, that we must leverage the better background that our students arrive with, and that CS faculty must learn new ways.

Denning and McGettrick (2005) identified a chasm between our historical emphasis on programming and the contemporary concerns of those choosing careers. Their diagnosis concerns the persistent image that computer science is a field of programmers, that CS is about coders. This is a narrowing of computing to computing = Java. There is a large communication gap. CS is much more than Java programming. It's service sciences, ethics, and particularly design. (I think we should be turning out designers, not programmers). What Denning and McGettrick say is extraordinary. The core practices of Computing are programming, systems thinking (which I put at the heart of the ICT degree), modelling (which is the basis of design) and innovating (which demands creativity, thinking outside the box, risk, trying out new ideas, working in uncertainty). They propose giving innovation a greater role in curricula (which would also make graduates more competitive in a global market). A spirit of innovation should permeate CS and IS courses. They divide undergraduate education into the creation of ideas and the adoption of ideas, through a 'Great innovations of computing' approach. This gives rise to a whole new syllabus including, for example, environmental computing and e-government.

These articles clearly indicate a need for major changes in curriculum to meet the changing needs of the technical environment. Computing departments that will survive are those that engage with the changing social environment and diversify their offerings.

Incidentally, Howard Gardner's management book on the five minds (Gardner,2007) for the future provides an interesting classification of the areas that should appear in computing curriculum. The first, the disciplined mind, involving mastery of a particular area is often all computing courses are about. Such a convergent approach leaves our students lacking breadth and exposed to obsolescence in a rapidly changing technical environment. The synthesizing mind draws sources together, considers the whole, and takes in the wider picture. At the heart of this is the essential need for systems thinking. The creative mind is able to think divergently and come up with new ideas. Our analytical approach drives this out of students. The respectful mind understands other cultures and points of view. This is a problem for technically focussed people who are unable to see things from other people's point of view. Emotional and social intelligence are vital in the computing student's armoury, in relating to users and relating to other cultures are part of managing off-shore outsourcing. Finally, we must amplify our teaching of ethics and our understanding of the ethical issues around computing in order to develop our students' awareness. Ethics is not just a nice extra topic but a core driver in what we do in computing.

The upside down degree: Design First

Students arrive on CS degrees with experience of using complex systems and putting together little systems from components. They are used to instant gratification and the quick building of a variety of computer applications. They are used to seeing complex outcomes on the screen as a result of simple programming activity. Then, in the first year, we expect them to go back to learning programming starting with 'hello world'.

Perhaps a degree is needed that starts with the design. That develops whole systems in the first year using components. Then the computer science theory, the compiler theory, the low-level programming become the final year modules.

The complex issues in computer science, I believe, centre round design, not implementation. Those design issues involve understanding customers, problems, and environments, both local and global. Studying design issues will also involve expanding the students' creativity and innovation. The new design and innovation course at Cambridge provides an example of the type of course that should be pursued.

It may be worth exploring different programming approaches, perhaps starting with Starlogo.

The Outward-Looking Degree: Application

CS is an applied science not a pure science, invented by man not God. Hence the roots of development should go back to tackling application problems. Babbage, Turing, Von Neumann (See Von Neumann (2000)), were all driven by application needs and problems in other disciplines that they faced. Von Neumann spent his post-war years considering problems in meteorology and nuclear physics.

These should be reflected in our courses. We should start students on bioinformatics. We should create new degrees – Environmental computing, Biocomputing, Creative Computing – which should link to interdisciplinary research programs which cross boundaries.

The outward looking degree should develop a broad understanding of how computing has intervened in human activities, what the potential is for further intervention in systems biology, in education, in business. Such application area exploration and problem identification can then lead to developing an understanding of problem solving and then using skills in computation to solve those problems.

Computing and the Changing Social Environment

I think that a major part of the problem is cultural. The stereotypical portrayal of computer experts as geeks has an underlying truth. That engineering unawareness of how people feel, that social awkwardness, is a characteristic of people, many in computing including me. I am merely a contained, reined in computer geek who has buried the programming manuals..

The problem is that social expectations have changed. Acceptance of dysfunctional social behaviour is less. What would be overlooked as just non-standard behaviour is

now diagnosed as a syndrome and becomes less acceptable in society. The increasing numbers of people with Aspergers, for example, may not derive only from similar people in the Silicon Valleys of California and Cambridge marrying, but in the changing nature of society in which even minor deviations from the norm become classified as conditions to be treated and may lead to people even to being labelled and ostracised.

It is within this social environment that the characteristics associated with computer experts become less acceptable and do not attract people to the discipline. Old computing cultures, with their social and psychological characteristics become isolated, bounded, separated from society in a kind of personality version of multiculturalism.

Hence the battle to get people interested in computing again becomes more cultural than technical. And we are all aware that the difficulties of changing culture even in an organisation are much greater than the simple matter of changing technology. Failures such as are found in aspects of NPFiT derive from inability and unwillingness of IT professionals to engage with the culture, to connect with users and carry the users along with them. I suspect it has little to do with the software, the hardware, or the programming bugs.

Such cultural connection, in actor-network terms, will involve understanding the interests of those outside the computing social circle and showing how their interests are served by computing, how applications and the technology of those applications are of interest to them.

We cannot sit in our churches, behind stained-glass windows and expect the unchurched to come to us. We must go out into the market-place, the media and take the message to them.

That message is not getting through. Not only is it not getting through in schools, on the web, in the media. It's not getting through in industry. Business leaders do not understand the value of ICT, they see it as a cost-sink, not a source of competitive advantage and innovation. However, a response that demand that pupils, teachers, business people, become like us, immersed in functional programming will get us nowhere.

Searching for New Paradigms: The relationship with Biology

There is a sense that computing (that's CS and IS) is in the position Physics was in early in the twentieth century. It's all done – or at least it seems that way. It's all done and school children can't see that there's anything new or exciting about the underlying technology. The specialist has become the familiar and the everyday. The interest is in the content, the usage, the applications not the media which supports those applications. Like physics it's all known.

But what shifted physics was the development of quantum theory. The certain became uncertain, the measurable became immeasurable, the known unknown. Weird effects

were discovered. Things appeared to be in two places at once. Electrons behaved as particles and waves, forming diffraction patterns even if fired through slits one at a time.

So where are the new paradigms for CS? Certainly not in pen computing. It seems to me we have two paradigms at the moment. The computational paradigm relies on systems being reduced to containable parts which can be mathematically proved to be correct: hence the search for the provable compiler.

This requirement for provability requires that any social, business or personal phenomena be reduced to a set of rules. This is hard to do in straightforward business situations where the rules may be simple but rapidly changing. In the case of complex psychological and social phenomena like trust, it is hard to see how it can be accurately represented in a set of computational rules.

I think that attempts to hold on to certainty, provability, will be defeated by uncertainty and the complexity of the business and society we support and the global systems.

The alternative paradigm of evolutionary computing, which seems random, a shotgun approach which may generate solutions, which work but we don't understand why, may elicit a feeling of loss of control. Even cellular automata only provide a particular kind of search space. Wolfram's "New Kind of Science" (Wolfram, 2002) suggests that there is a finite set of cellular automata, universal patterns from which we can only passively select.

Between the simpler certainty of software engineering and the complex uncertainty of evolutionary computing, I think there is a black hole. It is in that black hole that new paradigms may lie.

There is an intimate and long standing relationship between computing and biology. Right back to Von Neumann and Turing, biology was a source of inspiration for computer systems. Turing and Von Neumann were as ready to write about neuroscience and cellular development as about algorithms and analogue computers.

Yes, computing can support biology through the discipline of bioinformatics, but the basic designs of biology should provide metaphor, analogues, and new theoretical directions for computing.

Key questions are now being asked in biology. The sequencing of genomes gives rise to new information and new problems. Far from solving biological conundrums, the sequence information opens up a whole new fields including systems biology which not only give rise to new computing requirements and problems, but also can inform the development of computing.

How can a genome of only 25,000 protein-coding sequences give rise to the complexity and variation of humans?

How can a genome 99% the same as humans give rise to chimpanzees?

What is the role of the breaks or introns in those coding sequences?

How can certain homeotic genes control whole structures?

How are biological organs kept the same in all animals? Why is a liver always a liver?

What are the roles of small non-coding RNAs?

How do networks of control, networks of development lead to stable, reproducible biological structures?

How can individuality be generated within the context of a stable species which such a minimal specification?

The genome does not appear to be coding for a fixed structure, but for a development program, located in time which leads to structure which themselves are constantly altering and developing. Could we envisage programming developmental systems?

At a theoretical level the search should be on for completely different ways of doing things: Finding new problems, finding new ways of solving them.

Actions

I was criticised for not providing any positive ways forward. I admit it is much easier to be negative. From my view as an outsider here are a few suggestions:

Schools

Visit Schools and set up School programs. These need to be led from the top, with each professor making at least 3 school visits in the Autumn Term. This should be followed up by a review of ICT teaching in schools and suggestions for new syllabus. Schools and ICT departments should be adopted and advised. But not in a dictatorial style : “I’m the big expert coming to tell you what to do”, but (once we’ve done our emotional intelligence courses) with a listening attitude. A willingness to sit in on lessons, understand what concerns ICT teachers, what enthuses and turns off their students (this probably won’t involve introducing functional programming).

Research

Develop an Open Approach to Research. Create forums for silly and off the wall ideas. Promote diversity. Harold Thimbleby’s paper (2005) gives a brilliant analysis and should be not only read but actioned. Feyerabend (1993) stated that the proliferation of theories is beneficial to science and every ideas has the capability of improving our knowledge. CS needs a proliferation of new ideas, of innovation and creativity. There need to be forums for ideas that don’t fit into classic conferences or curricula to be given a hearing. As an outsider, it seems to me that the Grand Challenges has lost its impetus and needs revision and renewal.

Engage more with the users of Computing

Gain more understanding of the use of computers in all social environments. Develop creative agendas to support new applications. Develop modules if not courses within CS degrees that address application – medical, educational, environmental, biological, social.

Create new theoretical frameworks to support CS in industry. Investigate user problems in integration, services, modelling and programming constantly changing business processes. Put much more effort into what the user sees, into for example, researching user-centred design and usability engineering.

Start writing popular texts.

Support any effort by your staff to write popular text, radio programs articles for newspapers. The concentration of papers for the RAE, read by an elite few has, I think, not helped CS's reputation. This problem is not unique to CS but an outcome of the RAE. Following the award of the Orwell prize in political science, a commentator pointed out the rarity of writing for a general audiences. Academics in politics "take in each others' dirty washing", write for themselves and fail to engage with a professional audience. Excellent ideas for reform in public administration go unheard of by the politicians or the journalists.

The Call

What started off for me as an innocent piece of publicity for a local undergraduate course has during into a global furore. My attitude then evolved from worry and fear – I don't like upsetting people or facing threats of being sued – to annoyance. I like computing. It is an exciting subject flooded with ideas. IT has changed the world and yet has not even scratched its potential. It changes people's world. It is a people discipline touching people, connecting, serving people. The cross-disciplinary potential of computing is endless.

So I'm very worried by the complacency that casts computer science as the Queen of Sciences. There is a worrying inability to step outside the safety of computational certainty, and the limited theoretical foundations. There is, I think, a need to embrace uncertainty, to pursue creativity, complexity theory, emergent behaviour. More attention must be paid to addressing the foundational threats of the 21st Century and to applying computing to social issues, to global warming, to education and to developing nations.

Responses to this that pretend the threat to computer science will go away, that consider suing me or discrediting me are unacceptable. Academic disciplines are meant to be about debating ideas, not community censorship and the closing of ears. Such debate should be welcomed. It's how disciplines develop and change. Indeed, colleagues and acquaintances in other disciplines have been surprised at your response in THES.

My question to you all is then, 'What are you going to do about this?' We cannot stick our heads in the sand. We must engage more with users. We cannot keep pretending to be a pure science like physics, biology and chemistry and must recognise we are an invention of man with an audience to serve. Klawe and Shneiderman (2005) said that computer scientists must acknowledge there is a crisis and begin to address it. I wonder how far the acknowledgement goes let alone the willingness to address it. Like me, they call for interdisciplinary teams and the reorientation of computer science to deal more directly with societal problems.

Keith Mander is right in suggesting that an alliance is needed which has not been seen since the early days of computing – but an alliance with whom? An alliance that shuts the doors and shuts out other communities? I think the alliance is with those outside the computer science community - with our customers – the IT departments within retailers, IT users within the NHS, parents at home trying to understand their children's use of ICT.

We are three years off 2010. I learnt my computing in the 1970s. When I look at many first year computer science syllabi I see nothing that would have been out of place in 1977. What does this say about academic computing education? How can we catch up with the social and technical change which is racing ahead of us? It took me two years to get a degree up and running which is neither revolutionary, nor radical, nor different but just catching up with industry .. and this was called 'fast-tracking'! We haven't got that sort of time.

The challenge is the greatest of the last 50 years. The need for change in what we teach and research is obvious. The responsibility to change is yours.

Regards

Neil McBride, PhD
School of Computing
De Montfort University

References

Cater-Steel, A and McBride N. (2007) IT Service Management Improvement - An Actor-Network Perspective. European Conference on Information Systems.

Denning,P.J. and McGettrick,A. (2005) Recentering Computer Science. Communications of the ACM, 48, 11, 15-19.

Feyerabend,P. (1993) Against Method Verso

Furst, M and Demillo,R.A (2003) Creating Symphonic-Thinking Computer Science Graduates for an Increasingly Competitive Global Environment

Gardner,H. (2007) Five Minds for the Future. Harvard University Press.

IT Now (2005) University Challenge

<http://archive.bcs.org/bcs/products/publishing/itnow/onlinearchive/nov05/> see p 24-25

Klawe,M and Shneiderman,B. (2005) Crisis and Opportunity in Computer Science. *Communications of the ACM*. 48,11 27-28.

McBride, N K (1995). The Role of Executive Information Systems in Organisations: An Interpretive Approach. *Proceedings of the 28th Annual Hawaii International Conference on System Sciences*, III, 110-119.

McBride, Neil (2001). IT at Whose Service? *Computer Bulletin*, September 2001, 24-27.

McBride, N (2002a). A Biological Metaphor for Adaptive Evolutionary Information Systems. Feyerabend Workshop on Biological Framings of Problems in Computing, Santa Fe Institute, New Mexico, 17-19 April 2002.

McBride, N (2002b). Using Molecular Biology as a Metaphor for Generating Architectures for Adaptive Evolutionary Information Systems. Presented at the EPSRC Network on Evolvability in Biological and Software Systems. Symposium on Software Evolution and Evolutionary Computation, 7-8 February, University of Hertfordshire, UK.

McBride, N (2002c). Individuality, Control Networks and Computer Systems. Presented at the EPSRC Network on Evolvability in Biological and Software Systems, Symposium on Evolvability and Individuality, 18-20 September, St Albans, Hertfordshire,UK.

McBride, N (2003). A Viewpoint on Software Engineering and Information Systems: Integrating the Disciplines. *Information and Software Technology* 45, 281-287.

McBride, N (2003). Actor-Network Theory and the Adoption of Mobile Communications. *Geography* 88, 266-276.

McBride, N. (2006) Towards a metaphor-driven information systems design process. *Proceedings of the First International Conference on Design Science Research in Information Systems and Technology*. Claremont Graduate University Feb. 24th – 25th 2006.

McBride,N and Brown,S. (2007) Towards Computer Systems to Support Creativity. Working Paper.

Patterson, D.A. (2006) Computer Science Education in the 21st Century. *Communications of the ACM*. 49(3) 27-30

Thimbleby,H. (2005) Valuing Diverse Computing Science Research. Available at www.cs.swan.ac.uk/~csharpold/tick/csresearch.pdf

Von Neumann (2000) *The Computer and The Brain*. Yale University Press.

Wolfson,S. (2002) A New Kind of Science Wolfson Media Inc.

© Neil McBride 2007