Capturing the Past, Inspiring the Future

Dr Andrew Herbert OBE

Interviewed by

Mark Jones

4th April 2019

At

Dr Herbert's Home in Cambridge

Welcome to the Archives of Information Technology. It's the 4th of April 2019. I'm Mark Jones, an interviewer with Archives of IT. Today we're in the Cambridge home of Dr Andrew Herbert.

[00:00:13]

Andrew retired in 2011, after a long and eventful career in IT. He has been a lecturer and researcher, an entrepreneur, a manager of SMEs, and finally Director and then Chairman of Microsoft Research EMEA. Innovation is one of the many threads which runs through his career, but I believe Andrew also has a keen interest in history. And I'm very much looking forward to capturing his stories today. I'd also like to hear his thoughts on the future direction of our industry, and where inspiration might lie for people just starting out on a career in IT.

Andrew, thank you very much indeed for agreeing to be interviewed by the Archives.

My pleasure.

[00:51]

In time-honoured fashion, can we start walking through your schooling. I remember virtually nothing about my primary school days, I have to admit. But do you have a particular memory of those days?

I have two, and, I guess they both in a way relate to how my career progressed. My first primary school, we had an American, young American woman as a teacher, and she organised a trip to go and visit what was then a US Air Force base at a place called West Malling, near my family home. And, we were given a tour round the airfield, and, shown aircraft, and some of the things happening in avionics and radar and so forth. And that was quite good fun. I, I can't remember much of the detail, but I still have that sort of visual impression of an airfield and some fast jets and, and lots of technology. And outside computing, aviation is a, is another passion which in retirement I can indulge in, so, one of the seeds was planted quite early. And my father worked in the avionics industry, that kind of reinforced that. The other memory which sticks in my mind is, we were subjected to one of the cycles of New Maths when they came through. I think it was the first time that phrase was used. Because we must be talking of, something like about 1962 or 3. And one of the things they

taught us was binary arithmetic, because these new-fangled computers used it and it might be useful to understand different number systems. And I remember being completely intrigued that you would use just ones and zeros to hold numbers and calculate in them. And although I had never seen a computer and knew nothing about computers, that just stuck in my mind. So, another seed I think was planted at quite an early age.

[02:44]
So, did you have a conscious yearning towards science and technology then, or, just an interest really?

I was always kind of a geeky kid. And as I got through to grammar school, chemistry was, was really my passion, followed by physics, and maths was a kind of necessary evil if you wanted to do sciences. And, I think again that comes from the fact my father was an engineer, and, he was in the avionics industry. He would bring things home for me to play with. I had the obligatory chemistry sets and electoral sets to play with as, as toys. So I think, yes, there was a lot of encouragement to go down that scientific/engineering path. And I just lapped it up.

And did you have much exposure to computers in those days at school?

No. My, my first interaction with computers was, I think either in the, the very end of the fifth form, just after doing O Levels as they then were, or else it was the very start of the sixth form. I encountered my maths master, a Mr Stokes, who it turned out had come into teaching after starting his career in programming for Elliott Brothers, who were quite strongly in minicomputers in the Sixties. He, it was the time of one of the General Elections, and he was writing curious runes on squared paper, which he explained was something called FORTRAN, and he was doing analysis of the voting results. And, that intrigued me, and to make a long and rambling story short, he ended up finding himself running a computer club, and it was the classic model of, we wrote our programs out in squared paper, we sent them in an envelope to the local technical college, and a week later you got back a paper tape and a list of errors. [laughs] So you very quickly learnt to write correct code. And that was, that was really the start of it. And I was doubly fortunate that I persuaded my school that I

was, less use on the rugby and the cricket field than if I was to cycle up to the college on games afternoon, taking the school programs, running them, and bringing them back. And, the, the college welcomed me there, it was their games afternoon also, and I had the run of the computer suite. So I ran the school programs. And that gave me the chance obviously to correct and tinker with mine, and to discover there were other programming languages, and there were other machines. The place had two machines, an Elliott 903 which they were kind of pensioning off, and a brand new sparkly ICL 1901A. And between them they had FORTRAN, ALGOL, COBOL, assemblers. And on Wednesday afternoons I had the run of the computer suite, all the manuals, and just, completely dived into the hardware and the software, and, tinkered with everything.

I think I probably first met ICL computers at the same time. I remember the school organised a trip to Windsor I think, or Maidenhead, somewhere around there, and we played with an ICL for an afternoon. And that's probably my first sight of a real computer, shall we say.

[06:06]
Yes. Yes, and the, the 903 was essentially a bare bones machine, whereas the 1A had a tiny minimal operating system with just the executive, but you know, that opened my eyes to a computer that could organise itself. And I think, the thing which fascinated me all the way through was that you could just make this box do kind of, anything. You could program all kinds of things. I wanted to, well my main piece of programming was building a program that did... Oh, I'll try and start again. If you were doing A Level chemistry, you had, in those days you had to do a lot of volumetric analysis of working out what mixtures of chemicals and things you had, and it was all fairly routine stuff. So I wrote a program where you typed in the numbers, and it produced a lab report for you. And I thought that was really rather nice, that the machine could do all the work.

Mm.

So I think, aspects of automation, the machine taking over, doing boring tasks, that kind of really excited me. At the same time all that was going on, my father's career

progressed into avionics and that was starting to go digital, and so he was talking about what computers were doing in controlling aircraft. And, he got me summer jobs in those last years of fifth form and sixth form, at Elliott's and with the computer people. So I was involved in writing flight control software and other things. And so got to realise then, computers could not only model the world and simulate it, they can control it.

[07:40]
So it was that experience that led you to your degree course I guess, computational science.

Yeah. Much to the annoyance of my school headmaster, who I think saw me as a potential Oxbridge chemist. I very proudly told him halfway through the first year sixth form, I want to do computer science. And at that point, neither Oxford or Cambridge had an undergraduate computer science course. Cambridge did actually start a one-year computer science option. But other places were offering three years of computer science, and I was so besotted with computing, that was where my choices went. And, I looked through the universities, and was attracted by the offerings at Leeds and Lancaster if I recall, Heriot-Watt and Loughborough I think; the other strong candidate might have been Manchester. But their course looked rather more hardware and electronics oriented than I wanted. And for me it was all about software and compilers and operating systems that got me, got me jazzed.

[08:42]
I think it's hard for us to remember now, but in the early Seventies computing wasn't really a mainstream career option for many people, was it? It wasn't thought of in those terms.

It wasn't thought of in those terms. And, if I'm honest, when I got to Leeds, I discovered in the first year, it wasn't three years of computer science. So our first year was padded out with a lot of maths and statistics, which is all good useful stuff to have in your armoury, and a lot of the course was oriented towards numerical analysis and applications, core computer science, computer architecture, operating systems and languages. Really you could cram all that into, into a year if you wanted to. But I

think that, that breadth was good. From a career point of view, the option was there, and one of the things I explored... My parents weren't wealthy people, and yet they were concerned about how well I'd survive on a university grant, and whatever. So they encouraged me to look for sponsorship, and certainly companies like IBM and CDC were running programmes, essentially grant programmes, to undergraduates who took computing or were interested in computing. But I didn't really at that stage want to tie myself to a particular company and a particular career. I wanted to sort of, know more about the subject. But I certainly remember going to CDC's centre in London, and interviewing there.

[10:12]
Looking back now, do you think what you learnt in your degree course was useful in your later career?

Oh absolutely. I mean there's, throughout my career there have been people who have been particularly helpful and moved me to the next stage. And I think, I talked about Mr Stokes of the, of the grammar school who started the maths club, and encouraged us. At Leeds, the head of department, a guy called Professor Mike Wells, and one of the lecturers, a guy called Dave Holdsworth, spotted my enthusiasm and talent, and encouraged it. So, for example, at that time at Leeds there was one mainframe, the big ICL 1906A, that was used by university researchers, and we students had a KDF9, and bless it, another 903. And there were some languages and facilities on the 6A that just weren't there on the, the older machines. And so between them, you know, they, they allowed me to have an account on the mainframe, which undergraduates weren't supposed to even know existed. And so, those kind of opportunities were really quite important. And Leeds wrote their own version of MOP, which is ICL's multi-access system. They wrote their own version of that, and, and Dave shared a lot of that with me and talking about it, knowing again that operating systems and languages were my, my passion. So yes, I... But in terms of the course itself, I haven't used a huge amount of the numeric analysis and optimisation I was taught, because I didn't go into the applications world. But the grounding in architecture and languages systems, well that got me to Cambridge to do the PhD, which, that kind of defined the rest of my career. So, so yes. I think it was, it was a beneficial course. So many of the things you would teach in computer

science today didn't even exist then. Well I think we, we signed off artificial intelligence in about four lectures, and... [laughs]

Yeah, I think that...

It's a bit more complicated now.

[12:26]
That's right. But, in a way, computers fundamentally haven't changed that much, much of the basic science of computing is still, very similar isn't it?

Well, yes and no. [pause] At a very high level it's still the von Neumann model, and all programming languages are kind of descendants of ALGOL or, ones we use today. But, if you start to look inside modern operating systems and processors, their architecture, it's very very different to the kind of instruction sets and architectures we were looking at. And a whole set of engineering trade-offs have completely swung around now. Like, who does paging to slow disks any more?

Mm.

Because core was small and slow, and, you know, memories can be huge.

Mhm. Mm.

[13:25]
So... Yes, what I was taught has seen me through, but I've had to refresh that knowledge really quite frequently. And I think that's one of the, the challenges, you know. On the coffee table here is a, I've been doing some things with Raspberry Pi recently, and, Eben Upton and some of his colleagues have written a book, *Learning Computer Architecture with Raspberry Pi*. Most of the things talked about in that book have been invented since I did a computer architecture course. And, you know, because I've stayed current with them. I know that stuff. But it makes it very hard to explain how a Raspberry Pi works compared to having a 1906A or 903 explained to us.

Mm.

So yes, knowledge gets you into the field, but you've got to keep it current. And one of the contributory decisions to retiring when I did, as well as being in a fortunate position financially to do that, and have some hobbies I wanted to give attention to, was a feeling of a, actually I didn't really want to go through another cycle and refresh, and hence the interest in old computers, and, [laughs] going back over the old knowledge.

[14:36]
OK. So, we're at the point of going to Cambridge now for a PhD. How did that come about? Did you always want to do that, or did you think, I could get a job, or I could do a PhD?

No. Again, you know, my, my parents both left school at fourteen, so there was no history in my family of anyone even staying on to sixth form let alone going to, to college. So it was kind of, you know, a surprise and a novelty to all of us when I got through to, to university. And, so I had no idea there were PhDs and research students. When I was at Leeds there were two or three research students there who I got to know, and it looked like really rather a pleasant life. You get funded for three or four years to do a project of your own choosing. One of the guys was for example writing a compiler for ALGOL 68, which was a, a challenging new language at that time. And that looked like a fun thing to do. So I thought, yes, I'd like some of that. And went and talked to Mike Wells, and said, you know, 'What's this PhD stuff? [laughs] How does it work?' And, yeah, he, he explained what it's about, and, what kind of things might I be interested in. He recommended a batch of universities I ought to go and visit, and see what they were doing in their, their programmes. And, you know, very kindly wrote letters – this is before email – wrote letters of introduction. All these things were arranged by post and telephone. And, I visited, Newcastle, which, which I really liked. Sir Brian Randell was professor there, doing interesting things on reliable systems, and with the background I got from the avionics work that I had done, that was an interesting topic. I went to Edinburgh. They had done a lot of systems work. They had built their own operating system for ICL

System 4s and had moved that to ICL's New Range that a number of universities were having pushed upon them at that time. But they seemed to have run out of steam. They had done EMAS, and they'd done a programming language called IMP, and nothing very new was happening. And AI was a much stronger thing at Edinburgh, and I wasn't really interested in, in progressing that. I went to Warwick, where Colin Whitby-Strevens had an interesting group dong things around multiprocessor operating systems on ITL Modular Ones. That was of interest. And, I came down to, to Cambridge. And if I'm completely honest, a lot of the initial attraction of, one of the reasons for interview at Cambridge, was, actually going to go back to my headmaster and say, 'Neh, I got to Cambridge doing computer science anyway.' Because he was really quite critical of my decision to do computing, a new-fangled subject, he didn't think it was real. And wandering off to some provincial northern university. I think it ruined his batting average. So I was quite looking forward to going back to an old boys' dinner and saying, 'Neh.' And of course, I got there a year early, because I didn't have to do the extra year of sixth for, which was, to get through the Cambridge entrance exams.

Yes.

So there was a certain amount of that going on. Mike Wells spoke very highly of Cambridge. He had been there. He had used EDSAC 2 to do X-ray crystallography, which was his subject, and he knew Wilkes very well, he knew Roger Needham and the team who had worked on the, the Titan operating system. And Cambridge had this new project, the Cambridge Capability machine. They had just built a new machine, and were thinking about operating systems for it. And they were programming it in ALGOL 68, Cambridge had their own ALGOL 68 compiler. [18:38]
So when I got there, I really enjoyed meeting Wilkes and Needham who interviewed me. The possibility of a new machine to play with, writing in a programming language that I thought was the best language on the planet at that time, and at the place where, it doesn't get better than Cambridge. Oxford didn't really do much computing at that time; it was mostly numerical work. They were starting to grow the theoretical side, which is their big strength today. So, very rapidly decided Cambridge was where I wanted to be. Newcastle were very keen to have me, and

pushing me to make a decision. I remember basically ringing Cambridge up and saying, 'Look, are you going to give me an offer or not? Because, otherwise, I'm going to go to Newcastle.' And, Roger Needham tells... Unfortunately he died a few years back, but he always enjoyed telling two stories about me. Apparently, I was the only PhD candidate who's held them to ransom. 'Send me an offer or I'm going elsewhere.' And they thought that was amusing. And the interview with Wilkes and Needham was, was terrifying. Wilkes sat in, behind his desk in a chair, and just scribbled on a notepad, hardly asked any questions. And Roger's style was to pace around the room. And, so they kind of asked me a question, 'What are your interests? And I said, 'Operating systems and compilers,' and there was kind of, silence. So I thought, OK, I'll talk about what interests me in languages, and I talked about what I thought was interesting in ALGOL 68. And what I liked about it was, the very clean split of the concepts. They called it orthogonality. You could apply any operator to any data type and it made sense of it, and so forth. So I prattled on about that. And, more scribbling, and more pacing around. So, so I talked about what I thought were some of the difficulties in compiling those languages, and why that might be interesting. And, still more silence. OK, time to change tack. You know, I'd, the project I might be involved in here is an operating system one, so I'll talk a bit about operating systems. And commented that I had enjoyed the real-time systems stuff I had done particularly in the, in the summer jobs. And, you know, with the KDF9 and the 6A, they have been kind of time-sharing operating systems, more job queue oriented, they hadn't really been very real-time oriented. And that was starting to show up problems, particularly with networking, which was what Mike Wells had got into, where you kind of what the network to schedule the machine. So I started talking about how you might design an operating system and think about those issues. And, by then I think we had got through about two hours of interviews, and at the end of it they said, 'Thank you very much, that's very interesting,' and got some research student to show me round.

So you had...

I didn't exactly pump myself dry. But Roger told me afterwards, you know, when they finished, they looked at each other and said, 'Gosh, he's keen and knows a lot. We'd better take him on.' [laughs] They didn't share that with me. [laughs] So I left

wondering, you know, how that interview had gone. They had been terribly quiet. I had been shown round, and, 'Thank you very much,' and, you don't hear anything. [laughs] So I was kind of, you know, quite convinced when I phoned to say, 'Look, am I going to be getting an offer,' to be told, no. So when it came through, I was, I was really excited.

[22:00]
So Wilkes, Needham and David Wheeler, the three real key pillars of computing in the UK, weren't they?

They were. By...

Very different characters, completely different sorts of people.

Completely different characters. By the time I got to Cambridge, which was '75, Wilkes was head of department, and not really deeply involved in anything technical. He had been part of conceiving the CAP project. And CAP grew out of, all the challenges they had had building the Titan operating system, and Titan was a cut-down Atlas. They wrote the operating system in assembly code, and had all the problems you have if you're writing large chunks of assembly code, things scribbling in the wrong place, and, being unable to track down bugs and whatever. So, CAP was conceived of with a capability-based memory architecture that basically made it impossible to scribble on things you shouldn't. And if you were rebuilding Titan, that was the right path to go down. The, the sad aspect of, of the CAP project was, by the time we had finished it, and we kind of, should have realised it ourselves, we wrote most of our systems in a high level language, and half those problems went away, if you used a respectable language. And, you know, that's one of the things we might pick up in technology trends is, you know, the hardware does less and the software does more if you've got verification. So, it was kind of, solving an old problem, and, and the problem went away with the new technology, rather than having to be solved. But Wilkes had conceived the project, and a couple of companies, Plessey had built a capability machine for the telecoms market, and there was obviously scope for innovation and thinking about how you apply that in a, in the context of a mainframe operating system.

[23:51]

So Wilkes had kind of conceived of the project, but didn't really understand how operating systems are developed, didn't really understand modern programming languages. I remember many attempts to explain object-oriented programming to him and not really getting anywhere. 'It's just libraries Herbert,' he would say. And sometimes it is, but it's a bit more than that. So, Wilkes was, a very good manager of the department. Very good at keeping people focused on doing challenging projects. So a really good leader. And if I talk to people who worked on some of the earlier projects, you actually get the same message. Wilkes's skill was conceiving of really excellent timely projects, and putting together a team who were better than him to deliver them. So that was kind of Wilkes. He was a bit remote. Everyone called him the old man when I got there, and if you were summoned to be, you know, to, to the old man, it was a bit like visiting a headmaster, you got, oh shit, what have I done? Sometimes he just wanted a chat, to know what was going on in his department. And actually, I got to know him quite well and we became good personal friends.

[24:56]

Wheeler was very enigmatic. He was very quiet. He would pick problems that interested him. He had designed the hardware for CAP, so we would occasionally have to go back to him if there were issues with the machine, how things worked. He designed this machine. He had a, a double-sided sheet of A4 summary which was his description of the machine that he gave you. And it was a mixture of schematics some of the more interesting circuits; some timing waveforms. so you understand the order in which things happen; a list of registers and various ad hoc notes. And indeed, all the information was there. If you had crossed out any one line or figure, you wouldn't have understood the machine, but nothing was explained. But, yeah, you were supposed to be, have a brain like Wheeler's and just interpret this stuff. Did occasionally talk to David about technical things, and sometimes he could give quite valuable insights, you know, just ask a question that might come up you think, ah, yes, and you move off in a different direction.

[26:04]

Roger was leaving the CAP project, so we were, you know, much much closer working together on the design of the operating system, and trying out ideas. Roger and I focused a lot on virtual memory and paging and algorithms and so forth. Eventually discovering that, you're better off having a bigger memory. [laughs] It

solves the problems. But, yeah, so that was a good relationship. And Roger became head of department in '78, and I was essentially his, his sidekick. There were two of us. There was a guy called Andy Hopper, who had worked more with Wilkes and Wheeler and done the Cambridge Ring. He kind of was the hardware guy, and I was kind of the software guy. And we were the, yeah, the two protégés. [laughs]

[26:47]
So after your PhD, you were employed by Cambridge I guess as a lecturer, is that right?

Yes. Strictly my title was, demonstrator, or assistant lecturer. It's... And those were fixed term, three-year appointments. Modern day they would be called a postdoc. So you're encouraged to continue your research, but you pick up some teaching duties. So I taught some courses. I remember I did comparative programming languages, because those still interested me. And I shared with Roger some of the operating systems course. And by then we had started work on building the Cambridge Distributed System, which was kind of a, a mainframe built out of lots of machines attached to a Cambridge Ring, so we thought of it as a shared service. Modern-day age we would have called it cloud computing. A whole lot of computers hidden away from you, you access from your device, and the work was shared out, and whatever else.

[27:49]
So was that to solve a particular problem that was current at the time, or was that just a good idea for research?

[pause] We recognised that mainframes were kind of running out of steam. We saw that the, at that time it was the home computers, the personal hadn't happened, were really very attractive for doing small-scale work, and they're always available and interactive. And so, our model, what we wanted to explore, was, could you build a better approach to time-sharing by putting a home computer on every desk, which was used for mail and editing and kept your files, and, and was yours, and you could get, maybe, carry it home and that kind of thing. And then you would back that up with a whole host of computing resources, on which you run your big computations, like

compiling your programs, or if you're running a big maths analysis. And that those machines could be a spectrum of, yes, you still have your mainframe there with the job control system, and you put things in and take things out. You might well start using some of the superminicomputers that were around at that time, the early VAXes were coming out, and you could use those in a kind of time-shared mode. And, we also wanted to see... There were, a lot of single board 16-bit minicomputers were on the market. It's all part of that transition from, from mini to micro. And, we wondered what you might do if you have a lot of those, because they're really rather cheap, and you could imagine putting a few hundred in a machine room.

Mm.

And then, all the problems we had to fight with in doing mainframe operating systems go away with a single user machine. So it looked like an alternative way of building mainframes really. An alternative way of building a computing service. And if you think of Cambridge's history, up until that point, all the projects had been about the next generation computing service. You know, EDSAC was the first computing service. The work that Wheeler did on the libraries and programming was all about making it easy to program and use. EDSAC 2 took them to the next stage, the focus was still on making it easy to use and be computable. Titan brought in time-sharing and, brought that in. CAP was looking at building a better Titan. It did provide a small departmental computing service. And then this whole processor bang Cambridge Distributed System was kind of looking at what you might do with networks.

Hm.

Can we take a pause a minute? Make a run for the...

[pause in recording]

[30:43]
So, we're in line for a change of direction soon. I think you had seven or eight years as an assistant lecturer and lecturer, didn't you?

I did. I really enjoyed being an academic. And there was kind of a big change in Cambridge that rather than just doing its own thing, kind of more engagement with the wider community. And there were two key aspects to that. Roger Needham spent a year on sabbatical at Xerox PARC, when they were doing the Alto and all that early office automation, and as a consequence of that Xerox gave the Computer Lab a lot of kit. So we had an Ethernet as well as a Cambridge Ring. And we had, the next machine after the Alto, it was called a Dandelion if I remember correctly, and a whole batch of those. And Wilkes of course had retired, and he had gone off to Digital. And Digital gave us both a couple of 750s, so, big machines, and a whole lot of MicroVAXes to build a processor bank out of those. And of course we got to work with, you know, scientists and engineers in those companies, and got a lot more visibility of what was happening in the US. And I think, you know, that very much stopped Cambridge looking inwards and looking much more externally. So it was fun to be there, all this technology to play with, to be working with people in those companies and those labs exploring really leading-edge ideas at that time.
[32:17]
By then I, well I had married as a research student; child number one came along, and, wife wanted to be a stay-at-home mum. University lecturer salary doesn't really hack it. [laughs] So I had started doing a certain amount of consulting and so forth. And I never felt comfortable, as it were, having an academic job but using half the time to do consulting for other people, it was, just taking time away. Salvation came with the Alvey Programme, which was a big Government round of funding in the mid-Eighties, to compete with the Japanese who claimed they were doing a fifth generation, and it was all going to be AI and this, that and the other. Roger and his wife Karen and others had been on a panel that had kind of conceived of the programme, and one of the concepts was large demonstrators to prove the individual results. And one large demonstrator was supposed to pull together all the various pieces of technology into an Alvey workstation, it was called the Alvey Workstation Architecture Project. And it was kind of, your aim there was competing with Sun, and Sun workstations that were really taking over at that point. And, I was invited to make a sort of start on defining that. And fairly quickly said, actually, building a computer is not the right thing, learning from the Cambridge experience. What you want to do is, network devices together and design for a very heterogeneous world,

and think about connectivity, then you could have very specialist boxes, and use different tools for different jobs, and integrate it all together. So, early concepts of application integration. And, the Alvey directors were really excited by that. I was appointed as chief architect for the, what became the Advanced Network Systems Architecture Project, which was what I had termed the [laughs] allsap Alvey Workstation Architecture Project into.

So was APM a vehicle for doing the sort of work you wanted to do, presumably? Did you want to do this?

Yes. The Alvey directors basically wanted a shared laboratory in which this work was done, by, which they would co-fund with industry. So the simplest way to do that was to create a company. I was, I was hired as chief architect; another guy was hired as project director to kind of run it, managerially. So we found premises, and started to build a team.

[34:55]

The first managing director was replaced by, a succession. The one who was with us longest, a colleague called Mike Eyre did most of the, the final contracts that were signed, and he very carefully and very sensibly arranged that APM itself had rights to the IP that could generate it, as well as the industry partners, provided we didn't use it in competition with the partners. And that gave us the ability to spin up new business.

[35:25]

So literally a mix of things, APM did, didn't it?

Yes.

For a private company to get involved in so much standards work and research work was, probably unusual at that time.

It was unusual. So... I think it was, it was several things driving that. First of all there was the commercial imperative. We actually wanted to, yeah, live very comfortably, and so generating income [laughs] was quite important. I saw it initially as a sort of contract research lab, for those industrial partners, and the research we

were doing was what today would be called transformational or applied research. We were taking ideas that had developed in academia worldwide and making them work in the context of present-day practical systems, and putting multiple ideas together. You discover when you do that, all kinds of things don't fit. And there's actually a lot of innovation in making the bits fit. And indeed, you know, what the boundaries are between research innovation is a bit unclear, but for me, research is exploring ideas; innovation is making them practical. And, you know, some research ideas never get to the innovation phase.

Yes.

[36:40]
So... Yes, so that the industry sponsors were paying for research and basically, technology transfer into their own engineering organisations. The standards work came about, partly because at that time governments thought standards were important. I remember the industry in Europe was very fragmented, Bull, ICL, Siemens or whatever. The feeling was, if they could agree common standards, then they, they could make and compete against those standards, and everyone else in the world would have to fall into line. So it would make for a level playing field. So, Alvey being a Government programme, you know, standards one of the boxes that had to be picked. It turned out in the companies, changes was, was hard in, in those companies. They had, they were starting to build up legacies, they had, you know, big projects going on consuming resources. They weren't doing that well in business terms. So adopting new technology was sort of the last thing the management wanted to hear. But if this new technology was an ISO standard, they knew they had to do it. So there was a certain amount of, you know, the, the senior engineers in these companies saying, 'Could you go and do this thing in the standards world, and then we can follow you?'

Right.

Which was kind of, bizarre politics.

[37:58]

So APM was eleven years I think of your life.

Yes. So, yeah, the Alvey project evolved into a big esprit project. That generated spinoffs as various subsets, and the partners wanted to take the technology into different application areas. One of the most successful was called end-to-end security, where with Hewlett-Packard and Swiss Bank we did a big business-to-business system by which HP and its business customers, you know, manage repairs and the billing and, goodness knows, that, using smartcards and, you know, online credit card transactions, browser interfaces, and getting all the security of that right. That was quite fun.

[38:41]
So 1996, time for another change. Was that a planned change, or happenstance?

[pause] Sort of a planned change. We had got through that whole era of, of the contract research and the various projects and so forth. It was obvious to me the research was starting to slow down. The central problem of building distributed applications and whatever, that was, that was solved, and, commercial technologies were out there to do that job. So I couldn't see that continuing forever. And some of the partners were starting to fade away. I wasn't attracted by the possibility of just running a consulting company. We had got a nice software consulting business, but you know, I was in my mid-forties by then, doing that for another 20 years didn't look very exciting. Because I was turned on by, by technology and research. And the whole Internet thing was, was taking off. And to be blunt, by then the managing director and I were facing in different directions. He was near to retirement. His ambition was to top up and protect his pension fund, and I was ready to throw the dice again.
[39:56]
So... And myself and a group of the engineers sat down and brainstormed, what could we do in the, out of the things we were doing, what might be something where we could jump on this Internet bandwagon? And there was some stuff around securing applications running in browsers. The chairman of the management committee for the research side, he was in ICL, he was looking for a new opportunity. I think he had the same sort of frustration with his company. And he introduced us to an American who

became the CEO and first investor, and we essentially spun out Digitivity. We paid the managing director to go away, he got his pension pot, so he was completely happy. We spun up Digitivity. We kept the consulting business, and my deputy ran that, which was nice because there was always revenue coming in, which is unusual in a start-up. With the help of Scott Metcalf, who was the American, and Chris, we raised funding to build this Internet start-up that built the product we had talked about. So we did all the engineering in Cambridge; we put our headquarters in Silicon Valley so we looked American. And a lot of my academic colleagues by then were working in industry, quite a few in Sun who were king of the hill at that point with Java. So, it was, it was a good time to do that. And it was, it was definitely a gamble. I remember conversations with my first wife, because by then we had, certainly two if not three of our children, you know, I'm going to throw the dice again, it's all uncertainty, but the last time I threw them was taking the Alvey thing on, and that doubled my income, and bought us a new house. So, yeah.

[41:41]
So that was two or three years of Digitivity, was it?

Yes. It was, it was three years. We got the product out there. Our sales team, all three of them, found that, they were continually bumping into people from a company called Citrix. And what Citrix's business was, was remote access from branch offices, the head office, to mainframe and, and large minicomputer systems. That was all dial-up technology, and they realised they needed to start embracing the Internet, that was going to replace the phone system for data. And so, we started a conversation with Citrix about, could we become one of their channel partners, and would their, you know, through that channel, could we sell our technology? And we had a couple of meetings. And at the end of the second meeting their CTO, a guy called Ed Iacobucci has turned round and said, 'Look, can we buy you? What do you want?' And Scott and I looked at each other, invented a silly number, and they said, 'We'll think about that.' And they came back the next day with half the silly number. We expected them to come back with a third of the silly number. [laughs] So we said yes. And the deal was done.

Right.

So they literally just bought us out. Because they wanted... They weren't so much bothered about the product, but they wanted the, the expertise and the, the Internet knowledge.

[43:03]
So what sort of scale was Citrix at that time?

They were, sort of, the largest of the medium size companies. So they weren't in the league of, you know, Sun and IBM and Microsoft, a bit below Oracle. But doing quite well, you know, a serious company. They, their business had been very successful, with, MetaFrame was their flagship product. But they were certainly, you know, having to add new capabilities to it. And they were anticipating some of the developments in cloud computing, and, what companies might do.

So did the Digitivity idea survive into Citrix?

The product itself, CAGE, no. It withered away quite quickly. But, well the team still exists, they're here in Cambridge, Citrix have a big office in Cambridge. The security vein of work is still the main part of what they do. And yes, staying abreast of Internet technologies, and integrating those with Citrix's products, and doing very well, and the company's still very strong.

[44:10]
So how did you feel about working for a, very large company, having just set up a quite small company?

I enjoyed it. I mean I, I've enjoyed... Yeah, I've had a bunch of careers as it were. I have enjoyed them each equally. Any job has its pluses and minuses. So it's been nice to sample the different styles. When we were doing the, the ANSA work with, you know, the industrial consortium, that gave me a lot of taste of kind of the corporate environment where there are lots of players and lots of politics, and tensions and whatever. So, that wasn't a huge transition. Citrix themselves, yeah, they had been a start... They, they were in that transition from young company to mature

company, and so there was still a lot of energy and enthusiasm, and freedom to go charging off at all kinds of directions. So, I enjoyed the energy of Citrix. I enjoyed the rewards of working for a successful company in the industry at that time.

So did you split your time between Cambridge and the States?

Yeah, it was pretty gruelling actually. I was sort of, half the time there and half the time here. And they, they made me Director of Advanced Technology, so, a big part of the job was going around telling all the, all the barons who owned the various bits of the Citrix empire, you know, what they had to do to join the Internet age, and you know, some were enthusiastic about that, and others less so.

It's kind of hard to imagine that now isn't it, because we're so, the Internet's so embedded in everything. But, mid-Nineties, it was just completely revolutionary wasn't it?

It was completely revolutionary. And, you know, big banks, insurance companies, you know, and their data centres, had rooms full of modems, and dial-up connections, to, to connect their systems together. Remember, the Internet, yeah, only really started to take off as a, a general communications mechanism, '93, '94. It was, it was well into the 2000s before it was ubiquitous.

[46:10]
Mm OK, so, a couple of years at Citrix. And, then the move to Microsoft. How did that come about?

So you have to wind the clock back a little bit. Microsoft came to Cambridge in 1967, and Cambridge was their first overseas lab, so quite an experiment for them. And, they appointed my old boss, head of department and PhD supervisor, Roger Needham, as the founding director. By then he had been university Pro-Vice-Chancellor of Research, so there wasn't really a role for him in the university to go back to. And Cambridge forces you to retire at 67, and that was on the horizon. Microsoft came along. A lot of the people he had worked with in DEC and Xerox by then had ended up in Microsoft, Microsoft Research had collected a lot of those people, as, as DEC

and Xerox disappeared off the map. And, in fact it was them who had recommended that Rick Rashid, who headed Microsoft Research, and to Bill Gates, that, you know, you should a) to Cambridge, and b) you should get Roger Needham to run it. And, Roger wanted me to be his deputy. The, the irony is, he rang me the morning after I had closed the funding for Digitivity, offered me that job. And I said, 'Roger, if you had done this 24 hours earlier.' [laughs]

It would have been a lot easier.

It would have been a lot easier. So, yeah, I, I couldn't walk away from the Digitivity by then, so I had to stick with that. But there was that kind of, damn, you know, getting back into research and being with Roger would be great. So I did the Digitivity thing, which I thoroughly enjoyed, no regrets about that at all, and, and Citrix. And then, well sort of five years on by then, Roger rings me again. Someone had been his deputy, a guy called Derek McAuley. It's a lovely dance actually. Derek wanted to run a laboratory, he was sort of pushing and shoving for Roger to retire, which, Roger didn't want to. I had been approached by Marconi, who wanted to open a lab in Cambridge, which was a telecoms lab, and I'm a systems guy, and Mac is a telecoms guy. Roger got wind of this, and he knew the Marconi people. So he spoke to them and said, 'Look, I've got a guy who wants to run a lab who does your kind of stuff, and there's a software guy you're talking to who will be much better off working for me. Why don't we just sort this out?'

And what happened to Marconi, that was probably not a bad result, from your point...

For me, personally, it was a very good result. And Mac has survive quite well, he's a professor at Nottingham enjoying what he does, and very happy. But yes, this was all done over the space of about three days. I remember Roger rang me up and said, 'Look, I know you're talking to Marconi, but why don't you come to me and they can have Mac and I'll sort this out.' And I said, 'Well this time, [laughs] the answer is, yes.'

[49:07]

So Microsoft, a very different company to the ones you had worked for previously I would think.

[hesitates] Yes. And, running a, an overseas research lab in an organisation that is very Redmond-centric, had a positive and negative. The negative is, you know, you're not at the centre of what's going on, so, you can sometimes not, not be prepared when things are happening. You're not necessarily visible, so when it comes round for budgets and things, it's a bit harder. On the other hand, you can fly a few kites before they notice. And, one of the privileges I had in running that Microsoft lab was to spin out some things that would have been impossible to do in Redmond, because people would ask too many questions about why you're doing it and what the short-term benefits were, which I didn't know but it seemed like a good punt at the time. So, one of the things I'm most proud of setting up in the Cambridge lab is a whole computational science group, looking at how computing has been changing science as we've got into modern computing and machine learning and simulation and, you know, what you can do with databases, and so forth. And that's, that's been very exciting. And it's certainly had benefits for Microsoft as they've grown, a healthcare business, and how they play with pharma companies and so forth. And I also started a user interface group who were very focused on the way gadgets and social media were changing society. And again, that was a big punt in 2003, 2004 or 5, when I did that. That would have been really hard to do in Redmond, because I'd have had to have persuaded a lot more people.
[50:57]
Running the Cambridge lab, Rick's model was very simple. It's your lab. Here's your budget. You know, stay within the law, do what you think makes sense, with the resources you've got, and what you can tap into, being in Europe, and where, you know, there's a different talent pool, and there are different questions and problems that interest people. And his, you know, his only requirement was, if there's a problem, he wanted to hear about it from me first, rather than someone else. And it was great, you know, if there were problems, you know, he would provide me with air cover. There are sometimes conversations of, 'OK, if you really want to do that, you know, you're more than welcome to; it's your head in the noose, but I trust you.' [laughs] And that was a great relationship. So I had a huge amount of autonomy with that.

Mm.

[51:41]
So I really enjoyed it. I enjoyed being in Redmond, and, research was held in very high regard by the senior execs, and it was Bill and Steve Ballmer at that point, the people heading the, the big product groups, Windows and so forth. So you had the ability to talk to, you know, really senior, influential people in the company, and, help them set direction. I, I really enjoyed that, and where that kind of finally ended up was, in the last few years of my job, as well as looking after research, I worked for a man called Craig Mundie, who was the chief technological officer, and we were doing technology strategy and policy for the company, and I was kind of looking after the, the EMEA end of that, so lots of talking to, to governments and other companies and things. And I found that absolutely fascinating. So, yeah, I got to play with the big boys in Microsoft. Didn't have to get involved in the turf fights about products and budgets and all that kind of stuff, which was quite nice. So yes, I really enjoyed it.

[52:49]
So is Microsoft engineering driven, marketing-driven? Is it a nice balance between all the imperatives?

[pause] It is actually marketing-driven, but it holds its engineering to a very high standard. So it's marketing-driven in the sense of it knows what markets it's going after, but it has an arrogance that it knows what technical solution that market wants. And that sometimes comes unstuck. So, it's, the answer is, both actually I think.

Mm. It's interesting, because, obviously Microsoft and the PC, democratisation of IT, huge benefits to millions, tens of millions of, hundreds of millions of individuals around the world, and yet, sometimes, in the earlier days of Microsoft, I felt it's perhaps the end of computer science engineering you know? Because all of a sudden, end users were writing code. They didn't know how to, so things didn't work very well. Whereas, the kind of 'professional', I put that in quotes, software developers, did things properly and tested things. I just felt, there's a bit of a, an unfortunate consequence if you like.

Yes. It was a, there was a big cultural change in some sense. The industry was de-professionalised. But actually, most of the professionalism was around Chinese armies and project management, and advancing in very very tiny steps to increase your chances of detecting screw-ups and fixing them, and that kind of pushed back on innovation, and doing new things. So when I joined Microsoft, actually it was a fascinating time, because that kind of, home computer mindset of just, write the software and get it out there, was starting to get the company into trouble. There had been the whole antitrust thing; the whole Internet security thing, and viruses, and blue screens, was, was causing the company real difficulties, because serious players had built up big dependences on Microsoft technologies. And that's where research was really able to help. So, I mean out of the Cambridge lab we gave Microsoft a functional programming language, F#, which is state of the art in programming techniques. We did a huge amount of work on software verification. We built a device driver verification toolkit. So rather than device drivers being approved by somebody in Microsoft sort of looking at them with a rubber stamp, they actually had to go through a whole software-based verification system. And suddenly blue screens went away. And in the operating systems world, there's doo cloud computing; much of what we've been doing in operating systems, kernel development and networking in Research, that was leapt on. All the 3D graphics that went ultimately into Xbox and the games systems, that all came out of Research. So the company was receptive to picking up Research ideas when they could see it solving business problems for them.

[55:55]
Mm. I saw the lecture you gave at Wolfson in 2015, I found it on the Internet. There's an interesting piece there about Alan Turing, and, proving, being only the, I think you said at the beginning of the twenty-first century, being able to have software proving techniques. Is that what happened at Microsoft, one of the things you did at Microsoft?

Yes. [bell sounding] Sorry, can we pause?

[pause in recording]

[56:21]

So, yes, in the Wolfson lecture, you talked about, it wasn't until the 2000s that some of Alan Turing's work, you felt became of real practical purpose.

That's right. I mean people have been looking at verifying software using mathematical techniques. Well Tony Hall was doing it at Oxford when I was in Cambridge. I remember Tony coming to visit Roger and myself. But at that point they were essentially pencil and paper methods, and the scale of system that you could analyse was quite small, just, you know, because of the limits of what a human being can do. And, if you analysed a system, and then someone changed it, you sort of, had to start again. And you know, the spec and the system weren't tied together. What's happened in the beginning of this century is something called model checking, which essentially is mathematical theorem proving about pieces of software, whether they're in a high level language or a machine code. And so it suddenly became possible to have a batch of theorems which were statements you wanted to be true about a piece of software, like, you know, access certain data, or it runs in a certain way, and with a model checking system, you could throw the software at the model checker and it would say, yes or no. And the, the innovation that the model checking people put in was to make that scale up to very large systems. So the first practical version I met of that was the device driver verifying system that I talked about done by a colleague called Byron Cook. He now works for Amazon and he's doing verification on Amazon's cloud systems. So yes, it keeps scaling up.

[58:14]

The other... There were two other things that changed the landscape from my perspective at that time. Again, looking at Byron's work, model checkers sometimes can't check everything. There are just some cases they can't crack open. And so it was seen as a limited technique. Byron's approach was to say, well let's ask the programmer at that point what their intentions were, and so kind of embed heuristics so that, yeah, the, the model checker would check what it could formally. Things it couldn't check, it asked for a human verdict. And that, that also meant you could tackle larger and more complex pieces of software, and still have reasonable confidence, even if you didn't have the, the mathematical rigour for every aspect of it. Or, the model checker could tell you what it wanted changed, so you could analyse it

rigorously. And so suddenly, you weren't having to use fancy specification languages, you could work in your productive programming languages, and get confidence guarantees about the software.

[59:18]

The other thing which Byron did, he's one of my best hires, there's a whole list of best hires but he's certainly one of them, was, one of Turing's results was that in the general case, a computer can't prove a piece of software will terminate. It's known as the halting problem. And for years that has stopped people proving that software terminates; although actually in the case of an operating system like Windows, you want to prove that it never terminates, because a termination is a crash. [laughs] People had just avoided that problem, because Turing said it couldn't be done. Byron turned it on its head and said, OK, in the general case you can't do it; what are the special cases and are they interesting? And it kind of turns out, most software written in model languages, in modern operating systems, and all the structures we use, fit the special cases. And that has opened up a whole wave of, of further possibilities. And so, yeah, these days, if you are running a large software engineering project, I would certainly expect to have a battery of, of verification, no longer specification tools but verification tools, to convince me of the quality of the software. And certainly Microsoft invested very heavily in that. And, you know, companies like Amazon and, well one of Byron's colleagues is a guy called Peter O'Hearn, he's at Facebook, you know, across the big venders. And it means that they can now churn out huge amounts of software very very quickly with high confidence it's going to work. And in the modern Internet world, particularly in areas where they're competing, you know, they want to try new ideas overnight on quite big chunks of the community, and they can't have it crash their big clouds. Those cloud computer systems are very vulnerable, you know, in a way all the eggs are in one basket. So it has to be very carefully managed and very secure. It's, it's a mainframe all over again, in a much more challenging environment.

[1:01:19]

Interesting. OK. 2011, you retire.

Play with aeroplanes. [both laugh]

Yes. So, it sounds like you really enjoyed your time at Microsoft actually.

I did.

Endless stream of problems to solve and think about. And a genuine research thread through it as well by the sound of it.

Yes. I mean, problems in the sense of, challenging intellectual things to think about, whether it was, playing with my researchers on their research ideas, and, and challenging them, and getting involved in some of those discussions. And the technology strategy and policy stuff, and talking with, with Government. Oh it's, it's, it's interesting to spend an afternoon with a European commissioner talking about privacy, for example, especially when you're wearing a Microsoft badge. So yeah, there was, there was a lot of fun in all of that. I got to the point where, I think, I am one of the world's starters, not one of the world's finishers. I have kind of accomplished what I wanted to in that world, and it was fun, and I got to level flight, and then I start getting a bit bored. My kids by then had all grown up and left home. And Jane and I, you know, we were thinking about what we wanted to do. And want to spend more time, and do stuff together. I want to spend more time playing on aeroplanes, which is my other passion. So I thought, actually it was a good moment to get out. I had successors in training, and you know, when you've trained your successors, the kindest thing you can do is, get out of the way and let them succeed. So I did.

[1:02:51]
A current interest is the National Museum of Computing I see.

Yes. So, the day I finished at Microsoft I was approached by Andy Hopper who at that point was the head of the Computer Lab that the Cambridge Maths Lab had become. He and various others had come up with the idea of building a reconstruction of EDSAC, and putting it in the National Museum of Computing. A feasibility study had been done that said, yes you could still get the kind of bits you needed. A guy called Chris Burton had built a reconstruction of the Baby at

Manchester, and Chris was willing to be part of the project. So I was asked, would I be interested? And I thought, that sounds like fun. So, I, working through the Computer Conservation Society, and generally networking, built a team to work on EDSAC. Through my network of wealthy Cambridge alumni and Silicon Valley [laughs], raised the quarter of a million pounds that we needed to buy all the bits and bobs. And we might have a machine working some point this year. It's getting quite close now. And, yeah, we, we started the project at the National Museum of Computing. So I got to spend a lot of time there. And, they invited me to become a, a trustee with a view to becoming chairman. The then chairman, good guy, Andy Clark, is a professional IT lawyer, and his caseload meant, you know, he just didn't have time to do the job. And they wanted to bring in some fresh blood. So I, I said yes. There was a pause for a couple of years, I, my first wife died in a traffic accident, and that rather put life on hold for a bit. When I resurfaced, I became a trustee, and about a year later became Chairman of the Trustees, and that's been great fun. It's, it's a lovely museum. A lot of interesting artefacts. So we've got the world's oldest computer, still working, the WITCH, dates back from '53. We'll have EDSAC. We've got the Colossus reconstruction. In the last year the Turing Bombe reconstruction has come into our museum, so we've got all the wartime code-breaking machines. We've got an ICL 2966. And now, we go through to the modern age, with Raspberry Pi's and smartphones.

The PDP-11/34 I saw when I went there. that was a special thing.

PDP-11/34. And PDP-8. I mean we've got so many PDPs stuffed away now, our storage archive. And so a lovely museum with lots of nice stuff. It lives on a, a shoestring. It's, you know, it's totally reliant on donations and revenue coming in through the door. It runs a great schools programme, which has been very successful. Has a wonderfully committed group of volunteers, a lot of expertise. So it's a pleasure to, to work with. We've, we've got more challenges. We need to raise more, more money. We're tenants of the Bletchley Park Trust. If we want to expand or grow the museum, then that's going to need some, some changes. I don't think that's my delivery. [laughs] So yes, I, a lot of my thinking is, is about the future of the museum, looking at ten, fifteen years ahead. Day-to-day operations are wonderfully handled by our manager, Jacqui. I'm typically there a day a week working on

EDSAC, and maybe another day a week doing things for the museum. And it's, it's great fun.

[1:06:22]
So, let's talk about the future a bit. I mean I, I think of myself as being fortunate to have been involved in IT at such an interesting time. So many things happening, so many job opportunities, so many things I could have done, and, maybe I didn't make all the right choices, but I still had a good career. Are you optimistic about the next generation of IT professionals?

Absolutely. [pause] Computers sort of goes in waves, and every now and then there's a trough and you think it's all done, and then there was a kind of, you know, wave at the end of mainframes, and then bang, the Internet came along. And then you had mobile devices, and goodness knows what. So there's just a whole raft of, of stuff. I think, while it's very heavily hyped, what's going on around AI, machine learning, deep learning, there's some really valuable things to be done in, in that space. Wearing my Royal Academy of Engineering hat, I was visiting a company called Darktrace last week, who are applying these techniques to defending systems and doing network security and health monitoring and so forth, which is a completely new approach, and that was fascinating to, to hear about. The, the potential things you can do with big data analysis in spaces like healthcare, a whole batch of areas. So I think there's tremendous potential being unleashed there. Robotics I think are finally starting to get traction outside of, you know, things like manufacturing. So, that kind of gets me really excited.
[1:08:04]
In my own area, programming languages, computer architecture and operating systems. These big cloud computing data centres, making those run efficiently, effectively. Computer architectures, we start looking at different kinds of processors, particularly that handle some of the AI applications where graphics processors are better at it than central processors. How we deal with the challenge that, you know, Moore's law has given up on this now, we can't just cram more transistors on a chip; we've got to start being a bit more... [laughs] I think someone once appraised it as too many transistors and not enough ideas. [both laugh] We have now used up all the transistors, and so, you know, the ideas have to, have to flow. So there's, there's a lot

of interesting things, I think, going on in, in that world, which, you know, I think are equally fascinating.
[1:08:57]
The, the other thing which I find very compelling is, with those computing resources, we can now simulate things that lets us do science – lets us do experimental science, faster than we can do it in the laboratory, let alone in the real world, and lets us, in some sense, do impossible science. And this, this was a theme that came out of the computational science work I started in, in Microsoft. In fact I, I've remarried in recent years, and the woman I have married [laughs] is one of the scientists I hired into that group. That wasn't quite on the agenda at the time. What she is doing, she calls executable biology, she essentially, she's interested particularly in cancer, she models the oncological signalling that's going on between the cells in a cancer and the surrounding organ, as computer programs. Because you can think of them as little state machines sending each other chemical messages. So, she maps the biology. She talks to experimental biologists, and takes their knowledge, and turns that into, effectively computer programs. So, a very high level abstract kind of programming language she uses. And then, she can do two things. She can run simulations, and see how these things evolve, or, she can throw model-checking at them and prove properties of these things. And, so... And one very nice, very sort of high level view, is, if she models a treatment, and a cancer, and does a parallel composition of the two, and proves termination, that says that treatment stops that cancer.

Mm.

Now, being able to do that, when you start adding in what we're getting from the world of, you know, big data analysis, and personal genetic profiles, the possibility of a sort of clinician's assistant that does very personalised treatment regimes for you, and is running a simulation of you ahead of time, sort of, you know, a digital avatar, is a very exciting prospect. And that idea comes across a whole raft of, of places. So I think, you know, computer simulation.
[1:11:19]
And you know, the games world is driving a huge amount of that. Big data and machine learning, and, novel architectures for large-scale computing, are very compelling new directions. And they've changed the face of computing.

[1:11:34]
The hardest thing I think, in terms of motivating, particularly kids for careers in computing, is, all that stuff is in the woodwork these days. It's the other side of the Internet. They see their tablets and their phones, and don't realise how much stuff is propping up, you know, men buying their music from Amazon, or, playing their network game from their Xbox or their, whatever the Sony one is, PS4 at the moment. They just don't see all that stuff behind. It's kind of disappeared. And so I think an important part of, and it's what we're trying to do without education in the museum, and what other people are doing through things like the British Computer Society's Computing at School's programme, is, expose the kids to what's going on behind the gadgets they're used to, and the opportunities in that space to be part of it.

[1:12:26]
Mm. Do you have any fears about security? Do you think we're going to... Because it seems to me sometimes we're a bit on a knife edge you know, it's OK, but it's kind of, bubbling around.

Yes. And there are all kinds of, concerns about our dependence on technology. But no, it kind of goes all the way back. We're very dependent on power stations.

Yes.

All the UK gas comes ashore at Bacton.

Mm.

Without gas, Brexit will be the least of our problems. [laughs] So... As a species, we have become very dependent on technological infrastructure, which is exposed to risk. Certainly there's a lot we can do in computer science itself to minimise those risks, the verification of software, some of the, the hardware protection techniques, the kind of stuff Darktrace are doing in monitoring systems, are all good technologies to deal with that. And it's, you know, it's as it had ever been. You have to have defence in-depth, and you know, some of those defences are, are going to come through law and regulation and policing by Government. And, and yes, the military are developing

cyber capabilities, because, cyber has become part of the, the warring between nations landscape. Yes, it's, it's big, huge concerns. But I think, you know, we can't go back. And so, part of the responsibility is to give the best tools we can to those who will defend our interests. And I think the same issue is there with AI. The AI community is starting to pick up on the fact that they need to develop ethics, and people are talking about that. And, that, you know, in understanding those ethics, they've got to think about issues like bias. A lot of big data is, is coming unstuck because the datasets are biased. For example, many exclude people of colour or females. And so your face recognition for white males is there and done; other people it's not so good. So we've got to fix biases in the data, that's a kind of ethical issue. When you come to robotics, and autonomous cars, you've got to look at the ethics there. But I think you have to ask the right question. Everyone likes, in autonomous cars, coming up with, you know, you have two choices, you can swerve left and kill a toddler, or serve right and kill a grandma. What's the right thing to do? I think what you have to look at is, if we can make autonomous car systems work, and reduce the total number of accidents down from 5,000 a year to 1,000 a year, then, you know, it's unfortunate if you wipe out the toddler or a granny, but if by going to autonomous cars you reduce the number of deaths overall, that's an improvement.

[1:15:25]
Mm. Yes, well thank you ever so much. I think we've probably come to the end of our time. I'd just like to thank you again for talking to the Archives, and, one final thought from me. You've obviously done a huge amount of thinking and research over the years. Is there one particular idea which you just wished had lasted longer, or perhaps a missed opportunity, anything you regret not carrying through to the end?

[pause] One of the things I regret that didn't really find the light of day was, when we were doing the, the ANSA work, we developed a programming language for writing distributed applications, and it was object-oriented. If we had a following wind, it could have been Java. [MJ laughs] But I think with a lot of these things, you know, the timing and where you stand are as important as the idea and success. And we were early, and we weren't standing in the right place. But I'd love to have seen that

be more successful; that's, there's a frustration in that. It was a lovely little language we designed, but, it never really got to change the world.

Good. Well, thank you again. Thank you very much.

My pleasure.

[End of Interview]