When I heard I had this assignment you were the first person I thought of because that discussion we had last week where you said you were coding in C back in the day and doing assembly language stuff, that was just so exciting to me and made me think, man, you must have seen so much over the years in the IT industry. I was wondering if you could tell me a bit about what it was like in the early days, how things took off, and what you've seen and done?

Well, I started my career working for IBM straight out of university, so I was involved in systems development for IBM and that really required us learning some low level languages early in the day. So that was assembler, there's some languages there called PL1, just a step up from what assembler is. So I sort of did that for 3 or 4 years and then straight away got into the consulting game, and I went to work for, they're now called Accenture, but at the time they were called Arthur Anderson. They went bankrupt in the Enron process. Part of that you start developing systems for clients. So I found myself part of teams which went out and developed systems for banks, which went out and developed systems for government agencies. When I was back at Arthur Anderson we developed a very big system called COPS for police which is still in use today, for example, right, and we actually developed that ground up, from scratch, and I was on a project with over 100 people. So these projects tend to be quite big in terms of the sort of systems that I got involved in. You find yourself in projects that involve 20 or 30 programmers and just as many testers and management. So my career went from there to the point where I started managing techs more than anything else. So you go from the technical resource into the management of the technical resources, and that continued for me when I went to Price Waterhouse which is another big 4 consulting firm, and I was doing much of the same thing. I mean my career at the time sort of spanned more into sales as well, but also what we call delivery, which is the management of those technical teams and the delivery of those solutions to the clients at the time.

So you went from uni to coding and then almost straight away you were supervising people and implementing projects. It sounds like you were riding the wave of IT entering into industry and business.

At the time I probably only coded for around 3 to 5 years and then I went into the management part of technical resources. I've had different jobs where you are either managing large scale projects or you're getting involved in line management of IT solutions, so the ongoing operational maintenance of them, for example, and management of the resources that look after the systems once they've been built. But at the time it was the first time people were talking about GUI, graphical user interface technology, so C and C++ which was the first real object-oriented language at the time. So object-oriented was a big push in technology at that time and developing a solution for what was at the time OS2 or Microsoft Windows was C but underpinning a different sort of graphical user interface language as well. So when I started it was just in that transition of moving from your traditional batch programs that were taking millions and millions of records and consolidating them overnight, to more transactional-based GUI transactions. That was a big change in technology then. At the time there was a real push to get off mainframe technology, if you look at assembler and you look at the history of IT it was all about mainframes. IBM got big in the world by selling mainframe computers. That's how they did it. And when you're talking about mainframe computers, you've still got Westpac today running mainframe computers for example. Client server technology was a way of getting all that technology off mainframe computers and on to the PCs because the PCs suddenly had all this processing power and no one knew what to do with them. "Oh we better go for some sort of distributed model", where you've got your data in your core systems and you've got your actual application processing taking place on the PC almost. So lots of changes have happened in the industry in the time that I've been in it. Ironically these days what you see is almost a move back to that mainframe technology in the sense that everyone is talking about cloud these days. So again they are talking about centralisation of data, talking about centralisation of functionality, taking that off premise.

Yeah I was just reading today about a financial institution in the US whose gone 100% cloud, and I was wondering, you just said Westpac is still on mainframe technology, does that make you say Westpac is a bit backward?

Yeah, well banks are very, very complicated IT shops, right? And it's because, historically, in a bank, every product that they've ever sold constituted a separate IT solution to support. So if you go inside a bank, the mess that you're presented in terms of the architecture underpinning all of the products they sell, it takes ages to get across. So when I say they are still on mainframes, what I mean is, there are still mainframes being used, but that doesn't mean their entire footprint is on mainframes. They are obviously trying to get off that and trying to get into the cloud, and put a lot of the data back to the wide usage rather than just what they call 'on-prem'. So mainframes are still around but they're trying to get rid of them again.

You see this irony in the IT industry where first of all everyone was on mainframe and they thought "Well no we can't, we've got to distribute all our computing power because we've got all these PCs", so they started pushing it all towards PCs, developing all the technology for it. But now, it's almost going back the other way, they're not going to mainframes, but they're trying to get the technology, the processing and the data off PCs, back into the cloud. So it's almost going back the other way. I mean I've just got off a meeting now where we're trying to cloud-base a whole bunch of SAP solutions. So you see the big renders, the big ERP service providers, you know that term Enterprise Resource Planning? SAP is probably the most common system used by enterprises today across the world. German-based solution, which is an ERP system, and what they're trying to do is basically take all the standard functions of what an organisation does and deliver it in packaged software, so that organisations don't necessarily have to grapple with the heartache of doing a lot of bespoke development themselves. They're essentially saying that "we've done all that for you, just buy our products." Those big vendors today, Oracle and SAP being the two biggest, are all cloud based, it's very hard to buy one of those solutions which is 'on-prem' anymore. So they're doing everything in the cloud, everything. So I've been involved in the development of technology and the management of technology and also the sales of technology.

Yeah, across every aspect of it by the sounds of it!

As you go further you get into projects which involve IT as only one element of the project. For example I did a job at NSW Treasury where I was Overall Transformation Director of all their IT solutions, but all their business processes as well, underpinning the technology, which involves structural change to the organisation. Different roles, different positions being implemented as a result of the technology. Huge consequences to the way they are reporting to government. Huge consequences to the way they are receiving information from other agencies, so the reporting of budgets and things to them. These days, the projects that I run these days, IT is sort of more of a smaller element, and I'm spending more time in the business transformation space, rather than I am in the IT space. But it's always there, it's always in the background, and what you're generally trying to do in these large scale programs, is you're trying to minimise the amount of change you are doing to software. You are trying to ubiquitise every organisation out there as best you can because every time they make a change to their software to accommodate some requirement that they have internally, that constitutes some kind of ongoing cost for them, some sort of maintenance capacity which becomes a nightmare for them. The guy that knows that code goes somewhere else and he takes that knowledge with him and it's not documented properly. They will be teaching you all about how to document the code that you're coming up with properly, but no one ever does out in the real world. One thing I'll tell you is that the standards you are learning there in university never really get applied to the degree which your expectations are set at at university. Never. There's a lot of cowboys out there.

When you're talking about ubiquitising the software, you're really thinking about the client there, and not having any ongoing costs. I imagine there are some operators out there who want to have clients under the thumb and have them coming back because that means guaranteed income stream.

Yeah, when I say there's no ongoing costs I mean there's no ongoing development costs. You will be paying a licence fee to that software provider. So if I go out and buy SAP today I will pay an annual subscription to them for the use of their software. I will get some sort of enterprise licence and I will pay them that in perpetuity while I'm using their software, but the moment I want to modify that software to do something which specifically my organisation wants to do I've got some choices about how I deal with that situation. I can either ask SAP to deal with that and get them to maintain that code themselves on my behalf or I can have someone internally to actually tailor it 'on prem' as we say, on premise, which has its own raft of challenges as well. What businesses are generally trying to do these days is reduce their IT costs as much as possible, which generally means not having to maintain a whole heap of software. That's what they're trying to do, and push it back to the vendors because the vendors offer pretty good pricing for long term sign ups. So there is a business case that organisations are following. And in that sense the project work becomes more interesting than the operational work. In IT you can talk about a project, you can talk about all the steps that are required to get a piece of software live and then you can talk about all the stuff that needs to be looked after once it is live. The project is interesting and requires people and that's where I play more than anything else these days.

Okay so thinking about your average week, what do you spend most of your time doing?

I'll give you an idea, I've just got off a meeting right now where we're looking to move a whole bunch of integration software, third party integration software, to the cloud, specifically that. And so it was a whole heap of architects presenting to me, so I was effectively spending that time challenging the basis of why they want to do it. So there is a lot of my work which is about on-the-fly business cases if you like. A lot of my work is "Why are we doing this?", "What are the benefits we hope to achieve from it?", "Does it make sense?", "If it does, how do we get approval for it and how do we actually fund it?" So I need to have the technical knowledge to be able to understand what the hell they're talking about, but I play much more of a oversight role these days. Much more of a "Why are we doing this, is it worth it?" Every little IT initiative has to stand on its own legs in terms of why you are doing it for an organisation. Especially, my client at the moment is a government client, I'm a freelance management consultant right now. I'm a contractor, I don't work for them, but they're my client so a lot of what I have to do is convince New South Wales Treasury that the money we want to spend is justifiable and will deliver a benefit back to the State. Banks have similar processes but they're not quite as rigid as government processes.

So I do a lot of that sort of work, but then I also do a lot of oversight or management of projects in flight. So I'm a program manager in there so that means I look after something like 12 projects, each of which have project managers which report to me on a weekly basis really, about all the issues that are going on in their project. So I attend all those steering committees, and a lot of what I do is stakeholder engagements, so if there is a person in the business at a high level. The other day I was talking to the Head of Corrective Services, who are one of my clients. Corrective Services being gaols, and I'm on the steering committee to try and basically explain some of the problems we were having and explain some of the risks associated with what they are trying to do. So I do a lot of that top level stakeholder engagement as well.

How do you go explaining the really technical stuff to someone like that who might not be very technical?

That's a good question because as you go up you lose the breadth of technology that you once had. I'm no longer a specialist in any technology. I used to be able to program C. I probably still could but I certainly can't talk about complex integration patterns or APIs or infrastructure. There's a lot of things that I understand only in a certain level, so if I ever have to go to a meeting like that and explain the depth of that, first of all it's pretty rare that you actually have to explain technology in such detail at that sort of a meeting but if I do I always take a very good architect with me.

So that's an interesting point, you talk about programmers, these days they're called developers, they're not called programmers anymore. And architects are the kings. So architects are essentially your system designers and architects are in hot demand because design is recognised as something which is far more valuable than actually doing the programming. These days the development is something which happens a lot quicker than it used to. It all comes back to having the right architectural advice and design in place. There are two types of architects, there's an enterprise architect that says something like "OK we need a system to do tax returns for the courts, or something along those lines" and he will work with the business to put something in place for that. And then there are solutions architects which look at the thing on a much lower level. They'll say "OK if we're going to build a system like that, we need that module, that module, that module and that module, therefore we need 3 or 4 programmers to go and do that." So it's a level thing, right.

Yeah I see it, so you've got your big visionary and then you've got the person who solves the problem to make the vision happen, and then the programmers are under that just getting it done.

That's it, so if you're doing an IT course, architecture is more in demand these days than just being a developer. Although being a developer is more fun.

Ok, good heads up. I know you said to me last week that if I was fully specced up in HTML you could get me a job just like that.

Yeah, because there's a lot of demand for HTML, especially in government. There's a lot of demand for Pega developers. There's a lot of demand for customer relationship software. Third party integration tools, like we use one called Milsoft. It's very hard to find specialist resources these days. Everyone is so specialised, OK. But yeah HTML is always in demand.

So what sets apart a good employee and a not-so-good employee in the IT world these days?

Oh god, that's a very interesting question. Well a good IT professional is one who doesn't, I hope you don't mind the expression, one who doesn't bulls<>t. A lot of, I can tell, even though I'll talk to people who understand the technology a lot better than me, but I can still tell when they are trying to tell me a fib, I can tell. It just comes from having been a technician at one point in my life. You'd be the same in your job, you'd be able to tell if a junior was, exactly the same right.

So a good IT professional will always tell you when he's got a problem that he can't actually solve himself and be open and transparent about it, instead of necessarily trying to hide it. A good IT technician is one who will cover all bases and highlight where the design is insufficient, where he actually thinks that they could do better in terms of tightening the software. A good IT technician will test his own software before he hands it over to someone else to test. That's a big one, that last one is a big one, because so many developers who are a bit gung-ho, they write exactly what the spec says to do and says "here, it's your job to test it". And that's where in testing, you need very good testing and you need very good testing processes, but you can minimise the amount of time software spends in testing if the developer is a bit more cautious about these things upfront and actually highlights "well ok I know the spec says this, but I think if we do it this way, this may make it better and will make it a bit more robust in terms of the amount of bugs it's going to throw up or the amount of errors it's going to create in production. All software has errors. So coming back to my first point of "I'm not trying to pull wool over your eyes", if a programmer comes to me and says "I'm finished" and I say to him "Right, how many bugs did you find?" and he says "Nothing, we didn't find any bugs" then I'll say "You're lying to me", or I'll say "You haven't even looked" because all software has bugs. All software. It's just a question of how many of them do you find. Even production systems running your bank account applications today have bugs that they know about, and some they don't know about, it's just the way it is. We have very rigid testing processes these days to cover those things.

Yeah I was very thankful to my daughter this week, just before I handed an assignment in, it was a children's-based adventure story, and she tested it and found a bug an hour before I was going to hand it in, and I was very thankful.

It sounds like you're positioned right at the heart of the modern IT industry. You're transitioning all these government departments and businesses and catapulting them into each step of the modern IT industry, is that fair enough to say?

Yeah look it is, but it's not always so glamorous. You may or may not have heard about 4 or 5 years ago Commonwealth Bank was engulfed in an anti money laundering scandal that they had to pay zillions of dollars. I was right in the middle of that. I was working for Commonwealth Bank at the time and I was running one of the projects that was actually fixing some of the issues that they were dealing with. I can assure you that's not glamorous. High pressure work that you have to do as a result of a government directive. A lot of the work you do as IT is non-sexy work, trust me, a lot of it is boring stuff. I think a lot of IT professionals have that expectation of sort of coming out and always being able to work on new systems and always being able to push the boundaries of where technology is going and all this sort of stuff. That's a lovely dream, and sometimes you do get to do that. I've developed some wonderful systems over the years, but it's not always like that.

I developed a system for New South Wales Parks and Wildlife years ago when I was working for Price Waterhouse Coopers. It was about the controlling of the culling of kangaroos in Western New South Wales. It was a wonderful thing to work on. It was something that really helped the situation in regards to farmers and what they were trying to achieve, and at the same time we were using some pretty innovative GPS and space technology to actually come up with the solutions. So very often it's pretty exciting but very often it's not, OK? Very often you're doing work that people don't want to do.

Yep, ok, so when the Commonwealth Bank was coming down hard on you, you were thinking "I wish I was still doing the kangaroo stuff!"

That's right. That's the hardest I've ever worked in my life, that particular job, because the pressure was relentless. There was regulators all over our backs and they had to deliver results very, very quickly and we were pushing the boundaries in terms of hours we were working. I've never worked more hours in my life. At the moment my job is very flexible, very flexible. I'm not really working that many hours at all to be honest. But I'm a contractor, which means I only get paid for the hours I work. So that's another thing for upcoming IT Professionals to think about, whether or not they're more interested in contract work or permanent work. I've done contracting for the last 20 years, ever since I left those big companies. I've just been a freelance consultant.

How do the architects react when they come to you with their big plans and their ideas and your job is to be devil's advocate and really question it? Are they a bit shocked?

Oh yeah yeah, they get annoyed. I mean architects are classic personality types that expect the organisation to just spend money on their whims without necessarily the demonstration of results. Architects can come up with some wonderful ideas that can really do fantastic things for the organisation but very often you'll get architects being a bit surly about their work, "Oh why do I have to explain that to him" and this sort of stuff. You get a lot of attitude from architects. They sometimes think that they simply know everything about the organisation and the organisation should just do what they want them. So there's a fair bit of pushback on them that you basically have to do. I've rejected 3 or 4 things in the last month that they've come up with I suppose. And when I say rejected I don't mean to imply that I've got all the power to reject it, but I've sent them back to the drawing board to start again a few times. And yeah that can ruffle some feathers but that's part of the job. Part of being manager of any IT project means being a bit tough about things. You try and get on with people as best you can but there are times when things aren't done to your satisfaction and you need to deal with it. It's just the way it is.

And what about treasury, how do you go interacting with them?

Now?

Yeah.

So I used to work for treasury, I was Transformation Director there. I put in place some new systems that they're currently using to manage the New South Wales budget. I was working for Mike Baird at the time. He was the Treasurer at the time. Now I deal with Treasury, so now I'm not working with Treasury, but deal with them. They're a lot easier to deal with than they are to work for.

I've worked in the UK, I've been a CIO which is a Chief Information Officer in a software company in England. So that's a very different focus. Rather than going out and developing software for clients, you're the guy back at the ranch trying to develop software which then your sales people will sell to a wide variety of clients. I was involved in a company in the North of England doing that, who made all sorts of software for supermarkets over there. So I basically ran the development team, the development of the product, got involved in some customer implementations but was I more of a back office boy there, who was involved in the development of the software rather than the customisations of it for the clients.

I worked for Microsoft for a while. I have worked for some big names in my life. I'm lucky.

I would love to see your resume, it sounds amazing!

You're welcome to see it. Microsoft, it's not the sort of sexy job you can imagine though. I had a job which was about contracts, so I got involved in IT contracts. A lot of "OK, we've done the deal, let's put in place a contract with this client." And IT contracts have a language unto themselves. A lot of it's about "Well who's going to own this software when it's finished, and how are we going to deal with the intellectual property that we actually develop?" and those sort of things. There's a lot of nuances to an IT contract. So, I've done a bit of that.

Working the UK was good. Very, very good. I enjoyed that.

Alright Paul this has been just amazing. The amount of content you've given me, it's just so good. I cannot wait to write the report up about this, and I'm so grateful for your time. Thank you very, very much. I'll let you get back to your afternoon.

No worries