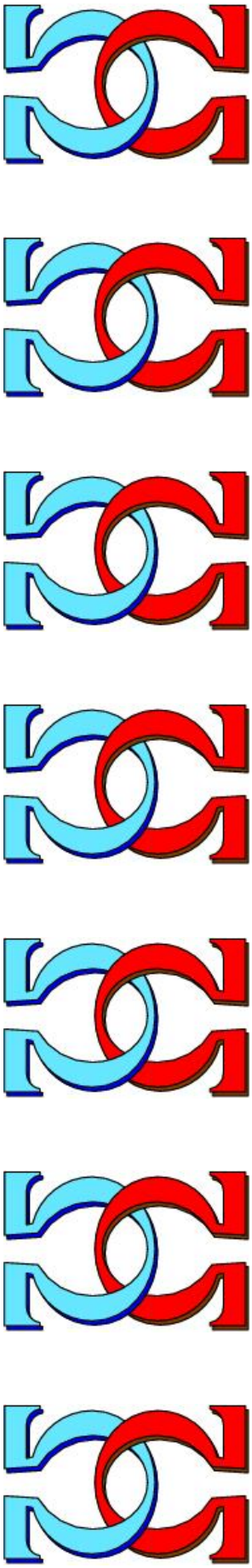

CDMTCS
Research
Report
Series

An Overseas Experience with Hypertext and Packet Switching

Peter Cashin and Brian E. Carpenter

School of Computer Science, University of Auckland, New Zealand

CDMTCS-577
February 2024

Centre for Discrete Mathematics and
Theoretical Computer Science

An Overseas Experience with Hypertext and Packet Switching

Peter Cashin and Brian E. Carpenter

February 2024

1 Introduction

This report records an interview of Peter M. Cashin by Brian E. Carpenter. This interview was conducted by email during February 2024.

Peter Cashin was born in Liverpool England, in 1944, and his family moved to Christchurch, New Zealand when he was 5. He says "All my schooling was in NZ, and I think of myself as a Kiwi. NZ is my home, and I have a place here, even though I am now living back in Canada". He graduated from the University of Canterbury in Christchurch. In 1970, he obtained a Ph.D. in Electrical Engineering there. From 1970 to 1974 he was at the National Physical Laboratory (NPL) in Teddington, England, working on the pioneering Scrapbook networked hypertext system [2]. From 1974 his career continued at Bell Northern Research in Canada, where he contributed to the DATAPAC network [1] and much more.

2 The Interview

How did you first get interested in computers and programming?

In Engineering School at Canterbury University. My first computer program was a class exercise, written in Fortran and run on an IBM 1620, it must have been around 1965.

Later for my PhD studies, in the early days of AI with Prof. John Andreae as my supervisor, I did lots of work on a digital computer in the electrical engineering department that was part of a hybrid computer. I had a lot of fun developing a visual display with my own fonts by writing to an oscilloscope. The scope had a

storage screen that kept a visible trace of the beam, so I could write anything, but I used it mostly for a "page" of teletype text lines, that could be quickly erased to read the next "page".

I don't think there was any real operating system, and my programs were in assembler code, so I learned a lot about how computers really work at the very lowest level. From that time on software and computing became an obsession that continued through my working life and on into retirement.

How did you end up at the NPL? Was that intended to be a short "Overseas Experience" like every young Kiwi should have?

Yes, exactly, travel to UK was for an OE! I was very lucky to get a job at NPL, and was very happy working there. I could have stayed in England for a lot longer except that the pay at that time was minimal, and buying into a house anywhere near London looked to be impossible.

As it happens, I visited NPL in 1969 while I was a Ph.D. student at Manchester, for discussions about speech recognition with somebody whose name I've long forgotten. I thought it was a much more interesting environment than the Computer Science Department at Manchester. What was your first impression of NPL?

It was my first real job, and I didn't know what to expect, but the work and the people were great. I was lucky that Mike Woodger became my mentor, he was very kind to me. I was fascinated to learn about his Algol 60 committee work with many of the great names in computer science. And hearing his memories of working with Alan Turing was amazing.

Did you have a lot of contact with Donald Davies? What was he like as a person?

No I didn't have much direct contact with Donald Davies, although he did my final interview, and gave me the job. He made sure I understood that I needed to switch from AI research into software for packet switching, and I was happy to do that. I remember him as a real leader, he had a clear vision and a quiet confidence that earned him great respect from everyone.

Same questions for Derek Barber. I think he was still at NPL then, although working on the European Informatics Network.

Yes, I remember Derek Barber, although I didn't ever work with him directly. It was his team with Roger Scantelbury and others that built the NPL packet network. They were on the next floor up in the same building that I worked in. The network was up and running when I arrived in 1970, but it had no real users. I think the Scrapbook project that I worked on was the first big user application.

You worked with David Yates and Malcolm Robinson. I'm guessing that was your first experience of working with Poms. How did that go? That was before the UK joined the Common Market and dropped Commonwealth preferences. Many New Zealanders still referred to the UK as "home". Did they treat you as a mere colonial?

Yes I was definitely a "colonial"! But that also worked to my advantage. I was given the benefit of the doubt whenever I didn't know the rules of the establishment!

I traveled all over UK and Europe, and saw many things that I had learned about in school. Very much the traditional Kiwi OE. Yes, I do remember that people used to refer to UK as "home". But I was a Kiwi, and I never felt England was my home, even though I was born there.

David Yates was a very polite Englishman, mild and reserved. He set the broad objectives, and just let us get on with it. Not that he wasn't involved, we had lots of deep discussions with him, but I never felt we were being "managed". It was more like a university research department.

I worked very closely with Malcolm Robinson, he was a new graduate employee too, and we became good friends although we lost touch after I left NPL. He was an Oxford (or Cambridge?) graduate. He was very bright, and easy to work with, but I don't think he had any experience with computing before he joined NPL.

Had the Scrapbook project already started, or was it just an idea when you arrived? Were you already aware of the concept of hypertext, or of packet networking for that matter?

When I arrived, Scrapbook [2] was only a general idea to build a computer application to run on the NPL packet switching network. David Yates had the vision of an easy to use information retrieval system, and he built the team, but the design and implementation were a blank slate.

Malcolm and I knew nothing about packet switching. But we quickly learned a whole lot about the NPL packet network. I became a total datagram enthusiast! The elegance and simplicity, and the flexibility to work into system software seemed so natural. I couldn't imagine any better way to design a computer network. But we were end users of the network, we were not involved in the NPL switching software itself.

How did the Scrapbook design process go? Was it a group effort or did one of you take the lead?

It was a design team of two, just me and Malcolm, working into David Yates. To run the computer we had good help from Vince Hayward.

At that time computing was still considered something that was mostly reserved for numerical work, or control systems. The idea of people using a computer for text editing of their own personal files was still quite novel.

David Yates wanted Scrapbook to be for everyone to do anything, over the network. He saw it as an information system. It turned out to be quite disruptive when the researchers started to write their own reports – there was a whole organization of administrative support with typists who were not so happy.

Scrapbook used URL-like links in the form

```
*<3==doc/scrapguide>*
```

Do you know if they were copied from an earlier hypertext system? Did you have contact with other groups working on hypertext?

I did have some knowledge of hyper-text. I was lucky enough to have visited Doug Engelbart at SRI. It must have been the late 1960s. I was just a PhD student then, but he ended up talking with me for most of an afternoon, and then invited me home for a meal at his house.

I was really interested in his NLS system, and of course I got introduced to the mouse that he had just invented, one with binary chord coded buttons.

I don't remember how we came up with the Scrapbook hyper-link text format. I do remember we put a lot of thought into how hyper-links should work, and we never really resolved some of the issues. The links were obviously references, but how exactly should the name-space be resolved, and how fine grained should the link targets be, and how could a user distinguish immutable from mutable references (such as the latest version of a document)? We made a lot of pragmatic decisions to make it as easy as possible for people to use them, and I think that worked. But I know there were a lot of fundamental issues that we thought about, but never managed to properly resolve.

Scrapbook was coded in BCPL on CTL Modular Ones. Who made those choices, and why? (By the way, the homebrew CERNET network at CERN, developed in the late 1970s, also used BCPL, but on Modcomps.)

The decision to use a Modular-1 computer had already been made when I arrived, I always assumed it was probably a political decision to support UK computing industry.

I think the decision to use BCPL was mine, but that was an easy choice. There was no C compiler, it was before the C language existed, and the debate about the efficiency of a programming language against assembler language machine code was mostly won by then. We needed a systems programming language

and the Modular-1 had a BCPL compiler, so the only real choice was between assembler or BCPL.

Both Malcolm and I had to learn how to program in BCPL and to use the Mod-1 computer via paper tape.

By the end of the 1970s, the idea of OSI-like layers was well developed, and discussed at length by Davies, Barber et al in their 1979 book "Computer Networks and their Protocols" [3]. But in 1970, did the network provide any well-defined programming interface (what we'd call a transport layer API today)? Or did you have to code network operations at the packet level?

I can't really remember, but I know there was a lot of discussion about layers of abstraction. I can remember Mike Woodger thinking about that in terms of how to structure software. He talked about the concepts that could and couldn't be expressed in layers that were built on top of each other.

The NPL network team must have done a good job because I don't remember any real problems with using the network. In essence we had to build our own Scrapbook operating system, and I think we must have used a quite low level datagram interface, but it worked very well and interfacing into the network was not a big problem at all.

How was the coding work shared between you and the others? Did you also work on the packet network code itself?

Malcolm and I did everything from conceptual design on a black-board through to writing and debugging the software, it was all part of the job. But we did have Vince Hathaway for tech. support to help us run the software on the Modular-1. I don't think Vince was involved so much in the conceptual design of Scrapbook, but he knew a lot, and it was great to have him as part of the team.

No, I was not involved in any of the software for the packet network itself. The Scrapbook development was a user application running over the network. The switching software was running on a Honeywell 516 on the floor above us as I remember.

What else would you like to add about Scrapbook?

Scrapbook was designed to be an information retrieval system. That required a text editor and a sort of data base file system which we built for the purpose. All very primitive by today's standards, but quite new and novel at that time.

The whole idea was to make it as easy as possible for users to access text "pages" that could be used for any purpose with hyper-links between them. This led naturally into collaboration with personal messaging and simple notice-boards

without any special extra design.

As more people started to use Scrapbook we ran into early examples of the social impact too. For example, I had some work files that I thought of as personal notes, as if they were in my office file cabinet. So I was upset when I discovered that one of the other researchers had been poking through my notes – he would have never dreamed of snooping into my office file cabinet! But he saw my files on the network, and to him that was totally different.

Scrapbook was used for all sorts of things, often for writing reports or sharing ideas. Some people saw this as great, but others saw it as rather trivial, and not the sort of serious research that NPL should be doing.

I left NPL in 1974, but Scrapbook was already in daily use by many people. I gather it grew into multiple servers and spread across NPL. Then it was spun out as a commercial product and ended up with quite a number of users including Shell and some UK government departments, and also the EU.

At some point it was ported off the Modular-1 machines and onto PDP-11s. So the use of BCPL was a big win there.

The UK National Computing Museum at Bletchley Park now has a project to restore Scrapbook. They have a copy of the Scrapbook software, but first they have to restore a PDP-11 with a compatible disk drive. That turns out to be a major feat, even for a team of engineers with professional experience in PDP-11 maintenance.

I believe you were involved in the INWG (International Network Working Group) around the time that the concepts of internetworking (a.k.a. the catenet) were being created by Pouzin, Cerf, and others. Any memories of those meetings?

Yes, I remember lots of fun meetings with plenty of interesting discussions, and great restaurants. I remember quite a few of the personalities, Louis Pouzin and Vint Cerf in particular. It would have been between 1970 and 1974. I know I attended quite a number of meetings and workshops, but exactly how many, when, or where, I can't recall.

Oddly enough I remember one long intense discussion with Vint Cerf on a train somewhere in Europe. We were debating the intricate details of how to design a virtual terminal. I have no idea why I should remember such an oddity.

I must have met Rémi Després too, because I do remember some of the long on-going debate between datagrams and virtual circuits that was almost a feud between Louis and Rémi.

I also remember visiting Vint Cerf at Stanford, and I must have been at some of his early protocol design meetings because I remember meeting Carl Sunshine

and discussing window flow control. I also visited Alex McKenzie and Dave Walden at BBN to talk about the IMP software, but exactly when that was I can't be sure.

In 1974 you moved on to BNR in Canada. Did you also look for opportunities back in New Zealand? (I can tell you that there were no interesting networking projects here then; that's the year I joined Massey University in Palmerston North, and we did start a project called Kiwinet in 1975, faced with the choice between ARPANET-like and X.25-like solutions.)

At the time I saw the BNR job as a stepping stone on the way back to NZ, but as it turned out that didn't happen till 25 years later! BNR was just getting established, and it was built along the same lines as the famous Bell Labs. BNR was an R&D lab to design products for Northern Electric with Bell Canada as the prime customer.

At BNR you designed the first datagram network product for Bell Canada, and it became DATAPAC. Your 1976 paper on "DATAPAC network protocols" [1] was quite widely cited. Was DATAPAC really based on connectionless datagrams despite offering X.25 virtual circuits? Was it a commercial success?

Yes, Datapac was a pure datagram core network. It was more than that because we designed the operating system from the ground up to work with processes and inter-process message sending as the foundation. This was before Bell Canada decided that it was going to support X25.

The choice of hardware was based on an evolution of the SL-1 product being manufactured by Northern Electric, and the customer was Bell Canada, but all the R&D technical design was in the hands of BNR.

The first problem was that the SL-1 processor was too slow, so we needed a multi-processor system. The SL-1 processor was a nice little stack machine with a proprietary programming language, no problems there. But I knew enough, from reading Dijkstra's papers, that without hardware that could support a semaphore a multi-processor system was going to fail. I would have liked to use a commercial mini-computer, but that was not an option. It was a bit of a surprise that it turned out that using our own SL-1 processors was a great idea!

My boss knew that we had to use SL-1 hardware for political reasons, but he was very smart, and when he understood my problem he had a brilliant solution. He simply said: Peter, why don't you design your own hardware message queue processor to allow the SL-1 processors to communicate, and that could help you with the datagram traffic too!

So we did exactly that, a multi-computer operating system with a special purpose hardware controller (with redundancy) to handle queues of message address pointers. The whole system was a datagram machine!

I was the lead designer, and building a multi-computer operating system was a dream job for me. We kept it simple and clean, and it worked very well. Very robust and reliable from the start.

When Bell Canada decided that they wanted an X25 network we simply built that as the external interface on top of the datagram core network. Bell was concerned about the service interface for their customers, initially for the Canadian banks. But they did not insist on an internal virtual circuit architecture. So we just kept quiet about that!

I left the Datapac development after the first systems were running, but there was a lot of development work after that. The product was a success with sales not only to Bell Canada but also in USA and Germany.

I knew that Bell Labs were working on a data network product at the same time, and I was very proud to think we had beaten them to the punch, little BNR was starting to grow.

Then I think you moved on to digital switching, and it was commercialised by Nortel. So that was just in time for the last gasp of POTS (plain old telephone service), on the 20 or 30 year timescale of the traditional telcos. How did it relate to ISDN and ATM, both of which were supposed to be the future at one point?

This was before 1980, and at that time Northern Electric had a computer controlled telephone switching product called SP-1, but the switching network itself was electro-mechanical. The big new opportunity was to interface the digital trunks into a pure digital switching network. This was the start of the DMS Digital Multiplex System product line.

It was long before the last gasp of POTS! DMS was new product opportunity that was a multi-billion dollar product line that had explosive growth, and led to Northern Electric breaking free of Bell Canada and transforming itself into Nortel which eventually became more USA based than Canadian.

ISDN and ATM came later, well after the Nortel DMS product line was well established. I led the original core development team for DMS computer systems. At that time the Northern executives saw "data" as a four letter word, and Datapac was a distraction from the main opportunity with DMS for digital telephony.

It was only much later that ATM was developed to expand digit switching from voice data to computer data traffic. By that time the Internet had shown that it could handle voice traffic. The advent of optical fibre and higher bandwidth came on fast, and the tide had turned, computer data was the business, and packet switching had won!

The reason I worked on DMS had nothing to do with network architecture, as far as I was concerned datagrams were the right answer. But the DMS project was a huge technical challenge. Not just on the computer software front, but the digital switching core and the digital conversion of individual subscriber lines was a huge advance. DMS used proprietary silicon chips for the analogue to digital conversion of individual subscriber telephone lines.

For me it was an amazing opportunity, my team had to design a new non-stop operating system, and we could design our own high level programming language, and the architecture to support telephony switching applications. Everything from the CPU and the firmware for the instructions, through to the top level software architecture. The system also worked with a network of peripheral processors that controlled the interface to lines and digital trunks.

The initial DMS core computer software team was less than 50 people. We developed the Protel programming language and the SOS operating system. Software modularity and source control management were critical infrastructure that we had to design from scratch.

Over the next 20 years there were hundreds of Protel programmers who built up a 10M line code base. There were many different DMS products built on the same software platform, and DMS systems are still in use today.

A huge challenge was the requirement that a DMS system must have less than 30 minutes downtime over 40 years! The downtime is cumulative and includes everything from hardware and software failures through to software upgrades and database migration.

The first DMS system was for trunk switching, which is the most basic telephone application. The first system was delivered to Bell Canada in 1979. At that point my job was done and I moved out of DMS to develop the next generation of computing for BNR.

That was the XMS project, and the idea was to build a distributed computing system for next generation products. The project was technically successful, and we even developed our own in-house computer workstations. We thought we had a better solution than Unix, with nice multi-computer inter-process messaging.

The XMS project grew to about 100 people and we built a lot of new things. Many of the developments were pick up by other product development groups, but the vision of a distributed operating system with a kernel message passing core nucleus never got fully exploited. We also developed a fibre optic token ring, and a distributed fault tolerant name server, and many other things. On top of the message passing kernel we also developed RPC (remote procedure calls), and all the developers had their own XMS software development workstations running as a distributed computing network.

But the writing was on the wall, the use of C and Unix was spreading, and the commercial computing industry was going to be able to supply the needs for telecom. The necessity for our own hardware was gone, and the advantages of our own systems software was shrinking. It was a computer industry opportunity, and Nortel made the decision that it was not getting into the computer business.

Around that time I remember Bill Gates made a personal visit to talk with us about the XMS distributed operating system. It was an interesting discussion, but nothing came of it.

After the XMS project ended I ran some other product developments in BNR and was a VP responsible for a group of more than a 1000 staff. At that point I had no time to do any research or design work. There were still a lot of very interesting technical decisions that needed to be made, but a management role was not my idea of fun, I wanted to get back into research.

I was very lucky to be given the opportunity to start a small Computing Technology Lab. This was great fun, and I had the opportunity to work with other research labs, and university researchers. Even with research groups of Nortel's competitors. For example, I collaborated with Joe Armstrong and the Ericsson computing research team in Sweden. That was the start of Erlang and the BEAM, now running the Elixir programming language. A great success for Ericsson, and an even bigger success for open source.

At long last software for multi-processors and distributed computing is starting to be seen as important. The idea of non-stop systems that keep running for ever, with evolution and self repair, are still not widespread. The Erlang development established the real-world reality of fail-fast, light-weight processes that could restart and recover, as the basis for building robust systems. It is not just something that is important to telecommunications.

The core idea goes back to the Actor model, named by Carl Hewitt at MIT in the early 1970s, and championed by Alan Kay at Xerox PARC with Smalltalk. I knew about Carl Hewitt's work very early on, but some of his abstractions were beyond me. However, the basic idea was exactly what we were doing, in my case inspired by the datagram model. Individual processes that do not share anything, and communicate only with message passing.

In the mid 1990s I had an expat period in Australia working for BNR and running a small R&D lab based at the University in Wollongong. At that time the WWW had arrived, and the commercial Internet was exploding, but the glory days of BNR as an independent R&D lab were over, BNR was absorbed into Nortel.

The world of research often needs to be collaborative, but this has become harder and harder as corporations build walls to protect their proprietary IP. The Uni-

versity researchers want to become entrepreneurs to disrupt the corporations rather than collaborate with them. And inside corporations the advanced research team is often seen as a threat rather than an opportunity. In the quest for profits many innovations that need collaboration between fundamental research and product R&D become road-kill. Open source software is an exception, but no one seems to have solved the problem of how to reward the inventors and innovators with a fair share of the profits that can result from their work.

I returned to Canada and retired from BNR (well technically Nortel) in 1998, and finally returned to NZ in 2000. It had been a rather long OE!!

In 2000 you became a winemaker back in New Zealand. Probably that was a good year to leave the circuit switching business, because after the .com crash, the era of connectionless packet switching really started. No regrets?

I did learn to be a winemaker, and some basic farming and viticulture skills. But it was a steep learning curve, and although I enjoyed learning a lot of new things, I soon began to miss working as a software geek. I now have great respect for farmers. They have to run a business, and they have a lot of hard work, with a tough and erratic schedule dictated by the weather.

As a retirement project, building a vineyard was not a good plan! It's far too much work! After 10 years I was happy to have sold the vineyard and fully retired to work on hobby software projects at my leisure. Software and computing continue to fascinate me.

Thank you very much!

References

[1] P. M. Cashin. Datapac Network Protocols. In P. K. Verma, editor, *Proceedings of the Third International Conference on Computer Communication, Toronto, Canada, August 3-6, 1976*, pages 150–155. International Council for Computer Communication, August 1976.

[2] P. M. Cashin, M. G. Robinson, and D. M. Yates. Experience with SCRAPBOOK, a Non-Formatted Data Base System. In J. L. Rosenfeld, editor, *Information Processing, Proceedings of the 6th IFIP Congress 1974, Stockholm, Sweden, August 5-10, 1974*, pages 1012–1016. North-Holland, 1974.

[3] D. W. Davies, D. L. A. Barber, W. L. Price, and C. Solomonides. *Computer Networks and their Protocols.* Wiley, 1979.