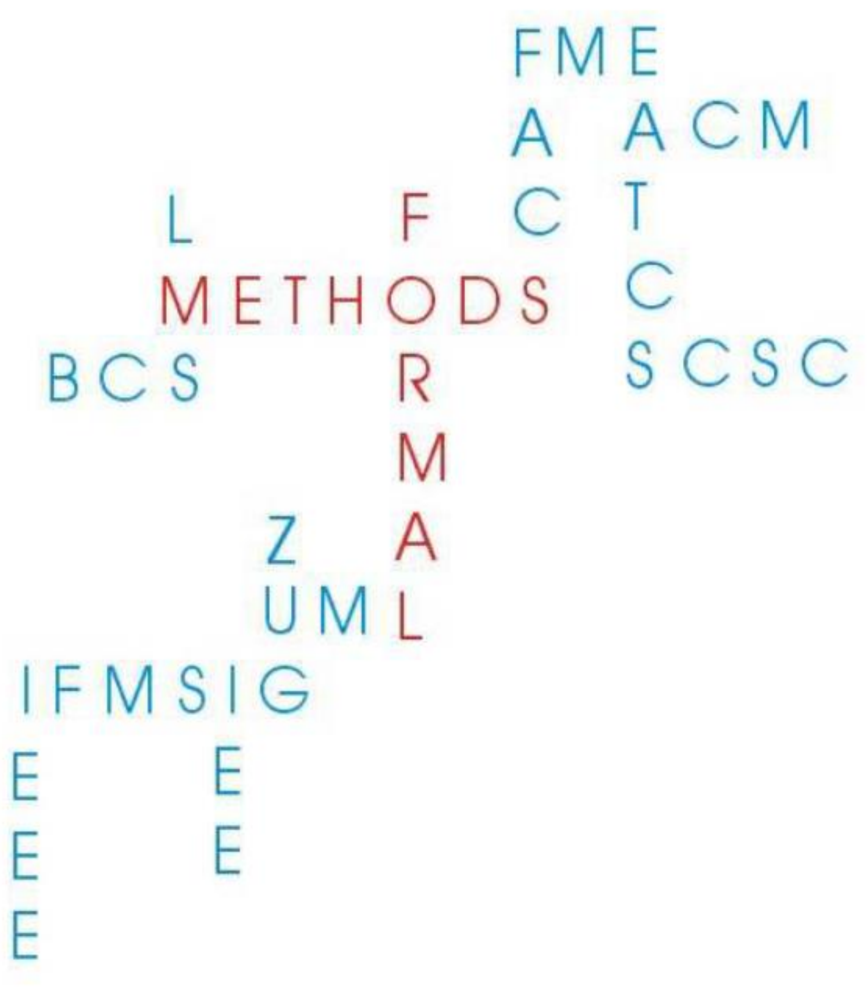
Issue 2024-2
July 2024

FACS FACTS



Formal Aspects of Computing Science Specialist Group

bcs The Chartered Institute for IT

The Newsletter of the Formal Aspects of Computing Science (FACS) Specialist Group

ISSN 0950-1231

About *FACS FACTS*

FACS FACTS (ISSN: 0950-1231) is the newsletter of the BCS Specialist Group on Formal Aspects of Computing Science (FACS). *FACS FACTS* is distributed in electronic form to all FACS members.

Submissions to *FACS FACTS* are always welcome. Please visit the newsletter area of the BCS FACS website for further details at:

https://www.bcs.org/membership/member-communities/facs-formal-aspects-of-computing-science-group/newsletters/

Back issues of *FACS FACTS* are available for download from:

https://www.bcs.org/membership/member-communities/facs-formal-aspects-of-computing-science-group/newsletters/back-issues-of-facs-facts/

The *FACS FACTS* Team

Newsletter Editors:

Tim Denvir	timdenvir@bcs.org
Brian Monahan	brianqmonahan@googlemail.com

Editorial Team:

Jonathan Bowen, John Cooke, Tim Denvir, Brian Monahan, Margaret West.

Contributors to this issue:

Jonathan Bowen, Jifeng He, Cliff Jones, Bill Roscoe, Joe Stoy, Bernard Sufrin, Tim Denvir, Keith Lines, Alvaro Miyazawa, Brian Monahan, Andrei Popescu.

BCS-FACS websites

BCS:	http://www.bcs-facs.org
LinkedIn:	https://www.linkedin.com/groups/2427579/
Facebook:	http://www.facebook.com/pages/BCS-FACS/120243984688255
Wikipedia:	http://en.wikipedia.org/wiki/BCS-FACS

If you have any questions about BCS-FACS, please send these to Jonathan Bowen at jonathan.bowen@lsbu.ac.uk.

Editorial

Dear readers,

Welcome to Issue 2024-2 of the *FACS FACTS* newsletter. This is the second issue of 2024. The flagship in this issue is a tribute to Professor Sir Tony Hoare, FRS, in honour of his 90th birthday earlier this year. Six eminent computer scientists have contributed personal, informal memories of Tony Hoare: personal and informal because there have been a good many "official" *festschrifts* for Tony over the last few years. These previous tributes are listed in the introduction to the six contributions in this *FACS FACTS* issue. The contributors here are Jifeng He, Cliff Jones, Bill Roscoe, Joe Stoy, Bernard Sufrin, and our chairman Jonathan Bowen. We urge readers to read and enjoy these memoirs.

Since the last issue of the newsletter, *FACS* has held four seminars. We can't help feeling rather pleased with this record: The annual *FACS-LMS* seminar, *Formalising 21st-Century Mathematics*, by Laurence Paulson FRS in January; *The SI Digital Framework: Underpinning FAIR measurement data,* by Jean-Laurent Hippolyte in February; *Scott models for probabilistic computation* by Abbas Edalat in March; *Verifying system-level properties of neural-network robotic controllers* by Jim Woodcock, in collaboration with the RoboStar Centre, University of York. Many thanks to Andrei Popescu and Keith Lines for organising the first two of these, respectively, and to Alvaro Miyazawa for organising the talks by Abbas Edalat and Jim Woodcock. Short reports of these follow our Tony Hoare tribute.

Immediately following on from our meeting reports, we also have a book review from Brian Monahan, *A Brief History of Mathematics for Curious Minds* by Krzysztof R. Apt; a report on the SETSS 2024 Springer School from Jonathan Bowen; and also from Jonathan, a reflection on *The development of the book cover for The Turing Guide and generative AI.*

Finally, our Back Issues (including FACS Europe and FORTEST reports) can be downloaded here. Recent FACS seminars can also be viewed here.

We greatly appreciate and look forward to contributions, including letters and comments, from you, our readers. We hope you enjoy *FACS FACTS* issue 2024-2.

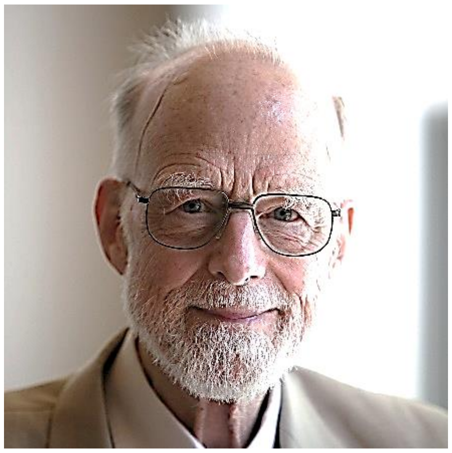
Tim Denvir
Brian Monahan

Table of Contents

Tony Hoare @ 90

Dear readers,

Professor Sir Charles Antony Richard Hoare, FRS[1], reached his 90th birthday in January this year. Tony Hoare, as he is better known, was knighted for his services to theoretical computer science, and was one of the earliest computer scientists to be made a Fellow by the Royal Society. I believe the first computer scientist to be elected FRS was Maurice Wilkes, and for some years only he and Tom Kilburn — famed for the Manchester "Baby" and other historic computers — were Fellows. It took several years for more to be elected, and Tony himself was one of the first.


Sir Tony Hoare, FRS
(Source: Wikipedia)

Since those days, the Royal Society has recognised computer science as a "proper" science, and there are now a respectable number of CS Fellows. Soon after Tony Hoare's election, Robin Milner was elected an FRS, and following that, "the gates opened" and several more have joined their ranks. For example, Laurence Paulson[2] FRS — who did his post-doctoral work with the late Mike Gordon[3], FRS — gave our most recent annual FACS-LMS seminar this year, with Tony Hoare himself giving an earlier FACS annual Peter Landin Semantics Seminar[4]. Just recently, Glynn Winskel, who gave the 2021 Peter Landin Semantics seminar, was elected an FRS. For many years, Tony Hoare was series editor for the Prentice-Hall "Red and White" computer science series. For these and many other reasons, FACS felt that we would like to pay a tribute to Tony Hoare in our newsletter.

There have been a number of *festschriftten* for Tony. I am grateful to several of our contributors for references to these, as listed below. In the light of that collection of technical material, the *FACS FACTS* editorial team invited a number of eminent computer scientists to give some more informal and personal recollec-

[1] https://en.wikipedia.org/wiki/Tony_Hoare
[2] Reported on by Andrei Popescu in this issue.
[3] Mike Gordon worked widely within the theorem-proving field, including mechanised support for proofs in Hoare logic.
[4] Denvir, T. (2013). Report on: Peter Landin Annual Semantics Seminar: Professor Sir Tony Hoare (December 3rd 2012). FACS FACTS, 2013(1):5–7, December 2013

tions of their work in collaboration with Tony or guided by him. It is a pleasure to record that six contributors we invited responded with enthusiasm: Jifeng He, Cliff Jones, Bill Roscoe, Joe Stoy, Bernard Sufrin, and the *FACS* chairman, Jonathan Bowen.

Tony Hoare read *Greats* at Oxford (*Literae Humaniores*, thanks to Bill Roscoe for this illumination!) and learned Russian during his national service with the Royal Navy. He then spent a year at Moscow University studying under the genius mathematician, Andrey Kolmogorov[5], who is famed for his original work on probability, statistics and intuitionistic or constructive logic, among much else. After that spell in Moscow, Tony moved to Elliott Bros. London Ltd., a British computer manufacturer, which is where I first met him, in my first job after graduating. On learning of Tony's professional path to that point, I was and still am amazed by the fact that he absorbed brand new technical and theoretical topics in mathematics and theoretical computing, in Russian, when he had only recently learned that language.

Since all the other contributors to this tribute to Tony refer to his later work in academia, I thought a brief note on his earlier 'industrial' time might be in order. I was at Elliott's from 1962 to 1965, Tony from 1960 to about 1968. Tony was designing and supervising the implementation of an early ALGOL 60 compiler[6], one of the first. I was rather envious of the ALGOL team, not being a member of it, but Tony orchestrated departmental seminars as in academic environments, which were very stimulating. Elliott's was at the same time bringing out a new computer. At that point, in the early 1960s, it didn't have a name. The computer was called "Project 41". We were sworn to public silence about that. The computer was to have a fuller-fledged time-sharing system than most of its predecessors. Individual user programs could proceed at the same time, and demands made by peripheral devices, such as card/paper tape readers, could take place simultaneously without seriously suspending the user programs. I was to program the kernel of the OS that would handle this time-sharing. This was a challenging task, but under Tony's guidance, I persevered. His core idea was that there should be two levels of interrupt, one less urgent but still needing attention in due time, which he called a 'hesitate', and the other, more urgent, which would inexorably lose information if not rapidly attend-

[5] https://en.wikipedia.org/wiki/Andrey_Kolmogorov

[6] Some 8 years later I too designed and implemented an ALGOL 60 compiler while at RADICS Ltd. I don't think I could have done so without the stimulus of witnessing Tony Hoare's earlier implementation. See Tim Denvir, More Recollections of ALGOL 60, in Resurrection No. 52, Autumn 2010, ISSN 0958-7403

ed to, a proper 'interrupt'. Our mutual manager, Sheila Quinn, long since deceased, almost warned me, "He's absolutely brilliant, and he expects everyone else to be the same!". Project 41 became the Elliott 4100 series, and I feel gratified that a few lines of crucial OS code that I wrote lie within those machines, if any of them are still in use.

Enough of this from me. I commend the personal memories from our six noble academic contributors, which follow on from here.

Tim Denvir
FACS co-editor

Festschriftten

C A R Hoare and Cliff B Jones. *Essays in Computing Science*. Prentice Hall, 1989.

Bill Roscoe, editor. *A Classical Mind: Essays in Honour of CAR Hoare*. Pearson Education, 1994.

Cliff B Jones, A William Roscoe, and Kenneth R Wood, editors. *Reflections on the Work of C. A. R. Hoare*. Springer Science & Business Media, 2010.

Cliff B Jones and Jayadev Misra, editors. *Theories of programming: the life and works of Tony Hoare*. ACM, 2021.

Cliff B Jones. *An Interview with Tony Hoare: ACM 1980 A.M. Turing Award Recipient.* ACM, 2015. https://amturing.acm.org/pdf/HoareTuringTranscript.pdf

Jim Davies, Bill Roscoe, and Jim Woodcock, editors. *Millennial perspectives in computer science: Proceedings of the 1999 Oxford-Microsoft Symposium in honour of Professor Sir Tony Hoare.* Palgrave, 2000.

CARH retirement symposium, Oxford, 13–15 September 1999

For Tony’s 90th Birthday

He Jifeng

Distinguished Professor, Tongji University, Shanghai

I was the visiting scholar in Oxford University from November 1983, when I first met Tony; he warmly welcomed me and gave me a tour of the computing laboratory. Under the recommendation of Bernard Sufrin, I officially joined PRG as a research fellow in November 1984 to participate in a project led by Tony Hoare. In a short period of time, he turned me from a newcomer to core member of the team, giving me the opportunity to participate in different projects, and encouraging me to participate in international conferences and summer schools, thereby exposing me to forefront of technology development. The academic exchange activities led by him gave me the opportunity to meet peers from different countries and that helped me in building an international network. In the more than 15 years of collaboration with Tony, he has always been a mentor to me, he had great skills as an applied logician; we had many meaningful discussions around software theoretical research and application technology. Additionally, he has this extraordinary ability to guide us in exploring links between fundamental research and engineering applications. He organized series of seminars to define the goal of the research, and discuss new approaches to improve the trustworthiness of software. We also had weekly meetings in which he inspired ideas from different groups, embracing a diversity of thoughts, and promoting a culture of openness and inclusion within PRG. Under his leadership, our projects with IBM Development laboratories and with microprocessor company Inmos culminated in technologically advanced products, and were recognized by Queen’s awards. Based on our experience in the development and application of logical theories, we undertook a project to broaden the theoretical models to cover more aspects of Computing Science and its application to system design. In 1998, we published the results of ten year’s research under the title “Unifying Theories of Programming”.

On a personal level, our two families have many years of friendship. At Oxford, we enjoyed dinner at each other’s home, and shared our respective cultural and personal experiences. Since leaving Oxford, we stayed in close contact and shared with each other our latest work findings. He came to China a number of times to support my new role of Dean of Software Engineering Institute. We last saw each other in 2018 in London at BCS symposium where we discussed the new research topics on Unifying Theories of Programming.

Till today, I still have many fond memories of the time that Tony and I shared, his wisdom, insight and kindness made a significant impact to my career and my life.

Congratulations Tony, to your remarkable lifetime achievement and highest international regard in the Computing Science field.

Happy 90th birthday.

Jifeng He, Bernard Sufrin, and Tony, at a party given for Chinese scholars at St. Hilda's College. Oxford in 1984, by the Ambassador of the People's Republic of China. (The ambassador took the photograph himself.)

Jifeng He and Tony at the UTP 2016 conference in Reykjavik, Iceland

Group photograph during the UTP 2016 conference in Reykjavik, Iceland.

Tony is in the centre with Jifeng He to the left and Leo Freitas, Ana Cavalcanti, and Huibiao Zhu to the right. Behind Tony to the right are Jonathan Bowen, Jim Woodcock, and Yixiang Chen.

Jifeng He and Tony were UTP 2016 keynote speakers. Jonathan Bowen and Huibiao Zhu were the proceedings editors.

Jifeng He and Tony at the BCS London office, 2018

Thanks to Tony Hoare

Cliff B Jones
School of Computing, Newcastle University

As well as the debt all computer scientists owe to Tony for his insightful scientific contributions, many of us have experienced his personal kindness. As someone who has benefited enormously from his support, I'd like to offer a personal note of thanks to this collection.

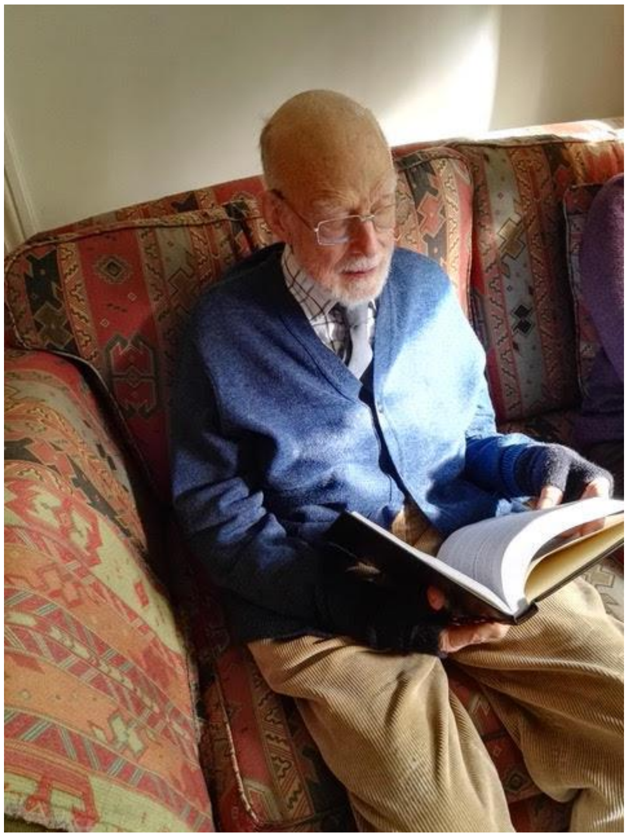
My most obvious debt to Tony is his arranging that I was accepted at Oxford in 1979 to undertake a doctorate with him. Since I had dropped out of Grammar School pre "A-levels", I did not have the normal prerequisites for post-graduate studies — but at the foot of the page detailing the requirements, the Examination decrees state how to get an exception approved — there's an organisation that has learned over centuries that to any rule there have to be exceptions. Tony presumably based the justification on publications and technical reports from my time in IBM. I should also add that Tony's college offered the perfect environment for a "mature student" and Wolfson remains a special place for me.

Pursuing my DPhil with Tony in Banbury Road was one of the most exhilarating phases of my long research career: in 1979 Tony was deeply into the evolution of CSP and did gently encourage me to consider it as an approach to the problems in which I was interested — but when he saw that I had my own approach to compositional design of concurrent programs he offered encouragement and support. I think I was the first of Tony's students to "be allowed to supplicate" for a DPhil in Oxford — my young sons sat with Tony during the ceremony and asked why Latin was being used — Tony was probably one of few people who understood it without difficulty but replied with humour "because the text is such nonsense".

A technical bonus of lunching with Tony at Wolfson College was a discussion over coffee with Robin Gandy when I was briefly considering using Temporal Logic for my rely-guarantee ideas: Robin was dismissive of TL and I didn't regret accepting his prompt.

However, Oxford was certainly not the first occasion for which my thanks are due. I'm almost certain that Tony prompted my invitation in 1973 to become a member of IFIP's prestigious Working Group on Programming Methodology (WG 2.3). We had certainly met as early as April 1969 when Tony presented his Axiomatic basis to WG 2.2 in Vienna. That part of Austria was important to both of us. I was able to be an "observer" at the WG 2.2 meeting because I was on a two-year assignment to the IBM Lab in Vienna. Tony had looked at their huge

VDL operational semantics of PL/I in his role on the ECMA standardisation committee; but well before that he had made a crucial comment at the 1964 Formal Language Description Languages conference in Baden-bei-Wien that foreshadowed his hunt for an implicit way of describing semantics.



Tony in their Cambridge house studying `Theories of programming: the life and works of Tony Hoare'

Photo by Joanna Francis reprinted with her permission

It is also a pleasure to recall the crucial role that Tony's famous "Red and White" series of Prentice-Hall books played in the development of what many of us call "formal methods". My own 1980 book was key to the description of those aspects of VDM that relate to the design of general programs. But when it became obvious that LNCS 61 was not going to be reprinted by Springer, Tony immediately invited Dines Björner and I to update the material on programming language aspects of VDM's denotational approach and produce a new volume (published in his series in 1982).

In 1981, I was appointed to a chair in Manchester. One of the Manchester professors who interviewed me confided that mine was the only name in the intersection of the sets of recommended candidates and applicants. Since this was prior to my DPhil viva, I have little doubt about who made the recommendation.

A delightful personal recollection was our joint visit to China in April 1983. Since Tony had recently been elected an FRS and, of course, had received the 1980 Turing Award, our mutual friend Prof Zhou Chaochen had arranged that the Chinese Academy of Sciences invited Tony for a lecture tour; I was clearly the "support act" but the technical sessions worked well because Tony and I sat through each other's lectures; this was important because at that time it was considered impolite for members of a Chinese audience to ask questions during lectures but we were both besieged in the breaks — sitting in the other one's lectures gave us time to hone our next talks.

What made the Beijing/Nanjing/Hangzhou trip so special was that we were both accompanied by our families. My sons were only seven and nine years old and were periodically "kidnapped" to appear in Chinese photos as the "golden-haired boys" (= not completely black hair). Tony and Jill's daughter, Joanna is tall with vivid red hair and attracted many curious Chinese who in many places had seen few Westerners — remember 1983 was not long after the "cultural revolution". We were all (eight of us) taken on many interesting visits and during a trip to, for example, a communal farm Tony would ask "How many members of the communist party are there here?". Puzzled, I finally queried why Tony wanted to know to which he replied with a smile "I don't, it's just interesting that they always know"! (Remember that he had earlier spent time in the USSR.)

There was an amusing coda to the 1980 book (SDRA): Henry Hirschberg and Helen Martin from Prentice-Hall chose to mark the 50th book in the above-mentioned series with an annotated book of Tony's papers, and they invited me to edit the 1989 volume. They also wanted to display all of the series in the window at Blackwells' wonderful store. Despite Henry having stuck his neck out by causing over ten thousand copies of SDRA to be printed, Prentice-Hall could not find one to display. Ian Hayes to the rescue: having complained many times about the cost of textbooks in Australia, he was able to buy two copies – at bookshop prices – and post them to Prentice-Hall at their expense!

There are many other happy personal links to Tony and his family, but I should close these words of thanks by returning to my debt to his inspirational research; I know that this observation could be echoed and amplified by enormous numbers of computer scientists.

The photograph was taken during that April 1983 trip:
Family Hoare is sitting in front of the Nine-Dragon wall in Beijing.

References

1. C. A. R. Hoare and Cliff B. Jones. Essays in Computing Science. Prentice Hall, Inc., 1989.

2. Bill Roscoe, editor. A Classical Mind: Essays in Honour of CAR Hoare. Pearson Education, 1994.

3. Cliff B Jones, A William Roscoe, and Kenneth R Wood, editors. Reflections on the Work of C. A. R. Hoare. Springer Science & Business Media, 2010.

4. Cliff B. Jones and Jayadev Misra, editors. Theories of programming: the life and works of Tony Hoare. ACM, 2021.

5. Cliff B. Jones. Professor Sir Tony Hoare: ACM Turing Award Winner 1980. Online, 3 2016. Interview.

Working with Tony Hoare

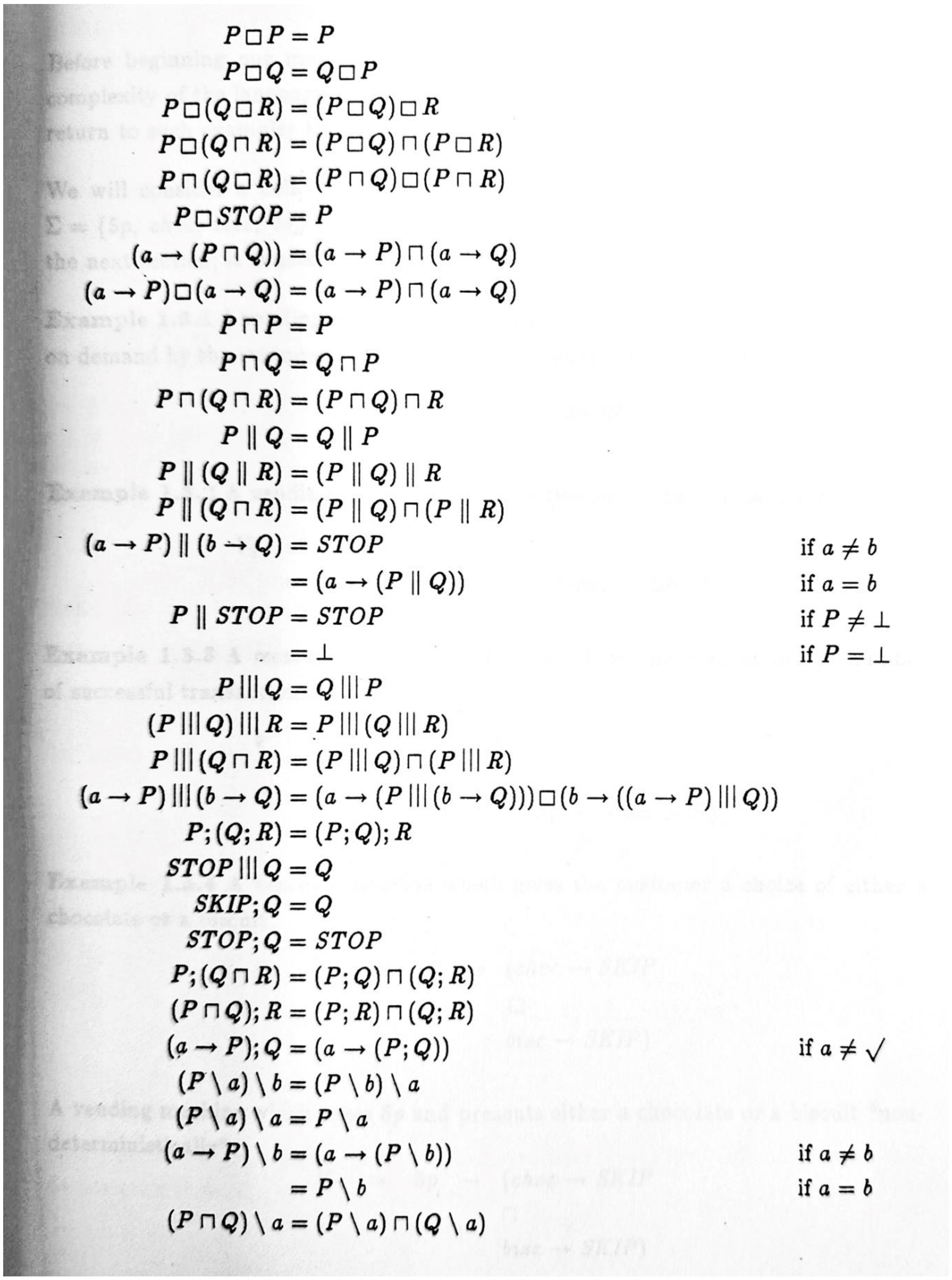
Bill Roscoe
University of Oxford

In this memoir, I recall my interactions with Tony from Autumn 1978, when I first met him, up to 1989 when we completed the first phase of hardware verification with Inmos, together with more general memories. Much of the history of my projects with him over this period — the development of process algebra CSP, the semantics of `occam` and their application to hardware verification — has been well documented in my and others' previous writings such as [1,2,3,4] so here I am going to concentrate on what it was like to work with him and be supervised by him.

Tony has an amazing intuition for seeking a simple and elegant way of attacking problems. When I first met him, I had just emerged from my Oxford Maths degree — and delighted in creating complex structures in what most people would describe as pure maths. The Scott-Strachey approach to programming language semantics is certainly elegant, but as I learned it as a final-year undergraduate it seemed anything but simple. It therefore took me some months to realise the power of Tony's quest for simplicity. In particular, my first year of working with Tony and Steve Brookes on CSP showed how his design of the language itself was driven by his quest for the most elegant laws and models.

He was certainly driven by the desire that the CSP we were developing satisfies algebraic laws. These laws were, at the time, what you might call healthiness principles: pleasing properties that you would expect an elegant structure to satisfy. They were not motivated by the immediate desire to create an algebraic semantics in the style of ACP or as was later done for CSP by Brookes [5] and myself [6,7]. Laws of this second sort tend to be directed at transforming finite programs into a restricted normal form, whereas Tony's original laws such as those set out in [8], and reproduced elsewhere, have a more subjective feel to them. In developing Timed CSP with me, Mike Reed gathered together a large collection of Tony's laws, which you can find in [9].

Although Tony did not study mathematics as an undergraduate, he studied logic as part of the philosophy components of his Greats (*Literae Humaniores*) degree and told me a few years ago how much he valued his logic tutorials with the then young philosopher and logician John Lucas, a junior research fellow at Merton and later a tutorial fellow there. Tony was also one of a group of Merton students who regularly met to discuss mathematical topics.



$$
\begin{aligned}
P \Box P &= P \\
P \Box Q &= Q \Box P \\
P \Box (Q \Box R) &= (P \Box Q) \Box R \\
P \Box (Q \sqcap R) &= (P \Box Q) \sqcap (P \Box R) \\
P \sqcap (Q \Box R) &= (P \sqcap Q) \Box (P \sqcap R) \\
P \Box STOP &= P \\
(a \rightarrow (P \sqcap Q)) &= (a \rightarrow P) \sqcap (a \rightarrow Q) \\
(a \rightarrow P) \Box (a \rightarrow Q) &= (a \rightarrow P) \sqcap (a \rightarrow Q) \\
P \sqcap P &= P \\
P \sqcap Q &= Q \sqcap P \\
P \sqcap (Q \sqcap R) &= (P \sqcap Q) \sqcap R \\
P \parallel Q &= Q \parallel P \\
P \parallel (Q \parallel R) &= (P \parallel Q) \parallel R \\
P \parallel (Q \sqcap R) &= (P \parallel Q) \sqcap (P \parallel R) \\
(a \rightarrow P) \parallel (b \rightarrow Q) &= STOP && \text{if } a \neq b \\
&= (a \rightarrow (P \parallel Q)) && \text{if } a = b \\
P \parallel STOP &= STOP && \text{if } P \neq \perp \\
&= \perp && \text{if } P = \perp \\
P \mathbin{|||} Q &= Q \mathbin{|||} P \\
(P \mathbin{|||} Q) \mathbin{|||} R &= P \mathbin{|||} (Q \mathbin{|||} R) \\
P \mathbin{|||} (Q \sqcap R) &= (P \mathbin{|||} Q) \sqcap (P \mathbin{|||} R) \\
(a \rightarrow P) \mathbin{|||} (b \rightarrow Q) &= (a \rightarrow (P \mathbin{|||} (b \rightarrow Q))) \Box (b \rightarrow ((a \rightarrow P) \mathbin{|||} Q)) \\
P; (Q; R) &= (P; Q); R \\
STOP \mathbin{|||} Q &= Q \\
SKIP; Q &= Q \\
STOP; Q &= STOP \\
P; (Q \sqcap R) &= (P; Q) \sqcap (Q; R) \\
(P \sqcap Q); R &= (P; R) \sqcap (Q; R) \\
(a \rightarrow P); Q &= (a \rightarrow (P; Q)) && \text{if } a \neq \surd \\
(P \setminus a) \setminus b &= (P \setminus b) \setminus a \\
(P \setminus a) \setminus a &= P \setminus a \\
(a \rightarrow P) \setminus b &= (a \rightarrow (P \setminus b)) && \text{if } a \neq b \\
&= P \setminus b && \text{if } a = b \\
(P \sqcap Q) \setminus a &= (P \setminus a) \sqcap (Q \setminus a)
\end{aligned}
$$

Laws from Mike Read's DPhil thesis (1990)

A few years later, at the start of his career, Tony studied in Moscow in the school of Kolmogorov, one of the greatest mathematicians of the 20th century, and who was extraordinarily influential in developing modern probability. I only discovered recently that the standard axioms of probability[7] that I learned as an undergraduate were created by Kolmogorov about 25 years before Tony's work with him. It seems reasonable to wonder whether his spell in Moscow influenced Tony in favour of laws. Of course, while he was there Tony created what has been described as the first probabilistic algorithm, namely Quicksort — and reading Tony's paper [10] shows he was well aware of the probabilistic properties it has, so it would seem that there are two clear potential influences from Kolmogorov.[8]

Tony created the traces model of CSP and many of its laws, thereby demonstrating the utility of the search for simplicity. While we were well aware of its deficiencies it clearly demonstrated how wise it was to search for simplicity. One might characterise this as saying that it is better to find a solution that is too simple and extend it rather than tackle everything at once. Thus, he guided Steve and me to find minimal extensions to traces that had the required properties including respecting the laws. We had many discussions in Tony's office at 45 Banbury Road, either just the three of us or alongside other students such as John Kennaway, visitors such as E-R Olderog and David Park, and of course Dana Scott. This was very much an environment in which everyone was interested in what everyone else was doing.

My own interactions with him often took one of two forms: either I would be finding mathematical justifications for his intuitions, or he would be pressing me to find more elegant ways of presenting my own. An example of the first was him claiming in a lecture (1979) that a counter $Count_0$ defined by infinite mutual tail recursion, was equal "by induction" to the clever recursion

`Zero = iszero -> Zero [] up -> Pos;Zero`, *where*

`Pos = down -> Skip [] up -> Pos;Pos.`

I immediately realised that while this was a true equality, it could not be proved by conventional induction. However, determined to justify his insight, by the

[7] https://en.m.wikipedia.org/wiki/Probability_axioms

[8] Kolmogorov has always been most closely associated in my mind with the Strong Law of Large Numbers, namely that the mean of repeated independent samples of a distribution converges to the mean of the distribution with probability 1. That result is very important in applications like blockchain.

next day I had formulated the metric theory of Unique Fixed Points and constructiveness which became central to CSP reasoning.

An example of the second form of interaction came a few years later (perhaps 1984/5) when he had asked me to create an algebraic semantics for occam, to which we had now switched our attention — I recall that our later discussions in 45 Banbury Road, which we left in Summer 1982, focussed on occam.

1990 Presentation of Queens Award for Technological Achievement for Inmos Ltd and the Oxford University Computing Laboratory

I was very proud to have created such a semantics (laws, normal form and reduction strategy) for occam that was congruent to my earlier denotational semantics. However, influenced by occam's syntactic structure, it was a syntactic mess containing too much ellipsis ... He made it clear that this was not good enough. So, I went back with my tail between my legs to think again, and eventually we came up with the form now in the paper [11].

That paper led to our famous project on the creation of the occam transformation system and the verification of the T800 FPU. Tony was a wonderful mentor: by that time he was happy for me to take the lead and influenced the project only through occasional suggestions — such as the idea of building the transformation system — and taking part in conversations with David May and others at inmos.

By that time, he had moved on from CSP, though the Laws of Programming project was itself influenced by the occam work. And so, his original creation of CSP was left to me to take forward, a typically generous move on his part, as was the later involvement with occam and the transputer.

My own close research collaboration with him thus came to an end in about 1989, though he was always very interested the work of my group over the years, who of course usually revered him. His sheer enthusiasm for the young and their work shone through. I remember that about 15 years ago he came back to Oxford for a big dinner in Trinity College. He asked me to organise a research meeting first thing the next morning, something the proposed participants were not keen on, as it would be so soon after the previous evening's celebration. I dragged my feet, but he organised it anyway. This was one of several occasions where Jill unjustly criticised me for "forcing Tony to work so hard".

I have been lucky enough to have been made to feel a part of Tony and Jill's family, ever since visiting them a number of times at their home in Chalfont Road when I was a research student. It is a wonderful feeling to be an honorary Hoare in that way, largely because of Jill.

Tony and Jill Hoare – Picture used for the Hoare Room, Oxford University

Tony was always extremely loyal to his protégés, and the "family" he built at the PRG stayed together around him for many years, with a number, including me, appointed to permanent academic positions.

I do not think I ever saw him touch an actual computer until shortly before he left Oxford in 1999. Microsoft had given him a PC. We should have seen the writing on Oxford's wall.

References

1. Jones, C. B., & Roscoe, A. W. (2010). Insight, inspiration and collaboration. *Reflections on the Work of CAR Hoare*, 1--32.

2. Brookes, S. D., & Roscoe, A. W. (2021). CSP: A practical process algebra. In *Theories of Programming: The Life and Works of Tony Hoare* (pp. 187–222).

3. Hoare, C. A. R. (1991). The transputer and occam: A personal story. *Concurrency: Practice and Experience*, *3*(4), 249–264.

4. May, D. (2021). CSP, Occam, and Inmos. In *Theories of Programming: The Life and Works of Tony Hoare* (pp. 271–284).

5. Brookes, S. D. (1983). *A model for communicating sequential processes* (Doctoral dissertation, University of Oxford).

6. Roscoe, A. (1998). *The theory and practice of concurrency*, Prentice-Hall.

7. Roscoe, A. W. (2010). *Understanding concurrent systems*. Springer Science & Business Media.

8. Brookes, S. D., Hoare, C. A., & Roscoe, A. W. (1984). A theory of communicating sequential processes. *Journal of the ACM (JACM)*, *31*(3), 560–599.

9. Reed, G. M. (1990). *A uniform mathematical theory for real-time distributed computing* (Doctoral dissertation, University of Oxford).

10. Hoare, C. A.R. (1962). Quicksort. *The Computer Journal*, *5*(1), 10–16.

11. Roscoe, A. W., & Hoare, C. A. R. (1988). The laws of occam programming. *Theoretical Computer Science*, *60*(2), 177–229.

Tony Hoare

Joe Stoy
Formerly of the University of Oxford

The first time I met Tony was at MIT in 1974, when I was on sabbatical there. Barbara Liskov organised a symposium called "Languages and Systems to Support Structured Programming", which he attended. I remember his saying, to the surprise of some there, that if compilation were instantaneous, the "linking phase" would never have been invented — logically we should go straight from a text management problem to an executable program. This was of course long before the web, HTML, hypertext linking, and all that.

When Christopher Strachey died in 1975, I became the only established ("permanent") member of the Programming Research Group. We spent the next year getting Strachey's ad hominem professorship turned into a permanent ("statutory") chair. Many people and bodies were a great help in this: Leslie Fox, a numerical analyst and Head of the Computing Laboratory; the Mathematics Faculty Board; the head of the University's principal academic committee (the General Board); and several others. I actually attended the (very brief) meeting of Congregation which established the Chair.

After that a Board of Electors was appointed, and they eventually invited Tony to accept the Chair. Tony and Jill kindly invited me to spend a couple of days in Belfast to discuss the job. This was during the "troubles", and some Oxford friends suggested I should not go, or if I did, not to leave the airport. In the event I had a very pleasant stay at their home, talking about the PRG, and Oxford more generally (to which Tony was actually no stranger, having read Greats at Merton). Later in 1977, after Tony had accepted the post, he spent a couple of days with us, meeting various people and "casing the joint".

When Tony took up his post in September 1977, the PRG was almost empty of research staff. The graduate students were there, of course, and so were a couple of programmers; a new cohort on the Diploma course was arriving (for the Diploma in "Advanced Mathematics and Computation"). Dana Scott was also around as Professor of Mathematical Logic, albeit in the Philosophy Department; he had been a great support during the interregnum, and continued to be so. The ethos of the PRG still survived: seminars in the garden during good weather, and frequent conversations on the stairs of the house, which helped keep the group unified. But with Tony's arrival the research focus of the group gradually shifted, from functional programming and denotational semantics, to imperative programming and predicate semantics. The practical work also gradually shifted, from work in BCPL on the PRG's own computer (by then an Interdata 8/32 machine) using our own hierarchical quasi-functional operating system (OS6, or the published version OSPub), to work on a new batch of DEC LSI-11 workstations running UCSD Pascal. A taught MSc course started in 1979 replacing (after a gap) the previous Diploma course.

Visitors also began to reappear, some supported by the SRC research grant, drawn to the new focus. For example, Cliff Jones, with whom I had had previous discussions of the difference between the Vienna Definition Language (operational) and the Vienna Development Method (denotational), arrived and began DPhil study under Tony's supervision, on the addition of "rely" and "guarantee" clauses to the pre- and post-conditions of Predicate Semantics. Nevertheless, Peter Henderson, a functional programming person, was appointed as a Lecturer in 1980 to run the new MSc course, and stayed until 1983. Mary Sheeran was his DPhil student, and Geraint Jones his research assistant, who worked on Peter's Lispkit system. Jean-Raymond Abrial was also here during that time, and his work on "Z" dominated much of the research conversation. He eventually left: I think Dana criticised Z as being too prolix, not mathematically concise enough to be really useful. In research, I myself became increasingly a loner, though Tony was always supportive. For example, when I wrote a piece for his Festschrift thirty years ago, he wrote me a detailed comment on it, which I greatly appreciated. Gradually, however, most of my research collaborators were at MIT. I went back there on sabbatical for the year 1981-82; but before I left, things at PRG were greatly overshadowed by sadness at the death of Tony and Jill's youngest child, Matthew, from leukaemia. Dana also left for CMU in 1981.

Teaching was a different matter. When I returned in 1982, the atmosphere had completely changed. For one thing, we had moved into the Keble Road site, along with the rest of the Computing Laboratory (the numerical analysts). That at once made things less personal and homely than it had been at Banbury Road. For another, the Thatcher Government had published the Alvey Report, and had proposed a great expansion of Computer Science teaching at British universities, including new posts for the PRG.

This would not be the PRG's first foray into undergraduate teaching: I had for several years offered courses in Functional Programming among the second-year options for Mathematics undergraduates, with accompanying tutorials. Bill Roscoe had been attracted into computing by attending these lectures (with a slight complication because he also wished to go to my wife's lectures on

Group Theory in an adjacent lecture room at the same times — she brought him to my room, and we all agreed a slight revision of the timetable); he later became a research student at PRG and subsequently Head of the Department. Another person attracted to computing in that way was John Launchbury, later the founder and now the Chief Scientist of Galois Inc.

But now the undergraduate teaching would go into a much higher gear. The Computing Laboratory had been allocated four new lecturerships initially, each accompanied by ten additional undergraduate places. These were to be "joint" tutorial posts, which implied that colleges had to be found for each of them. I myself had been taken care of, having been elected (with Leslie Fox's support) to a Research Fellowship at Balliol in 1975 — when the Honour Schools eventually began in 1984 and my Fellowship became Tutorial, I actually became Balliol's junior tutor and Senior Tutor in the same instant. But colleges had to be wooed for the newcomers. Three of the first tranche of appointments, in 1983 went to people who were already at PRG in one guise or another: Bill Roscoe, Bernard Sufrin and Ib Sørensen; the fourth went to Richard Bird from Reading. The ten additional places were a great bribe for the college administrations, but a college's support depended in large part on its Mathematics tutors. The college election committees had their own set of stories (for example, at those colleges which had hoped that the Computing tutors would be able to spend most of their time teaching classical Applied Mathematics. Fortunately, Tony had learnt the art of academic politics during his time at Belfast, but even so he was initially only able to get two colleges offering to give tutorial fellowships — to Bill at Univ and Bernard at Worcester. Ib, whose special remit was industrial liaison, joined Tony at Wolfson, and Richard went initially to a graduate college, St Cross, where Peter Henderson had been, and replaced him as Director of the MSc course; five years later, though, he moved to a Tutorial Fellowship at Lincoln. That set of appointments at one blow tripled the academic staff of the PRG. But other tranches followed. In 1984, John Hughes (St Edmund Hall) and his wife Mary Sheeran (Lady Margaret Hall), both no strangers to PRG, were appointed, and so was Ian Page (initially at St Hugh's, though when Richard moved to Lincoln, Ian, preferring to be at a graduate college, took his place at St Cross). In 1985 Mike Reed and Jeff Sanders replaced John and Mary at SEH and LMH, while Carroll Morgan (Pembroke) and Bill McColl (Wadham) were new appointments.

So for those years, Tony had a particularly heavy load of academic appointments to manage, as well as the usual graduate intake. But there was also the new undergraduate program to design. It was decided early on that we should not initially attempt to put on a single-subject Honour School — in fact, the

Honour School of Computation did not start until 1994. There were to be two joint courses, Mathematics and Computation, and Engineering and Computing Science (ECS). Alongside ECS there was another joint course, Engineering and Materials Science. It was also decided that the first years of both our courses should be entirely devoted, along with Mods at the end of that year, to the other subjects, with their existing first-year syllabuses. That gave the new staff an extra year of grace to prepare classes and tutorial sheets for the new courses, many of which were given in both Schools. But before that, the legislation establishing the new Schools had to be put through the Faculty Board and the General Board, and the second-year syllabuses designed, and then approved by the Faculty Board and its Subfaculty of Computation. Tony was on the Faculty Board and saw that through. All this meant that our periodic group meetings became weekly and more formalised (e.g., regular minutes): Tony's meticulous mind was invaluable. And of course, once the courses started in earnest, the tutors were often away from the Computing Lab giving tutorials or doing other college things: Wednesday afternoons in term time was the accepted time for college meetings.

The Maths and Computing School flourished: even when the Computing School started it remained (with Tony's strong approval) one of our flagship programmes. ECS, on the other hand, dwindled and eventually faded away. I think the Engineers didn't understand the way we taught programming: for example not using C, but preferring to start by using languages (Haskell and Modula) in which it was easier to teach good programming principles. Later they did better on their own when they appointed their own Professor of robotics and computing (Mike Brady) — he later did a stint as Head of the Engineering Science Department.

After the initial upheaval, the Lab continued to flourish and grow. A second Chair was established in 1988 -- Joe Goguen for ten years, followed by Richard Brent. My informal contact with Tony became less and less — no doubt the college and University administrative work I was doing didn't help (for example, at various times chairing the Faculty Board and a couple of University committees). But I know that the Computing Laboratory was in excellent shape when Tony handed over its leadership and left Oxford for Microsoft in 1999.

The last time Tony and I met was in Seattle, at the Federated Logic Conference in August 2006, where we (and Dana) were among the invited speakers. The three of us enjoyed a long on-deck conversation on the conference boat excursion around the Bay. I remember Tony's talk vividly — he spoke for about an hour, in elegant English, with not a single slide or use of the blackboard, to the bemusement of some in the audience, who were not used to presentations like that: "A Classical Mind" indeed.

Early Days at the PRG …

Bernard Sufrin
Emeritus: Worcester College and
Department of Computer Science, Oxford University

… anecdotes from the Programming Research Group, for Tony Hoare's 90th birthday

It’s a pleasure to be able to write here about Tony, and about our early days at the Programming Research Group; and it's a relief that it's going to be read by people who already know Tony's work, and for whom yet another technical appraisal of his oeuvre would be *de trop*. The invitation said “Don’t be afraid to make it personal.” So I haven’t been. My goal is to show how strange the Oxford environment felt at first; and how adaptable, tolerant, and magnanimous Tony was.

The Interview

I met Tony Hoare and Joe Stoy for the first time in mid-1978, when I came to be interviewed at the PRG for an SRC research fellowship at the PRG that was associated with a Wolfson College fellowship. Tony had arrived as Professor of Computation about a year earlier.

At the time the PRG still occupied a large semi-detached house at 45 Banbury Road, and the first challenge for the three interviewees had been to believe that they were in the right place, for there was little or no indication of that on the outside of the building.

To our surprise, we all arrived at the house at exactly the same time, and Tony seemed completely unfazed that one of us was wearing swimming shorts and a singlet[9] “Do you mind all being interviewed at the same time”, he greeted us without flinching, “like in the Civil Service?”

The ostensible goal of the project we were competing to staff was to publish the texts of high-quality software. I imagined that this would be along the lines

[9] For years afterwards he would tell people that it had been me wearing the swimming shorts: but it wasn’t – my own noticeability stunt had been to arrive on a huge motorbike; and the third candidate is still memorable because he was wearing a tweed suit and waistcoat on the hottest day of that year.

pioneered by Strachey and Stoy when they published an annotated BCPL text of their operating system OS/Pub in 1972[10].

The reason I say "ostensible" is that in those days project proposals could promise insight without having to commit to a timetable of deliverables. And in those early years we were all to learn from Tony that progress in the general direction of a project's goals would nearly always generate enough insight to please its sponsors, and to carry us into the next project or two.

In fact the interview was a round-table discussion: though it never became clear what the role of the four grandees present beside Tony and Joe was, for they hardly spoke. Questions soon arose about what kinds of program should be published and what programming languages should be used.

I said that I thought we would need to publish in the highest-level language possible: "efficiency be damned." Whatever programs we would publish, it was the ideas behind them that mattered; and I recall saying that these would be more straightforwardly expressed in Lisp, because of the ease with which (what we would now call) domain-specific control and data abstractions could be described. Having not a clue about the extensive work initiated by Tony at Belfast to improve Pascal[11], when it came up as a candidate language I was pretty forthright about what I thought were its shortcomings as anything other than a pedagogical tool[12].

At the time I was effectively ignorant of the emerging disciplines of data and program refinement that Tony did so much to teach us about, so it had been the reputation of the PRG that had attracted me.

Almost all my research[13] had been conducted in the spirit of the "invisible college of Strachey." Most of the large programs I'd built had either been written in Lisp or BCPL[14]. I had also been an early and enthusiastic subscriber to

[10] BCPL is best known as a precursor of C. To the best of my knowledge, this was the first time the complete text of an operating system written in a high-level language had been published.
[11] Pascal Plus – a language whose relationship to Pascal was analogous to that of Simula67 to Algol60. See: https://onlinelibrary.wiley.com/doi/abs/10.1002/spe.4380091109.

[12] Tony and Joe seemed to like this: but my co-interviewees really didn't, and competed vigorously to show who was the better defender of Pascal. The man in swim shorts also turned out to be an enthusiast for Algol68; and it is (now) clear that I wasn't the only one there who wasn't completely *au fait* with Tony's work.

[13] At the University of Essex, CMU, Bolt-Beranek and Newman, then Essex again.

[14] For example, I'd built a fast Gedanken implementation inspired by Landin's "mechanical evaluation of expressions" in BCPL soon after reading John Reynolds's paper. I'd completed an implementation of BBN-BCPL for PDP-11s when I worked in the USA, and I'd also implemented a

the PRG monograph series, and had followed the developing work on mathematical semantics, despite feeling ill-equipped to understand its theoretical foundations.

Junior?

Soon after the interview I got a job offer from Tony, and a letter from Wolfson College saying that I had been "elected to a junior research fellowship" there. Being a Sheffield and Essex alumnus I wasn't used to this terminology; and being 31 at the time, and the tenured head of a small group at Essex, I didn't really consider myself "junior". Tony's very first professional kindness to me was to explain in our subsequent phone call that I mustn't take the "junior" too seriously, that it was his intention to build a proper academic department, that it would be advantageous to get in on the ground floor, and that I could have a few years to prove myself[15].

The PRG

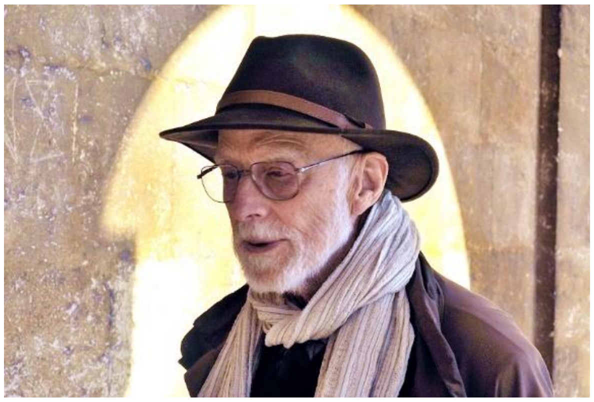
I arrived at 45 Banbury Road on my motorbike on the first of September in 1978, to join a PRG that consisted of two academics (Tony and Joe), service staff (two programmers, a caretaker, and a PA), and around half a dozen first year D.Phil students.

I had packed and sent a hundred or so books and papers to the PRG in advance of finding somewhere to live; and was astonished to find that the staff had added them all to the machine-readable PRG library catalogue. They included a few grocery invoices, several books on radical US politics, and a dozen anti-war pamphlets; every item had been allocated an accession number, and given the corresponding library sticker. Service indeed!

During working hours Tony seemed to spend an inordinate amount of time in administration: as Professor of Computation he had inherited multifarious responsibilities, including chairing the management committee of the University Computing service, and overseeing the work of the Computing Teaching Centre – a large operation teaching not-for-credit courses to anybody in any department who was interested. I learned later that much of what I'd taken to be routine administration had actually been working through Oxford's ramified committee structures to divest the post of these responsibilities.

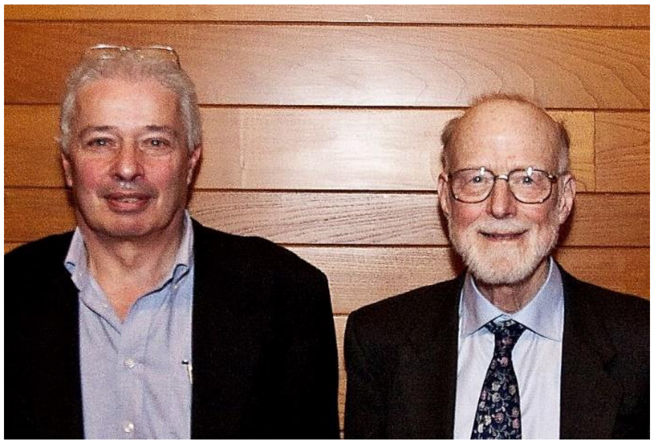
multithreading variant of BCPL. Other large projects had included a typesetting system at the core of whose operation was a microprocessor inspired by Strachey's GPM.

[15] The advantage of having been interviewed with the other two was that I was sure by the end of the day that if one of them were offered the post then I'd have been misreading Tony's body-language and couldn't have done the job anyway.



Tony outside my Magdalen College room, late summer 2015

At that time, and for years later, he did his creative thinking and writing at home: disappearing at five o clock, then returning in the morning with long manuscripts that he'd often share with us. His ability to switch from administration to scientific work without losing his focus was extraordinary. And his heuristic to avoid being overburdened by the pile of internal correspondence on his desk was audacious: "if I can't deal with something immediately, then I add it to the [discard] pile; if the senders think it important then they'll send it again."



Tony and me at Wolfson College after I had delivered (ventriloquising him) the second part of his 2011 Haldane Lecture: "Applied Logic".

Tony and I would walk to Wolfson for lunch nearly every day. It really was a free lunch, but the main attraction was having Tony to myself. He walked very quickly, and though a keen and strong walker myself I found it impossible to maintain his pace without breaking into a trot from time to time. I had asked him on my first day what he wanted me to do, and he had said that I should please myself, but that if I felt like it, I should teach a course. Once I decided that I was going to teach a course on programming language implementation I was able to run some ideas past him – I'd done some thinking about targeting (what I called) tree interpreters implemented in microcode[16] and was keen on exploring these in a course.

His counterproposal was that I instead talk about targeting stack machines implemented in the same technology. This was not to be the last time during our work together that he said his piece then deployed his trademark phrase: "I

[16] User micro-programmable bit-slice processors had started to emerge a few years earlier.

hope you make the right decision[17]." It was the first of countless examples I recall of him providing clarity, without giving instructions: one of the real signs of academic leadership.

The most difficult thing about giving the course was the incredulous reaction of the University Examination Schools to my request for a lecture room to teach it in. "But you are not a matriculated member of the University", they said, "so we cannot." Being a matriculated member of two other Universities cut no ice with them. "Get your college to matriculate you", they suggested. But my college was no more helpful "But do you teach for us?", the senior administrator asked. Being told that most of the Doctoral students in the PRG who would be coming to the course were at Wolfson also cut no ice: unless I *already* taught for them, they would not matriculate me.

Tony wryly told me that he could do absolutely nothing about any of this and recounted his own difficulties in getting any resources at all from the University for the group; let alone stable commitments that would let us move in the direction of establishing an undergraduate degree[18].

[17] I suppose I must have, for I still own the front panel and four boards of a High-Level Hardware Orion: a machine designed and commercialized by two of the attendees at the lectures, along lines I preached about in the course. For a short time – before it was eclipsed by Sun – this machine was a *sine qua non* for UK CS researchers, particularly those working in functional and logic programming.

[18] I have written elsewhere about Tony's role in the PRG's long march to proper department-hood. See: https://doi.org/10.1145/3477355.3477366.

Second cohort of [M.Sc students] and their teachers.
***From rear, LTR:** _ _ Abrial Sufrin [Bill Richardson] [Ian Cottam] _*
***Front**: Stoy _ Jones Hoare _ [Mary Sheeran] _*

Joe’s contribution was to send me an elegantly written nineteenth-century squib on the subject of not knowing what, exactly, it was that one had joined when one joined Oxford; but this only reinforced my views about the weirdness of the place, and the soundness of Tony’s advice to be patient. Eventually I got a hint that I could “squat” in a lecture theatre in Atmospheric Physics; and the course went ahead.

Operating Systems Wars

In a tradition established by Strachey, the group would meet weekly around Tony's dining table or in the library upstairs, and discussions ranged from scientific to technical. The technical discussions were mostly about how to replace the superannuated small Modular One that provided our *only* in-house computing facility.

Its operating system had evolved from OS/Pub. It was a two-phase cooperatively-interactive system – in the sense that either *everybody* had to be editing, or *everybody* had to agree to stop editing and let the queued batch jobs all run to completion[19]. The phase-change negotiations entailed much shouting up and down the four flights of stairs[20].

Tony chaired the technical meetings with an amused detachment that I understood once he told me that he had no need for a computer. His detachment was to be seriously tested during the "Operating System War" that followed us buying a large microprogrammed Interdata 8/32 machine, and inviting Richard Miller as a visitor – he had done the first port of Unix to another architecture, and it was for the 8/32. There were two camps: the utilitarians who wanted to use Unix on it; and the idealists, who had planned to microcode the BCPL abstract machine, and to run a derivative of the in-house system on it. It must have been difficult for the latter not to take matters personally. Joe, with the support programmers, had done a tremendous job in microcoding the BCPL abstract machine so that each user would have their own virtualised abstract machine; and Joe himself was admirably patient with the hot-headed utilitarians during this episode[21].

Tony eventually brought peace by making the judgment of Solomon: the working day would be cut in half: the in-house system would run for one half, and Unix would run for the other half[22].

Editors

During one of our walks, I spoke of my interest in interactive text editors (of which I had already built too many), and Tony suggested that I choose a well-known one and present a "rational reconstruction" of it. I struggled for a month or so, and eventually came up with something a bit like the Unix editor e, written in a highly modular form in Ada. I wasn't really satisfied with the outcome, and nor was Tony.

[19] One of its quirks was that when someone did a complex search on a large document, all the other terminals would stop echoing until the search had completed.

[20] A couple of the DPhil students were a bit too enthusiastic about the shouting. I'd been used to quieter surroundings and much better computing facilities and quickly decamped to a room in the High-Energy Physics particle-tracking lab, where I used their DEC10. I am still astonished that Peter Mosses had managed to build his **Semantics Implementation System** on the PRG system, and that Joe Stoy had used it to write his monumental work on Denotation Semantics. See: https://scholar.google.co.uk/citations?view_op=view_citation&hl=en&user=fIK8JS8AAAAJ&citation_for_view=fIK8JS8AAAAJ:DBa1UEJaJKAC.

[21] One of us had described the PRG as "like a computing Galapagos Island."

[22] A slight complication was that in the afterglow of the publication of Joe's book on Denotational Semantics we had somehow convinced ourselves to buy a line-printer with a λ where the X should be, so reading "raw" UNIλ printer outputs could be disconcerting.

He suggested that we ask Steve Brookes, one of his DPhil students, to do the same, and Steve very quickly came up with the editor semantics in the denotational style. His key modelling decisions were to give the semantics of *whole sessions* with the editor, and to incorporate the filestore into its state. I thought this an impressive feat, but it seemed clear that the path from a continuation-based whole-session semantics to an actual implementation would not be straightforward. My response was to make several attempts at giving a "narrative" account of the editor based on successive approximations to the (intended) reality; but it was only after Jean-Raymond Abrial and Cliff Jones arrived at the PRG that I was to learn enough about abstract specifications and program and data refinement to make a successful attempt at this.

Jean-Raymond Abrial & Cliff Jones

Jean-Raymond Abrial & Cliff Jones arrived in time for the following academic year: J-R as a research fellow funded by an individual grant, and Cliff to do his first degree (a DPhil under Tony's supervision). This happened to be the first year of the M.Sc. in Computation that Tony had persuaded the University to offer despite the PRG not really having enough staff to give it. He had deployed an argument that I saw him use a few times while we worked together – we may not have had enough permanent staff to give the necessary courses, but had enough visiting and research staff.

Jean-Raymond gave a course in Program Specification, where he introduced the variant of Z that he, Steve Schumann, and Bertrand Meyer had written about earlier that year. Cliff and Tony gave a course on Programming Language Principles – Cliff using VDM for his part[23]. Tony encouraged me to go to J-R's and Cliff's lectures, and it was at these that I learned the basis for the most powerful and liberating intellectual tools I ever used.

STL, and 2G Z

In the spring of that academic year, Tony drove Jean-Raymond, Ib Sørensen (Joe's DPhil student), and me from Oxford to meet Bernard Cohen at Standard Telecommunications Laboratories at Harlow to discuss a project that they would fund to demonstrate the effectiveness of formal specification techniques in communicating ideas about systems[24]. The project was called CAVIAR[25].

[23] Joe Stoy and I gave courses on Functional Programming and on Programming Language Definition and Implementation.

[24] It was a terrifying journey: Tony drove fast, with what might have been called verve had he not also kept turning round to talk to the two of us in the back of his car.

[25] Computer Assisted Visitor Information And Resources

It was to specify a system to be used by STL to manage room scheduling and resource provision for the (very many) visitors to its buildings. J-R, Tim, Ib and I spent the next few months refining the existing **Z** notation, and thinking about CAVIAR with it. We'd meet every afternoon in the smoke-filled office[26] occupied by Jean-Raymond and Ib and our non-smoking research fellow, Tim Clement. And when we weren't changing the specification we'd change the language. Jean-Raymond had the big language ideas, and worked frighteningly fast, but listened to anybody with critical ideas. What emerged from our deliberations was the "second generation" **Z**, with semantics rooted in a generically-typed set theory, and a "basic library" presented in the style of Bourbaki.

We were pleased with the overall intelligibility of the notation, though our "class'", and "class-function" notions – intended to promote incremental and discursive specifications – soon turned out to be quite unsuitable for rigorous reasoning.

By then Tony's scientific energies were focussed on CSP; and for personal reasons he wasn't in a position to engage properly with our work as well. But he trusted us to make our own way, and at his suggestion we sent the language draft to Dana Scott. Dana's response was warm but unrelenting: the language was too verbose – for example it could take half a page to write down a simple quantified predicate. Of course, he was right: we had made too many concessions to avoiding making the language *appear* too mathematical, so as not to frighten programmers in flight from mathematics.

We soon understood that anybody who was going to use the language seriously would have to be able to think mathematically, and we began the retreat from our pretty-but-verbose notations to the more orthodox forms from which the "third generation" **Z** eventually evolved. This was to become, for a while, one of the pillars of the reputation of the PRG.

Skating on safe ice

I was "unpartnered" at the time and would have been indescribably lonely during my first Christmas in Oxford had it not been for Jill and Tony's invitation to join their three children and a couple of close friends for lunch on Boxing Day. It was a freezing cold day and gloriously sunny.

In those days Port Meadow, a large water meadow by the Thames in North Oxford, would flood and freeze for a couple of miles for a few days every year; and since the water was never more than a foot or so deep it was very safe to

[26] Those were the days!

skate on it. After lunch the children insisted that we all go skating. I had been an ice-hockey player until I was 18 and was very keen to explore the meadow: I'd never skated on anything larger than an ice rink.

But what about skates for me? Although the Hoares turned out to have enough skates to shoe a hockey team, my feet were unhoareishly huge. Here the law of spontaneous availability, known to all packrats, came to my rescue: if you don't throw old things away then one day they'll come in useful, if you can find them. The family had a mid-Victorian pair of skates that could be clamped to my shoes, and the attic they were hiding in was soon found.

The scene when we got to the meadow was like something out of Breugel. North Oxford was out on the ice: some folks skittering along on chairs, others falling about on skates. Tony himself skated joyfully and at breakneck speed in his everyday uniform of thigh-length grey coat and hat[27].

[27] The clamp-on experience wasn't completely satisfactory – they were quite blunt – but it was good enough to get me to revive my old hockey skates; and later to buy a more modern pair. I still skate on the frozen meadow whenever I can.

Linda Forrest (my partner), Tony and Jill Hoare: Addison's Walk, Magdalen College, late Summer 2015

My wife and I have felt part of the extended Hoare family for very many years now, and I've never forgotten the warmth of the welcome they gave me that day.

Recollections of Tony Hoare

Jonathan P. Bowen
Emeritus Professor, London South Bank University

Here I record some recollections of Professor Sir Tony Hoare [4]. As has been said elsewhere “Recollections may vary”, but these are my memories as I remember them. Apologies now for any infelicities.

I first encountered Tony for a job interview in 1984 at the Programming Research Group in Oxford to work as a “Research Officer” (or research assistant in normal UK university parlance) using the Z notation, which was entirely new to me then [7]. We initially met in the Common Room at 11 Keble Road where he made me a coffee and chatted informally before the interview, which certainly helped me feel at ease. I was then formally interviewed by Tony and Bernard Sufrin. Tony asked me what I knew about garbage collection, and I said I had heard of it. This seemed to be enough to be offered the job, although I guess there must have been some other questions too! I had the background of an “elevated engineer” rather than a “fallen mathematician” (an important intersection for computer science in my view!), but enjoyed the connection of theory and practice, as Tony famously did too.

I initially worked on the Distributed Computing Project at the PRG with Carroll Morgan and Roger Gimson, specifying network services using Z. Tony was more involved with CSP at the time, so I did not see him so much for my first few years at the PRG, but the door of his office was always open, with the possibility to see him at any time. An early memory was dinner with Tony and Jill in their Chalfont Road home, with others such as Cliff Jones.

Then in 1989, Tony invited me to dinner at St John’s College with the instruction to meet at the main gate of St John’s [5]. There I met for the first Dines Bjørner, from DTU in Denmark and a larger-than-life character as anyone who knows him will tell you, Cliff Jones, then at the University of Manchester after working at the PRG with Tony and others, and Hans Langmaack, based at the Christian-Albrechts-University of Kiel in northern Germany. Each had “henchmen” in attendance for discussions on what was to develop into the European ESPRIT ProCoS I and II projects [2,10] and then a Working Group [11] on “Provably Correct Systems”. I was to become very much involved with helping Tony on these projects, although I was employed on the parallel UK DTI SAFEMOS project on “Totally Verified Systems”, collaborating with Mike Gordon at the University of Cambridge, David May and David Shepherd at Inmos, and Roger Hale and John Herbert at Cambridge SRI, with Tony as the Oxford investigator. The research results were later more modestly recorded in a book entitled "*Towards Verified Systems*" [3]. Visits with Tony to Cambridge included meeting Roger Needham and David Wheeler. An important discussion was how to travel between Oxford and Cambridge, with Roger Needham favouring the four "Bs" route via Bicester, Buckingham, Bletchley (halfway), and Bedford, before reaching "C" (Cambridge). This may also be a reason for the location of Bletchley Park in World War II.

Meetings for the ProCoS projects took place in various European locations. I took on the role of Tony's "minder" during journeys, helping to ensure that Tony and his briefcase remained in the vicinity of each other and on one occasion ensuring that he boarded the train going in the right direction! Early in the project, Tony produced a typically beautifully elegant approach to compiling specifications in a relational style [17]. More modestly, I was able to show how easily these could be transformed into a Prolog logic program, for compiling and even for decompiling a high-level Occam-style program to/from Transputer-like machine instructions. On reflection, my job as Tony's minder may have been more important in the scheme of things, helping, in a small way, Tony to have higher thoughts.

As the PRG expanded, some of us were "banished" to 2 South Parks Road, still in the Science Area and not so far from the main location at 11 Keble Road but nicknamed "Tasmania" by Carroll Morgan (as an Australian!) due to its remoteness. Despite this, Tony regularly visited us to discuss research progress, when I was in a shared office with Paritosh Pandya and Mark Josephs. I had programmed my Sun workstation to issue me with audio reminders, recorded by my then very young daughter Alice (now a chemistry lecturer at the University of Manchester!). One pleaded, "Please come home Daddy" and of course, this was played back during one of Tony's visits towards the end of the day. Tony obligingly terminated the discussion on hearing this!

Other memories of Tony include his early use of electronic mail. Emails to Tony would be printed by his secretary, Julie Sheppard, and handed to him on paper. He answered these in his beautifully clear handwriting, and Julie typed the replies before emailing them back.

At one point, Tony gave a lecture course on CMOS circuits to undergraduates. These were beautifully modelled in a relational style at the transistor level, presented on handwritten acetates. With this approach, it was easy to transliterate the specifications very directly as logic programs, so I helped Tony by creating the practical laboratory worksheets based on Prolog for the students. I remember Tony sitting in front of a workstation near the start of the course and asking, “What do I do?”! Of course, all his handwritten specifications worked perfectly in Prolog, and it was a delight to help in a small way on the course, demonstrating for students during practical sessions.

At one point, just before Tony went on sabbatical leave, a large cardboard box containing a PC, with a Visual Basic manual, was to be seen in his office. I never dared ask Tony what this was doing there. Tony and the PC disappeared for a year. On his return, he confided that he had opened the box, turned on the PC, and written a Visual Basic program. The only problem was that he did not know how to save it, so we will probably never know what the program did.

In 1995, I left the PRG and became a lecturer at the University of Reading. We held the last ProCoS Working Group meeting there in 1997, which Tony attended. In 1999, I attended Tony’s retirement symposium in Oxford [13]. Such is Tony’s reputation that there were many ACM Turing Award winners there. Don Knuth lay in the front row, apparently asleep, perhaps due to jetlag after flying from Stanford in California, but often asking a very pertinent question at the end of presentations. Ole-Johan Dahl gave a wonderful presentation during which the lights and overhead projector failed due to a temporary power cut. From the gloom of the emergency lighting, Ole-John said, “I think I can go on”, which he duly did without slides, seamlessly continuing when the power returned to restore his slide presentation.

Tony subsequently moved to Microsoft Research in Cambridge, where he had to answer his emails on a computer, perhaps due to the lack of industrial secretarial support! In 2006, I was invited to interview Tony in Cambridge, for the Computer History Museum in California [8]. As an amateur concerning interviewing, the experience could have been nerve-racking for me, but the questions were well-prepared, and Tony put me at ease as he was able to do with most people in his always modest way. An abridged version later appeared in the *Communications of the ACM* [18].

Tony also attended the celebratory ProCoS reunion workshop in 2015 [15], which I helped to organize as Chair of FACS at the BCS London office, then in Covent Garden. Dines Bjørner presented his memories of how the ProCoS project started, including notes that Tony wrote in Dines’ notebook on ideas to form the basis of research on the project. Tony kindly authored the foreword in the subsequent proceedings, written in his characteristically generous and eloquent style.

Walking with Tony was always an experience. He and I both like walking and at reasonable speed too. My last walk with Tony was in Reykjavik, Iceland, for the

UTP (Unifying Theories of Programming) Symposium there in 2016 [12]. He and his longtime collaborator Jifeng He, especially on UTP, were both keynote speakers. Tony's mind was as sharp as ever in our discussion.

The last time I saw Tony was in 2019 for the celebration of the 40th anniversary of FACS and the 20th anniversary of Tony's book on *Unifying Theories of Programming* with Jifeng He [16]. Jifeng gave the main talk, with Tony delivering some introductory commentary and Jim Woodcock providing a summary at the end [6]. It was gratifying that they could meet on this occasion, with Jifeng visiting from Shanghai in China.

After my job interview with Tony at the PRG in 1984, I spent the summer working at the then start-up company Silicon Graphics, in Mountain View at the heart of Silicon Valley in California. My Engineering Science tutorial partner at Oxford was an American and he returned to study for a master's degree at Stanford University in the US under Jim Clark, who invited him to become one of the seven founder members of the company. At the time I was also a research assistant in the Wolfson Microprocessor Unit at Imperial College, with the offer of a job at the PRG (at about a quarter the salary of that at Silicon Graphics). Should I stay on after three months at Silicon Graphics, receive stock options, and potentially become a millionaire? Or should I return to work under Tony Hoare at the PRG in Oxford, also my birthplace? Naturally, I chose the latter, such was the draw of Tony and Oxford! In any case, I would probably have lost my millions in the dotcom crash, knowing my lack of business acumen. This tribute is to thank Tony for that major decision point in my career. And of course, to congratulate him on his lifetime of contributions to computer science, always done in a very modest way, which I admire immensely.

Happy 90th birthday, Tony.

References

1. Bjørner, D. (2017). *ProCoS: How It All Began – as Seen from Denmark*. In [15], pp. 3–6. https://doi.org/10.1007/978-3-319-48628-4_1
2. Bjørner, D., Hoare, C.A.R., Bowen, J.P., He, J., Langmaack, H., Olderog, E.R., Martin, U., Stavridou, V., Nielson, F., Nielson, H.R., Barringer, H., Edwards, D., Løvengreen, H.H., Ravn, A.P., Rischel, H. (1989). A ProCoS project description: ESPRIT BRA 3104. *Bulletin of the European Association for Theoretical Computer Science*, 39:60–73. https://researchgate.net/publication/256643262
3. Bowen, J.P. (ed.) (1994). *Towards Verified Systems*. Real-Time Safety Critical Systems, vol. 2. Elsevier.

4. Bowen, J.P. (2001). Hoare, C. Antony R. In: Rojas, R. (ed.), *Encyclopedia of Computers and Computer History*, pp. 368–370. Fitzroy Dearborn Publishers.
5. Bowen, J.P. (2019a). *A Personal Formal Methods Archive*. ResearchGate. https://doi.org/10.13140/RG.2.2.31943.65447
6. Bowen, J.P. (2019b). FACS Events, 2018–2020. *FACS FACTS*, 2019(1):7–18, December 2019. https://www.bcs.org/media/5204/facs-dec19.pdf
7. Bowen, J.P. (2022). Review on Theories of Programming: The Life and Works of Tony Hoare. *Formal Aspects of Computing*, 34(3–4):1–3. https://doi.org/10.1145/3560267
8. Bowen, J.P., Hoare, C.A.R. (2006). *Oral history of Sir Antony Hoare*. Tech. Rep. X3698.2007, Computer History Museum, California, USA, September 2006, Catalog Number 102658017. https://www.computerhistory.org/collections/catalog/102658017
9. Bowen, J.P., Hoare, C.A.R., Hansen, M.R., Ravn, A.P., Rischel, H., Olderog, E.-R., Schenke, M., Fränzle, M., Müller-Ulm, M., He, J., Zheng, J. (1994). Provably Correct Systems – FTRTFT'94 tutorial. In: *Third International School and Symposium, Formal Techniques in Real Time and Fault Tolerant Systems*, Lübeck, Germany, September 1994. ProCoS document [COORD JB 7/1]. https://researchgate.net/publication/2420842
10. Bowen, J.P., Hoare, C.A.R., Langmaack, H., Olderog, E.R., Ravn, A.P. (1996). A ProCoS II project final report: ESPRIT Basic Research project 7071. *Bulletin of the European Association for Theoretical Computer Science*, 59:76–99. https://researchgate.net/publication/2255515
11. Bowen, J.P., Hoare, C.A.R., Langmaack, H., Olderog, E.-R., Ravn, A.P. (1998). A ProCoS-WG Working Group final report: ESPRIT Working Group 8694. *Bulletin of the European Association for Theoretical Computer Science*, 64:63–72. https://researchgate.net/publication/2527052
12. Bowen, J.P., Zhu, H. (eds.) (2017). *Unifying Theories of Programming: 6th International Symposium, UTP 2016, Reykjavik, Iceland, June 4–5, 2016*. LNCS, vol. 10134. Springer. https://doi.org/10.1007/978-3-319-52228-9
13. Davies, J., Roscoe, B., Woodcock, J. (2000). *Millennial Perspectives in Computer Science, Proceedings of the 1999 Oxford–Microsoft Symposium in honour of Sir Tony Hoare*. Cornerstones of Computing, Palgrave.
14. Denvir, T. (2013). Peter Landin Annual Semantics Seminar: Professor Sir Tony Hoare (December 3rd 2012). *FACS FACTS*, 2013(1):5–7, December 2013. https://www.bcs.org/media/3083/facs-dec13.pdf
15. Hinchey, M.G., Bowen, J.P., Olderog, E.-R. (eds.) (2017). *Provably Correct Systems*. NASA Monographs in Systems and Software Engineering, Springer. https://doi.org/10.1007/978-3-319-48628-4
16. Hoare, C.A.R., He, J. (1999). *Unifying Theories of Programming*. Prentice Hall International Series in Computer Science. http://www.unifyingtheories.org

17. Hoare, C.A.R., He, J., Bowen, J.P., Pandya, P. (1990). An algebraic approach to verifiable compiling specification and prototyping of the ProCoS level 0 programming language. In: Directorate-General XIII of the Commission of the European Communities (ed.), *ESPRIT '90 Conference*, pp. 804–818. Kluwer Academic Publishers. https://doi.org/10.1007/978-94-009-0705-8_65
18. Shustek, L. (2009). An interview with C.A.R. Hoare. *Communications of the ACM*, 52(3):38–41. https://doi.org/10.1145/1467247.1467261

Jonathan Bowen and Tony Hoare during the FTRTFT'94 conference in Lübeck, Germany, September 1994 [9].

Tony Hoare and Jonathan Bowen at dinner after Tony's BCS-FACS Peter Landin Semantics Seminar in London, 3 December 2012 [14]. (Photograph by Sue Black.)

Jonathan Bowen, Tony Hoare, and Jifeng He at the UTP Symposium, Reykjavik, Iceland, September 2016 [12].

Joint LMS / BCS-FACS Seminar 2024

Formalising 21st-Century Mathematics

Lawrence C. Paulson FRS
University of Cambridge
https://www.cl.cam.ac.uk/~lp15/

15th January 2024

Reported by: Andrei Popescu, University of Sheffield

LMS Webpage: https://www.lms.ac.uk/events/lms-bcs-facs-seminar-2024

Video: https://www.youtube.com/watch?v=nxlpYp8bKdc

Webpage: https://www.bcs.org/events-calendar/2024/january/webinar-formalising-21st-century-mathematics/

Abstract: The formalisation of mathematics is an ongoing process that arguably started as early as the 19th century, intensified with the foundational crisis at the start of the 20th century, and since the 1970s has been conducted with the help of computers. Recent decades have seen the machine formalisation of lengthy and technically complicated proofs, but some have argued that even these were not representative of modern mathematics. Recent achievements by a number of different groups are starting to challenge this scepticism. The speaker will outline some of these, while also noting some of the remaining trouble spots.

Biography: Lawrence Paulson is a Fellow of the Royal Society and a Professor of Computational Logic at the University of Cambridge, a Fellow of Clare College. He has made fundamental contributions to the science and art of theorem proving and functional programming (including pedagogically, as the author of the textbook "*ML for the Working Programmer*" which many functional programmers grew up with).

After being involved in work on the LCF family of provers based on the ideas of Dana Scott and Robin Milner (and creating the so-called "Cambridge" variant of the LCF system), in 1986 Paulson created the generic interactive theorem prover Isabelle. He has been developing Isabelle ever since together with Tobias Nipkow, Makarius Wenzel, and several students and postdocs.

Today, Isabelle, mostly through its Isabelle/HOL object logic, is one of the most widely used theorem provers, and really one of the most widely used formal methods tools in computer science and mathematics. Paulson has opened or co-opened several important directions in the field of theorem proving, including resolution-based higher-order proof management, the cooperation between interactive and automatic provers (leading to Isabelle's Sledgehammer tool and other "hammers" it inspired), the semantic approach to the mechanisation of inductive and coinductive definitions, and the inductive method for certifying cryptographic protocols. As a perhaps not so widely acknowledged contribution, his Generic Isabelle theorem prover with its technique for representing object logics was one of the first successful incarnations of Higher-Order Abstract Syntax, developed at about the same time as systems such as Edinburgh LF.

Paulson also formalised deep results in set theory and logic, including the logicians' favourite: the first-ever mechanical proof of Gödel's second incompleteness theorem. Recently, Paulson has developed the MetiTarski prover specialized on real-valued functions, encompassing wisdom from the practice of working mathematicians. Through his recent Advanced ERC grant "Alexandria", Paulson and his team were able to pursue in a very focused manner Paulson's long-standing commitment: making theorem provers capable of tackling state-of-the-art contemporary mathematics, and turning them into tools that will eventually support the day-by-day creation of mathematics.

In 2017, Paulson received the prestigious Herbrand Award for Distinguished Contributions to Automated Reasoning.

Summary: Paulson's talk was a fascinating journey through the history of capturing mathematical thinking by formal rule-based systems going back to Euclid and culminating with today's state-of-the-art powered by mature and sophisticated proof assistants for machine-checked proof development.

In the past few years, Paulson has led the ERC project "Alexandria" (https://www.cl.cam.ac.uk/~lp15/Grants/Alexandria/) on formalising modern mathematics, and building infrastructure for such formalisations, in the Isabelle/HOL proof assistant. His talk gave an overview of this work, which includes impressive formal developments covering areas such as Grothendieck schemes, transfinite combinatorics and random graph theory. He discussed the infrastructure required to formalise the sophisticated mathematics involved in these areas, noting that Isabelle/HOL's simple type theory foundation turned out to be sufficient. In this context, Paulson also touched upon alternative foundations based on dependent types that underlie other mainstream proof assistants such as Agda, Coq and Lean. Part of the work developed within Paulson's project seems to have been stimulated by a fruitful rivalry with formal developments in Lean led by the mathematician Kevin Buzzard.

The talk lasted for one hour and was followed by a lively Q&A of almost half an hour. In addition to the above main topics, other topics addressed in the talk or the Q&A were the relevance of mathematicians in a presumptive world with fully decidable mathematics, AI's (current and future) role in theorem proving, representation independence, formal translations between different representations, and equational proofs versus natural deduction.

Both the talk and the Q&A are available online (see the links above) and we warmly invite anyone interested in formalised mathematics to watch it.

The SI Digital Framework: Underpinning FAIR measurement data

Dr Jean-Laurent Hippolyte,
National Physical Laboratory

20th February 2024

Reported by: Keith Lines, National Physical Laboratory

Video: https://www.youtube.com/watch?v=ui1ZzsaqVGQ

Webpage: https://www.bcs.org/events-calendar/2024/february/webinar-the-si-digital-framework-underpinning-fair-measurement-data

Abstract: The International Bureau of Weights and Measures (BIPM) was established in 1875 to coordinate matters related to measurement science and measurement standards worldwide. It is the home of the International System of Units (SI), the recommended practical system of units of measurement, and of the international reference time scale (UTC).

Through a framework known as the “International Committee for Weights and Measures Mutual Recognition Arrangement (CIPM MRA)”, the comparability of measurements and measurement standards between all the participating institutes is assured, along with the recognition of measurement results and calibration certificates between these national institutes.

As recommended by the General Conference of Weights and Measures in its Resolution 2 of 2022, the BIPM is developing an SI Digital Framework to provide machine-actionable reference points to support the international measurement system in the digital era.

This talk will introduce the general principles and the building blocks of the SI Digital Framework, link them to essential resources of the measurement community and demonstrate their machine-actionability.

Biography: Dr Jean-Laurent Hippolyte is a senior scientist in the Data Science department of the National Physical Laboratory, the UK’s National Metrology Institute, where his research focusses on modelling and quality management for measurement data.

He has over 10 years of experience as a researcher in object-oriented design, distributed computing and semantic technologies applied to interdisciplinary engineering and scientific challenges.

Introduction

The National Physical Laboratory's work with ontologies for use in metrology (the science of measurement) has progressed steadily since Clifford Brown gave a presentation for FACS in 2019 on Ontologies for Data Provenance and Curation [1].

In February of this year Jean-Laurent Hippolyte presented a webinar on the SI Digital Framework [2], which has been launched recently by the International Bureau of Weights and Measurements (BIPM [3]). Jean-Laurent has been seconded from NPL to BIPM, to be a member of the framework's development team.

An ontology providing a digital representation of the International System of Units (SI) lies at the heart of the framework.

The webinar began with some background information to provide context. An overview of the framework was then given, which included a brief demonstration. The evening ended with a lively question-and-answer session, during which Jean-Laurent was assisted by colleagues from BIPM.

Feedback on the SI Digital Framework will be greatly welcomed.

Background: Metrology

The webinar began with a brief overview of NPL, the UK's national metrology institute, and then outlined some important concepts from metrology. For example, traceability chains ensure the validity of measurements through an unbroken sequence of calibrations. International intercomparisons, coordinated by BIPM, ensure that measurement results provided by national metrology institutes are consistent with each other.

The International Bureau of Weights and Measurements (BIPM) is an intergovernmental organisation, established by 17 nations signing the Metre Convention in 1875. Britain signed the convention in 1884. There are currently 64 member states and 36 associate states and economies. The work of the BIPM is overseen by the International Committee for Weights and Measures (CIPM), which is in turn elected by the General Conference on Weights and Measures (CGPM).

In 2022 the CGPM recognised the requirement for a digital representation of the SI [4], anticipating that "…maintaining and building confidence in the accuracy and global comparability of measurements will require… a full digital representation of the SI, including robust, unambiguous, and machine-actionable representations of measurement units, values and uncertainties".

Background: Ontologies

A brief description of ontologies was provided, as it was not assumed the audience was familiar with the subject. Ontologies are shareable, reusable and computable representations of knowledge based on set theory and first-order logic.

The underlying concept is the subject-predicate-object triple. Ontologies are typically visualised as directed graphs, illustrating how nodes representing subjects and objects are linked with arrows representing predicates. Figure 1 provides a small example from the ontology developed for the SI Digital Framework.

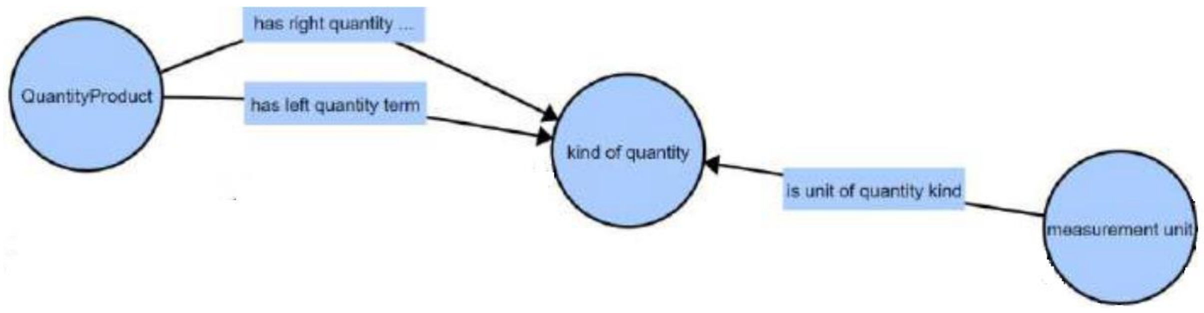


Figure 1: *An example from the SI Digital Framework SI ontology*

The Terse Resource Description Framework Triple Language (Turtle or TTL) syntax of the Web Ontology Language (OWL [5]) was used to write an ontology representing the SI. The TTL files are available on a Git repository [6].

The SI Digital Framework

The SI Digital Framework consists of the following components: a set of OWL ontologies, Web APIs for machine access and Web front ends for human access.

The root URI https://si-digital-framework.org can have suffixes appended as illustrated in figure 2. For example, the URI for the SI units is https://si-digital-framework.org/SI/units and the URI for the defining constants of the SI is https://si-digital-framework.org/constants

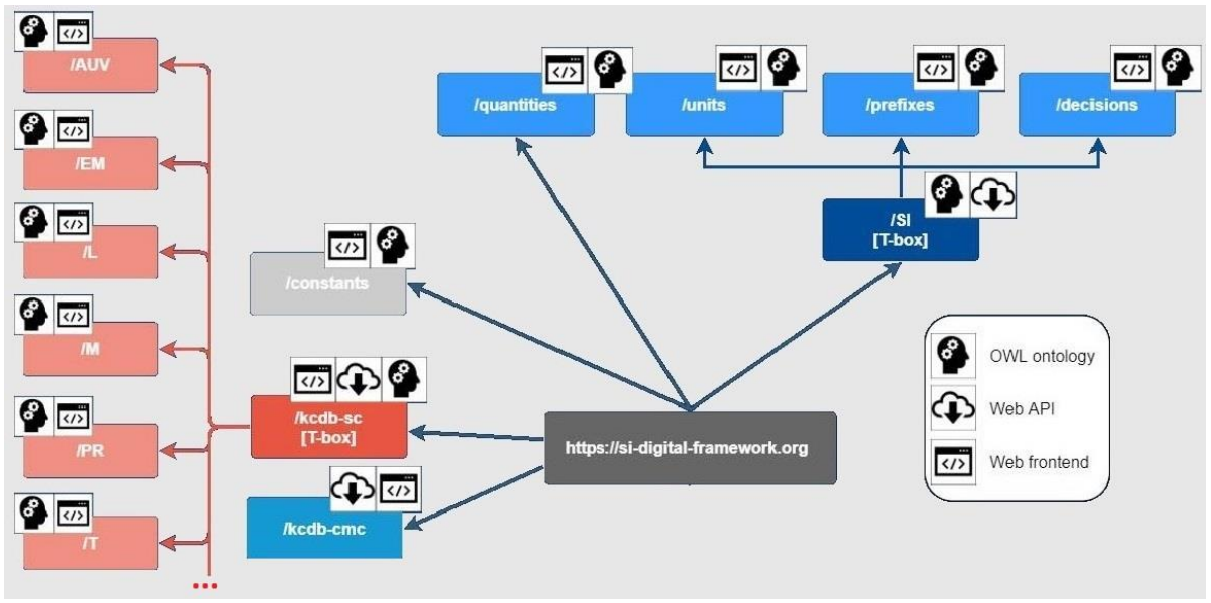


Figure 2: *A section of the SI Digital Framework namespace hierarchy*

Examples of how the SI Digital Framework helps support the FAIR [7] principles include:

- **F**indability through resolvable URI-based persistent identifiers.
- **A**ccessibility through standard web languages and protocols (e.g., OWL, SPARQL [8]).
- **I**nteroperability through relying on existing FAIR ontologies where possible.
- **R**eusability through acting as an authoritative digital reference for the SI.

Classifications of service categories

Jean-Laurent has largely focused on the classifications of service categories published by eight of the Consultative Committees of the CIPM. Machine-interpretable classifications of service categories across metrology areas have been developed: https://si-digital-framework.org/kcdb-sc

Question and answers

The presentation ended with a Q+A session. Topics covered included whether this framework could one day replace the SI Brochure [9],

References

1. FACS FACTS, Issue 2019-1, December 2019: https://www.bcs.org/media/5204/facs-dec19.pdf
2. SI Digital Framework: https://si-digital-framework.org/
3. International Bureau of Weights and Measurements (BIPM): https://www.bipm.org/en/
4. Resolution 2 of the 27th CGPM (2022): https://www.bipm.org/en/cgpm-2022/resolution-2
5. OWL, web ontology language: https://www.w3.org/OWL/
6. The SI Digital Framework public repository: https://github.com/TheBIPM/SI_Digital_Framework
7. GO FAIR Initiative: https://www.go-fair.org/fair-principles/
8. SPARQL Protocol and RDF Query Language: https://www.w3.org/TR/rdf-sparql-query/
9. SI Brochure: https://www.bipm.org/en/publications/si-brochure

Scott models for probabilistic computation

Abbas Edalat
Imperial College, London

26/03/2024

Reported by Keith Lines, National Physical Laboratory

Video: https://www.youtube.com/watch?v=2Bd_VNdfk5A

Webpage: https://www.bcs.org/events-calendar/2024/march/webinar-scott-models-for-probabilistic-computation/

Abstract: Scott domains, or more generally, continuous domains, have been the traditional framework for semantics of programming languages.

It has however been a key open problem since 1980s to develop a model of probabilistic semantics based on continuous domains – the obstacle being that no appropriate Cartesian closed category (CCC), closed under the probabilistic power domain, is known to exist.

In this talk, bypassing the latter stumbling block, we provide a solution to this long-standing problem.

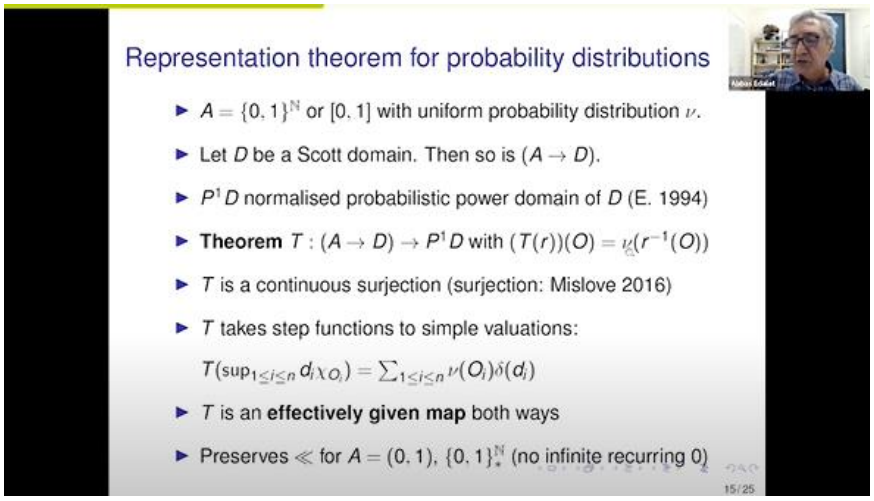
We show that the probabilistic power domain of a Scott domain can be equally well represented by the Scott domain of random variables from any standard probability space to the given Scott domain: there is an effectively given, surjective and Scott continuous map from this domain of random variables to that of the probabilistic power domain of the underlying Scott domain.

This map simply takes any random variable on the domain to its associated probability distribution in the probabilistic power domain, crucially by preserving canonical basis elements. By enriching the category of Scott domains with a partial equivalence relation – to capture the equivalence of random variables – we obtain a CCC.

We can then develop four canonical commutative monads for constructing random variables from four standard probability spaces to objects of this category. We show that all basic probability distributions on finite dimensional Euclidean spaces can be denoted by their corresponding random variables in this framework.

Biography: Abbas Edalat is a Professor of Computer Science and Mathematics at Imperial College, London. He came to Imperial having worked previously at Sharif University of Technology in Iran. In 1990s and 2000s, he uncovered connections between domain theory and several branches of mathematics, including measure and integration theory, fractal geometry, computational geometry, exact computation, differential calculus and ODEs, leading to new computational models and algorithms in these subjects.

Overview: Probabilistic computation has an important role in subjects such as machine learning and quantum computing. The task of providing a formal semantics for languages that implement such computations is therefore increasingly important.



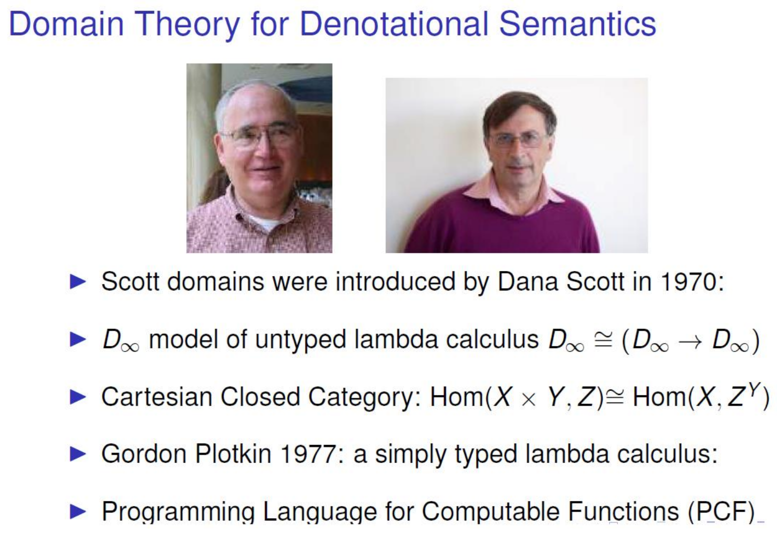
Representation theorem for probability distributions

- $A = \{0, 1\}^{\mathbb{N}}$ or $[0, 1]$ with uniform probability distribution ν.
- Let D be a Scott domain. Then so is $(A \to D)$.
- $P^1 D$ normalised probabilistic power domain of D (E. 1994)
- **Theorem** $T : (A \to D) \to P^1 D$ with $(T(r))(O) = \nu(r^{-1}(O))$
- T is a continuous surjection (surjection: Mislove 2016)
- T takes step functions to simple valuations:
  $T(\sup_{1 \leq i \leq n} d_i \chi_{O_i}) = \sum_{1 \leq i \leq n} \nu(O_i)\delta(d_i)$
- T is an **effectively given map** both ways
- Preserves $\ll$ for $A = (0, 1)$, $\{0, 1\}^{\mathbb{N}}_{*}$ (no infinite recurring 0)

15/25

Figure 1: *Representation theorem*

Abbas Edalat described the novel mathematical work combining existing elements of probability theory and denotational semantics (Scott domains) to create a semantic model for probabilistic computation. This work was carried out with Pietro Di Gianantonio of the University of Udine, who was also present online.



Domain Theory for Denotational Semantics

- Scott domains were introduced by Dana Scott in 1970:
- D_∞ model of untyped lambda calculus $D_\infty \cong (D_\infty \to D_\infty)$
- Cartesian Closed Category: Hom$(X \times Y, Z) \cong$ Hom(X, Z^Y)
- Gordon Plotkin 1977: a simply typed lambda calculus:
- Programming Language for Computable Functions (PCF)

Figure 2: *Scott domains*

The presentation began with a description of the axiomatization of probability theory by Andrey Kolmogorov. It then moved on to a detailed overview of Scott domains[28], which for over 50 years, have been applied to providing denotational semantics for non-probabilistic computation.

A more complete description of this important work can be found here [1]. Further developments will hopefully include a Haskell implementation.

References

1. Pietro Di Gianantonio, Abbas Edalat. *A Cartesian Closed Category for Random Variables*, June 2024, arXiv:2402.11727v2

[28] A useful, introductory discussion of domains can be found in the July 2021 edition of *FACS FACTS*: *Domain Theory – Revisited* by B. Monahan, *FACS FACTS*, Issue 2021-2, pp 5–29, https://www.bcs.org/media/7577/facs-jul21.pdf

Verifying system-level properties of neural-network robotic controllers

Jim Woodcock, University of York

28th May 2024

Reported by: Alvaro Miyazawa, University of York

Video: https://www.youtube.com/watch?v=2dcaHTvgOWM

Webpage: https://www.bcs.org/events-calendar/2024/may/hybrid-event-verifying-system-level-properties-of-neural-network-robotic-controllers/

Abstract: Verifying learning-enabled robotic systems is challenging. There are existing techniques and tools for verifying artificial neural networks (ANNs), but they are focused on component-level properties.

We verify robotic systems with ANN control components. Currently, we focus on trained, fully connected ReLU neural networks for control. We use RoboChart, a domain-specific robot modelling and verification framework.

In RoboChart, we combine behavioural and ANN models, and we combine traditional and ANN-specific verification tools: Isabelle/HOL and Marabou. We report on exploratory work. In future, we will address the training phase, other ANN classes, and more sophisticated activation functions.

We collaborate with Ziggy Attala, Ana Cavalcanti, and Matt Windsor in the RoboStar Centre (robostar.cs.york.ac.uk) and colleagues at the Chinese Academy of Sciences.

Biography: Jim Woodcock attended the University of Liverpool, where he was awarded a BSc (Hons) in Computational Science (1977), an MSc in Operational Mathematics (1978), and a PhD in Computation (1980). He worked at the GEC Hirst Research Centre from 1980 to 1984, where he rose from Research Scientist to Principal Research Scientist and GEC Research Fellow.

In 1984, he joined the Department of Electrical and Electronic Engineering at the University of Surrey as a Lecturer in Information Technology. In 1985, he moved to the Programming Research Group at the University of Oxford as a research assistant to work with Tony Hoare and Ib Holm Sørensen on a collaborative project with IBM Hursley formalising the CICS transaction processing system.

This won the Queen’s Award for Technological Achievement in 1996. During this time, he was a Junior Research Fellow at Wolfson College (1985–87) and an Atlas Fellow at Pembroke College and at the Rutherford-Appleton Laboratory (1987-91). He was appointed as a Lecturer in Computation in 1994, promoted to a Readership in Software Engineering in 1997, and appointed to a personal chair in 2000. He has been a Fellow of Kellogg College in Oxford since 1994.

In 2001, he moved to the University of Kent and in 2004 he moved to the University of York, in both cases as Professor of Software Engineering. He has been head of the Department of Computer Science at York since 2012. He was appointed to a Fellowship of the Royal Academy of Engineering in 2011.

Summary: The RoboStar Centre and FACS hosted a hybrid seminar on the verification of neural-network robotic controllers by Jim Woodcock of the University of York. The seminar had 29 online attendees and 12 in-person attendees.

The talk started by discussing the rationale behind verifying artificial neural networks (ANN) and the importance of verifying system-level properties as well as component-level properties. Several motivating examples were explored, and the pros and cons of using neural networks as robot controllers were discussed.

The vision of the RoboStar Centre was presented, identifying the use of models and model-based software engineering techniques as the basis for achieving a multitude of results, including simulations, tests, and proofs. RoboChart, the core notation supporting the RoboStar vision, and the RoboStar modelling stack were briefly discussed.

The modelling stack showed different aspects to be modelled (control, operational requirements, physical models, etc.) in the horizontal dimension and different contexts and paradigms (reactive, cyclic, simulation, and deployment) in the vertical dimension. Furthermore, a flow chart showed how these different notations, tools and techniques come together to form a cohesive and comprehensive development approach.

Jim discussed the extensions to the RoboStar visions that support the modelling and verification of systems with neural network components. The requirements for replacing standard controllers with neural networks are established, and the verification approach is explained. The approach relies on the standard semantics of RoboChart models and the novel semantics of RoboChart ANN and uses a special notion of conformance to compare standard and ANN controllers. The verification of conformance is done via a combination of the theorem prover Isabelle/UTP and the neural network verification tool Marabou.

The approach is explained in more concrete terms, and a simple example is discussed in more detail. Furthermore, an approach for specifying ANNs is developed in terms of reactive contracts involving pre-, post-, and periconditions[29], and the definition of the conformance relation is given.

The talk concluded with a summary of the main contributions and future work and a discussion with the audience about the potential future directions for the area.

RoboStar made a Research Centre

The University of York's Department of Computer Science is proud to announce the establishment of RoboStar, a cutting-edge research centre dedicated to Software Engineering for Robotics. Launched five years ago with generous support from the Royal Academy of Engineering, the UK Research and Innovation Council, and industry collaborators, RoboStar has emerged as an international hub of excellence.

Situated at the forefront of innovation, RoboStar addresses the most pressing challenges in robotics. The university's recent recognition of RoboStar emphasises the centre's commitment to improving design and verification methodologies for roboticists. Their work will contribute to safety standards and reduce the overall development and operation costs of mobile and autonomous robots.

With branches across six countries and collaborations involving 13 academic and industrial organisations, RoboStar has become a global force. Members of RoboStar are grateful for the widespread contributions from colleagues worldwide who actively participate in the advancement and application of RoboStar technology.

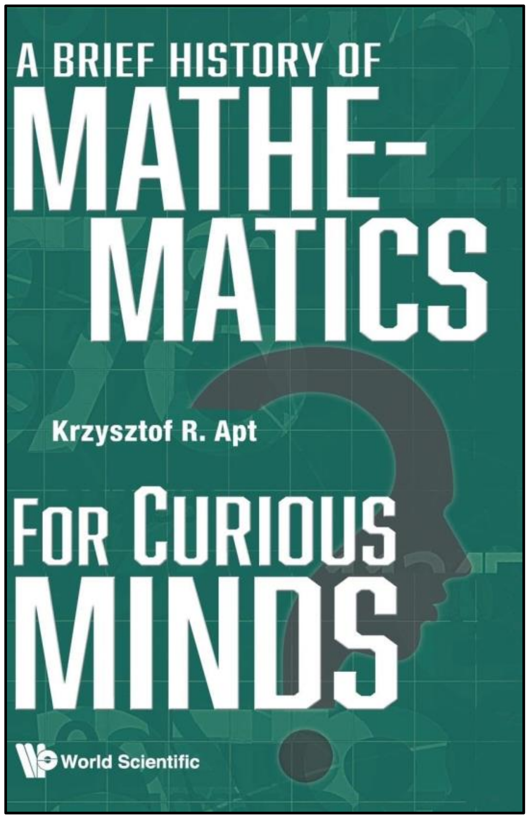
[29] A *pericondition* is a term used within the UTP world for a certain kind of reactive system invariant. See the paper: *Unifying theories of reactive design contracts* by Foster, Calvacanti, Canham, Woodcock and Zayda, TCS, vol 802, pp 105–140, Jan 2020, https://doi.org/10.1016/j.tcs.2019.09.017

Book Review

A Brief History of Mathematics for Curious Minds

by Krzysztof R. Apt

Reviewed by Brian Monahan



A Brief History of Mathematics for Curious Minds
by Krzysztof R. Apt, World Scientific Pub., 2023, 210pp, **ISBN: 978-9811280443**

A senior member of the international formal methods community strongly recommended that *FACS FACTS* take a look at this book—and I'm very glad that he did! It is written by Krzysztof Apt, a professor emeritus at the University of Amsterdam, well known for his computer science work in program verification and constraint programming but also, more recently, at CWI, for his work in game theory and economic mechanism design.

The title declares what this book is all about. The part about "Curious Minds" hints that it may not be quite the somewhat dry and straightforward account of mathematical history that one might have otherwise expected. Indeed, not only is it delightfully packed full of all sorts of curious and surprising facts about the mathematicians who brought mathematics to light, but it fortunately also contains many short appendices (32 in all!) explaining various mathematical topics and theorems at an intermediate level. I can think of nothing more frustrating or tedious than a book discussing mathematics at some length – but then comprehensively neglecting to do any! Fortunately, that particular criticism *doesn't* apply here.

Although reflecting its historical development, it is the mathematics itself that drives the content of this book. From the above, you might be forgiven for thinking that the book is merely a historical compendium, running through the history of mathematics with some potted biographies of various notable mathematicians. Well, that remark is only occasionally true – yes, the book is historically organised, but it is also far from a series of "potted biographies". The author's approach is to generally find something unusual, notable and interesting to say about the various mathematicians being written about.

To begin with, in Chapter 1, we find some of the ancient origins of numbers and so on being discussed – but there are no individuals to speak of there. It is only in Chapter 2, focusing upon the Greeks, that we do come across particular individual mathematicians – but even there, more often than not, it is particular *communities* of mathematicians that are (all too) briefly discussed, focussing upon a diverse range of topic areas, such as geometry, astronomical calculations, the nature of Infinity, Greek number notations and so on. All of the well-known greats of Greek mathematics (spanning nearly 1000 years from the 6th century BCE onwards) are mentioned here – Euclid, Pythagoras, Archimedes, Ptolemy, Diophantus, beginning with Thales of Miletus (his theorem is discussed in Appendix 1).

Inevitably, the account here includes some non-mathematicians in addition to strict mathematicians – because mathematics wasn't entirely separated from other science-related pursuits; this only happened much later with the rise in applications and their technical complexity. A good case in point is the extraordinary mathematician and engineer *Heron of Alexandria*[30], who appears to have taught a wide range of subjects at the *Musaeum* in Alexandria, which included the famous Library. Mathematically, Heron is famous for his formula relating the semi-perimeter length and sides of a triangle to its area: if a, b, and c are the lengths of the sides:

$$\text{Area} = \sqrt{s(s - a)(s - b)(s - c)}$$

where s is half the perimeter, or $(a + b + c)/2$. What's more, he is credited with writing about how pure geometry can, via early surveying instruments like the *dioptra*[31], show how two teams can tunnel from opposite sides of a hill and ensure a close enough meeting in the middle. Quite remarkable.

But the Greeks weren't the only developers of mathematics in ancient times. Chapter 3 briefly discusses ancient Chinese and Indian mathematical developments. The author observes that Chinese mathematics focuses mostly on empirical matters, very different from the Greek focus on principles and deduction. Even so, there is evidence that Chinese mathematicians knew of and had demonstrated for themselves Pythagoras's theorem – but clearly far *later* than the Greeks had. Of course, Hindu mathematicians discovered and developed the Hindu-Arabic numeral notation. However, because of its wider ramifica-

[30] https://en.wikipedia.org/wiki/Hero_of_Alexandria
[31] https://en.wikipedia.org/wiki/Dioptra

tions, this topic is discussed in detail in the following chapter on Roman and medieval mathematics, spanning from the 5th century to the 15th.

The next three chapters cover the period from the 15th to the 19th century and briefly discuss, for example, the introduction of algebraic notation, complex numbers, the general solution of equations of third and fourth degree – but not the fifth, the birth and development of calculus, probability theory, and the emergence of more systematic algebraic notions such as Group and Galois theory, by many great mathematicians including such names as Fibonacci, the Bernoulli brothers, Euler and Gauss. That's a lot of mathematics!

Chapter 8 brings the topic up to date by discussing mathematics in the 20th and 21st centuries. Because of mathematics' extraordinary super-exponential growth, this discussion tends to be sketchy, as expected. Even so, computer science is briefly represented here by John Von Neumann, Alan Turing, Stephen Cook, Edsger Dijkstra, Donald Knuth, Tony Hoare, and Leonid Levin. Although both John Von Neumann and Alan Turing get a few paragraphs, the other mentions are far more perfunctory – for example, Tony Hoare is mentioned solely for *quicksort* – but not for Floyd-Hoare logic or, indeed, for CSP or UTP!

It is impossible to compactly mention all the topics and mathematicians that are all too briefly touched upon in this book – it is certainly an achievement to capture such a broad range of topics, albeit in somewhat restricted discussion. After all, this book is clearly the author's personal take on mathematics in a historical context overall. Consequently, this book is primarily about *classical* mathematics and tends to emphasise calculation over the conceptual. To that end, as far as I could tell, modern areas and approaches such as *lambda calculus, topology* and *category theory* were not even mentioned *anywhere.* For instance, although Henri Poincare is indeed substantially referred to in connection with the n-body dynamics problem and the stability of the Solar System, there was still no mention of topology, even when the famous Poincare conjecture is very briefly alluded to when discussing Gregori Perelmann's solution!

This book does not claim to be either comprehensive or definitive. However, it will no doubt prove useful as a reference guide to (classical) mathematics for the interested layperson who may have heard about some mathematical topic in another context. It is a shame that it doesn't extend to some of the more modern topics, but, of course, the author must finish somewhere! It is quite remarkable what has been packed into these 210 pages, with copious references, notes and even a smattering of colour photos and diagrams. For me, there is always the great charm in the 32 appendices discussing actual mathematics, i.e., what it's all *truly* about.

Report on the SETSS 2024 Spring School

Jonathan P. Bowen

第六届可信软件系统工程（国际）春季学校
THE 6th SPRING SCHOOL ON ENGINEERING TRUSTWORTHY SOFTWARE SYSTEMS (SETSS)
15-21

Background

The School on Engineering Trustworthy Software Systems (SETSS) series was established ten years ago by my colleague Prof. Zhiming Liu in the summer of 2014 at Southwest University (SWU) in Chongqing, China. SETSS provides extended lecture courses on computing topics by lecturers from around the world. Since 2014, there have been further Schools held in the spring due to the more clement weather, annually from 2016 to 2019. SETSS is aimed at Ph.D. and Master students, from around China and elsewhere, as well as being suitable for university researchers and industry software engineers. The series has become like a Chinese version of the European Marktoberdorf Summer School held annually in Germany.

Due to the COVID pandemic, the School was not held during 2020–2023. However, it has now been re-established in 2024. All previous Schools have had post-proceedings published in the Springer Lecture Notes in Computer Science (LNCS) series (volumes 9506, 10215, 11174, 11430, and 12154) and this is planned to continue. I have been involved with the School since its inception, as a course lecturer in 2014, evening seminar lecturer in 2016 and 2018, and co-organizer and main editor of the proceedings from 2016. I have been an Adjunct Professor at Southwest University since 2017 and I am on SETSS 2024's academic committee. The 6th School on Engineering Trustworthy Software Systems (SETSS 2024) was held during 15–21 April 2024, again at Southwest University, Chongqing, China.

SETSS 2024 was organized by the School of Computer and Information Science, in particular the Centre for Research and Innovation in Software Engineering (RISE), at Southwest University (SWU), providing lectures on current research in methods and tools for use in computer system engineering. The SETSS aims to enable participants to learn about state-of-the-art software engineering methods and technology advances from experts in the field. Five days of extended lecture courses were augmented in 2024 for the first time by a two-day workshop, with additional invited and submitted presentations.

The opening session of SETSS 2024 was chaired by Prof. Zhiming Liu, head of RISE in the School of Computer and Information Science. A welcome speech was delivered by the Vice General Secretary of Southwest University, Prof. Yufeng Xia (see right), translated into English by Prof. Zili Zhang, followed by an introductory briefing for SETSS 2024 by Prof. Liu. The session finished with a photograph of participants at the School (see below).

第六届可信软件系统工程（国际）春季学校
The 6th Spring School on Engineering Trustworthy Software Systems (SETSS 2024)
中国·重庆 西南大学
2024年4月15日—4月21日

Group photograph at SETSS 2024. Seated front row, fourth left to right:

Xiaohong Chen (Workshop invited speaker), Kuldeep Meel (School lecturer), Cláudio Gomes (School lecturer), Jean-Pierre Talpin (School lecturer), Jim Woodcock (School/workshop committee), Yufeng Xia (Professor, SWU), Jonathan P. Bowen (School committee), Shmuel Tyazberwicz (School committee), Zili Zhang (Professor, SWU), Zhiming Liu (School/workshop committee).

The following lectures courses (each consisting of one to four 1½-hour lecture sessions, with breaks) were delivered during the SETSS 2024 School:

- Zhiming Liu: *Introduction to Mathematical Logic and Logic of Programming*
  Chair: Shmuel Tyszberowicz (two sessions)
- Cláudio Gomes: *Introduction to and Deployment of Digital Twins*
  Chair: Jim Woodcock (four sessions)
- Jean-Pierre Talpin: *Theories of Contracts and Their Applications*
  Chair: Naijun Zhan (four sessions)
- Martin G. Fränzle: *AI Components for High Integrity, Safety-Critical Cyber-Physical Systems: Chances and Risks*
  Chair: Lijun Zhang (three sessions)
- Moshe Y. Vardi: *What Came First, Math or Computing?*
  Chair: Jonathan P. Bowen (one session, online)
- Youcheng Sun: *Software Engineering for Explainable AI*
  Chair: Zhiming Liu (two sessions)
- Kuldeep Meel: *Distribution Testing: The New Frontier for Formal Methods*
  Chair: Jonathan P. Bowen (four sessions)

In addition, there were shorter presentations during a two-day workshop immediately following the lecture courses. A selection of papers associated with these are planned to be included in the proceedings, to be published in the Springer LNCS series as a post-proceedings, as for previous SETSS Schools. The following section presents summaries of the main lecture courses, edited from information provided by the lecturers.

SETSS 2024 Lecture Courses

Introduction to Mathematical Logic and Logic of Programming

Lecturer: Prof. Zhiming Liu, Southwest University, China

Biography: Zhiming Liu is a professor at Southwest University, Chongqing, China. His research interests lie in software theory and methods, with a particular focus on the modelling, design, and verification of software and systems. He is renowned for his work on the Transformational Approach to Fault-Tolerant and Real-Time Systems, Probabilistic Duration Calculus for System Dependability Analysis, the theory of semantics and refinement of object-oriented programming, and the rCOS method for component and object system modelling and refinement. Zhiming Liu's recent research revolves around the computational model, models, and refinement of software systems for human-cyber-physical systems (HCPS) and trustworthy autonomous systems (TAS). Together with his colleagues, he has recently proposed a model of human-cyber-physical automata (HCPA) for HCPS to characterize interactions, concurrency, and coordination behaviour of human agents, intelligent machine agents, and physical objects and processes.

Zhiming Liu studied mathematics at university and holds an MSc in Computing Science from the Institute of Software, Chinese Academy of Sciences (1988) and a PhD in Computer Science from the University of Warwick (1991). Before joining Southwest University in Chongqing as a full-time professor in 2016, he worked at the University of Leicester as a lecturer (1994–2005), at the United Nations University – International Institute for Software Technology as a research fellow and then a senior research fellow (UNU-IIST, Macao, 2002–2013), and at Birmingham City University as the Chair Professor of Engineering (2013–2015).

Overview: Mathematical logic and semantic theory of programming form the core foundation for research and teaching in the thematic areas of the SETSS School. This short course offered introductory knowledge to participants on the basics of mathematical logic and programming theory. It covered the formalization of logic, formal logic systems, formal languages and semantics, computational models, and the semantics of programming languages, all in a unified setting. The aim was to prepare the participants for a comprehensive understanding of the rest of the School's lectures.

Introduction to and Deployment of Digital Twins

Lecturer: Prof. Cláudio Gomes, Aarhus University, Denmark

Biography: Claudio Gomes is an Associate Professor of Software Engineering & Computing systems in the Department of Electrical and Computer Engineering at Aarhus University. His research focuses on the co-simulation and the engineering of digital twins — including Simulation, Digital Twin, Models, Cyber-Physical Systems, Algorithms, Semantics, Standards, Interface Standards, and so on. He is the author and co-author of over 80 papers and received the Runner-Up Best Paper Award at the ANNSIM Conference in 2023. He is the guest editor for the SIMULATION journal and the reviewer for eight different journals during the previous two years.

Part 1: Introduction to Digital Twins

Overview: Cyber-Physical Systems (CPSs) are becoming increasingly complex and generate large amounts of data. Analysing such data provides insight into a given system. The digital twin concept emerges as an attempt to seamlessly integrate the data and insight to improve system performance. It enables applications such as visualization, monitoring, state estimation, and self-adaptation. This lecture was split into two parts:

1. The construction of a digital twin exemplified by an incubator system, including the benefits and challenges of each application, was discussed. The result is a description of the building blocks of a digital twin as well as an example of self-adaptation.
2. Formal verification of self-adaptations, and their role in avoiding the re-certification of reconfigured CPS, was discussed. In addition, constructing a non-deterministic model which captures the uncertainties in the system behaviour after a self-adaptation was demonstrated. Signal Temporal Logic was used to specify the safety requirements the system must satisfy after reconfiguration and employ formal methods based on verified monitoring over *Flow** flowpipes to check these properties at runtime. This gives us a framework to predictively detect and mitigate unsafe self-adaptations before they can lead to unsafe states in the physical system.

Part 2: Building and Deploying a Digital Twin Service

Overview: This was a hands-on tutorial. The students were guided in setting up their own digital twin of the incubator, and then the exercise consisted of deploying their own DT service for the incubator. Where possible, the students installed the following tools in their laptops ahead of time: Docker Desktop (for virtualization of the time series database and message broker); Python v3.10 or higher (for running the DT services). The setup instructions are available online:

https://github.com/INTO-CPS-Association/example_digital-twin_incubator

Theories of Contracts and Their Applications

Lecturer: Prof. Jean-Pierre Talpin, INRIA, France

Biography: Jean-Pierre Talpin is a senior scientist with INRIA. He received a Masters degree in Theoretical Computer Science from the University of Paris VI and undertook his Ph.D. Thesis with Ecole des Mines de Paris under the supervision of Pierre Jouvelot. He then joined the European Computer-Industry Research Centre in Munich for three years before being hired by INRIA in 1995, where he led INRIA project teams ESPRESSO and TEA from 2000 to 2023. Among his 150 co-authored articles, Jean-Pierre Talpin received the 2004 ACM Award for the most influential POPL paper (for 1994) with Mads Tofte and the 2012 ACM/IEEE LICS Test of Time Award (for 1992) with Pierre Jouvelot, both for his early-career work on region-based memory management.

From his career-long studies in logic, type, and concurrency theory, and his experiences in program analysis and verification, avionics and cybernetic system design, his research interests have recently focused on novel and challenging topics such as end-to-end mechanized program verification, the design of advanced process calculi to model the logic of mobile, dynamic cyber-physical system networks and of compositional algebraic methodologies to verify them.

Overview: In computer science as in real life, a contract is a logic for the assumptions of one to meet the guarantees of another. Starting from the seminal works of Abadi et al. on “composing specifications” and of Benveniste et al. on a “metatheory of contracts”, this course aimed at structuring many instances of the concept of contract within the late algebraic works of Incer et al. Once some fundamental notions were established to architect an algebraic theory of contracts, the tutorial related and exemplified these fundamental notions for compositional design and verification in engineering fields as various as automated factories, monitoring biological or chemical reactions, schedules of real-time systems, optimisation of space missions, system architecture integration.

From Automata Models to Validated BCI-Based Cooperative Control – On the Viability of Rigorous Approaches to Human-Cyber-Physical Systems of Systems

Lecturer: Prof. Dr. Martin G. Fränzle, University of Oldenburg, Germany

Biography: Martin Fränzle holds the Chair of Foundations and Applications of Systems of Cyber-Physical-Systems at the University of Oldenburg in Germany, where he also was dean of the School of Computing Science, Business Administration, Economics, and Law as well as Vice President Research, Transfer and Digitalization of the university. His research interests are in modelling, verification, and synthesis of reactive, real-time, and hybrid dynamics in embedded, cyber-physical, and human-cyber-physical systems. His work spans the semantic foundations of high-level modelling and specification languages as well as decision problems and their application to verifying and synthesizing real-time and hybrid discrete-continuous systems, including settings subject to stochastic disturbances.

His contributions to the extension of SAT-modulo-theory solving to the undecidable domains of arithmetic constraints involving transcendental functions have generated one of the very few commercially successful automatic verification tools for hybrid state systems (iSAT, under distribution by BTC ES AG). Fränzle has co-authored more than 190 articles primarily in computer science, but also in communication engineering, applied neuroscience, and physics, and has edited several special issues of journals as well as proceedings volumes. He has been member of the board of the Transregional Collaborative Research Center SFB-TR 14 AVACS, as well as of the Research Training Groups DFG GRK 1765 SCARE, DFG GRK 1076 TrustSoft, SAMS (Safe Automation of Maritime Systems), and SEAS (Social Embeddedness of Autonomous Cyber-Physical Systems) at the University of Oldenburg. His research received funding from DFG, BMBF, BMWI, the State of Lower Saxony, VDA, Velux Fonden, and the EU, as well as through direct industrial research contracts with Volkswagen, Daimler, DENSO Automotive, and BTC ES AG.

Overview: Model-based design has become a standard means of designing software, hardware, and their combination with physical environments known as cyber-physical systems (CPS). In most of these applications, the model is considered a blueprint of the system to be such that the model is faithful and correct to the desired level because the latter system implementation is expected to adhere to the design model. Analysis results obtained on the model are thus meant to carry over to the implementation and are considered indicative of the implementation’s respective behaviour. Especially when it comes to safety analysis, it is expected that the model’s set of possible behaviours is (modulo adequate abstraction relations) a not necessarily strict superset of the implementation such that positive safety verdicts transfer. When considering human-cyber-physical systems, establishing such models becomes elusive: it is necessary to integrate CPS models with models of the human, with the latter being empirically validated at most and modelling a behaviourally stationary system. These models are thus approximate and may miss rare-event behaviour or inadequately represent the behaviour of a certain human subject on a certain day. Integrating such human models with CPS models into a unified design flow for a human-cyber-physical system (HCPS) thus poses the problems of semantically integrating models featuring different forms of validity and of deriving consistent behavioural predictions from such an integration. The lecture sketched the development of CPS models, some types of human models, the seamless semantic integration of the former two, and an example of a safety-critical application facilitated by rigorous behavioural analysis of such integration.

What Came First, Math or Computing?

Lecturer: Prof. Moshe Y. Vardi, Rice University, USA

Biography: Moshe Vardi is University Professor and the George Distinguished Service Professor in Computational Engineering at Rice University. His research focuses on the interface of mathematical logic and computation – including database theory, hardware/software design and verification, multi-agent systems, and constraint satisfaction. He is the recipient of numerous awards, including the ACM SIGACT Gödel Prize, the ACM Kanellakis Award, the ACM SIGMOD Codd Award, the Knuth Prize, the IEEE Computer Society Goode Award, and the EATCS Distinguished Achievements Award. He is the author and co-author of over 750 papers, as well as two books. He is a Guggenheim Fellow as well as a fellow of several societies, and a member of several academies, including the US National Academy of Engineering, National Academy of Science, and the Royal Society of London. He holds nine honorary titles. He is a Senior Editor of the *Communications of the ACM*, the premier publication in computing.

Overview: Computer science seems to be undergoing a paradigm shift. Much of earlier research was conducted in the framework of well-understood formal models. In contrast, some of the successful trends today shun formal models and rely on massive data sets and machine learning. A canonical example of this change is the shift in AI from logic programming to deep learning. I argue that the correct metaphor for this development is not paradigm shift, but paradigm expansion. Just as General Relativity augments Newtonian Mechanics, rather than replace it – humans went to the moon, after all, using Newtonian Mechanics – data-driven computing augments model-driven computing. In the context of Artificial Intelligence, machine learning and logic correspond to the two modes of human thinking: fast thinking and slow thinking. The challenge today is to integrate the model-driven and data-driven paradigms. The talk described one approach to such an integration – making logic more quantitative.

Software Engineering for Explainable AI

Lecturer: Dr. Youcheng Sun, The University of Manchester, UK

Biography: Youcheng Sun is a Lecturer in Cyber Security at The University of Manchester. Before joining Manchester, Youcheng was Lecturer at Queen's University Belfast and he was a postdoctoral researcher in the verification group at the University of Oxford. Youcheng is an expert in security and especially AI security. In particular, he pioneered several techniques on the assurance of AI safety. His research has been funded by companies such as Dstl, BAE Systems, Ethereum Foundation and Google. In the past, Youcheng led the source code testing and verification work for two UK domestic airborne software projects, SECT-AIR and AUTOSAC. He was a member of the EU H2020 project SAFURE investigating safety and security assurance in the design of mixed-critical cyber-physical systems.

Youcheng has a strong track record of publications on software engineering, formal verification, embedded systems, robotics and AI safety in top-tier academic conferences and journals, including IEEE S&P (Oakland), ICSE, ASE, CAV, TACAS, NeurIPS, ICCV, ECCV, IJCAI, ICRA, IROS, RTSS, ACM TOSEM, ACM

TECS, IEEE TR. He is an Associate Editor of ACM Transactions on Software Engineering and Methodology (TOSEM).

Overview: Artificial intelligence (AI), especially deep neural networks (DNNs), has been widely used, posing significant safety and security concerns. This course addresses these challenges by exploring the use of software engineering approaches and formal verification methods for diagnosing and fixing defects in DNNs. In addition to theoretical concepts, this course included a practical session on installing and running the software tools. Participants are encouraged to download and install the tools and follow the step-by-step instructions during the session.

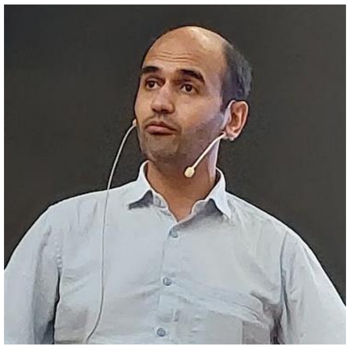
Lecture 1: Software Engineering for Explainable AI

The black-box nature of DNNs makes it impossible to understand why a particular output is produced, creating demand for “Explainable AI”. This lecture showed that testing techniques from software engineering deliver high-quality explanations of the outputs of DNNs, where an explanation as a minimal subset of features sufficient for making the same decision as for the original input is defined. The lecture presented software testing-driven explainable algorithms and a tool called DeepCover, which synthesizes a ranking of the features of the inputs and constructs explanations for the decisions of the DNN based on this ranking.

Lecture 2: Automated Repair of AI

Different from traditional software, the performance of DNNs highly depends on the data used to train the model, which is not exhaustively tested. Therefore, the repair of DNNs refers to fixing the failures of a neural network by modifying its architecture or parameters. Such failures could arise from multiple sources including training errors, adversarial attacks, backdoor attacks and distribution drift. In this lecture, recent techniques on automated AI repair to counter these potential risks, for ensuring the performance and reliability of AI in practice were covered.

Distribution Testing: The New Frontier for Formal Methods
Lecturer: Prof. Kuldeep Meel, University of Toronto, Canada



Biography: Kuldeep Meel is an Associate Professor of Computer Science in the Department of Computer Science at the University of Toronto. His research interests lie at the intersection of Formal Methods and Artificial Intelligence. He is a recipient of the 2022 ACP Early Career Researcher Award, the 2019 NRF Fellowship for AI, and was named AI's 10 to Watch by IEEE Intelligent Systems in 2020. His research program's recent recognitions include the 2023 CACM Research Highlight Award, 2022 ACM SIGMOD Research Highlight, IJCAI-22 Early Career Spotlight, Distinguished Paper Award at CAV-23, "Best Papers of CAV" (2020 and 2022) special issue in FMSD journal, Best Paper Award nominations at ICCAD-21 and DATE-23, 1st Place in Model Counting Competition (2020 and 2022). He is passionate about teaching, and most proud of being a recipient of university-level Annual Teaching Excellence Awards in 2022 and 2023.

Before moving to Toronto in 2023, he held NUS Presidential Young Professorship in the School of Computing at the National University of Singapore. Before joining NUS in Spring 2018, he received M.S. and Ph.D. degrees from Rice University, co-advised by Supratik Chakraborty and Moshe Y. Vardi. His thesis work received the 2018 Ralph Budd Award for Best Ph.D. Thesis in Engineering and the 2014 Outstanding Masters Thesis Award from Vienna Center of Logic and Algorithms, IBM Ph.D. Fellowship, and Best Student Paper Award at CP 2015. He graduated with a Bachelor of Technology (with honours) in Computer Science and Engineering from IIT Bombay.

Overview: The dominant guiding philosophy in the first sixty years of computer science was for designers to design systems that were always correct, and to accept nothing less as users. But times have changed: Users and designers are accustomed to systems with statistical components and behaviours. What does it mean for the formal methods community? This tutorial discussed how such a dramatic change in the acceptance and design of systems presents exciting opportunities to make fundamental contributions: it is necessary to rethink the notions and techniques for the design of specifications and verification methodologies. In particular, the lecture focused on the systems whose behaviours are not naturally captured by symbolic relations but instead require reliance on probability distributions. The tutorial discussed recent efforts in designing formal methodologies for testing whether a sampling subroutine generates a desired distribution. The challenges, opportunities, and rewards were covered.

In addition to the above, a selection of shorter papers associated with a two-day workshop held immediately after the lecture courses will be included within the planned proceedings. The workshop was organized by Cláudio Lomes et al. It included several invited talks, including one by Prof. Jifeng He of Shanghai, who recently celebrated his 80th birthday with a Festschrift Symposium (see Springer LNCS volume 14080, 2023).

SETSS 2024 included social events such as an evening boat trip on the Yangtse River in central Chongqing, and a dinner in a local restaurant serving “hotpot”, the very spicy local cuisine, with cooking by the diners at each table. It is intended that SETSS will continue on an annual basis each spring.

For further online information on SETSS 2024, see:

https://www.rise-swu.cn/SETSS2024/

The development of the book cover for *The Turing Guide* and generative AI

Jonathan P. Bowen

Forethought

The mathematician Alan Turing (1912–1954) has been considered by some to be the father of computer science (Bowen, 2017) and was also interested in early ideas relating to formal methods and program proving (Bowen, 2019). In 2012, I co-organized an event in Oxford to celebrate the centenary of Turing's birth. This was in parallel with celebratory events in Bletchley Park, Cambridge, and Manchester, all workplaces of Turing during his short life. Turing's connections with Oxford are more tenuous, but there are some interesting ties despite this (Bowen, 2022). After the centenary celebrations, a volume on Alan Turing's life and work, some based on presentations at the various 2012 events, was published by Oxford University Press (Copeland et al., 2017). Curiously, the proposal was rejected by Cambridge University Press, but with three of the four main co-authors having Oxford connections, perhaps this is appropriate! Jack Copeland, the lead author of the book, has previously published academic works on Turing with OUP in any case (Copeland, 2014).

Book Cover

The publisher OUP allowed some flexibility concerning the cover of the book. One possibility that was considered was a modern but relatively traditional 2014 portrait painting of Turing by Maxime Xavier. Eventually, this was included as a frontispiece in the book. I was keen for a striking cover that would stand out in a book display. As a fan of the American pop artist Andy Warhol (1928–1987) and his colourful multiple screenprint images of famous people, I thought that something similar would be a memorable and more unusual image for the cover of the book.

The Pictomizer website (http://pictomizer.com) allows the creation of Warhol-like images from a source photograph. Warhol produced his colourful images of various famous people largely during the 1960s and 1970s, including Jackie Kennedy, Marilyn Monroe, and even Mao Zedong, but never Alan Turing, who was not that well-known to the public at the time, especially in the US, largely due to the secrecy surrounding his groundbreaking World War II mechanized codebreaking work at Bletchley Park. It is interesting to speculate whether if Andy Warhol had been active a few decades later when Turing became more of an icon to the public, he might have chosen Turing as a subject. Figure 1 shows an initial mock-up artwork that I produced using the Pictomizer website (then known as the "Warholizer") and presented to the other main co-authors of the book.

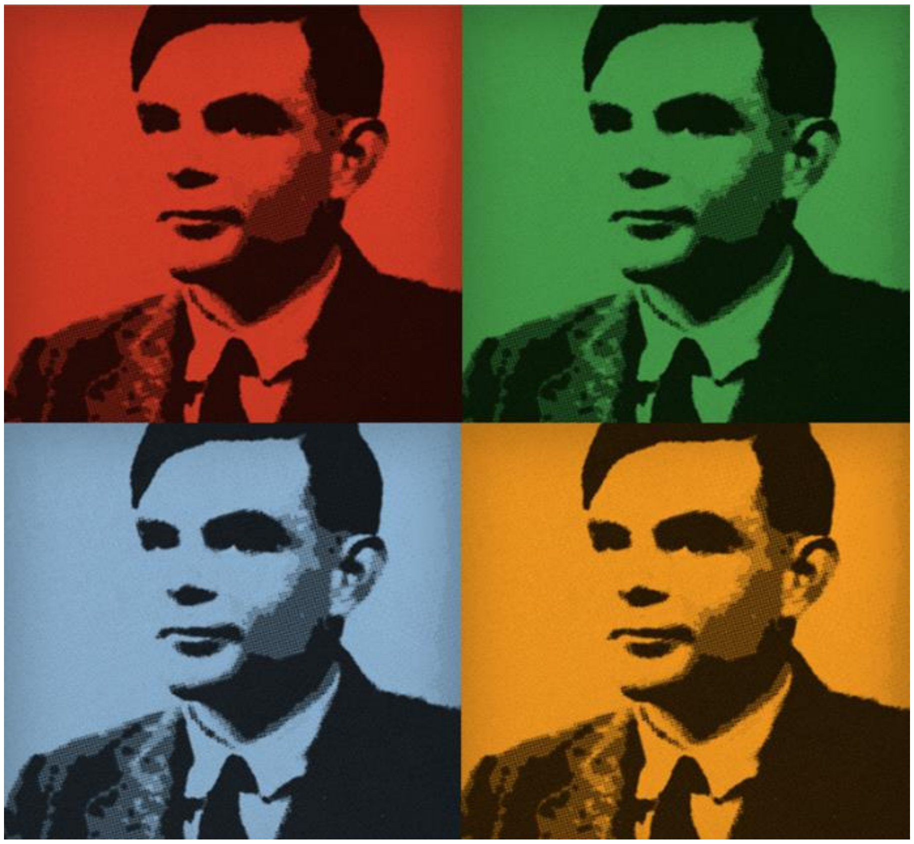


Figure 1. *Initial mock-up artwork for the cover of The Turing Guide (Copeland et al., 2017) inspired by Andy Warhol (Bowen, 2016). (Artwork by J. P. Bowen, 2012.)*

On 28 October 2013, I emailed the three other main co-authors of *The Turing Guide* (Copeland et al. 2017):

> *I am giving talks on The Turing Guide to the history of maths group at Queen's College, Oxford this afternoon and on Alan Turing at Gresham College in London on Thursday (both organized by Robin [Wilson]). These are useful preparation for my chapters, including finding possible illustrations. I have even done a mock-up of a Warhol-style cover art. Robin will see this afternoon and I will send to all in due course.*

The main coauthors liked the mock-up artwork, and it was decided that the concept could be the basis for the book's cover. This idea was worked up into a more professional version for the actual front cover by the book's lead editor/author Jack Copeland (see Figure 2), with the help of two collaborators, Peter Fitzpatrick and Vicki Hyde (Copeland et al., 2016). Grey monochrome images as well as colourful images were deliberately included because Turing's life was not entirely happy (Bowen et al., 2018). Jack Copeland has written the story of producing this artwork elsewhere, including the influence of the diptych of Marilyn Monroe by Warhol (Bowen & Copeland, 2024).

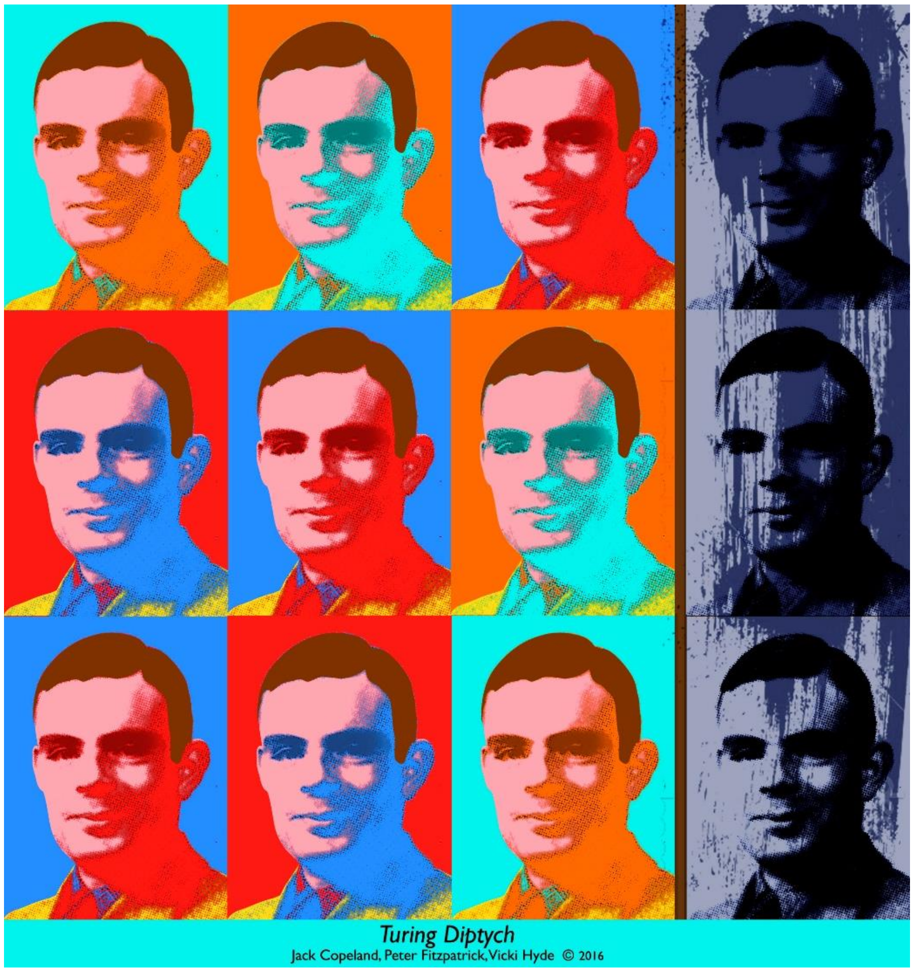


Figure 2. *The* ***Turing Diptych*** *artwork for* ***The Turing Guide*** *(Copeland et al., 2017). (Artwork by Jack Copeland, Peter Fitzpatrick, and Vicki Hyde, © 2016.)*

Finally, the artwork produced by Jack Copeland et al. was transformed into the book cover itself by the publisher, Oxford University Press, including the book's title, etc. (Copeland et al. 2017) – see Figure 3. The books in Figure 4 show *The Turing Guide* with its competing books on a bookshelf. We leave it to the reader to decide if the aim of making the book stand out has been achieved!

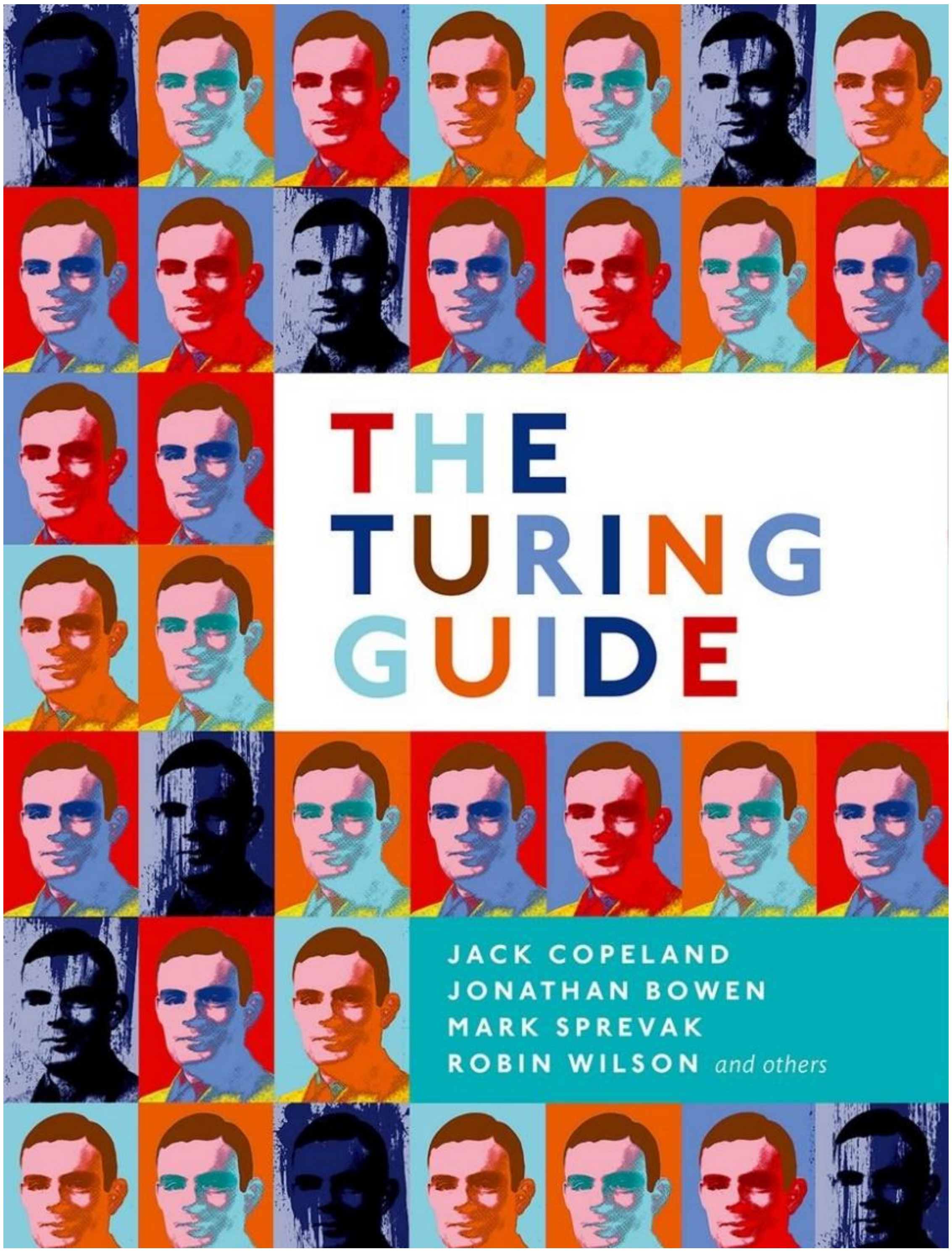


Figure 3. *The final front cover of* ***The Turing Guide*** *(Copeland et al. 2017).*
(Artwork by Oxford University Press.)

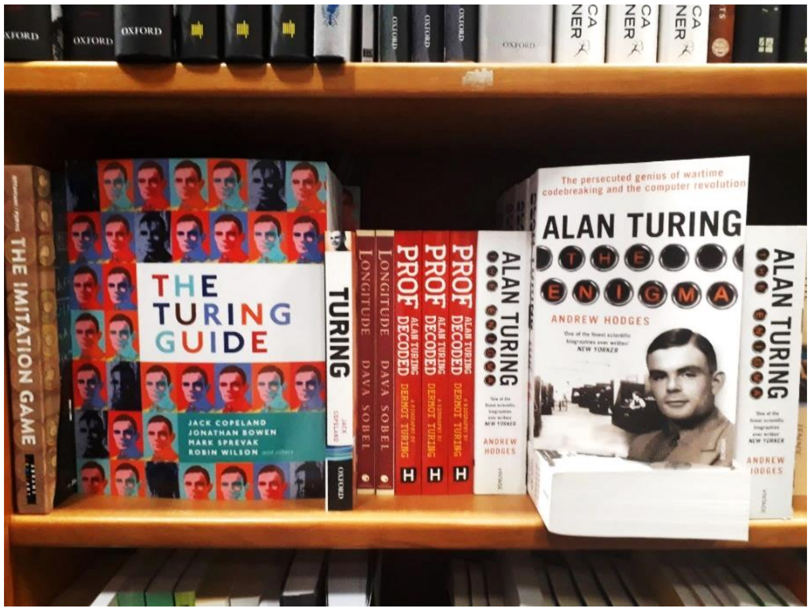


Figure 4. *Turing-related books including* ***The Turing Guide*** *on a bookshelf at Blackwells bookshop in Oxford. (Photograph by J. P. Bowen, 2018.)*

Although designed as a digital artwork (Copeland et al., 2016), the *Turing Diptych* was subsequently printed by the photographer Graham Diprose on archival paper and a high-quality printer. This was framed and exhibited as part of the BCS Specialist Group *Computer Arts Society* (CAS) Members' Exhibition, held in 2023 at the BCS London headquarters (Clark, 2023) – see Figure 5. The print has been donated to the Computer Arts Archive (https://www.computer-arts-archive.com; Bowen & Clark, 2023), associated with CAS.

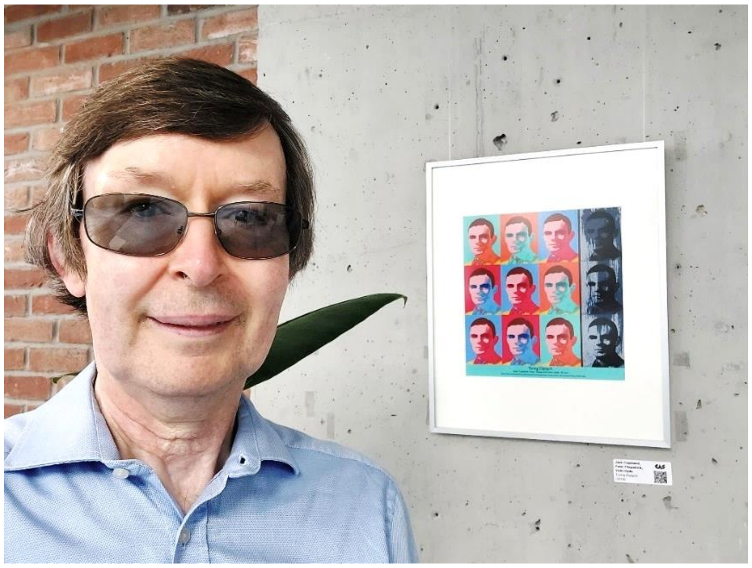


Figure 5. *A printed version of the* ***Turing Diptych*** *artwork (with the co-author) in the Computer Arts Society Members' Exhibition, held in 2023 at the BCS London headquarters. (Photograph by J. P. Bowen, 2023.)*

Generative Artificial Intelligence

Turing visited Bell Labs in the Greenwich Village, area of Manhattan, New York during World War II in the early 1940s (Giannini & Bowen 2017), where he was able to interact with Claude Shannon (1916–2001), considered by some as the father of information theory (Giannini & Bowen, 2017). These included discussions on early ideas relating to Artificial Intelligence (AI), which Turing later distilled in his groundbreaking philosophical article on machine intelligence, including the idea of the Turing test (Turing, 1950). Turing predicted that with increasing computing power and size, "machine thinking" could be achieved by the end of the 20th century (Turing, 1950):

> *I believe that at the end of the century the use of words and general educated opinion will have altered so much that one will be able to speak of machines thinking without expecting to be contradicted.*

Now, with recent and remarkable advances in machine learning, large language models, and generative AI, we are somewhat nearer to what Turing predicted, if slightly later than he suggested, through relatively simple algorithms that depend on large amounts of human-generated data. We have new and developing AI software such as OpenAI's DALL.E (https://openai.com/research/dall-e) that can produce computer-made "generative AI" artworks using simple text prompts. For example, the prompt "Alan Turing" produced the selection of four Turing images in Figure 6.

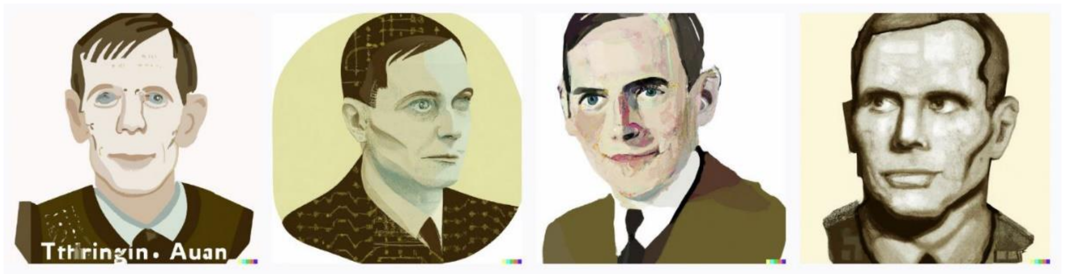


Figure 6. *Images produced by DALL.E AI software using the prompt "Alan Turing". (Produced by J. P. Bowen, 2022.)*

When used in 2024, the DALL.E generative AI software is still not much better although the images are more colourful (see Figure 7). Perhaps this indicates the wider online availability of colourised images of Turing. It is interesting to note that all the generated images of Turing present him in a tie and jacket since he is mostly dressed formally in the photographs that do exist of him.

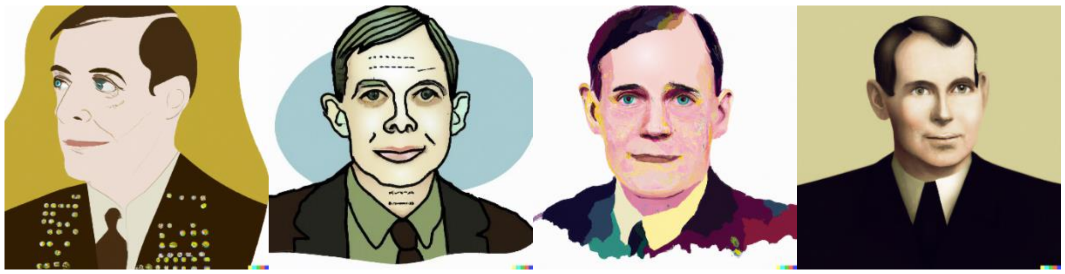


Figure 7. *Images produced by OpenAI's DALL.E using the prompt 'Alan Turing'. (Produced by J. P. Bowen, 2024.)*

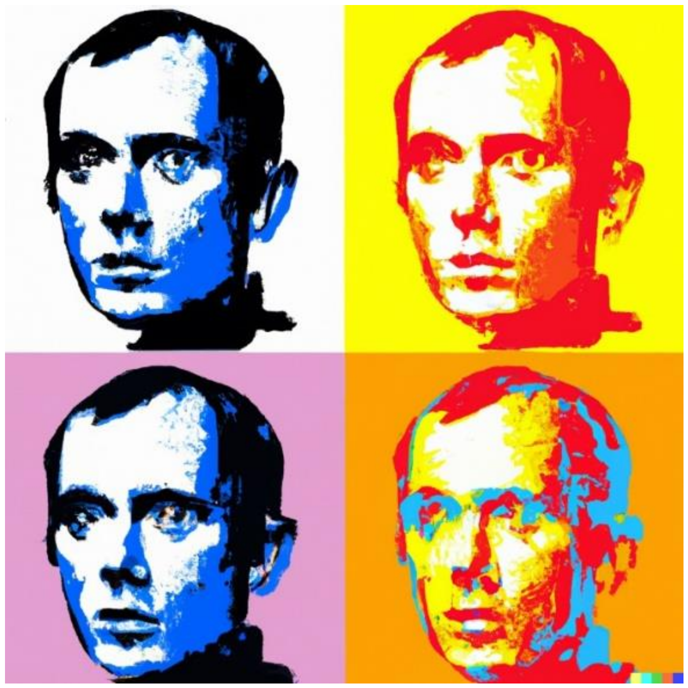


Figure 8. *Image produced by OpenAI's DALL.E 2 using the prompt 'Alan Turing in the style of Andy Warhol'. (Produced by J. P. Bowen, 2024.)*

Prompting DALL.E with 'Alan Turing in the style of Andy Warhol' produces yet more colourful images and one in the multiple-image style of Warhol with four images in a square formation (see Figure 8), although the likeness to Turing is still not very close.

All the known original photographs of Turing are monochrome, even though he lived until 1954 when colour photography was reasonably easily available. Generative AI software can be used to colourise monochrome images with increasing veracity. For example, see Figure 9, which shows the result of the Deep AI *Image Colorizer* using machine learning (https://deepai.org/machine-learning-model/colorizer) with additional enhancement to improve the quality of the monochrome image of Turing used for *The Turing Guide* cover.

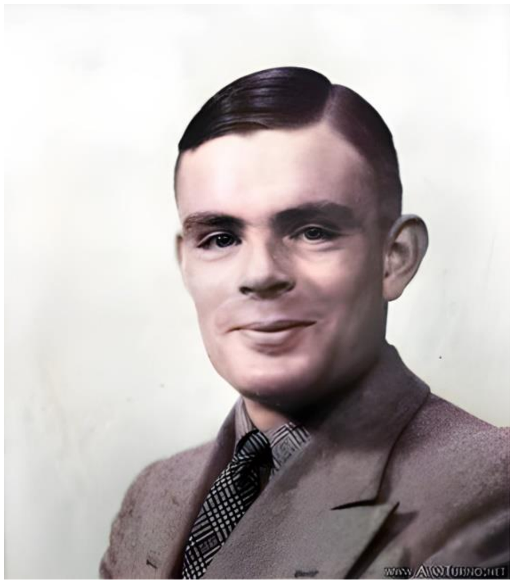


Figure 9. A colourised version of Alan Turing using the Deep AI Image Colorizer. (Generated by J. P. Bowen, 2024.)

The commercially available *Midjourney AI* software (https://midjourneyai.online) is more successful in producing artistic works, as illustrated in an AI-generated 'watercolour' portrait of Turing in Figure 10. This image could fool a human viewer into believing that it has been produced by another human rather than AI technology. It is interesting to note the permission information for this image that is included in the metadata information on Wikimedia Commons:

> This file is in the **public domain** because it is the work of a **computer algorithm** or **artificial intelligence** and does not contain sufficient human authorship to support a copyright claim.

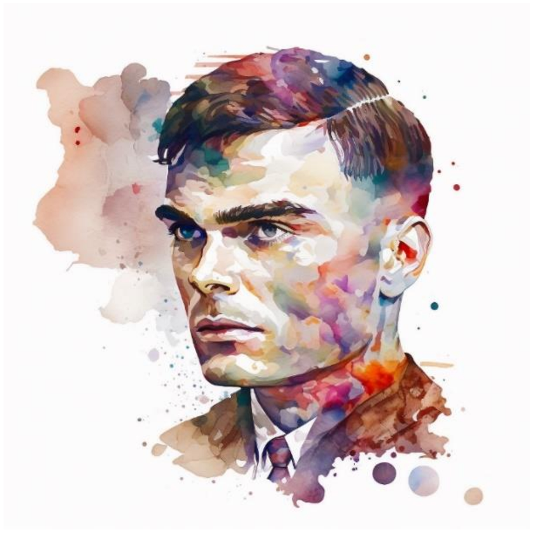


Figure 10. *Alan Turing in watercolour, generated using Midjourney AI. (Netha Hussain, 20 January 2023.) Wikimedia Commons,* https://commons.wikimedia.org/wiki/File:Alan_Turing_in_watercolour.png

Afterthought

Warhol was working in New York at his Manhattan studio, "The Factory" during the 1960s, only two decades after Turing was there, but sadly after Turing's death (Bowen, 2024). Perhaps if Turing and Warhol had been more contemporaneous and located in New York together, the two could have met, at the Factory for example. If they had, Warhol could have done his own version of a Turing screenprint. But in the circumstances, at least the *Turing Diptych* now exists as a digital artwork with a print in the Computer Arts Archive, as a tribute to both Turing and Warhol, two "geniuses" even if in entirely different ways and fields, namely mathematics and art.

References

Bowen, J. P. (2016). Alan Turing: Virtuosity and visualisation. In *EVA London 2016*, pp. 197–204. BCS, eWiC. https://doi.org/10.14236/ewic/EVA2016.40

Bowen, J. P. (2017). Alan Turing: Founder of computer science. In *SETSS 2016: Engineering Trustworthy Software Systems*. Springer, Lecture Notes in Computer Science, volume 10215, pp. 1–15. https://doi.org/10.1007/978-3-319-56841-6_1

Bowen, J. P. (2019). The Impact of Alan Turing: Formal methods and beyond. In *SETSS 2018: Engineering Trustworthy Software Systems*. Springer, Lecture Notes in Computer Science, volume 11430, pp. 202–235. https://doi.org/10.1007/978-3-030-17601-3_5

Bowen, J. P. (2022). Alan Turing and Oxford. *Resurrection: The Journal of the Computer Conservation Society*, 97:11–18, March. https://www.computerconservationsociety.org/resurrection/res97.htm#e

Bowen, J. P. (2024). Alan Turing: Breaking the code, computing, and machine intelligence. In T. Giannini and J. P. Bowen (eds.), *The Arts and Computational Culture: Real and Virtual Worlds*, chapter 3. Springer Series in Cultural Computing.

Bowen, J. P. and Clark, S. (2023). Recent Progress with the Computer Arts Archive. In *EVA London 2023*, pp. 64–65. BCS, eWiC. https://doi.org/10.14236/ewic/EVA2023.10

Bowen, J. P. and Copeland, B. J. (2023). A newly discovered mathematical manuscript by Alan Turing. *FACS FACTS*, 2023(2):6–9. https://www.bcs.org/media/10894/facs-jul23.pdf

Bowen, J. P. and Copeland, B. J. (2024). Turing, Warhol, and Monroe: Development of *The Turing Guide* cover. In *EVA London 2024*, pp. 11–17. BCS, eWiC. https://doi.org/10.14236/ewic/EVA2024.3

Bowen, J. P., Trickett, T., Green, J. B. A., and Lomas, A. (2018). Turing's Genius – Defining an apt microcosm. In *EVA London 2018*, pp. 156–162. BCS, eWiC. https://doi.org/10.14236/ewic/EVA2018.31

Clark, S. (2023) *CAS Members' Exhibition 2023, July–November 2023*, BCS Moorgate, London, UK. https://computer-arts-society.com/exhibitions/cas-members-2023.html

Copeland, B. J., ed. (2004). *The Essential Turing*. Oxford University Press.

Copeland, B. J., Bowen, J. P., Wilson, R., Sprevak, M., et al. (2017). The Turing Guide. Oxford University Press.

Copeland, B. J., Fitzpatrick, P., and Hyde, V. (2016). *Turing Diptych*. AlanTuring.net: The Turing Archive for the History of Computing. https://www.alanturing.net/turing_diptych.jpg

Giannini, T. and Bowen, J. P. (2017). Life in Code and Digits: When Shannon met Turing. In *EVA London 2017*, pp. 51–58. BCS, eWiC. http://doi.org/10.14236/ewic/EVA2017.9

Turing, A. M. (1950). Computing Machinery and Intelligence. *Mind*, LIX(236): 433–460, October. https://doi.org/10.1093/mind/LIX.236.433

Forthcoming Events

We have a new Seminar Organiser on the FACS committee, Alvaro Miyazawa at the University of York. If you have suggestions for future FACS seminar speakers or other events, especially if you are willing to help with co-organisation or even give a talk, please contact Alvaro on Alvaro.Miyazawa@york.ac.uk.

Events Venue (unless otherwise specified):

BCS, The Chartered Institute for IT
Ground Floor, 25 Copthall Avenue, London, EC2R 7BP

The nearest tube station is Moorgate, but Bank and Liverpool Street are within walking distance as well. The new Elizabeth Line is now very convenient for the BCS London office, by alighting at the Liverpool Street stop and leaving via the Moorgate exit.

Details of all forthcoming events can be found online here:

https://www.bcs.org/membership/member-communities/facs-formal-aspects-of-computing-science-group/

Please revisit this site for updates as and when further events are confirmed.

FACS Committee

Formal Aspects of Computing Science Specialist Group

Jonathan Bowen
FACS Chair and
BCS Liaison

John Cooke
FACS Treasurer and
Publications

Roger Carsley
Minutes Secretary

Margaret West
Inclusion Officer

Tim Denvir
FACS FACTS
Co-editor

Brian Monahan
FACS FACTS
Co-editor

Alvaro Miyazawa
FACS Seminar
Organiser

Andrei Popescu
LMS Liaison

Keith Lines
Government and
Standards Liaison

Ana Cavalcanti
FME Liaison

Brijesh Dongol
Workshop Liaison

FACS is always interested to hear from its members and is keen to recruit additional helpers. Presently we have vacancies for officers to help with fundraising, liaise with other specialist groups such as the Requirements Engineering group and the European Association for Theoretical Computer Science (EATCS), and maintain the FACS website. If you can help, please contact the FACS Chair, Professor Jonathan Bowen at the contact points below:

BCS-FACS
c/o Professor Jonathan Bowen (Chair)
London South Bank University
Email: jonathan.bowen@lsbu.ac.uk
Web: www.bcs-facs.org

You can also contact the other Committee members via this email address.

Mailing Lists

As well as the official BCS-FACS Specialist Group mailing list run by the BCS for FACS members, there are also two wider mailing lists on the Formal Aspects of Computer Science run by JISCmail.

The main list <facs@jiscmail.ac.uk> can be used for relevant messages by any subscribers. An archive of messages is accessible under:

http://www.jiscmail.ac.uk/lists/facs.html

including facilities for subscribing and unsubscribing.

The additional <facs-event@jiscmail.ac.uk> list is specifically for announcement of relevant events.

Similarly, an archive of announcements is accessible under:

http://www.jiscmail.ac.uk/lists/facs-events.html

including facilities for subscribing and unsubscribing.

BCS-FACS announcements are normally sent to these lists as appropriate, as well as the official BCS-FACS mailing list, to which BCS members can subscribe by officially joining FACS after logging onto the BCS website.