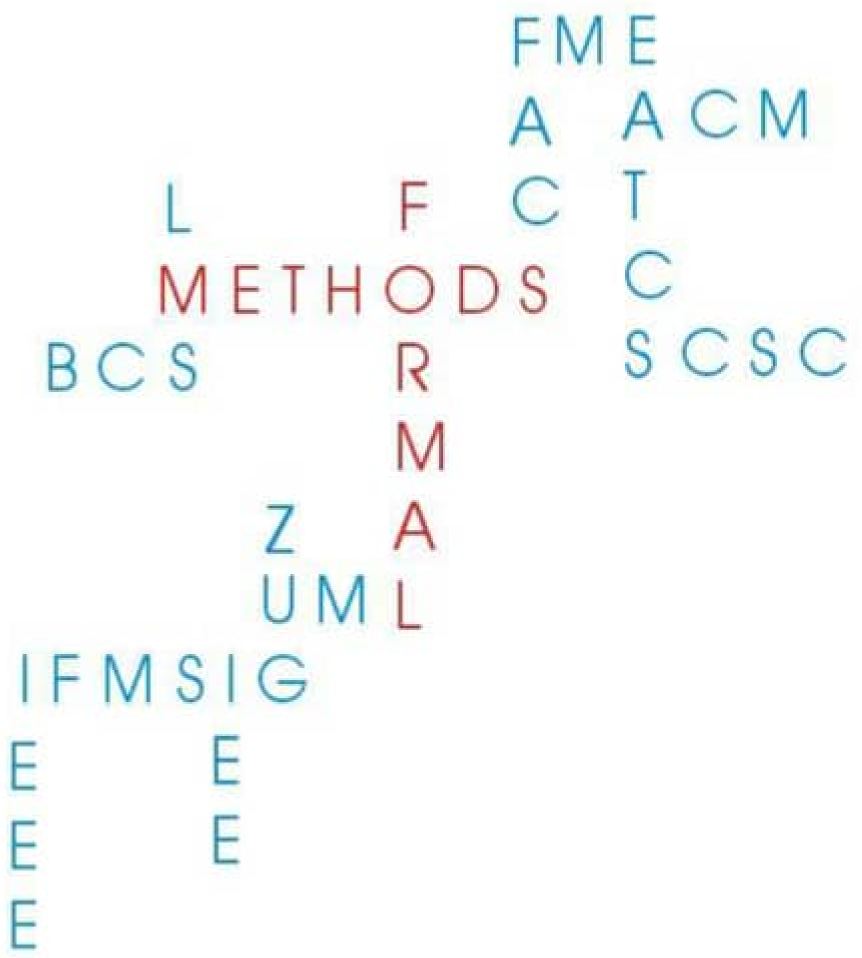
Issue 2026-1
January 2026

# FACS FACTS



Formal Aspects of Computing Science Specialist Group

**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 co-editors:

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

### Editorial team:

Jonathan Bowen, Tim Denvir, Keith Lines, Brian Monahan

### Contributors to this issue:

Bernard Amy, Jeremy Avigad, Jean-Louis and Odette Bernezat, Egon Börger, Jonathan Bowen, Michael Butler, Dominique Cansell, Tim Denvir, Henri Habrias, Stefan Hallerstede, Thai Son Hoang, Cliff Jones, with Joanne Allison, Christian Jullien, Guy Laffitte, Thierry Lecomte, Keith Lines, Brian Monahan, Andrei Popescu, Bernard Sufrin.

## BCS-FACS websites:

| | |
|---|---|
| BCS | **https://facs.bcs.org** |
| LinkedIn | **https://www.linkedin.com/groups/2427579/** |
| Wikipedia | **https://en.wikipedia.org/wiki/BCS-FACS** |

If you have any questions about BCS-FACS, please send these to Keith Lines at: **keith.lines@npl.co.uk**.

# Editorial

Dear reader,

Two highlights of this issue are changes to the FACS committee, notably the Chair but also the Treasurer, Newsletter Editors and other roles. Details can be found in the second feature, **Committee News**, by the incoming new Chair, Keith Lines, who also introduces himself there.

The second highlight is an extensive tribute to Jean-Raymond Abrial, who died on 26th May 2025. **Tributes to Jean-Raymond Abrial** by Jonathan Bowen and Henri Habrias introduces a further 13 personal recollections of Abrial by his fellow computer scientists and other associates.

In addition to those features, we also have the 2025 **Chair's report** for the BCS by Jonathan Bowen; a **Progress Report on editing Mike Shields' posthumous books** by Tim Denvir, which is a collaborative effort with Paul Krause, helped with comments from other FACS committee members. This is followed by a final missive from Prof. F. X. Reid, written by Mike himself shortly before he died.

There follows a short **In Memoriam: Rod Burstall (1934-2025)** by Tim Denvir: this gives a direct link to a fine tribute from four eminent computer scientists in the *FAC Journal* and urges readers to peruse that. One of your co-editors (Brian) was very fortunate to be a first-year postgraduate student of Rod's in Edinburgh – he was always immensely kind and gracious, who introduced his students both to functional programming (using HOPE), and also to basic category theory.

There is then a report from Brian Monahan and Keith Lines on the BCS FACS 2025 Annual Peter Landin Seminar, **Formal Methods: Whence and Whither?** given by Jonathan Bowen. This is followed by a report from Andrei Popescu on the annual BCS FACS/LMS evening seminar **Mathematics in the Age of AI** given by Jeremy Avigad, from CMU in the US. Finally, there a book review by Jonathan Bowen of **This is for Everyone** from Tim Bermers-Lee.

With this being said, we do hugely appreciate and look forward to your contributions, including letters and comments, from you, our readers.

We hope you enjoy FACS FACTS issue 2026-1.

*Tim Denvir*
*Brian Monahan*

# Table of Contents

# BCS-FACS Specialist Group 2025 Chair's Report

| | |
|---|---|
| **Member Group Name:** | FACS Specialist Group |
| **Year:** | 2025 |
| **Report By:** | Jonathan Bowen |

| | | |
|---|---|---|
| **Group Chair:** | Jonathan Bowen | |
| **Group Treasurer:** | John Cooke | |
| **Group Secretary:** | Roger Carsley (Vice Treasurer) | |
| **Group Inclusion Officer:** | Margaret West | |
| **Other Committee Members:** | Ana Cavalcanti | FME Liaison |
| | Tim Denvir | *FACS FACTS* newsletter co-editor |
| | Keith Lines | Vice Chair |
| | Alvaro Miyazawa | Seminar Organiser; Vice Secretary |
| | Brian Monahan | *FACS FACTS* newsletter co-editor |
| | Andrei Popescu | LMS Liaison |

## Successes

| Success | Additional Comments |
|---|---|
| 1. Continued evening seminars online, with recordings on YouTube | The BCS Zoom facilities and recording transfer to YouTube have widened access to FACS seminars. Thank you to Alvaro Miyazawa, Andrei Popescu, Keith Lines, and the Chair for help with organising FACS events in 2025. |
| 2. Publication of *FACS FACTS* newsletters by BCS-FACS and *Formal Aspects of Computing* journal by ACM | We aim for two major newsletters each year, published online in PDF format. Thank you to Tim Denvir and Brian Monahan for continuing with their excellent editing of the two 2025 newsletters. Thanks to Brian, newsletter formatting now uses LaTeX, which helps to improve mathematical content in particular. Contributions by FACS members are always welcome. The associated FAC journal is published by ACM with open access to papers. See `https://dl.acm.org/journal/fac` |
| 3. Online and hybrid events | In 2025, five evening seminars were delivered as webinars or hybrid events, including our usual highlight event, the Peter Landin Semantics Seminar at the BCS London office in December, delivered in hybrid format after the FACS AGM. |

## Plans

| Planned Activity | Additional Comments |
|---|---|
| 1. Continued online and some hybrid events | The BCS facilities for online and hybrid talks enable these modes of delivery. Alvaro Miyazawa of the University of York is the FACS Seminar Organiser. Andrei Popescu organises the annual joint talk with the LMS (London Mathematical Society). |
| 2. At least two *FACS FACTS* newsletters | We continue to aim for early and mid-year as times of publication (typically January and July). |
| 3. Collaboration with related organizations such as Formal Methods Europe (FME) and the London Mathematical Society (LMS). | Our LMS Liaison Officer, Andrei Popescu organized online FACS/LMS talks in January and November 2025 (see further details below). Alvaro Miyazawa and Ana Cavalcanti co-organized a joint hybrid event with FME in June. |

## Impediments

| Impediment | Description |
|---|---|
| 1. Succession planning for committee members | It is planned that there will be a change of FACS executive officers for 2026. Proposed new officers are in place and have been in vice positions for 2025. A change of editorship for the *FACS FACTS* newsletter is also planned in 2026. |
| 2. Limited BCS funding | Our funding from the BCS for the 2025/26 financial year continues at a much-reduced amount compared to pre-COVID times. Thus, our programme of seminars is now realistically reduced to a maximum of two to three physical events per year. That said, online seminars are much more cost-neutral, so we can continue with these more easily. |
| 3. Limited physical meetings | The limited number of physical meetings impedes networking by FACS members. On the other hand, online seminars are popular due to the reduced cost for attendees and provide a significantly wider national and international reach, both for the audience and in selecting speakers. The joint online talk with LMS in November was particularly well attended, with an American speaker and well over a hundred attendees. |

## Additional Facts and Figures

We aim for at least two ***FACS FACTS* newsletters** per year (with two in 2025, in February and July). We also aim for four to six online/hybrid evening seminars per year (with five in 2025, three online and two hybrid).

## Further Comments

For the record, FACS organised the following five events during 2025:

1. **Probabilistic Datatypes: Automating verification for abstract probabilistic reasoning**, by **Annabelle McIver**, Macquarie University, Sydney, Australia, 15 January. Online with LMS, organised by Andrei Popescu.
2. **Functional Programming and Dependent Types for Metrology**, by **Keith Lines** and **Andre Videla**, National Physical Laboratory, London, UK, 11 February. Hybrid at the BCS London office, organised by Keith Lines.
3. **Formal Methods and Tools in Railways: Recent successes and future challenges**, by **Maurice ter Beek**, CNR-ISTI, Pisa, Italy, 26 March. Hybrid with FME at the BCS London office, co-organised by Alvaro Miyazawa and Ana Cavalcanti.
4. **Mathematics in the Age of AI**, by **Jeremy Avigad**, Carnegie Mellon University, USA, 6 November. Online with LMS, organised by Andrei Popescu.
5. **Formal Methods: Whence and Whither?**, by **Jonathan Bowen**, London South Bank University, UK, 4 December. Hybrid Landin Seminar with FACS AGM at the BCS London office, co-organised by Jonathan Bowen and Keith Lines.

Thank you to all the FACS committee members for performing their various roles, as detailed above. New committee members and new ideas for activities/collaborations are very welcome, especially if interested in co-organising events.

# Committee News

Keith Lines
keith.lines@npl.co.uk

December, 2025

## Overview

The 2025 FACS AGM saw significant changes to the committee, combined with reassuring continuity. Although Jonathan Bowen and John Cooke have stood down as FACS chair and treasurer, they will continue on the committee, as (co-)editor of FACS FACTS and in an emeritus role respectively. Roger Carsley has replaced John as treasurer, with Alvaro Miyazawa replacing Roger as secretary.

Thanks to Brian and Tim who will be stepping down from the FACS FACTS editorship next year (but hopefully staying on in an emeritus capacity, please!). Thanks also to all other committee members. Ana and Andrei have done a great job connecting FACS to Formal Methods Europe and the London Mathematical Society respectively. Such links are vital to FACS, and I'm grateful that they are continuing in their roles. Thanks to Margaret for continuing to do a great job as our inclusion officer, a vital role that must help shape the future of FACS.

## Me / Myself / I

Without wanting to seem too egotistical I feel a few words of introduction, including the reasons I feel so strongly about the importance of FACS, are in order. In this year's Peter Landin talk, Jonathan has largely done the work for me in explaining the importance of formal aspects. Thinking about the work of FACS, the YouTube content, from so many excellent speakers, alone is a valuable resource. Note the Landin talks referenced below. Also, can AI help new tools to be developed that will continue to help formal methods reach their full potential? Can FACS help with this task? Exciting days lie ahead.

When it comes to theoretical matters more widely, I believe strongly there's a wonderful narrative of which everyone developing computer systems should at least have an appreciation (see Nicholas Wu's 2022 Landin talk [1]). It shouldn't be necessary to understand the finer points of denotational semantics to be aware of the importance of using a language with a formally defined semantics.

As far as my own background goes, my introduction to formal aspects began (unbelievably to me) just over 40 years ago as a Computer Systems Engineering undergraduate at the University of Kent. In the syllabus taught by the Computing Laboratory there was a strong emphasis on formal methods such as Z and CSP leading to the Occam language (weren't transputers a beautiful concept?). David Turner's development of the Miranda functional programming language was another powerful influence. David had a reputation as an excellent lecturer but, as he was on a Miranda-focussed sabbatical for a lot of my undergraduate days, I only got to experience that for myself some 30 years later when he gave the 2019 Landin talk [2].

Meanwhile, over in the Electronics Laboratory, Mike Shields taught a wonderful course on automata theory. In those days, formal methods and functional languages were poised to solve the software crisis. But, although it wasn't yet a formal part of the syllabus, there was also a lot of excitement about a new(ish) language called C++ and the associated object-oriented programming paradigm.

For my final year undergraduate project I wrote a Z specification of a system for uploading and processing student exams results. Supervised by Simon Thompson (who gave the 2016 Landin talk) it was an enjoyable piece of work, but my goodness, it got me some bemused looks in graduate job interviews. An encounter with Plessey remains memorable, and not in a good way! In fact, after graduation I joined Mike Shield's group at Kent for a couple of years working on an Alvey project sponsored by British Telecom. It was a wonderful opportunity for which I will always be grateful to Mike. But, once my contract was finished, an honest assessment of my strengths and weaknesses made it clear that academia was not for me.

The National Physical Laboratory, with its location on a crossroads between government, industry and academia (as well as in Teddington), was the right place. When I joined NPL in 1990 a thriving group of computer scientists worked with functional programming and formal methods on a variety of projects. These days there's a greater emphasis on collaboration with universities rather than undertaking such work in-house [3].

After years of knowing that I really should join the BCS, I finally joined in 2013. FACS was one of the groups I gravitated towards. But that's more than enough waffle about and name dropping from me. It is a tremendous privilege to be elected chair of FACS. I don't come from a PhD / academia background but I hope I've provided some reassurance about my background and that I have a strong interest in formal aspects. I am also well-aware that following on from Jonathan, I have big shoes to fill!

## References

[1] https://www.youtube.com/watch?v=y0EjeY02joI

[2] https://www.youtube.com/watch?v=ezFZIPuSQU8

[3] https://www.youtube.com/watch?v=CjKIxwv-z6U

## Thank you to John Cooke!

Before the main Landin seminar got underway, Jonathan gave a presentation acknowledging the many contributions of John Cooke, an early chair of FACS and our erstwhile treasurer for 7+ years.

Figure 1: Jonathan Bowen (with Keith Lines) awarding John Cooke [not present] with his well-deserved Certificate of Appreciation.

## Committee Rôles for 2026 in full

As planned last year, the committee has now moved forwards with new rôles as follows:

| | |
|---|---|
| FACS Chair: | Keith Lines |
| Tresurer: | Roger Carsley |
| Secretary: | Alvaro Miyazawa |
| Inclusion Officer: | Margaret West |
| Seminar Organiser: | Alvaro Miyazawa |
| Publications: | John Cooke |
| LMS Liaison: | Andrei Popescu |
| FME Liaison: | Ana Cavalcanti |
| Newsletter Co-editor: | Jonathan Bowen |
| Newsletter Co-editor: | Brian Monahan |
| Emeritus Newsletter Co-editor: | Tim Denvir |

All presently serving members continue to serve in some capacity, with three members gaining emeritus status:

Jonathan Bowen, John Cooke, and Tim Denvir

# Progress Report on Editing Mike Shields' Posthumous Books

Tim Denvir

December, 2025

The memorial piece for Mike Shields in the FACS Newsletter last July recorded that Mike was in the process of writing three books when he died. His colleague, Paul Krause, at the University of Surrey, had heard about these cut-short efforts of Mike's, but had not seen the material. The net result has been that Paul Krause and Tim Denvir have been editing Mike's material on "Whole Numbers" (always intended by him as a temporary title), and "Logic". Mike designated these two as "Book0" and "Book1". We have had helpful input from other FACS committee, especially John Cooke, Margaret West and Brian Monahan.

We have made considerable progress, with Paul Krause being the main driver. Our approach has been to maintain as much of Mike's original style and content as possible, making judgments about where to replace, where to add an editorial footnote etc., in terms of preserving Mike's own mathematical style while at the same time achieving clarity, comprehension and accuracy.

In this work of his, Mike's approach to the integers has been to take Peano's axioms for the Natural Numbers and extend them by adding a "predecessor" operator, thus producing a system that is symmetric between the positive and negative integers. The traditional method of extending natural numbers to integers has been to represent the integers as equivalence classes of pairs of natural numbers, whose differences are the same, the differences being expressed as multiple applications of the successor operation. Mike may possibly have thought this a contrivance, using concepts somewhat beyond Peano's starkly simple original. Certainly, Mike's alternative solution to axiomatising the integers seems far more in the spirit of Peano.

Mike's book on Logic ("Book1") may be seen as an adjunct to his book on Whole Numbers. If so, it might be sensible to combine the two into a single volume; both are quite short. We shall give more thought to this.

While Mike had reported that he was also working on a third book, on the Reals, he never shared this material with anyone we know, other than making a few one-sentence remarks about it in a couple of emails. Some efforts are in place to see if Mike's computer can be located and accessed. Meanwhile, we hope to have the other material, Whole Numbers and Logic (temporary titles) ready for publication in 2026. There have been a few suggestions for approaching a publisher.

# A late missive concerning Prof. F. X. Reid

Y

Received in late 2023

*Dear X.*

*You may recall a certain late-night (say-no-more) chat over a few (!) glasses of Glen-something-or-other, in which the subject arose of our old friend (?) F. X. Reid. Had he really (as some have darkly averred) defected eastward? Had he taken over some spurious political or religious sect? Had he retired to some secluded nook to translate* Finnegans Wake *into demotic Greek (as he had long threatened to do)? You may remember that our speculations become more and more outrageous as wee glass followed glass, and night glimmered into day.*

*But the truth, sad to say, is far more mundane.*

*It was while I was visiting my aunt in Xlendi ('X' is pronounced 'sh', here) that I first caught wind of F. X. Reid. A strange, dishevelled man was apparently squatting on a soap box in Valletta ranting that adherents of interleaving semantics were disseminating a false perspective on Concurrency Theory. Not only are they not right, they aren't even wrong (allegedly). What more could my aunt tell me?*

*Facts were unfortunately hard to come by, Reid was being held incommunicado, somewhere near Balzan by his 'secretary', Dr. F. X. Lurk, and is only seldom to be seen 'free'. It is on these occasions that the populace are treated to his denunciation of Algebraic process methods, of the so-called Oxford Mafia, of the Glitch phenomenon, of G*d knows what!*

*But what is really going on? Only the shadowy F. X. Lurk would seem to have the answers. Suffice it to say that there is nothing of which he has not been accused and there is no evidence of any of it.*

*Will there be more? Who knows?*

# In Memoriam: Rod Burstall (1934-2025)

Tim Denvir

We are sad to report that Rod Burstall (see his wikipedia page[1]), one of the most eminent British theoretical computer scientists, died in February 2025. Anything we could write about Rod in this newsletter would be dwarfed by the eloquent, comprehensive, warm and remarkably timely obituary written by four highly noted computer scientists, J Strother Moore, Gordon Plotkin, David Rydeheard and Don Sanella, in the Formal Aspects of Computing Journal, Volume 37, Issue 4, October 2025. Readers are urged to read this, since ACM publications are open access, the FAC Journal obituary for Rod can be read here[2].

An indirect link in that obituary leads to another FAC Journal article, an Ode to Rod Burstall, written in 2000 by Rod's secretary of 35 continuous years, Eleanor Kerse, to wish him well for his retirement. It was published in the Journal in 2002. It reads as hilariously today as it did when I first read it in 2002 [3]. For two people to work together for so long is a feat, especially perhaps, on the part of his secretary! I knew Rod quite well when he, together with Robin Milner, Gordon Plotkin, and Matthew Hennessy formed and inspired LFCS, the Laboratory for Foundations of Computer Science, at Edinburgh University. Rod's research there focussed on the more algebraic approaches to program semantics, leading to the use of category theory. At first Rod and his colleagues, Robin Milner in particular, seemed to consider program semantics in terms of modelling, perhaps a philosophical exercise in defining "what we are talking about" when we write computer programs. Accepting that semantics has a role in software development arose a little later, I think, possibly as a result of interactions with industry. I can also absolutely corroborate what the writers of the FAC Journal obituary wrote about Rod's personality. He was always kind, pleasant, encouraging and cheerful.

## References

[1] Wikipedia *Rod Burstall*
`https://en.wikipedia.org/wiki/Rod_Burstall`

[2] J. Strother Moore, Gordon Plotkin, David Rydeheard, Don Sannella *Rod Burstall: In Memoriam* FAC Journal, Vol 37, No 4, Article 27 (October 2025)
`https://dl.acm.org/doi/10.1145/3731974`

[3] Eleanor Kerse *An Ode to Rod Burstall* Formal Aspects of Computing, Vol 23, p194, 2002
`https://dl.acm.org/doi/pdf/10.1007/s001650200007`

# Tributes to Jean-Raymond Abrial

Jonathan P. Bowen
London South Bank University
United Kingdom

Henri Habrias
University of Nantes
France

December, 2025

On 25 May 2025, a conference on the scientific career of the formal methods pioneer Jean-Raymond Abrial (1938–2025) was organized in Nantes, France. The following day, we learned of his death on 26 May, at the age of 86. A tribute to Jean-Raymond ("J.-R.") appeared in the last issue of *FACS FACTS* [1], published in July 2025. We have now collected a wide variety of tributes by colleagues and friends of "J.-R.", which are included in this issue of *FACS FACTS*.

J.-R. was born in 1938 in Versailles, near Paris in France. He attended the Prytanée Militaire de la Flèche, then the École Polytechnique, and in 1960 became a Naval Engineering graduate, after having studied at Stanford University in the USA. He worked on the LTR (Real Time Language) 2 programming language at the Naval Programming Center in Paris, and then was invited to the University of Grenoble in 1969. There, in less than two years, with a team of three PhD students, he developed the Socrate DBMS. He implemented an approach that would define his subsequent work: specifying before programming. Georges Vigliano, who was a member of this team, presents J.-R.'s working method at that time. Christian Jullien, who was a student of J.-R, worked on the evolution of Socrate (which became Clio), which, through its various versions, was used for 30 years. His contribution is titled "Thanks to Jean Raymond Abrial".

J.-R. was also part of Jean Ichbiah's Green Team, which developed the Ada programming language. In 1974, J.-R. published *Data Semantics*, concluding with "We could merely consider it [The model we have described in this paper] a specification tool that could help people in writing programs", and wrote the first papers on the Z specification language. Z was not published outside of the EDF DER IMA (Department of Computer Science and Automation). However, Z was disseminated through its use in two books published in French: one by B. Meyer and Cl. Baudoin in 1980, and the other by Cl. Delobel and M. Adiba in 1982. Tony Hoare had suggested to Bertand Meyer that he publish the latter in English in the Prentice Hall International Series in Computer Science.

In 1979, Tony Hoare, who had heard J.-R. speak at a school organized by EDF (Électricité de France) at La Bréole in the Alpes-de-Haute-Provence, invited him to the Programming Research Group (PRG) at Oxford University. This marked the beginning of Z's golden age. In his text entitled "Memories of Jean-Raymond Abrial in Oxford and the Alps", Bernard Sufrin recounts how, from autumn 1979 to mid-1981, he worked with J.-R. and how the "Oxford Z" was developed.

J.-R. then worked on what would become B. In 1990, he returned to Oxford, where he made several visits. Ib Holm Sørensen (1949–2012) left the PRG for BP Research, developing the B-Tool, and then founded B-Core Ltd to support the B Toolkit. In 1985, Guy Laffitte, an engineer at INSEE (the French National Institute of Statistics and Economic Studies), attended one of J.-R.'s courses. This led to numerous exchanges with J.-R., who thanked him for his contributions to the B-Book, published in 1996. He applied B to the 1990 French census. He presented this in an interview by Henri Habrias.

In Paris, it was at the RATP (Parisian public transport operator) that the B-Method and Atelier B were used for the development of the safety software for the driverless metro trains of the new Line 14 (Météor Project), inaugurated in October 1998, and subsequently in several metro systems worldwide. Atelier B was then distributed and improved by the company CLEARSY. Thierry Lecomte, Research and Development Project Director at ClEARSY, presents the "Atelier B proof system", as developed for the commissioning of Météor in December 1998. This was the first industrial experiment of its kind.

Then J.-R. moved on to Event-B and finally to the Rodin Platform developed at the Polytechnic School of Zurich from 2004 to 2009, with, among others, Stefan Hallerstede who, in his text "Three Years in Zurich In memory of Jean-Raymond", explains how during this period "Event-B and Rodin were developed glove in hand to fine tune notation, method, and tool".

J.-R. then participated in several research projects. Michael Butler, who was introduced to Z in the 1980s, offers us "J.-R. Abrial, an appreciation", an account of his work interacting with J.-R. over decades, especially his work on the collaborative European RODIN project after his move to the University of Southampton. Another Southampton colleague, Thai Son Hoang, provides some further heart-felt memories of J.R., largely through his participation in the RODIN project.

Dominique Cansell began corresponding with J.-R. in 1997, then worked with him in the field of proof and related tools, as well as on Event-B, and accompanied J.-R. in his final work. His moving contribution, "My Jean-Raymond", reveals the strength of the exchanges he had with J.-R. He is the co-author of J.-R.'s last publication.

J.-R. was a great mountaineer and Saharan explorer. Bernard Amy, with Louis and Odette Bernezat, mountain guides and Saharan guides, who accompanied J.-R. on expeditions, attest in their testimonies to a talented mountaineer, a tenacious and equally talented explorer, and above all, a precious friend, steadfast in his loyalty and support. Formal methods readers may be less aware of this part of J.-R.'s life, and the photographs in this tribute provide a vivid illustration of J.-R.'s desert explorations in north Africa.

The contributors to this issue illustrate, through their personal accounts, the life of J.-R., who, during his lecture at the Collège de France in April 2025, introduced himself as follows: "There are two kinds of researchers: the prolific and the monomaniacal. I belong to the second category, because I have always practiced the same kind of investigation, namely the specification and verified construction of computerized systems."

We sincerely thank all the contributors for their tributes. A Festschrift event and volume is planned by colleagues including Egon Börger, who provides a tribute to his friendship with J.-R. here. Henri Habrias provides some thoughts on what he owes J.-R. We also include a comprehensive bibliography for J.-R. If any readers knows of any additions or corrections, please send them to Jonathan Bowen on **jonathan.bowen@lsbu.ac.uk** and Henri Habrias on **henri.habrias@univ-nantes.fr**.

## References

[1] Habrias, H. and Bowen, J. P. (2025). In memory of Jean-Raymond Abrial (1938–2025). *FACS FACTS*, **2025**(2):6–10, July. BCS.
https://www.bcs.org/media/yd4ocehl/facs-jul25.pdf#page=6.00

## Contributions for Jean-Raymond

## Testimonies from Bernard Amy, researcher and mountaineer, and Jean-Louis and Odette Bernezat, mountain and Saharan guides

Bernard Amy

November 2025

All those who knew and worked with Jean-Raymond Abrial will always remember his profound humanity in human contact, his capacity for empathy, and his social charm. Those he encountered in the field of IT will, of course, attest to this. But his international scientific reputation should not obscure his remarkable skills in his other fields of activity, particularly in climbing, mountaineering, and exploring the world's mountains.

As a mountaineer, he completed numerous major climbs in European massifs. But he also made his mark in some more distant massifs. His participation in several magnificent first ascents in the Cilo Dag mountains in Turkish Kurdistan is now part of the history of this massif.

This mountaineering expedition was to be followed by others in Alaska, South America, and Oman. Later, as part of the Bernezat camel treks, he discovered a new passion: the desert and its Hoggar Mountains.

There, too, he was able to demonstrate his climbing skills while proving himself to be a Saharan of stature in historic crossings. His Saharan friends particularly remember a particularly grueling Tamanrasset-Djanet camel trek during which he tirelessly provided physical and mental assistance to the Tuareg guides in accompanying the body of one of the group members who had died en route.

Our exploratory trip to the mountains of Turkish Kurdistan resulted in a book [1]. Our expedition to the Denali Mountains (McKinley) in Alaska was mentioned in two articles [2, 3]. Jean-Raymond took part in several major Saharan explorations in the Hoggar. His guides Jean-Louis and Odette Bernezat testify to this in the following excerpt of the message:

> "Jean-Raymond accompanied Yves Merle d'Aubigné on foot by walking behind the camel led by Entayent, a camel that carried Yves' remains. That was in April 2001. The death of Yves Merle d'Aubigné by heart attack took place during a camel ride with more than 12 participants going from Tamanrasset to Djanet, in the middle of the desert. Jean-Raymond moved away from the group for 2 days to follow the funeral procession while the rest of the group walked with the camels a few dozen meters away. Jean-Raymond made the beautiful crossing of the Isowan erg[1] with us in 2009."

To this testimony must be added two beautiful Saharan explorations in which Jean-Raymond participated in the immense erg of the Égédé de Mourzouk in the great Libyan south. After crossing the southern part of this erg in 1997, he became one of the members of the famous first north-south crossing of 284 kilometers, which was carried out in complete autonomy from 20 December 1998 to 4 January 1999.

[1] An *erg* is a large area of sand dunes.

In all these expeditions, Jean-Raymond showed his taste for exploration off the beaten track, this need for new paths that was also at the heart of his scientific career. Always with the little smile of connivance addressed to those who accompanied him.

Jean-Raymond: a computer genius,[2] a talented mountaineer, a tenacious and equally talented explorer, and above all, a precious friend, steadfast in his loyalty and support.

## References

[1] Bernard Amy. *La Montagne des Autres, alpinisme en pays kurde*, Ed. Arthaud, 1972.

[2] Bernard Amy. A Game of Patience: The Northeast Summit of the Rooster Comb. *The American Alpine Journal* (AAJ), 1972.

[3] Bernard Amy. À la face nord-est du Rooster Comb. *Revue La Montagne et Alpinisme*, n° 5, 1972.

## Appendix: Expeditions

**Murzuk Éguédé, Libya (December 1998):** This journey, in which Jean-Raymond participated, has been described as "Jean-Louis Bernezat's Saharan Everest" (Hommes et Montagnes agency). An éguédé (a Tuareg term) is larger than a simple erg, consisting of immense dunes forming a compact massif.

The geographer Robert Capot-Rey made three attempts to cross it from north to south in 1941, 1942, and 1943. During December 1998–January 1999, the team assembled by Bernezat, which included Jean-Raymond, successfully completed the crossing in 18 days with camels, five Tuaregs, and pack camels (no water source, no pastures). The participants took turns, in certain sections, creating a makeshift track for the camels. Shoveling away the soft sand was essential. (See Figures 1 to 3.)

**Crossing from Tamanrasset to Djanet, Algeria (March 2001):** It was during this crossing that a participant, Yves Merle d'Aubigné, died. Jean-Raymond insisted on accompanying our friend's body for several days, following the Tuareg leading the camel that carried his remains. (No photographs.)

[2]At the time when Jean-Raymond was in Grenoble, after creating Socrate, he was beginning to develop his idea of Z, while I was working in a small team led by Professor Louis Bolliet on program proofs. But we hadn't taken the same step as JR. We were still downstream of the program, whereas JR had understood that we had to position ourselves upstream. (Bernard Amy)

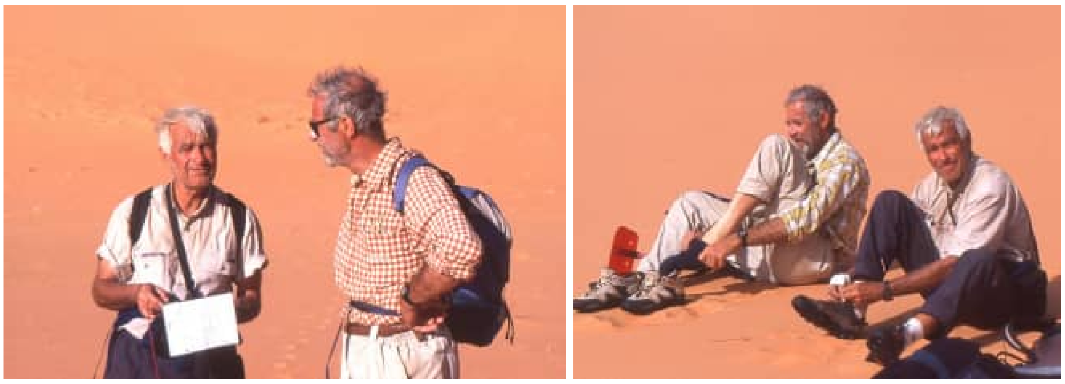


Figure 1: Jean-Raymond with Jean-Louis Bernezat, expedition leader. (Libya, 1998.)

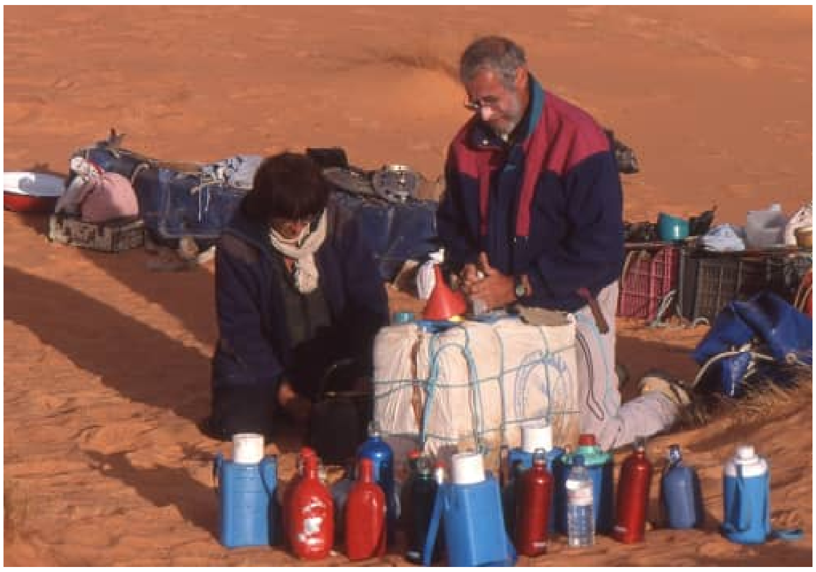


Figure 2: Every evening, Jean-Raymond was tasked with filling the participants' water bottles. Here he is with Odette Bernezat, who was responsible for keeping track of water expenses. (Libya, 1998.)

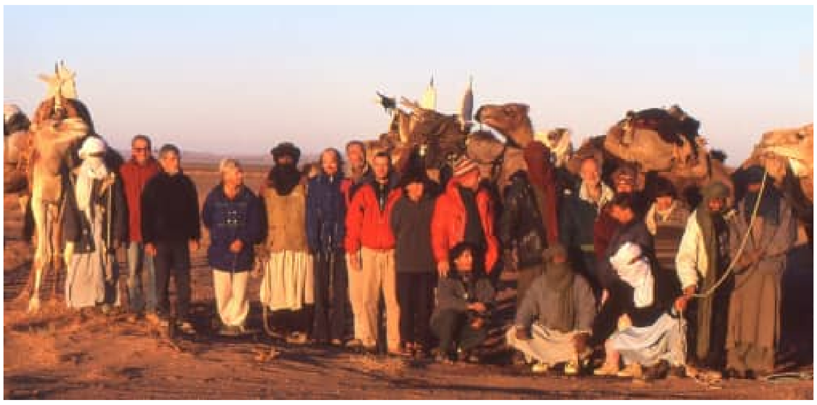


Figure 3: The group, Tuaregs and participants; Jean-Raymond is in the center, between the blue and the red jackets. (Libya, 1998.)

**A Historical Journey to the Amadror Salt Mines, Algeria (March 2003):** Amadror is a salt mine in the north-northeast of the Hoggar Mountains, where the Tuareg people of the Hoggar used to collect salt to trade for millet in Niger. If the trade was successful, they would exchange the surplus millet for dates in Tidikelt (In-Salah, Algeria). The entire journey took 10 months. For us, it was 20 days: Tamanrasset to Amadror. (See Figures 4 and 5.)

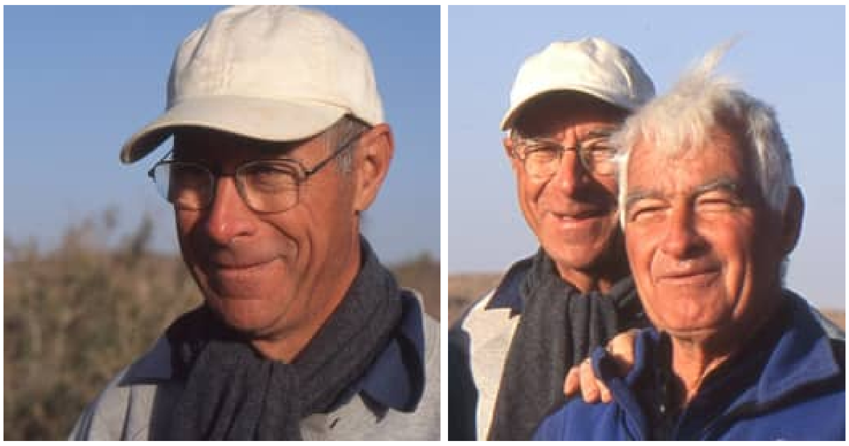


Figure 4: Left: Jean-Raymond Abrial. Right: Jean-Raymond and Jean-Louis Bernezat. (Algeria, 2003.)

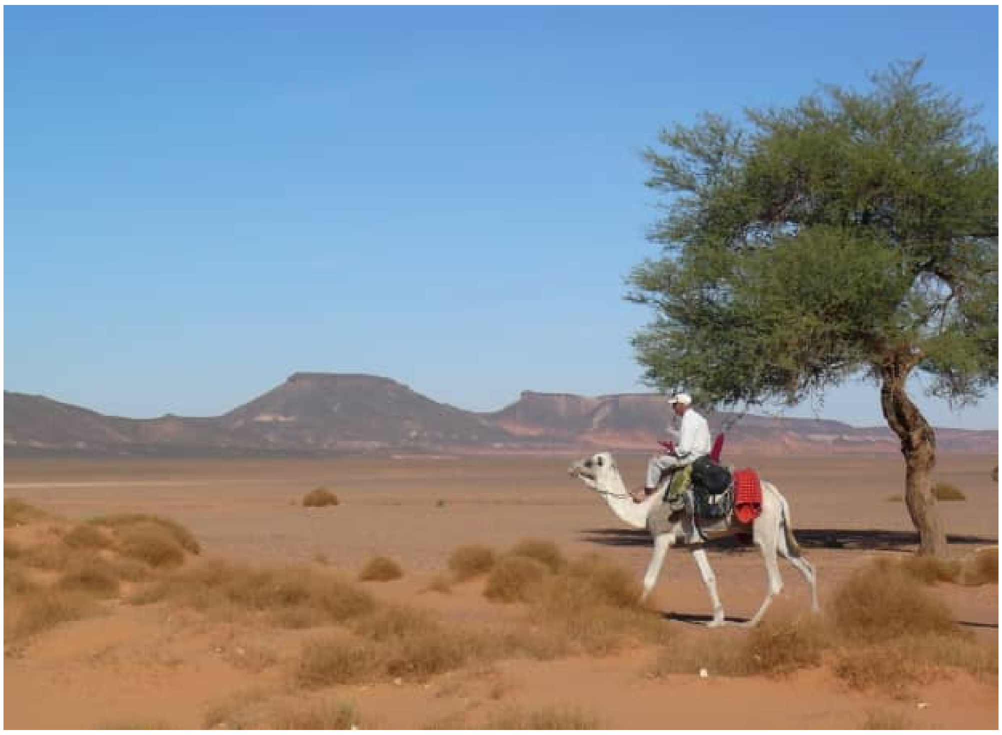


Figure 5: Jean-Raymond riding a camel in the desert. (Algeria, 2003.)

**Crossing the Erg Isoxan. Algeria (November 2009):** This erg had never been fully crossed. There were no major difficulties (the dunes were never high and compact enough to form a dune ridge). (See Figures 6 and 7.)

Figure 6: Jean-Raymond Abrial. (Algeria, 2009.)

Figure 7: Jean-Raymond in a caravan with the Tuareg and Jean-Louis Bernezat. (Algeria, 2009.)

# Jean-Raymond Abrial: A Rich Friendship

Egon Börger
Università di Pisa, Dipartimento di Informatica, Italy
boerger@di.unipi.it

When, at the beginning of the 1990s, my scientific interest shifted from pure logic to its applications in software engineering, I became acquainted with the author of the B-Book. In over 30 years, this developed into a close scientific cooperation, characterized by a common effort to develop mathematically precise state-based methods for the construction and analysis of reliable software-intensive systems. Here are some salient events we organized together.

- 1994–1996 Steam-Boiler Experiment
  - Dagstuhl seminar, *Methods for Semantics and Specification*
    https://www.dagstuhl.de/9523
  - Springer, LNCS 1165, *Formal Methods for Industrial Applications*
    https://link.springer.com/book/10.1007/BFb0027227
- 2001-2002 Summer Schools
  - *Formal Methods for Engineering of Software*,
    CISM, Udine (Italy), 24–28/9/2001. https://cism.it
  - *Software Technology* Lipari (Sicily), 2002.
    https://complex24.liparischool.it/previous-editions
- 2006 Dagstuhl seminar:
  - *Rigorous Methods for Software Construction and Analysis*.
    https://www.dagstuhl.de/en/seminars/seminar-calendar
    /seminar-details/06191
  - Festschrift https://link.springer.com/book/10.1007/978-3-642-11447-2
- 2006–2021, *ABZ Conferences* we organized to enhance the cross-fertilization of rigorous state-based software development methods:
  - Foundation of the ABZ conference series. https://abz-conf.org
  - 2006–2021, guiding the steering committee of ABZ.

Our cooperation made me know Jean-Raymond not only as a scientist but also from the human point of view, as a friend. This triggers me to refer to a famous distinction Cicero made between three categories of people: top-level scientists without human qualities, persons with the highest human qualities but without any scientific expertise, and those—very rare persons—who excel in science but also possess the highest human qualities. From what I know I can say that Jean-Raymond belongs to the third category.

# Jean-Raymond Abrial, an appreciation

Michael Butler
University of Southampton, UK

Jean-Raymond was a huge influence on my research from the very beginning of my career. While I was an undergraduate in Dublin in the 1980s, I was introduced to Z without being quite aware of his role in its development. I recall finding the idea of using set theory and logic to specify software requirements fascinating, and of course, Jean-Raymond was an early promoter of that idea. The precision and expressivity that Z offered compared with informal software modelling approaches appealed to me. The appeal was such that I decided I wanted to pursue a PhD in formal methods and ended up doing that in Oxford, the home of Z. Jean-Raymond had already left Oxford at that stage to work on the development of the B Method and B-toolkit with Ib Sørensen and others. I did attend a talk he gave in Oxford on B in around 1989. That was my first introduction to B, though I didn't make the switch from Z to B until after I finished my PhD in 1993. By that stage, Jean-Raymond had already moved back to France to work with Alstom and others on formal development of railway software with B, including guiding the development of what became the Atelier B tool, now maintained by CLEARSY.

In June 1995, I had the good fortune to attend the now-famous Steam Boiler workshop in Schloss Dagstuhl. I presented a development of the steam boiler case study using Ralph Back's action system formalism, work that was joint with Kaise Sere and Emil Sekerinski. (My PhD topic was a CSP semantics for action systems – supervised by Carroll Morgan – and I was working as a postdoc in Ralph's group in Turku during 1993-1995.) Jean-Raymond was one of the organizers of the workshop, and he straightaway saw the connection between our action system development of the steam boiler and his own B development – not surprising given the common heritage in Dijkstra's guarded-command language of action systems and B. I recall intense discussions with Jean-Raymond on the sidelines of the workshop, where he was keen to learn more about the basis and rules for action system refinement. That was the start of a collaboration and friendship that remains dear to me. I took a lecturing role in Southampton later in 1995 and continued correspondence with Jean-Raymond - initially by post as he was a late adopter of email! He invited me to visit Paris in 1996, where I spent happy hours explaining action system refinement to him, including the CSP semantics I had developed in my PhD with Carroll.

Jean-Raymond was interested in action system refinement because his thinking on how B could be used was evolving from modelling and reasoning about *software* to modelling and reasoning about *systems that contain software components*. The original refinement rules of B considered components as abstract data types with a one-to-one correspondence between abstract operations and refined operations, where component state is data-refined, and preconditions may be weakened in refinement. Action systems were developed to model and reason about reactive systems and provide more flexible refinement than abstract data types, including guard strengthening (for preservation of safety properties), multiple refinements of abstract events (representing different options), and refined events that are data-refinements of *skip*, i.e., refined events whose effect is not observable at the abstract modelling level. I recall Jean-Raymond showing me a B development he had constructed of a communications protocol, where an abstract operation represented the desired outcome of the protocol as a 'one-shot' event, and the refined model represented intermediate steps of the protocol as refinements of *skip* and strengthened the precondition of the refined one-shot operation. He knew this style of refinement felt right, even though it was different to normal B refinement, and he even had the intuition that the refined state provided a *variant* that explained termination of the intermediate steps. I recall his delight when I pointed out that action systems provided a semantic justification for his intuition (including the variant) and a precise formulation of the required proof rules. The desire to model systems, not just software, and the necessity of more flexible refinement rules led Jean-Raymond to evolve B to Event-B.

It is easy for researchers like me to get caught up in the details of a formalism and the nuances of guards, preconditions, semantics, etc., without considering any wider engineering value. Jean-Raymond had the knack of stepping back and appreciating the importance of the modelling and refinement style from an engineering perspective. When I heard the sad news of Jean-Raymond's passing early this year, I had a browse through his Event-B book again, and I was reminded of his determination that the importance of formal modelling and reasoning is broader than software correctness, as made clear by this quote from the book's prologue (written in the didactic style that he often adopted):

Programming is the activity of constructing a piece of formal text that is supposed to instruct a computer how to fulfil certain tasks. Our intention is not to do that. What we intend to build is a system within which there is a certain piece of software (the one we shall construct), which is a component among many others. This is the reason our task is not limited to the software part only. In doing this as engineers, we are not supposed to instruct a computer; rather, we are supposed to instruct ourselves. To do this in a rigorous way, we have no choice but to build a complete model of our future system, including the software that will eventually be constructed, as well as its environment, which, again, is made of equipment, varying physical phenomena, other software, and even users. Programming languages are of no help in doing this. All this has to be carefully modelled so that the exact assumptions within which our software is going to behave are known.

Over the years, I enjoyed many beautiful talks given by Jean-Raymond at various meetings, workshops, and conferences. His talks were always carefully prepared, with a clear message and flow (almost as if he applied a top-down abstraction/refinement approach to his talks!), and impeccable, engaging delivery. The photograph below from his invited talk at ABZ 2018 in Southampton, where he is expounding on the importance of abstraction and incremental refinement, is evocative of his style, with even the lighting and hand gestures giving it the air of a sermon.

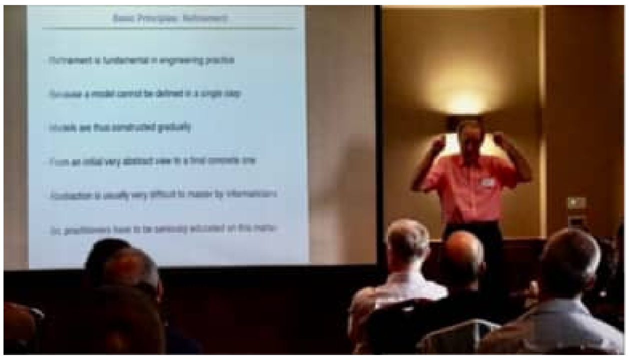


Figure 1: Jean-Raymond, giving an invited talk at ABZ 2018 in Southampton.
(Photograph by Chenyang Zhu.)

After my move to Southampton in 1995, my research interests became strongly focused on B, influenced, of course, by my growing collaboration with Jean-Raymond, but also by the attraction of the B-toolkit and Atelier B and the growing industrial uptake of B. I started to work with colleagues in Southampton on linking B to other notations, including combining CSP and B, and working with Colin Snook on the development of what became UML-B, which uses UML class diagrams and stage machine diagrams to provide graphical representations of B state and control structures. UML-B diagrams include the concept of diagram refinement, and we were strongly influenced by Jean-Raymond's insistence on incremental refinement in the design of this feature of UML-B. Michael Leuschel, an expert in constraint logic programming, joined Southampton not long after me, and he persuaded me that constraint logic programming could be used as the basis for a model-checker for B, leading to the development of the ProB tool. Michael subsequently moved to Düsseldorf, where he has led the evolution of ProB into an industrial-strength tool now regularly used on large railway projects.

Much of the funding that supported my research in Southampton was from EU-funded projects, and I have Jean-Raymond to thank for helping me to enter into the complexities and rewards of EU research programmes. In 1999, he was working with Thierry Lecomte from CLEARSY and Traian Muntean from Aix-Marseille on a proposal for an EU project on system-level modelling and reasoning with B. Jean-Raymond arranged for me to become part of the consortium, which also included Peter Ryan and Colin O'Halloran, who were then with DERA (now QinetiQ), and Kasia Sere from Turku. That proposal led to the MATISSE project, which was active from 2000 to 2003. Our involvement in MATISSE led in turn to our involvement in the RODIN project (2004–2007) and the DEPLOY project (2008–2012). During the MATISSE project, I recall giving an overview of the project at a workshop in Pisa that Cliff Jones and Sascha Romanovsky also attended, and discussions about common interests, especially on formal reasoning about fault tolerance, led to us working together on the RODIN proposal with Jean-Raymond and others. I recall proposal preparation meetings for RODIN in Brussels and Newcastle (in the days before online meetings) where it became clear that Jean-Raymond had a very bold vision for the project: I had anticipated that we would work on linking and evolving existing tools, whereas Jean-Raymond persuaded us that a completely new tool was required for Event-B. Hence, the development of what became the Rodin tool, initially developed in the RODIN project and further developed in the DEPLOY project.

Jean-Raymond was based in ETH Zurich during the RODIN and DEPLOY projects, where he assembled a strong tool development team, including Laurent Voisin, Stefan Hallerstede, and Thai Son Hoang. Jean-Raymond ensured that the new Rodin tool incorporated various innovations that he had identified and worked on over previous years. Of course, the key innovation was the support for the Event-B refinement rules that were more general than those of B, including new events (data-refinements of *skip*), multiple refinements of abstract events, and variants for termination of new events. Another innovation was an interactive prover with an intuitive graphical user interface that made it easy for a user to apply proof rules and navigate through a proof tree, mostly using mouse clicks. This prover interface was based on an experimental proof tool called Click'n'Prove that Jean-Raymond had developed with Dominique Cansell in Nancy.

Jean-Raymond also made sure that well-definedness was properly treated in Rodin with the introduction of pragmatic 'left-to-right' proof obligations for well-definedness, e.g., well-definedness of $x>0$ *&* $y/x=z$ is provable but not $y/x=z$ *&* $x>0$. A more general treatment of well-definedness would require more complex proof obligations (with disjunctions), leading potentially to a proliferation of cases to prove. Jean-Raymond's view, which I support, was that the compromise was justified as it largely corresponded to how a modeller would want to express a constraint in any case. Of course, I'm skipping over some of the nuances of the impact of well-definedness on proof rules, but these were addressed in the background in Rodin. This pragmatic approach to well-definedness was typical of Jean-Raymond's blend of elegance and pragmatism in order not to over-complicate the proof effort. In my view, he managed to find a sweet spot between elegance and pragmatism in various ways. Another example of this is the introduction of the *witness* statement in Event-B refinements. Similar to Z, data-refinement in B and Event-B allows for a relational correspondence between concrete and abstract states, leading to proof obligations with existential quantifications. These reduce the level of automatic proof as provers may struggle to instantiate existentially-quantified variables. The witness statement allows the modeller to explicitly provide witnesses for abstract variables, and these are used to simplify proof obligations and increase the level of automatic proof. While the motivation for witness statements is to support proof, they also increase the readability of refined Event-B models by explaining the correspondence to variables of an abstract model.

I occasionally got insights into the evolution of his workings prior to its presentation as an elegant top-down development or story. Once we were both due to present at a workshop on an automotive case study as part of the DEPLOY project. For reasons I have forgotten, Jean-Raymond was unable to attend, and he asked me to present his Event-B development of the case study. The week before the workshop, we talked through his development by phone, where he made clear how he wanted me to present it, and I tried to assure him I would do my best. Then, a day or so before the workshop, he sent me a new version of his development, with some significant improvements, and asked me to present the new version instead. I suggested to him that the audience would benefit from seeing both versions to gain insights into the iterative nature of formal development, but he was insistent that I only present the newer version. I decided to present both versions regardless, and I think the audience did appreciate that – though I don't think Jean-Raymond was entirely pleased when I told him afterwards!

I look back fondly at various technical discussions and debates I had with Jean-Raymond and, in some cases, can still remember where we had those discussions. I already mentioned intense discussions in Dagstuhl and in Paris. Another strong memory is when we took a transatlantic flight together to attend a conference in Florida. I had a problem proving that operations on a file store (such as *move directory*) preserved the hierarchy of the directory structure and didn't introduce cycles. Jean-Raymond took out his laptop and led me through an elegant reformulation and proof on the Rodin tool – definitely more engaging than an in-flight movie! Luckily, we managed to complete it before landing, and I was able to include the reformulation and proof outline in my talk. We both lectured at the Marktoberdorf Summer School in 2008, and I recall we took the opportunity to have detailed discussions on the sidelines about supporting an extension of the mathematical language and proof rules in Rodin. These discussions led to the *Theory* plug-in for Rodin. An early version of this was developed during the DEPLOY project with Issam Maamria, who did his PhD on theory extension with me. In fact, I continued to meet with Jean-Raymond until quite recently in the French EBRP project, led by Yamine Ait-Ameur, which improved and extended the Theory plug-in. Like Jean-Raymond, I attended the post-COVID meetings of EBRP online rather than in person, and the last time I met Jean-Raymond in person was in 2019 in Paris.

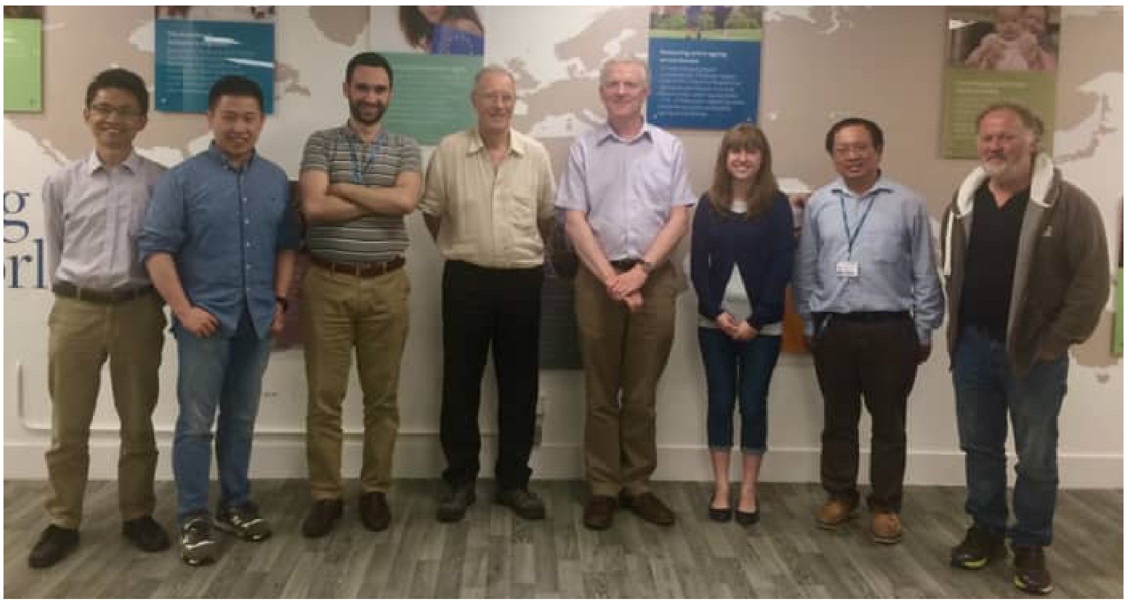


Figure 2: Jean-Raymond on a visit to Southampton in 2017.
Left to right: Keming Wang, Chenyang Zhu, Sadegh Dalvandi, Jean-Raymond, Michael Butler, Zoe Gathercole, Son Hoang, Colin Snook.
(Photograph by Asieh Salehi.)

As well as influencing my own work, Jean-Raymond influenced the work of many PhD students and colleagues at Southampton, such as Michael Leuschel and Colin Snook. Colin's PhD on UML-B was the first one that I supervised related to B. I counted up all the PhDs I've supervised at Southampton related to B, and I was pleasantly surprised that the total comes to 25 PhDs. Many of those students (some now colleagues) benefited from interactions with Jean-Raymond. He was always interested in the work of PhD students and generous with his encouragement and feedback. I know we all hugely appreciate his support and feel his loss.

# Mine Jean-Raymond

Dominique Cansell
Event-B Rodin Project (EBRP), Lessy
dominique.cansell@gmail.com

Jean-Raymond isn't mine. He belongs to everyone. If I use "mine", it's because this text is also my story: my account of the work I did with Jean-Raymond. I apologize for the phrasing. This is not a scientific text. The reader will easily be able to find the papers I'm talking about in this document.

## Meeting and Starting Work

I was extremely lucky to meet him, and especially to work with him. I wasn't really destined to meet him. In Metz, I did little research and taught algorithms to young students, using proofs quickly, with a preference for recursive algorithms. However, I didn't know it yet, but I had the mathematical background to do proofs in B.

In 1997, I decided to join Dominique Méry's team in Nancy. I didn't work directly on B. Since B wasn't a recursive language (I didn't know the methods), I foolishly decided not to use it. However, I did use Logic-Solver, a language used by Atelier B (STERIA, then CLEARSY). I sometimes called Jean-Raymond (but rarely) for advice.

In early 1998, Jean-Raymond gave a talk at LORIA (Nancy computer science research laboratory) on his predicate prover (PP), developed in Logic-Solver to automatically prove the rules of the Atelier B prover. During the summer of 1998, I wanted to learn object-oriented programming, and I chose PP as the program to develop. At the AFADL (Approches Formelles pour l'Assistance au Développement de Logiciels) conference in 1998, I talked to Jean-Raymond about my program. He had just moved to Marseille but suggested we discuss my program in more detail, and I could give him a demo in Paris, in Véronique Donzeau-Gouge's office at the CNAM (Conservatoire National des Arts et Métiers Paris).

I still wasn't doing B in research, but I had decided to teach it in Metz (master's program) and in the DEA program in Nancy (research master's) with Atelier B. To do this, I had to practice using Atelier B's interactive prover. It wasn't easy at first. We didn't always know where we were in the proof. It's understandable that many gave up. I taught myself so I could teach the students. The only secret to helping the prover is knowing how to do proofs. I was starting to use PP a lot during the interactive proofs. I also did my first fully proven B developments.

## Our Collaboration

In 2000, I went to Marseille with Dominique Méry, where I showed Jean-Raymond how I used PP in interactive proofs, and he showed us his development of the mobile agent (Luc Moreau). I taught it directly in my DEA program that September. Our collaboration must have started there.

Dominique Méry suggested we develop the leader election protocol (IEEE 1394). Working with Jean-Raymond is very difficult because you have to react quickly; otherwise, he does the work on his own. To be honest, even though I did some proofs on trees, I admit I was a bit behind.

Then I received a handwritten letter from Jean-Raymond with, among other things, the proof of Zermelo's theorem (his work with Guy Laffitte). Four days later, the models were fully proven with Atelier B. Jean-Raymond put his proofs on paper before moving on to the tool.

After all these proofs, Jean-Raymond asked me to work on the interactive prover interface to democratize the proof activity. This development (in Xemacs for the interface and Logic Solver) took us a lot of time (writing the program) and, in addition to our numerous exchanges (emails, phone calls), I went to Marseille at least once a month to take stock. Click'n Prove (or the Balbulette) was used extensively by both of us, researchers and students, before Rodin was developed.

At the end of 2001, I asked Jean-Raymond to participate in my day in Nancy dedicated to the B method (27 February 2002). He refused because his course at the CNRS school for young researchers had been taken away. This affected him greatly. I insisted, and in the end, I told him that only he could talk about B and disseminate his ideas. He finally agreed. Phew.

For all our developments, we began to introduce new concrete events into the abstraction to say that they respect the invariant and cause a variant to decay so that the Event-B proof obligations are generated by Atelier B, and this work led to our paper "Refinement and Reachability in Event-B".

In 2004, Jean-Raymond asked me to follow him to Zurich. I refused because I didn't want to leave my family, but I did a lot of proofs with Click'n Prove to certify the rules of Rodin's interactive prover. Jean-Raymond invited me at least twice. The second time, we worked on Simpson's "4-slots" algorithm (work published in his latest book on Event-B). We continued our developments with Click'n Prove until the end of 2006 or early 2007. After that, we only used Rodin, which I was already using with my students.

In 2008, I decided to officially stop my research (in conflict with the university and the policy of Inria management, both locally and nationally, which did not support me) and felt it was no longer worth investing in it. However, I always kept in touch with Jean-Raymond (or vice versa). I spent a lot of time rereading his book on Event-B before its publication in 2010.

When he returned to Marseille, I went there to see him and discuss our respective developments and proofs. We didn't have a joint project.

In 2017, Jean-Raymond was working on the Goodstein sequence. He had succeeded in proving the weak version but was stuck on the real Goodstein sequence, which required ordinals. He asked me to help him. Jean-Raymond thought he could prove it with Peano (induction over the natural numbers). He almost had a proof that seemed correct to me, but Rodin didn't agree. We almost gave up several times until we found the right structure (actually ordinals without knowing it) that allowed us to finish the proof. At the end of 2018, he made a big decision: he was moving into a senior living facility on 1 January 2019, because his bad legs made it difficult to climb the four flights of stairs. He told me there were possibilities for having guests. A word to the wise!

In 2019, Jean-Raymond sent me his direct model of a protocol (a variant of Leslie Lamport's Paxos protocol). He was desperate: he still couldn't implement the right invariants to formally prove the basic Paxos protocol. We took too long to find the last invariant, and we often almost gave up. When I found and proved the last invariant (on 19 December 2019), I made Jean-Raymond happy.

In 2020, Jean-Raymond started writing our paper on Paxos. He was taking longer than usual. For the first time, I wrote him parts with explanations in French as well as the Event-B models (events and invariants). During this period, there was also the pandemic. As he was often alone, I sent him photos of the flowers Isabelle had planted in our garden. I learned after his death that he had shared some with his loved ones.

At the end of 2020, Jean-Raymond asked me to participate in his instantiation of Rodin contexts to do mathematics as part of the ANR EBRP project (ANR: National Research Agency; EBRP: enhance Event-B, Rodin, and Rodin-PLUS). Initially, I refused because I no longer wanted to work with anyone other than Jean-Raymond, but as his health began to deteriorate, I accepted. It was an exciting time for me, but unfortunately, Jean-Raymond could no longer concentrate for long.

I went to see him twice in 2022 to show him the work he had asked me to do. He told me that in 10 years we had exchanged more than 6,000 emails (not including those from LORIA). He also asked me to answer emails about B from colleagues. I offered to take him for walks in his wheelchair. It was during these walks that I took my last two (and first) photos of Jean-Raymond. He told me it was the first time he had left the residence since he had been in a wheelchair, except for medical visits. He was modest and didn't want to impose his disability on others. It was his wish. He also gave me his old Mac. He didn't really tell me why. He just said, "Take it." I think it's because it contained our $\LaTeX$ files. I realized this when I had to complete his work on real numbers and our "Paxos" protocol. In fact, I'm writing this text on his Mac.

We continued to communicate for a long time by email and phone, but Jean-Raymond was very tired and could no longer hold a sustained conversation. His email advice was still sound. He never lost his head. In December 2024, I wrote a paper on his instantiation. I had put his name in the title, but he asked me to remove it. In early 2025, I spoke to him on the phone for a little longer; his voice was much better, and his problems at the senior center were resolved. I thought, great, I'll be able to see him again this summer. His last email was on 18 February, telling me there was still work to be done. He had read in *Le Monde* (which he read every day) that mathematicians were going to prove Fermat's theorem using Lean. His last text message was on 24 March 2025, telling me he had already seen Ireland vs. France in women's rugby and that there was no need to send him a link to the match summary. He loved rugby.

I sent him more texts and MMS messages after his death. The last one was a photo of a circular slide rule to ask if he knew it.

## Personal Reflections

Jean-Raymond also had many constants in his life: lots of stairs, the same restaurant (until it closed for good), and the same meals (sausage and pasta). To reach his apartment in Marseille, he had to climb four floors (five in Zürich). Then, after a good coffee, Jean-Raymond would write on sheets of paper the schedule of objectives to be achieved for my visit. At the end, he would cross out what had been done and then tear up the sheets, satisfied with the work accomplished. Jean-Raymond was also a great walker and a great climber. I remember (in 2002) short climbs in the Calanques with him and Christophe Metayer, and especially a long walk in the Calanques with him and Ralph Back. We had to keep up with him; he exhausted us. He was a very good skier and a desert enthusiast, and when he was there, I was on vacation; no email or phone. Others will undoubtedly speak about it better than I can.

Jean-Raymond was an outstanding speaker who took a lot of time to prepare his presentations. One always had the impression that his presentations were simple and clear. During the conferences I organized, there were always many people there to listen to him, and people came to Nancy from far and wide. He also loved to share his knowledge and to work with young people, as he did at the School for Young Researchers, in Zürich and, more recently, in Shanghai. He trained his students in proofs using Rodin and limiting the automatic rules of the prover. Li Qin will undoubtedly speak about this better than I can.

He loved dancing. In 2016, I saw a Luc Petton show with his Manchurian cranes. He absolutely wanted to see it. Luckily, a month later, the show came to Marseille. He went there with a cousin. "Dazzling," he said. During the pandemic, I sent him links to shows by Béjart and others.

Jean-Raymond loved paintings like Fernand Léger, Braque (he produced less than Picasso, but for him, it was of great quality), Klimt (which he admired in Vienna in 2014), and especially Vermeer. In 2015, he went to Dresden, where he had the pleasure of admiring two Vermeers, including "The Young Woman Reading a Letter at the Window". In 2017, in Frankfurt, he saw The Geographer at the Städel Museum. He has seen about ten Vermeers out of the 35 known. His regret is having missed the Vermeer and the Masters of Genre Painting exhibition at the Louvre.

In 2009, we went to the Centre Pompidou Metz to see the first exhibition together: Chefs-d'œuvre.

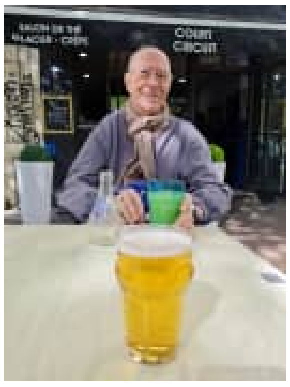
At each of my museum visits, I sent him photos of exhibitions like the one by Fernand Léger in 2017, the one by Braque in Rouen in 2019, and many others.

Jean-Raymond was very discreet. I don't know if he compartmentalized himself by not talking to me about his previous work (except for B) and colleagues, and/or the people who didn't like him. I think that for the former, he didn't look back and moved on, without forgetting that for the latter, he was like that, and it was probably also to protect me. In fact, I only knew the surface of Jean-Raymond. Reading the tributes dedicated to him, I discovered Jean-Raymond's submerged part. I thank Nora, his housekeeper, for taking care of him until the end.

On June 18th, I went to Dourgne in the Tarn region to see the grave where his ashes will be buried.

Not knowing what to do, I drew a B there with cypress cones from a cypress tree in the cemetery. Goodbye, Jean-Raymond. It was a real pleasure working with you. Thank you.

Here are my two photographs of Jean-Raymond and my “B” in Dourgne:



Hats off Jean-Raymond!

# What I owe to Jean-Raymond Abrial

Henri Habrias
Retired from Nantes Université, France
henri.habrias@univ-nantes.fr

When did I discover Jean-Raymond Abrial? Perhaps it was when I bought the proceedings of the Cargèse conference and read his article *Data Semantics* [1]. Or was it while reading the publications of EDF's DER-IMA (located in Clamart), especially on Z?

What's certain is that I discovered Z in Meyer Baudoin's *Méthodes de programmation* published in 1978 [2], "*the first ever published description of the Z specification language anywhere (as far as I know)*" (B. Meyer) and then in Delobel Adiba's *Bases de données et systèmes relationnels* [3].

When I started studying "business computing" (in the Computer Science department of the University Institute of Technology), I found what I was reading to be rather unscientific. "Information", "data", "section", etc.

Regarding "data", I learned the theorems of Jean-Louis Rigal, who was a professor at Dauphine University. Easy: Th. 1, "Data is not given" because it is the result of an abstraction; Th. 2, "Data is not given" because once given, it is no longer data (it no longer changes the state of your knowledge). Oh! I just introduced another term!). It's no longer a scoop. Journalists have understood this well. Th. 3 "Data is not given", it costs money, it's not free! Even with the Web. Someone pays.

In 1970, I read and reread *La science informatique* by Jacques Arsac [4]. I at least understood the difference between "information" and "knowledge" (according to J. Arsac's definitions).

Paul Namian of the CNAM (Conservatoire National des Arts et Métiers, Paris, where Jean-Raymond later became a professor, had written an article *Approche théorique du traitement des informations administratives*, [5] also trying to clarify things, and Claude Pair, had written "Information Structures" [6].

We talked about "methodology" (even though we weren't studying methods; today, methodology is more sophisticated than "method"! Who would dare publish a journal called "Work and Methods" (where I published quite a bit)?). We talked about "analysis", then it was... "Design", "conceptual analysis"—the word "structure" was everywhere, as was "model." But it was Michel Serres, invited to Nantes by Jean-Louis Gardies, who enlightened us with the example of the fable "The Wolf and the Lamb", [7] on Structure and Model.

This reminded me of primary school. A teacher (now called a "primary school teacher") asked us, "What do you want to be when you grow up?" One student answered "chartered accountant" (his father was an accountant). Another then answered "expert hairdresser" (his father was a hairdresser). In the 1980s, he would have answered "designer hairdresser", and nowadays I see signs with "Hairdressing Concept" and other variations.

Since I'm talking about primary school teachers, let's also talk about secondary school teachers. It was the philosophy teacher who talked to us about truth tables and sets. Admittedly, it didn't go very far! He explained the empty set using the symbol of the 'no parking' sign.

I mentioned Z. But was also the DBMS (I'm using an anachronism here, the term wasn't in use yet) Socrate by J.-R. Abrial, also discovered in Delobel Adiba. Socrate was used under that name and then as Clio at the Indret arsenal near Nantes.

When I arrived in Nantes in 1971, I taught the Cantor method. It had nothing to do with Cantor. But it went from studying existing systems to generating COBOL code. I had learned it during an internship at ICL, which used it.

Following my reading on Z, I bought the first volume of Bourbaki's book [8] with its hardback, cloth-covered binding, made to last. I didn't manage to get very far (but I must have gotten further than in Ulysses by James Joyce!). But at the end, there's the *Historical Note* and the *little results booklet*. And I was able to read it. Unfortunately, it's much more readable than some of the textbooks that were distributed when the math reform was launched in France. Jean-Raymond talks about this booklet in his lecture at the Collège de France [9].

Perhaps it was around this time that I became interested in the SETL language, which was developed at New York University.

I can't remember if I taught Z (the Oxford Z), before or after discovering NIAM (Nijssen Information Analysis Method) from Control Data in Europe. NIAM is the use of the binary relational model. I published the first book in the world (!) on NIAM (but it also dealt with the n-ary relational model and others) in 1988 [10]. I had spent a year on secondment to CMILACO (Crédit Mutuel Informatique Loire Atlantique Centre Ouest) where NIAM was used. In 1985, Jean-Pierre Giraudin and Monique Chabre-Peccoud published *For a Rehabilitation of the Binary Relational Model* in Bigre+Globule. It was Jean-Pierre who saved the 2025 conference by preparing a presentation on Z, Socrate in Grenoble [11].

In 1984, J. Raymond published *Spécifier ou comment matérialiser l'abstrait* (Specifying, or How to Master the Abstract) in [12]. The article begins with a boxed quote from Proust. Typical for someone who named a DBMS Socrate. But that's just not done! You have to read the article's introductory text by an academic!

> *Be careful, this article is different. [. . . ] Some theorists may smile at the simplicity of the calculations, while some practitioners will be put off by the formalism used. But the great qualities, both didactic and pragmatic, of this paper cannot be denied.* (Gérard Memmi)

Apart from the last sentence, the rest, and I haven't quoted everything, shocked me. The article is impossible to find these days. A plea to everyone! But I used the first two paragraphs for my introduction to the *Formal Specifications* chapter of my book *Introduction to Specification* [13]. All is not lost!

In 1985, Jean-Raymond gave a lecture at CEPIA in Rocquencourt. Guy Laffitte from INSEE[1] listened. And he would later frequent 26 rue des Plantes, Paris, 14th arrondissement, where Jean-Raymond lived. We could have crossed paths at one of those two places. It was probably in 1985 that I visited Jean-Raymond on rue des Plantes. I remember that before the dinner prepared by Hélène, he showed me his proof booster. I also remember that he spent long periods in Oxford at

[1] Institut National de la Statistique et des Etudes Economiques.

the time, long enough, he told me, to absorb the English accent. In the B-Book, Jean-Raymond wrote: "*G. Laffitte influenced this work by his careful reviews, his accurate criticisms, and the sometimes very serious rearrangements he proposed for some of the mathematical developments in this book.*"

I read in the bibliography of [13] *A small case study in program design*, November, J.-R. Abrial, 1986, unpublished. This must therefore be the first year of my exchanges with Jean-Raymond. *The Case of Proportional Representation Elections in LCP, Deductive Programming*, J.-R. Abrial's Method", IUT of Nantes. Claude Pair had addressed the case in deductive programming. For LCP, it was Alain Coulon and M. Koutchouk from Bull. And also *A Formal Introduction to Abstract Machines*" May 4, 1987, J.-R. Abrial, published by the author.

If I'm not mistaken, it was in 1987 that I gave as the subject of an engineering thesis *Four Specification Methods: SADT, Merise, F1 Formalism, Abrial's Mathematical Specification, Application to an Example: The Electronic Alarm Clock.* The formalism was then that of the TSI article. But JR sent several times a week his work in progress which was his progress towards B.. One day, perhaps the day of the defense, he came with his tools to give us a demonstration. He must have been working at CNAM in Paris at the time.

In 1988, the *B User Manual, Tech. Report*, Programming Research Group, Oxford University, was published.

In September 1989, I believe Guy Laffitte, then working in Nantes, came to my third MOACSI conference to present Jean-Raymond's text, *A Formal Approach to Software Construction.*

In 1994, I published a French translation of David Lightfoot's book *Formal Specification Using Z* [14] with supplementary material, such as a French–English glossary. It was my course on Z.

In 1996 (or 1995), the book I mention later [15] was published in 1996. I went to teach a course on NIAM at the Swiss Federal Institute of Technology Lausanne (EPFL). And I remember… that J.-R. Abrial had received a request to teach B there. A colleague went in his place. The course was published in a book [15], and a colleague from EPFL sent me my course rewritten with mathematical notation. This reminded me of a course I had given to math professors on the Merise method (the French equivalent of SSADM). During the break, one of them came to rewrite the conceptual data models (CDMs) in mathematical notation. I confirmed to him that he had understood everything!

In NIAM, we use a graphical notation with two symbols to express the 16 cases of binary relations: the V, which intersects a line, gives the universal quantifier (the "for all") for the totality constraint, and the line for the uniqueness constraint (as in Codd's n-ary relations, the key is underlined), and a few others for the constraints of equality, inclusion between relations, domains, codomains. What's important is that we should write, at best, two *subject-verb-object* sentences for the relation and its inverse. It's then easy to move to an "n-ary relational model", which in NIAM isn't called a "logical model" (why "logical"?!), but a "grouped model".

I see that in my 1988 book [10], I cite *Data Semantics* (1974) and *Proof in Specifications* [16]. I note that Nijssen was one of the organizers of the Cargèse conference.

In 1993, with a philatelist colleague, we were specifying the Yvert & Tellier Catalog in Merise, NIAM, and Z.

One day, Bill Stoddart, a colleague from Teesside University in Middlesbrough (the city that kept its transporter bridge while Nantes lost its), toured the universities. where Z was taught. He went to Grenoble and then to Nantes. Z was the lingua franca for several teachers. Through Erasmus, there were several meetings of teachers and students from different countries where we created Niam diagrams and then Z diagrams. And in the UK, Robin Milner's CCS. Then colleagues from Teesside (Bill Stoddart, Steve Dunne, Andy Galloway) taught B and published on B. And they went to see Jean-Raymond in Marseille.

In 1994, two years before the publication of the B-Book, Jean-Raymond Abrial came to Nantes to teach students in the APPC program (am I being anachronistic again? "Année Post Premier Cycle" followed "Année Spéciale"). The lecture was video-recorded at the IUFM (University Institute for Teacher Training) in Nantes, which had a studio, and distributed by the IUT of Nantes and the publisher Teknea in Toulouse. There are six VHS tapes introducing B and eight case studies, along with copies of the transparencies. It was a very pedagogical course. Jean-Raymond used "papygrams".[2] On that subject, I remember that when I provided students with the Merise notation, I told them to ask their contacts in companies to provide them with the papygrams. And that they would realize their interlocutors hadn't understood the "cardinalities". When I met them on the train, they reminded me of my advice. And that I was right. They had tested it. I remember when J. Raymond explained the refinement theorem very well. I was then able to give a B-level lecture.

I remember that I had told him about Niam's simple notation. But the Z notation was already widely used and had even become a standard. At the 2025 Nantes conference, I presented it. It uses only the stroke ———, the $\vee$, and the points $>$ or $<$. The less constrained a relation is, the fewer symbols are used. Any relation is represented by ——. The most constrained by <-$\vee$—$\vee$->. We called it Oc, in reference to Occam and his razor… and to Occitan.

1995: I had planned the first international conference on B Method in Nantes with Jean-Raymond. But the publication of the B-Book was delayed. So I changed the name of the conference. It was short for Z2B (it was fashionable at the time to use "2" to make "to"), and full for "Z Twenty Years On – What is its Future?" [17]. At this conference, I created the APCB, Association de Pilotage des Conférences Internationales B (Association for Organizing International B Method Conferences).

1996: In November, the 1st Conference on the B Method was held in Nantes [18].

1995–1996 were also the years of the BUG (the B User Group) frequented by users of the B Workshop. I remember Ranan Fraer, who helped many people with their problems. I met up with Ranan again at the EDF, CEA, INRIA Summer School at the Château de Bréau-Sans-Nappe from 12th to 23rd June 1995, where Fernando Mejia distributed a document entitled *The Method and the Language B*. Ranan was working on Atelier B day and night. And I learned from him that it was the first time he'd ever seen Atelier B![3]

[2]A *papygram* is a visual, diagrammatic way to show a mathematical relation between elements of the same set, using points and arrows to illustrate connections.

[3]On his page, still available: that's rare!
**https://www-sop.inria.fr/croap/personnel/Ranan.Fraer.html** I read: "Ranan Fraer Formal development in B of a Minimum Spanning Tree Algorithm distinguished with the 'Best Paper Award' at the First B Conference, Nantes (France), November 1996." So he must have gone home with the B-Book! And "*I'm also*

1996 was the year of the Steam Boiler Control Specification, held at Dagstuhl Castle in northern Saarland. It also saw the first international B conference in Nantes [18]. Jean-Raymond announced what would become the “event-driven” B method with his presentation entitled: *Extending B without changing it (for developing distributed systems)*."

In 1997, at the AFADL conference in Toulouse, J.L. Lanet spoke following Jean-Raymond’s presentation. Gem+ would use B for the smart card. And later, I went to spend a weekend in Marseille with Jean-Raymond after a student from the IUT (University Institute of Technology) defended their internship at Gem+. I haven’t forgotten. During the thesis defense, some PhD students discovered that the other student… was in a university institute of technology (IUT)! I also discovered the Calanques by following Jean-Raymond. We went as far as the Luminy campus and took the bus back. When we arrived at the Old Port, I had a lot of trouble walking. I spent the next day lying down. And Jean-Raymond went back to the Calanques in the morning. He had just returned from a desert expedition. Was it the particularly grueling Tamanrasset-Djanet camel trek where, on the very first day, one of the group members died and whose body had to be transported for the entire expedition?

In 1997, the computer science department of the Nantes IUT published the book [19]"La Méthode B, Concepts, Démarche, Exemple d’applications” by Karl Emeriau, written following his CNAM engineering thesis. Karl was self-taught in B programming thanks to the B-Book. All the applications are so-called “management” applications, developed using the B Workshop, right up to the automatically generated C code.

1997 was also the year of the 1st IEEE International Conference on Formal Engineering Methods, held in Hiroshima [20] (M.G. Hinchey, Shaoying Liu, eds.). I had submitted a paper co-authored with a colleague, *Formal Specification of Dynamic Constraints with the B Method*. I knew we had a problematic point (a theorem to prove, I believe). One day, I received a call from Jean-Raymond. I was mortified. I knew he was a member of the program committee. I immediately told him I shouldn’t have sent the paper. He had a problem. But Jean-Raymond said, *I’m calling to tell you that all you need to do is…* —a very small but crucial correction! I anonymously thanked a reviewer in the article. I had to cancel my trip at the last minute. I would have met up again in Japan with several English friends I had made through Z and then B.

In 1998, Claude Boksenbaum organized the second B conference in Montpellier in April. Didier Bert edited the proceedings, the first published in the LNCS series by Springer-Verlag.

In 2000, the “First International Conference of B and Z Users” took place at York in August–September. The editors were Jonathan Bowen, Steve Dunne, Andy Galloway, and Steve King.

In 2001, I published “Formal Specification with B” [21], Hermes Lavoisier, ISBN 2-7463-0302-2, in collaboration with Jean-Yves Lafaye and Marie-Laure Potet, with a foreword by Fernando Mejia (Head of Software Development Methods and Tools, Alstom Transport Signalisation), which begins as follows:

> “For some time, I had been experimenting with algebraic specification methods at what was then called Bull Systèmes, when, in 1985 or 1986, during a training session

*an enthusiastic supporter of Jean-Raymond Abrial’s B method. There’s a growing interest in B, as it has already been successfully applied to several large-scale industrial projects in France and the UK.”*

> given by Jean-Raymond Abrial, I discovered what would a few years later become the B language and method. It was love at first sight!"

He tells us the story of B as he experienced it.

In 2003, the 3rd Z and B User Conference was held in Turku, Finland. Turku, where Ralph-Johan Back taught at Åbo University, which is often mentioned in Jean-Raymond's articles. It was June, a time when the nights aren't dark. It's worth noting that it's a Swedish-language university. Of course, we met up with Jean-Raymond and many others there.

I have kept the exchanges on the B Forum, run by INRETS Forum B, from 2004 to 2008. Here are the names of the participants whose messages I have kept: Michael Leuschel (HH Univ., Dusseldorf), Samuel Colin (Loria), Simão Melo de Sousa (Universidade da Beira Interior), Guy Vidal-Naquet (Supelec), Georges Mariano (INRETS), Marc Guyomard (Enssatt, Lannion), Steve Dunne (Teesside Univ.), Dominique Cansell (Loria), Daniel Zingaro (McMaster Univ.), Ken Robinson (UNSW, Australia), Ib Holm Sørensen (B-Core), Elisabeth Ball (CSE, Southampton), Jeremy L. Jacob (Univ. York), Yann Zimmermann (Keesda).

2004: Start of the Rodin project with Laurent Voisin. Jean-Raymond was a professor at ETH Zurich until 2009. where he worked on Rodin with Laurent Voisin and others. He talks about it in his lecture at the Collège de France [9].

2008: In October, I retired. And I organized the conference "The B Method: From Research to Teaching" [22], where Jean-Raymond was present. On 13 September 2008, Jean-Raymond received a "doctorat d'honneur" from the University of Sherbrooke in Quebec, where Marc Frappier taught. In 2008, the first ABZ conference was held at the British Computer Society in London. In 2009, the second "The B Method: From Research to Teaching" conference took place.

2014: I attended the Toulouse ABZ conference and meet up with Jean-Raymond, who was returning from China. Also, there, of course, was Egon Börger (he must have attended every one) – who was a visiting professor in Nantes for a few months – Leslie Lamport, and Tony Hoare. When I went to visit the Toulouse Aviation Museum with my grandchildren... I recognized the restaurant where the gala dinner was held.

2015: Jean-Raymond's lecture at the Collège de France. I remember him telling me that his laptop had been stolen before the lecture.

2022: These were my last email exchanges with Jean-Raymond. He had just told me that he was in a wheelchair and living in a suitable facility. He was reading Montaigne's Essays and was interested in Japanese civilization. I offered to visit him, but he replied that he wasn't well. In her presentation at ABZ 2025, Dominique Cansell concludes with two photos taken during her last two visits with Jean-Raymond. That was in 2022. But Dominique communicated with Jean-Raymond in subsequent years.

25 May 2025: The J.-R. Abrial lecture, honouring a pioneer in the scientific development of computer languages and formal methods and their large-scale application in industry, took place as part of the Scientific Days at the University of Nantes. We learned on 3rd June of Jean-Raymond's death, the day before the event.

From 10th to 13th June 2025, the ABZ conference took place, where Dominique Cansell presented work done with Jean-Raymond a few years prior. It was 30 years after the first conference on Z and B in Nantes!

And I never imagined a year ago that, at almost 79 years old, I would be publishing a paper on B at a conference ("Formal Modelling of Information System Evolution using B")! I rediscovered this paper while preparing the presentation on Jean-Raymond. My colleague had to go back to the B Workshop because only fragments of the B specifications remained. And the tools have evolved since 2008! The paper deals with applications where the rules change according to phases. This is the case, for example, with the academic year. It is also the case with changes in legislation.

Thank you, Jean-Raymond, for everything you taught me, from Socrate to Z, to B, to Event-B, to Rodin, for your teaching skills, for sharing your work with so many people, and for your warm welcome, whether in Paris or Marseille, for your visits and teaching to Nantes. Thanks to you, I connected with many people who also spoke of how much you had given them. Thanks to you, I didn't work, I had fun.

## References

[1] Abrial J.-R. (1974) *Data Semantics*in Database Management (Klimbie, Koffeman, eds); North-Holland, pp. 1–59

[2] Meyer B., Baudoin C. (1980) *Méthodes de programmation*, Eyrolles, Paris, ISBN: 2-21201581-1

[3] Delobel C., Adiba M. (1983) *Bases de données et systèmes relationnels*, Dunod, Paris, ISBN: 2-040116328-1

[4] Arsac J. (1970) *La science informatique*, Dunod, Paris

[5] Namian P. (1966) *Approche théorique du traitement des informations administratives* in journal Automatisme, Volume XI, No. 10, October

[6] Pair Cl. (1971) *Les structures d'information et leur représentation en mémoire* Ecole d'été, Alès, 1971

[7] Serres M. (1976) *Le jeu du loup* in *Savoir, faire, espérer: les limites de la raison*, Facultés Universitaires Saint-Louis, Bruxelles

[8] Bourbaki N. (1990) *Eléments de mathématiques. Théorie des ensembles. Chapitres 1 à 4, nouveau tirage*, Masson, Paris, ISBN: 2-22581909-2

[9] Abrial J.-R. (2015) *Spécification, construction et vérification de programmes: le parcours d'une pensée scientifique sur une quarantaine d'années* Séminaire au Collège de France, 1 April, `https://www.youtube.com/watch?v=1yF6R-mjTnE&t=19s`

[10] Habrias H. (1988) *Le modèle relationnel binaire, méthode I.A. (NIAM)*, Eyrolles, Paris, ISBN: 2-22584367-8

[11] Adiba M., Giraudin J.P. (2024) *Du Big Data à l'IA 60 ans d'expérience en traitement des données, des informations et des connaissances à Grenoble*, UGA Editions, Grenoble, ISBN: 978-2377474882

[12] Abrial J.-R. (1984) *Spécifier ou comment maîtriser l'abstrait* in TSI, Vol. 3, N° 3, pp. 201–219

[13] Habrias H. (1983) *Introduction à la spécification*, préface de H. Gallaire, présentation de M. Jackson, Masson, Paris, ISBN: 2-22582768-0

[14] Lightfoot D. (1994) *Spécification formelle avec Z*, traduit par H. Habrias, Teknea, Toulouse, ISBN: 2-87717038-1

[15] Strohmeier A., Buchs D. (eds) (1996) *Génie logiciel: principes, méthodes et techniques*, Presses polytechniques et universitaires romandes, Lausanne

[16] Abrial J.-R. (1987) *Prototyping and Software Specifications*, Afcet Conference, INSA Lyon, January

[17] Habrias H. (ed.) (1995) *Z Twenty Years On – What is its Future?* 7th International Conference on "Putting into Practice Methods and Tools for Information System Design", proceedings, October, 10–12, Nantes in co-operation with ZUG and BUG, ISBN 2-90608219-8

[18] Habrias H. (ed.) (1996) *First B Conference*, Proceedings, November 25–26, IRIN, ISBN: 2-90608225-2

[19] Emeriau K. (1997) *La Méthode B, Concepts, Démarche, Exemple d'applications* I.U.T. de Nantes, ISBN 2-906082-26-0

[20] M.G. Hinchey, Shaoying Liu,(eds) (1997) *1st IEEE International Conference on Formal Engineering Methods*, Hiroshima

[21] Habrias H. (2001) *Spécification formelle avec B*,in collaboration with Jean-Yves Lafaye and Marie-Laure Potet, Hermes, Lavoisier, Paris, ISBN: 2-74620302-2

[22] Habrias H., Attiogbé C. (ed.) (2008) *The B Method: from Research to Teaching*, Journées Scientifiques de l'Université de Nantes, June, ISBN: 2-95124612-9

# Three Years in Zürich

*In memory of Jean-Raymond Abrial*

Stefan Hallerstede
Aarhus University, Denmark
sha@ece.au.dk

During the period from autumn 2004 to autumn 2007, Professor Jean-Raymond Abrial gathered a small group of researchers at ETH Zürich in order to finalize work on Event-B [7], work on its accompanying tool Rodin [4], and write a book about Event-B [3]. The work during this period was financed by the EU project RODIN. Abrial was in a perfect position for supervising the work; he had devised the B Method [1] and worked on a supporting tool called Atelier B [9] during the two preceding decades. Abrial had started the work on Event-B well before 2004 [8, 2, 6] but until then Atelier B and Click'n'Prove [5] had been used for writing formal models and doing formal proofs. The user interaction of Rodin is based on the experience gathered with Click'n'Prove. During the period at ETH Zürich Event-B and Rodin were developed glove in hand to fine-tune notation, method and tool: good tool support is essential if larger, more complex systems are to be mastered. For Event-B, this meant to blur the boundary between models and proofs: elements of proof were—after (mostly) long discussions—moved into models and explained in terms of the abstraction offered by the model.

I moved to Zürich after spending three years working in a startup in Grenoble, France, where I worked on verified hardware design using the B Method, Event-B, Atelier B and Click'n'Prove. The work resulted in a translator from B into register transfer level hardware descriptions that were synthesized into FPGAs. During this time, I had met Abrial occasionally during project meetings and meetings at various industry sites in France. Abrial had moved to Zürich to take up a post as guest professor at ETH Zürich. When a position became available, I contacted him, and he invited me for an interview. At the interview, there was an exceptionally friendly atmosphere, and there was no question that I ought to join. Friendliness also describes Abrial's style of leadership very well; the worst possible reaction one could evoke was a frown (which would last for several milliseconds). Professor Abrial preferred to be addressed as *JR*, just JR. By everyone.

At first, we were three: JR, Laurent Voisin, who had arrived before me, and myself. Laurent Voisin joined from a company called CLEARSY and brought with him the skills that were needed for professional tool development. At first, we occupied two offices in a larger building in Clausiusstrasse 59 that belonged to computer science. A little later, Thai Son Hoang, who had just finished his PhD at the University of New South Wales, and PhD student Farhad Mehta, who had just received his MSc at the Technical University of Munich, joined the group. Because of the increased space requirements, we moved to a separate house that was painted pink in Clausiusstrasse 47. It turned out to be a beautiful place that got as close to a family atmosphere as was possible for a working environment. After the different tasks for the development of the main components in Rodin were decided on, we sat together once per week in JR's office to discuss progress and agree on the next steps. Technical discussions concerning key concepts, the formalism, the methodology, and the implementation took place continually, so that decisions could be made during the weekly meetings. Although JR was quite good at managing the group, he was not keen on aspects that he appeared to consider more administrative. He focused on what was important, the work on

Event-B, Rodin, the lectures, and the book, quite happily delegating administrative toil.

Together with Professor David Basin, JR taught a course entitled “Functional Programming and Formal Methods”. One half of the course taught Haskell, the other half Event-B. The teaching in the course mattered a lot to JR, and the group invested a sizable effort in supporting it. He prepared the slides with utmost diligence, advancing the material in small, well-digestible steps and animations where they were of use. In order to demonstrate the example of the mechanical press that can also be found in the book [3], he implemented an animation in C with a “graphical” visualization in a block graphic in a terminal. All of this made his teaching didactic and authentic, and this showed in the students’ engagement in the exercises. One of the MSc students, François Terrier, joined the group to write his MSc thesis on implementing a theorem prover for Rodin.

The three years at Zürich had a profound effect on my thinking and work [9]. The approach taken in the development of Event-B and Rodin, the need to consider the practical use of logical concepts, the transparent relationship between all modeling artifacts that support a user’s understanding and how this can be done concretely, are well applicable in all sorts of verification and validation frameworks. When I prepare presentations or lectures, my approach is very much inspired by JR’s approach of worked-out, animated examples that explain the key points step by step to the audience or students. This is certainly a result of having seen and listened to him on many occasions. I am very grateful for this time.

## References

[1] Jean-Raymond Abrial. *The B-Book: Assigning Programs to Meanings*. Cambridge University Press, 1996.

[2] Jean-Raymond Abrial. Event based sequential program development: Application to constructing a pointer program. In Keijiro Araki, Stefania Gnesi, and Dino Mandrioli, editors, *FME 2003: Formal Methods*, volume 2805 of *LNCS*, pages 51–74. Springer, 2003.

[3] Jean-Raymond Abrial. *Modeling in Event-B: System and Software Engineering*. Cambridge University Press, 2010.

[4] Jean-Raymond Abrial, Michael J. Butler, Stefan Hallerstede, Thai Son Hoang, Farhad Mehta, and Laurent Voisin. Rodin: an open toolset for modelling and reasoning in Event-B. *STTT*, 12(6):447–466, 2010.

[5] Jean-Raymond Abrial and Dominique Cansell. Click’n’Prove: Interactive Proofs within Set Theory. In *Theorem Proving in Higher Order Logics*, volume 2758 of *LNCS*, pages 1–24. Springer, 2003.

[6] Jean-Raymond Abrial, Dominique Cansell, and Dominique Méry. A mechanically proved and incremental development of IEEE 1394 tree identify protocol. *Formal Aspects of Computing*, 14(3):215–227, 2003.

[7] Jean-Raymond Abrial and Stefan Hallerstede. Refinement, Decomposition and Instantiation of Discrete Models: Application to Event-B. *Fundamentae Informatica*, 77(1–2):1–28, 2007.

[8] Jean-Raymond Abrial and Louis Mussat. Introducing dynamic constraints in B. In Didier Bert, editor, *B'98 : The 2nd International B Conference*, volume 1393 of *LNCS*, pages 83–128. Springer, 1998.

[9] Stefan Hallerstede, Robby, John Hatcliff, Jason Belt, and David S. Hardin. Proof engineering in logika: Synergistically integrating automated and semi-automated program verification. In Anne Remke and Bernhard Steffen, editors, *Formal Methods for Industrial Critical Systems – 30th International Conference, FMICS 2025, Aarhus, Denmark, August 27–28, 2025, Proceedings*, volume 16040 of *LNCS*, pages 39–58. Springer, 2025.

[10] Thierry Lecomte. Proving B with Atelier B. In Simon Foster and Augusto Sampaio, editors, *The Application of Formal Methods: Essays Dedicated to Jim Woodcock on the Occasion of His Retirement*, volume 14900 of *LNCS*, pages 329–345. Springer, 2024.

Figure 1: Jean-Raymond Abrial (right) and me, Saas Fee, July 2007. (Photograph by Gabriela Hallerstede.)

Figure 2: Jean-Raymond Abrial, the passionate skier, Saas Fee, July 2007.

Figure 3: Jean-Raymond Abrial in the garden behind the pink house, Zürich, June 2006.

# Remembering Jean-Raymond Abrial

Thai Son Hoang

December 2025

There are no words that can describe my gratitude to an incredible person like Jean-Raymond Abrial.

I first met Jean-Raymond at the ZB2003 conference in Turku, Finland, in 2003. I was a PhD student attending my first conference, so I was understandably quite nervous. When my supervisor, Ken Robinson, introduced me to Jean-Raymond over a coffee break, I was at a loss for words. After all, I had been studying the B-Method for a few years, and to meet the person who invented the method is an extraordinary opportunity. The warm greetings from Jean-Raymond had made it very clear how approachable he is, and without a doubt, an inspiration to many of us. Needless to say, his talk titled "*B#: Toward a Synthesis between Z and B*" [1] set out the programmes for several projects and research directions for me, and it will continue to be the case for the foreseeable future.

After a brief meeting in Turku, I met Jean-Raymond again when he visited Carroll Morgan and Ken Robinson in Australia in October 2003. Another remarkable talk was delivered at the University of New South Wales (UNSW), and we (Jean-Raymond Abrial, Carroll Morgan, Annabelle McIver, Ken Robinson, and I) had a fantastic day out hiking in the Blue Mountains. Jean-Raymond was often leading the group, while I was most of the time the last one trying to keep up with the pace of the hike. At that time, I was still using an old film camera while Ken Robin had a much fancier digital camera. The advantage for me was that I had the date printed on the photo (Figure 1), but clearly, Ken knew better how to take a good picture (Figure 2).

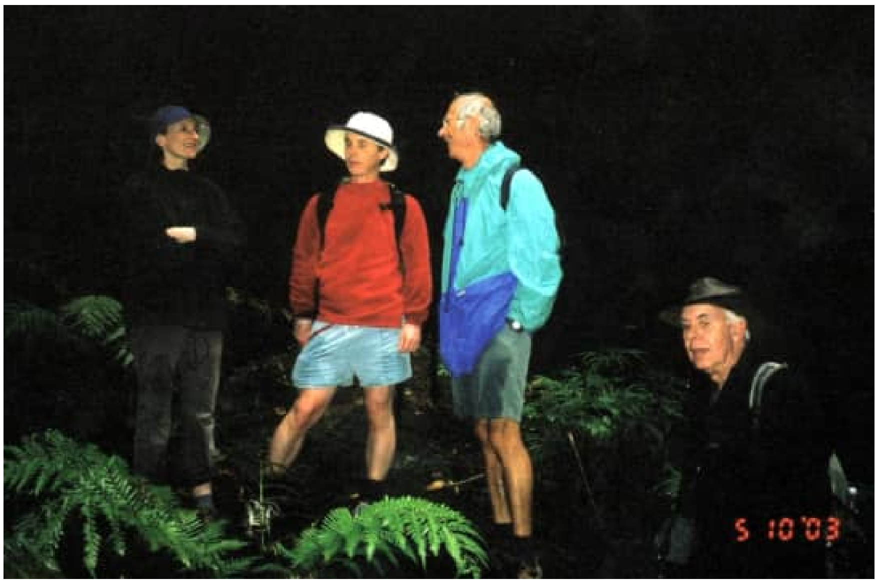


Figure 1: Blue Mountains (New South Wales, Australia), October 2003. From left to right: Annabelle McIver, Carroll Morgan, Jean-Raymond Abrial, Ken Robinson. (Photograph by Thai Son Hoang.)

Figure 2: Blue Mountains (New South Wales, Australia), October 2003. From left to right: Carroll Morgan, Thai Son Hoang, Annabelle McIver, Jean-Raymond Abrial. (Photograph by Ken Robinson.)

I was fortunate that the opportunity to work for Jean-Raymond came much sooner. In November 2004, a position to work on the RODIN project opened up. Without any hesitation, I applied for the position with the knowledge that the chance was quite small, given that I would still need to work on my PhD for a few more months, with the intention of submitting my dissertation in July 2005. To my surprise, Jean-Raymond offered me the job only after a few days and carefully guided me through the official process. The RODIN project opened up many opportunities to collaborate with Jean-Raymond and many other people in the community for me.

I met Jean-Raymond again at the ZB2005 conference in Guildford, UK, in 2005. By then, B# is officially Event-B, the modelling method that has been shaping my work ever since. The talk on "*Refinement and Reachability in Event-B*" [3] (his joint work with Dominique Cansell and Dominique Méry) illustrates many core features of Event-B that we are using today. Jean-Raymond also introduced me to Stefan Hallerstede as "*the guy who will join the RODIN group at ETH Zurich*", which I felt very welcoming.

I finally joined Jean-Raymond, Laurent Voisin, Stefan Hallerstede, and Farhad Mehta at ETH Zurich, just one week after submitting my PhD dissertation. I immediately felt at home with the support of the group for settling in. We are located in a two-storey house, with a basement (called the Pink House due to its distinct colour) on Claussiusstrasse. It was quite luxurious: Jean-Raymond, Farhad, and I on the 1st floor with our own office, while Stefan and Laurent shared a big office on the ground floor. The arrangement was incredibly convenient for us, as we often just come in to talk with each other and discuss without many problems. (20 years on, and things cannot be much more different with hybrid and online meetings). A common pattern was Jean-Raymond calling Stefan and Laurent to come up for our weekly group meeting from the top of the staircase.

The chemistry of the group was fantastic; each of us worked on a different part of the Rodin Platform, with Jean-Raymond as the project manager. Laurent designed and implemented the Rodin Core, Stefan worked on the static checker and proof obligation generator, Farhad was completing his PhD thesis on the proof support, and I worked on the user interface (UI) of the tool. Working on the UI was not an easy task. While Jean-Raymond was critical of my work, he was also incredibly patient in trying out my different implementations of how to input Event-B models and perform proofs.

Attending Rodin project meetings was always a joyful experience. During the workshops, I had the privilege of meeting other (senior and junior) researchers who share the same passion about the formal method, the tool, and their application. What is clearly evidenced during these meetings is Jean-Raymond's constant support for the different generations of researchers. I always look at the photo taken at the 3rd Rodin Workshop in Fontainebleau as an example of that: Jean-Raymond with 3 early-career researchers following his footsteps (Figure 3).

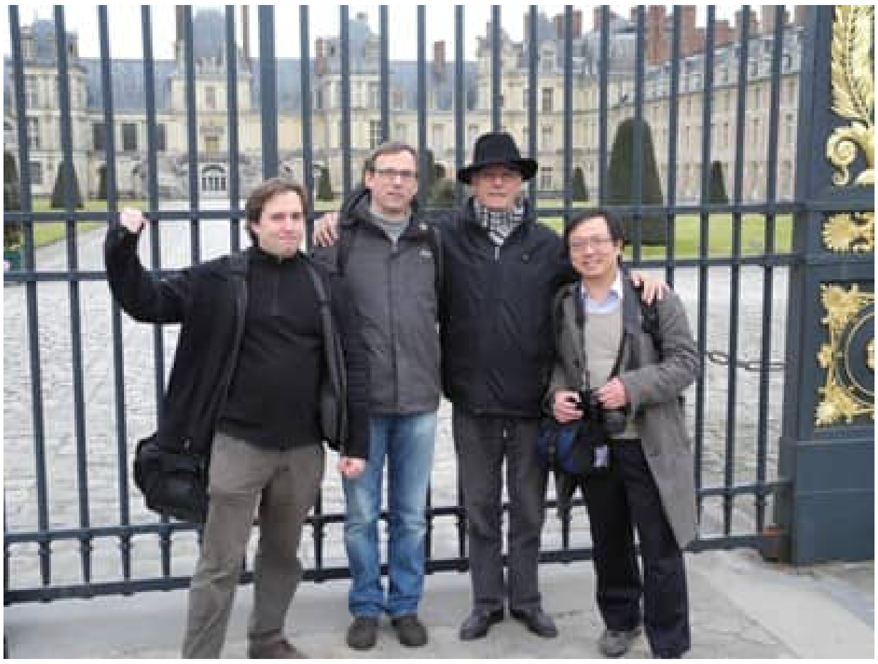


Figure 3: The 3rd Rodin Workshop (Fontainebleau, France), February 2012. From left to right: Neil Evans, Stefan Hallerstede, Jean-Raymond Abrial, Thai Son Hoang. (Photograph by Mike Poppleton.)

After the Rodin project, I worked with Jean-Raymond on the DEPLOY project. We had immense support from Prof. David Basin at ETH Zurich during this time, and were joined by a talented PhD student, Mathias Schmalz. I had one of the most productive periods, with most of my joint publications with Jean-Raymond being during this time. Jean-Raymond's encouragement and enthusiasm for new techniques and tools helped progress my career much further.

Jean-Raymond was always a very active person and often came into the office and talked to me about his weekends. In the summer, it's usually about taking the train and hiking along certain trails. Often, he continued a certain trail for several weekends until he completed it. In the winter, it's usually about skiing and the incredible places that he has been to. Even after his knee surgery, Jean-Raymond invented a new way to continue his passion for skiing, and proud to share that with us. Once, he came into the office and looked a little bit disappointed. Apparently, a lady had stood up to give him a seat on the tram, and he felt it was absolutely unnecessary. At the time, Jean-Raymond was over 70 years old.

After working with the DEPLOY project, I continue to have exchanges with Jean-Raymond. He told me about his three-month around-the-world cruise on a cargo ship, as well as his delight when he was working and receiving a medal in China. He came to Southampton to give a talk on the formalisation of Goodstein theorems (joint work with Dominique Cansell) in 2017. His enthusiasm and energy level were as high as ever, which inspired many of us at the talk. In 2018, Jean-Raymond gave an invited talk at the ABZ2018 conference at Southampton, UK, on "*On B and Event-B: Principles, Success and Challenges*" [2]. Once again, his vision and enthusiasm for formal methods and their applications shone through his talk, and once again, I can see how lucky I am to have the chance to collaborate with Jean-Raymond Abrial.

Not only at the professional level, but Jean-Raymond was also an exceptional friend to our family. He arranged our work to accommodate my bi-weekly travels to Lugano, Switzerland, where my wife studied for her master's degree. My daughter still remembers Jean-Raymond when he visited us in Kingston-upon-Thames on his way to Southampton in 2017. He often talked to us, sent encouragement messages, and was always happy to support our family as much as possible.

Until today, many fond memories of Jean-Raymond are still with me. And I would like to say to Jean-Raymond that his legacy will be with us forever and continue to guide us on our journey.

Thank you, Jean-Raymond!!!

## References

[1] Jean-Raymond Abrial. B$^{\#}$: Toward a synthesis between Z and B. In Didier Bert, Jonathan P. Bowen, Steve King, and Marina Waldén, editors, *ZB 2003: Formal Specification and Development in Z and B, Third International Conference of B and Z Users, Turku, Finland, June 4–6, 2003, Proceedings*, volume 2651 of *Lecture Notes in Computer Science*, pages 168–177. Springer, 2003.

[2] Jean-Raymond Abrial. On B and Event-B: Principles, success and challenges. In Michael J. Butler, Alexander Raschke, Thai Son Hoang, and Klaus Reichl, editors, *Abstract State Machines, Alloy, B, TLA, VDM, and Z – 6th International Conference, ABZ 2018, Southampton, UK, June 5–8, 2018, Proceedings*, volume 10817 of *Lecture Notes in Computer Science*, pages 31–35. Springer, 2018.

[3] Jean-Raymond Abrial, Dominique Cansell, and Dominique Méry. Refinement and reachability in Event-B. In Helen Treharne, Steve King, Martin C. Henson, and Steve A. Schneider, editors, *ZB 2005: Formal Specification and Development in Z and B, 4th International Conference of B and Z Users, Guildford, UK, April 13–15, 2005, Proceedings*, volume 3455 of *Lecture Notes in Computer Science*, pages 222–241. Springer, 2005.

# Jean-Raymond: a top-flight friend

Cliff Jones, with Joanne Allison

My thoughts on Jean-Raymond's contributions as a computer scientist will appear in "Formal Aspects of Computing" (probably in January), here we share a few anecdotes in the hope they allow those who knew him only as a scientist to appreciate a little of a warm and interesting man.

Jean-Raymond was top-flight not only as a scientist but in daily life as well: he insisted on a top floor apartment wherever he lived. When he was heading up the work on Event-B and the Rodin Toolset, I (Cliff) visited him and his team at ETH Zurich many times. Meeting in his apartment before dinner involved a climb up many flights of stairs (who on earth would want a lift?). Unfortunately, during his years as an avid skier, hiker and climber Jean-Raymond had picked up a serious knee problem which required surgery and a recovery period on crutches. Even though he had moved to a slightly shorter building, I still wondered how he would get to his (still top floor of course) apartment. I needn't have worried; he ascended the stairs on crutches at a speed that I could not match!

A move back to Marseille saw Jean-Raymond in another top-floor apartment. There, he suffered a few headaches but nothing he was too concerned about. The apartment was due an update though and he had the draughty widows replaced with double-glazed units that had a decent seal. I had arranged to phone Jean-Raymond at home one Sunday but got no reply and was surprised – even slightly concerned. Early that morning, we had been woken by the carbon monoxide alarm at our home in Northumberland: it was just a low battery alert but turned out to be one of the most bizarre cases of "synchronicity" in my life. A few days later, I did get to speak to Jean-Raymond and heard the story of how he had passed out that Sunday morning. His headaches had been caused by carbon monoxide but his old leaky windows had kept the concentration relatively low. His preference for top-floor living and an alert downstairs neighbour who heard his fall probably saved his life.

Jean-Raymond was both a walker and a talker. During his time in Zurich, he walked to Santiago de Compostela, covering a stage at a time then returning to Zurich by train, using the train later to return and pick up where he left off. He also trekked more dangerously in the desert and told wonderful stories of his experiences during these walks. We both had dinner at Da Bruno in Pisa during a conference and for Joanne especially it just wouldn't have been the same without Jean-Raymond's impeccable impression of the camels that had carried supplies for the desert trekkers. Always a charming and interesting companion over a meal; he is sorely missed by those who knew him.

# Thanks to Jean Raymond Abrial

Christian Jullien

I didn't work directly with Professor Abrial, but I owe him much!

## A brilliant and dedicated teacher

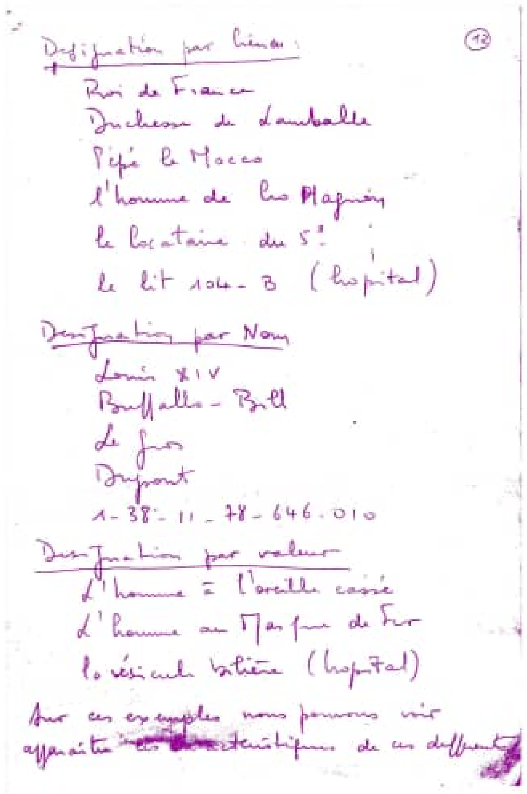
Firstly, he was my teacher (for two years) during my computer science master's degree at IMAG Grenoble (1971/72). In 1971, he gave us a strong basis in data management. The classes were fascinating; the lecture hall was truly captivated; the examples were "telling", despite being based on a rigorous theoretical foundation.

I still have hand-written lecture notes for additional courses that he gave for free at the university in the evening, after normal hours.

This document (215 pages) was produced with an alcohol duplicator. He used a lot of figures, graphs, and examples that related to our comics at the time.

With these simple and concrete examples, he nevertheless presented essential principles of computer science from the sole perspective of data, its representation, and its processing, all within a rigorous and original approach. Everyone could understand because the explanation was so clear.

It was very pragmatic and exciting.



12

Désignation par liens :

Roi de France
Duchesse de Lamballe
Pépé le Mocco
l'homme de la [illegible]
le locataire du 5e
le lit 104-B (hôpital)

Désignation par Nom

Louis XIV
Buffallo-Bill
Le [illegible]
Dupont
1-38-11-78-646.010

Désignation par valeur

l'homme à l'oreille cassée
l'homme au Masque de Fer
la vésicule biliaire (hôpital)

Sur ces exemples nous pouvons voir apparaître les caractéristiques de ces différents

Data structures, page 12:
Naming data

IMAG Labs where J.-R. Abrial was our professor and involved in SOCRATE project. At that time, he had a magnificent Peugeot 403 coupe (*like Colombo's!*).

In 1972, the course was mainly on the theory of automata and computability (much more formal – see the halting problem!), and reflected the particular focus that J.-R. Abrial was putting on calculability and proofs of programs.

## SOCRATE

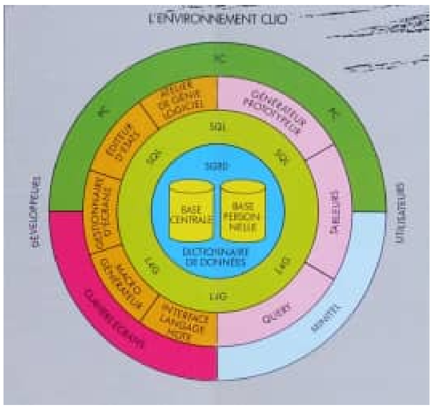
In 1968, J.-R. Abrial had been commissioned by IMAG to lead a major research project on the development of management applications. The topic was data processing. This project, called SOCRATE, gave birth to a software product (a DBMS – database management system) in 1970, with the cooperation of a few local organizations (hospital, social insurance). Industrialization followed at the CII company (which later became BULL), and by ECA Automation, a software house whose director, Pierre Thellier, had known J.-R. Abrial at the CPM (Naval Programming Center). The university team had largely joined ECA Automation for the first industrial project for Social Security in Grenoble. Among them, George Beaume and George Vigliano, 2 of the 3 main PhD students. At ECA Automation, the team was directed by S. Stepanian, an engineer who had also been at the CPM.

A few years later (1976), after some research activities and teaching in software engineering, I was able to join the team involved in SOCRATE. I will not report the whole story here, but in the first years, J.-R. Abrial went regularly visiting the team directed by Stephen Stepanian, Georges Vigliano, and Georges Beaume. With them, and for more than 15 years, I could study, develop, and put in software applications with the basic notions resulting from the work of J.-R. Abrial.

Basically, SOCRATE was a revolution: a practical way for high productivity in programming, and much more than a DBMS. Indeed, since 1970, Socrate has offered a 4GL language that provides management of persistent data in a manner that could free programmers from physical management considerations.

This software has been used for more than 30 years for a large number of applications in different areas (industry, utilities, defense, bank and insurance, . . . ) on various platforms (IBM, SIEMENS, minicomputers like SOLAR, MITRA, . . . , VAX, PCs, . . . ). Renamed CLIO – *due to a trademark dispute* – and anglicized, SOCRATE was sold in 1985 in the United States, to two customers for use until around the year 2000. Contacts were also made in Singapore, but due to a lack of commercial resources, sales were mainly domestic.

SYSTÈME DE GESTION DE BASES DE DONNÉES
CLIO
SYSECA
Télémécanique Informatique
SOCRATE



Moreover, notions coming with SOCRATE from the beginning (i.e., collections of data like tables, trees, lists, ...) have been easily mapped to the relational approach in 1992. An SQL engine has been integrated, conformant to the ISO 92 standards. It was not until 1988 that ORACLE brought PL-SQL in order to offer a comparable 4GL-type programming language!

Concepts identified by JRA in 1968 were really ahead of their time, and moreover, an important source of value creation. We owe a great deal to this highly creative genius.

Christian Jullien, Member of Aconit (**www.aconit.org**)
IMAG, Student of J.-R. Abrial (1970–1972)
SYSECA, DBMS Development Engineer (1976–1981)
CNET Labs, DBMS Research Engineer (1982–1989)
SYSECA, DBMS Project manager (1989–1993)
SYSECA, Head of unit Grenoble (1993–1998)
HARDIS Group, Senior IT consultant (1999–2008)

# Guy Laffitte and J.-R. Abrial

Guy Laffitte, interviewed by Henri Habrias

## Introduction

We provide an interview of Guy Laffitte by Henri Habrias before the JS2025 conference in Nantes, France.

## 26 Rue des Plantes

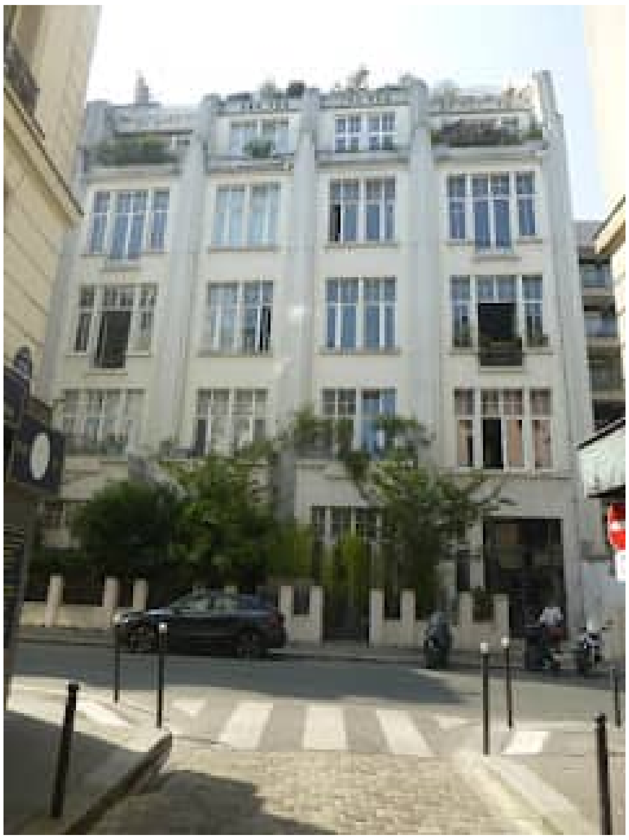


Figure 1: 6 rue des Plantes, Paris, France: Jean-Raymond Abrial's address as a consultant.

The wind was favourable. The BN (Biscuiterie Nantaise) factory gave us the good smell of biscuits. And Guy answered our questions while the grandson was at French football training.

- H.H.: You met J.-R. Abrial when you took his course at CEPIA on the Domaine de Voluceau in Rocquencourt. What year was it?
- G.L.: In October 1985.

- H.H.: I was there in October and spring as well. We could have met there. But we didn't know each other yet. What did you like then?
- G.L.: Strong point: instead of trying in vain to make a proof on any program, we start by making a specification that we can prove, and then we are only interested in programs implementing the specification.
- H.H.: J.-R. Abrial thanks you in the B-Book for your contribution.[1] You tell me.
- G.L.: I was working at INSEE boulevard Adolphe-Pinard in Paris in the $14^e$ and it wasn't far from home. He lived at the address that appeared in his articles, Independent Consultant, 26 rue des Plantes, an Art Deco style building, composed of duplex artists' studios with the studio in the lower part, bedroom and kitchen, living room on the mezzanine, with double-height windows. Max Ernst lived there.
- H.H.: Marx Ernst is the one who illustrated my "Logic Without Pain" (1966 edition) by Lewis Carroll, published by Hermann, the publisher of the first Bourbaki. You bring back memories. If my memory serves me correctly, there is a plaque in memory of Jean Moulin who rented a studio there.
- G.L.: Yes. I could walk from INSEE to the rue des plantes, so we were able to talk.

## Contributions of Guy Laffitte

- H.H.: What were your contributions?
- G.L.: Theoretical contributions.
  - Requirement of proof of initialization.
  - Loop Introduction on pp. 377 and 378 of the B-Book. Initial (dual) version of Th. 9.2.1, p. 379 with unions instead of intersections: drastic simplification of the assumptions.
  - B-Tool.
    - Improved the suite machine.
    - Expertise of non-plagiarism between the initial version (belonging to BP) and the new version (STERIA at the time).
    - Parsing a new version of the parser, an LR-inspired stack-based automaton replacing a recursive descent. Subtleties: generation of saturated reverse Polish and insertion of implicit operators (substitution and evaluation).

[1] G. Laffitte influenced this work by his careful reviews, his accurate criticisms, and the sometimes very serious rearrangements he proposed for some of the mathematical developments of this book.

## The General Population Census in 1990

- H.H.: And your application of B to the population census. You had made a presentation to the students. I remember the study of administrative geography. Can you tell me more?
- G.L.: Okay.
  - Tools for the 1990 Census of Population. [2]
  - Systematic use of the B-Tool for the parsing of tools and the code generation (Pascal).
  - TRANS xx. Data storage tool using virtual persistent memories and of sequences machines based on an initial idea by J.-R. Abrial.
  - Databases, systems, and applications concepts.
  - Concept of mirror database.
  - Development on Unix, laptop on IBM OS and DPS7.

[2]Bernard, P., Laffitte, G. (1995). *The French population census for 1990*. In: Bowen, J.P., Hinchey, M.G. (eds), ZUM '95: The Z Formal Specification Notation. ZUM 1995. Lecture Notes in Computer Science, vol 967. Springer, Berlin, Heidelberg

# Validating Atelier B Proof Capabilities

Thierry Lecomte
CLEARSY, Aix en Provence, France
thierry.lecomte@clearsy.com

Abstract

This article provides an overview of the work carried out in the 1990s to develop and validate the proof functions of Atelier B, with a focus on how Jean-Raymond Abrial contributed to this topic. These functions are at the heart of the B-Method. Even if they have not been formally developed, it is necessary to implement validation means adapted to safety issues.

## A Dedication

The author collaborated with Jean Raymond Abrial during the development of Atelier B in the 1990s and over several research projects related to B and Event-B. The numerous interactions were mainly linked to the development and validation of the Atelier B proof tools, but also to the modelling and tool support for Event-B.

Figure 1: Scientific discussion between Jean Raymond Abrial and the author, in Tokyo during the Event-B Day in 2014 (https://research.nii.ac.jp/eventb2014/)

## B Introduction

Mathematical proof lies at the core of the B-Method [1]. This constitutes both a major advantage, since it enables a level of software validation that surpasses conventional testing, and a limitation, as it demands specialized expertise [9] that is often scarce within industrial teams. With the industrialization of Atelier B in the 1990s, to be ready for the opening of the Météor Line 14 metro in Paris [3], means of verification acceptable to standard EN50128 had to be developed and implemented to ensure that there could be reasonable confidence in the validity of the automatic and interactive demonstrations obtained. This paper highlights the key elements of this development and validation activity, undertaken within an industrial framework. It does not include comparative scientific studies with other provers or similar tools, as the technical and scientific directions were largely predetermined by initial constraints.

The remainder of this paper is organized as follows. Section C presents the characteristics and constraints associated to the B proof obligations. Section refps addresses the proof system, its architecture, and its requirements, before concluding. For each section, the contribution from and interaction with Jean-Raymond Abrial is emphasized.

## C Atelier B Proof Obligations

Proof obligations (PO) are central to the B verification schema[8]. They are computed from B models (Fig. 2) to cover several aspects: correctness, overflow, and well-definedness.

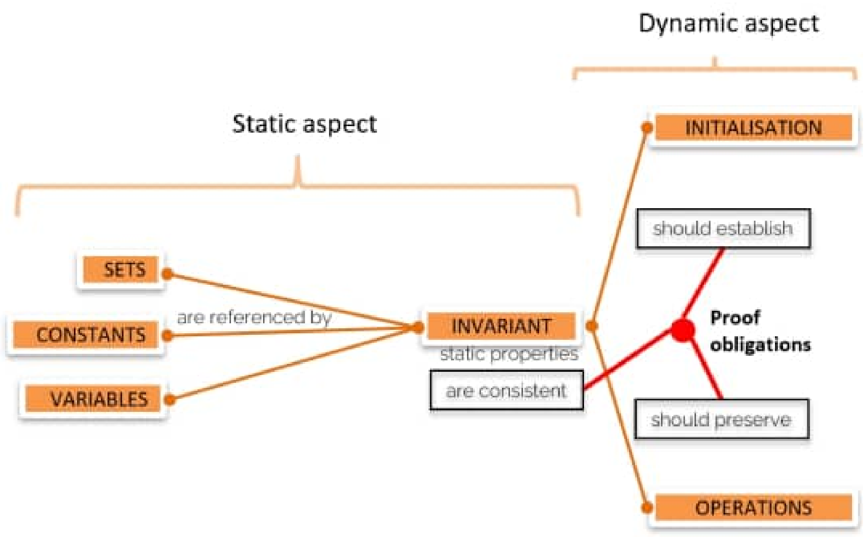


Figure 2: Proof obligations are linked with model clauses.

The first proof obligation generator developed in the 1990s by Alstom [5] contained many hard-coded optimizations, which made it difficult to understand how it worked, to optimize the generator, and to modify the B language. Atelier B historically also used custom data formats and specific programming languages. These formats have lasted for several decades because redesigning the formats and the programs would have been too costly. Specific programming languages (like an in-house Prolog-like language) have made it more difficult to convince engineers to work on this software.

In any case, once the first industrial project had been completed, Atelier B was obliged to keep the proof obligation process constant to avoid any proof regression. Any modification in the proof obligations shape, naming, or order would have had a dramatic impact on the effort (i.e. cost) required to maintain B projects proved over the years. The proof obligations are named after the B clauses to which they apply. Mathematical demonstrations are associated with the proof obligations of a clause. Heuristics allow us to continue to associate these demonstrations correctly with the proof obligations, when refactoring the B model for which we do not want to lose the interactive proof work already carried out.

**Contributions:** Jean-Raymond Abrial was involved in the process of designing and experimenting with proof obligation generation, including the normalization process (predicates are normalized to reduce the number of mathematical rules required) and automatic removal of obvious proof obligations (to reduce the burden of the ongoing proof process). The generation process was very sensitive to any modification to the mathematical rules, but also to the tactic used to select and execute the rules package.

## D Atelier B Proof System

When Atelier B was built, there was no prover capable of handling proof obligations that mixed set theory and arithmetic, and that could contain thousands or even tens of thousands of hypotheses. In addition, the processing time had to be a maximum of ten seconds on average for each proof obligation. Alstom initially developed a proof kernel limited to automatic proof, requiring to be extended. This section presents the proof system that was developed for this purpose.

### D.1 Automatic Proof

The Automatic Prover PR is made of two distinct parts: a loader and a solver.

The **loader** is designed to minimise PO loading/unloading in memory: PO could have many hypotheses, so unloading all hypotheses from memory when moving to the next PO is not optimal. The hypotheses are grouped into packets, corresponding to the different clauses of a B model. PR unloads from the computer memory only those hypotheses that are no longer used and keeps the others. The PO file format was structured accordingly.

The **solver** generates new hypotheses and transforms the goal in order to obtain $\top$ (represented with the predefined symbol *btrue*). If the solver is successful, the proof obligation is considered proved. If not, the proof obligation remains unproved. A B project is valid only when 100% PO are proved. The solver (Fig. 3) is itself composed of a hypothesis processor, generating new hypotheses, and of a goal processor, simplifying the current goal by adding local hypotheses derived from existing ones, or by replacing the current goal with one or several sub-goals. The two processors are executed in sequence: the goal is simplified while new hypotheses are generated. The loop stops when the goal cannot be transformed anymore.

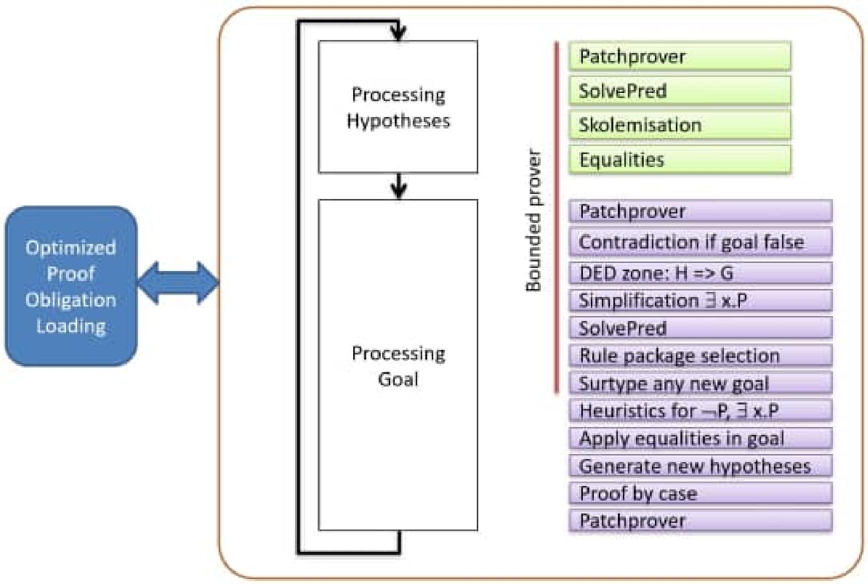


Figure 3: Structure of PR.

**Proof mechanisms:** The solver, actionable through the parameter *force*, ranging from 0 to 3, contains 35 mechanisms[1]. They were designed, improved then selected by engineers from different companies, based on experiments on several B projects representing a total of 3,000 POs. The selection was made based on both the automatic proof percentage and the complexity/difficulty of proving the remaining POs. There is another version of the Automatic Prover called MonoLemma ML. It comes without the optimized loader to demonstrate a single proof obligation. ML is used to power the Rodin platform as “Atelier B provers” plug-in.[2]

## D.2 Extending the Prover

Three kinds of rules can be added to Atelier B:

- backward rules transforming a goal into 0 or more sub-goals,
- rewriting rules working on expressions and predicates,
- forward rules generating new hypotheses, triggered when the Modus Ponens is activated.

Apart from the Atelier B rules database, user-defined rules may be added at the component or at the project level (PatchProver). The PatchProver is central to the proof performance: it has to be set

[1]The structure of the prover, the mathematical rules database, the mechanisms constituting both bounded and unbounded provers, and their parameters were designed, defined, and optimised based on the B models developed by the company Matra from 1993 until 1998. Hence, the proof performances were not guaranteed when the symbols and predicates used were quite different from those in Meteor models. To improve the automatic proof percentage, complementary mathematical rule databases need to be designed to address specific symbols or patterns.

[2]`https://wiki.event-b.org/index.php/Rodin_Platform_Releases`

up gradually based on previous completed projects and managed with a source-code management system. It constitutes the palette that each user has to know well. The addition of user rules in proof component files is only permitted if no (combination of) rules is available in the PatchProver. All rules have to be validated before being used in a project, at least with a manual demonstration.

**Proof-oriented script language:** It contains more than 30 commands allowing modification of the proof tree or the collection of information about it. These commands are able to activate the bounded and unbounded provers, the processing of hypotheses for any force. They are also able to add a hypothesis, initiate a proof by cases, trigger the Modus Ponens, start the Predicate Prover, or simplify a goal with the Set Solver, and execute an external prover. To avoid long waiting times, some commands are associated with deadlines – the execution of these commands will not last longer than this delay, which is a parameter of the command. Similarly, for provers that work better with a limited number of proof obligations, it is possible to specify that only typing hypotheses or hypotheses having a symbol in common with the goal are selected for the proof.

**Definition of proof tactics and their selective execution:** Proof-oriented commands can be saved as a demonstration. These are the demonstrations that are replayed often to ensure that the project is still provable. These demonstrations are supposed to be as generic as possible, to be applied to other similar proof obligations. In the interactive prover, when at least one PO has been demonstrated, the user is asked if these demonstrations could be saved in the User Pass, a specific part of the component proof files that only contains proof tactics (generic demonstrations). These demonstrations can be tried on the remaining proof obligations of the current component. These tactics are ordered: the first one is tried on all unproved POs, the other ones on the remaining POs once the previous tactic has been applied. Naturally, the most generic, efficient tactics have to be in the first places, while the more specific and longest to execute have to be positioned further down the list. In addition, it is possible to use filters to make the selection even more precise by specifying the name of the operation and/or the form of the goal.

### D.3 Validation

From 1993 to 1998, the Automatic Prover was reverse-engineered, documented, tested, and extended to be ready for RATP qualification. It was made available to the Community and presented at the occasion of B-User Group meetings. The mathematical rules database was published to enable academic research [4]. Proof mechanisms were validated in conformance with software development best practices: documentation peer review, source code inspection, and requirement-based test bench. Mathematical rules database (2,700+ rules) validation was decomposed into two activities: automatic proof with the Predicate Prover PP, and manual proof for the rules not proved or not handled by PP.

**Automatic proof:** The Predicate Prover PP was developed specifically to automate this error-prone activity. PP [2] is based on a semantic tableau proof system. The 116 inference rules are described by conditional rewrite rules, and their application is controlled by strategies (case-split, instantiation of quantified predicates, etc.). The specification of PP was made public to encourage formal research [7][6]. Its specification was also peer reviewed, and traceability with source code was established. Its validation consisted in developing an automaton able to replay the demonstration[3] to obtain a contradiction. Then, PP has been included in a professional tool, including checkers and a translator, to specifically support industrial projects. PP is efficient when the number of hypotheses is low, which is the case for mathematical rule validation. The use of PP for proof of OP requires hypothesis filtering to maintain its proof performance.

**Manual proof:** The mathematical rules that are not handled or proved by PP had to be verified manually. A form had to be filled with the source code of the rule, its manual translation into a mathematical predicate, the verification of its correct typing, and the absence of name collision between identifiers and quantified variables. Then a mathematical demonstration with no imposed format had to be provided. Each form was reviewed by a panel of about ten people from five different companies. The remarks made had to be taken into account by the author of the form, and then the rule had to be reviewed again, as long as the remarks remained. Finally, a third-party company was in charge of reviewing all forms and evaluating whether the demonstrations were convincing. The validation was completed when all manual demonstrations were deemed to be convincing. The forms and the evaluation report were added to the validation file.

The validation of the rules was a long process, which lasted much longer than the five years of Atelier B Meteor's development. Some errors were detected after 1998. Where possible, the rule was corrected, and industry users were invited to re-prove their projects. Most of the time, the rules were "almost correct". Often, a restriction on the application domain was missing. It was also detected that some rules were duplicated: as the validation of rules is unitary, this fact remained undetected for some time. Circumventing the problem required replacing the duplicated rule with a non-applicable rule.[4]

During the development of PP, a first attempt to verify the rules database occurred in 1995. The tool handled 1,535 rules; the other rules were not processed because of functional limitations (sequences not supported, several guards not yet translated). 59 were detected as false, 55 were corrected and verified by PP successfully, and 4 were removed because they were wrong (or too specific). A dedicated proof tool with a GUI was designed and integrated into Atelier B to provide a systematic verification framework. Finally, a few tens of rules have been corrected since 1998. Their detection is often due to incorrect models being proved successfully. Backward analysis allows to locate the origin of the error, if it is a rule. If the cause is a faulty mechanism, then the analysis is our responsibility, as the code of the mechanisms is not accessible through the tool, unlike the mathematical rules.

[3]A sequence of inference rules, with the instantiation value of related quantified variables if any.

[4]It was not possible to delete it as it would have modified the order (the name) of the rules, which can be called by their index during interactive proof.

**Contributions:** Jean-Raymond Abrial was heavily involved in the database validation process:

- as a mathematician in charge of checking mathematical rules and reviewing the rule demonstrations from other members of the validation team.
- as a developer of the Predicate Prover PP tool.[5] PP was used for the validation of the Atelier B mathematical rules. PP is also integrated into Atelier B interactive prover as an actionable proof command, and into the OPR tool to validate mathematical rules added to a B project by users.

## E Conclusion and Perspectives

This article briefly presents the Atelier B proof system, as developed for the commissioning of Météor in December 1998. This was the first industrial experiment of its kind. This project required the interactive participation of several companies and laboratories in order to reach a consensus on the validity of the proof process, which was not formally developed. Jean-Raymond Abrial's involvement was decisive in terms of technical choices and scientific orientations. After two decades of use, a few errors have been detected so far in the proof rules and the predicate transformation mechanisms, but their impact has been reduced since no false proof obligation linked to safety has been wrongly demonstrated because of these errors. This is the most important lesson of this industrial story.

## References

[1] Jean-Raymond Abrial. *The B-Book – Assigning Programs to Meanings*. Cambridge University Press, 2005.

[2] Jean-Raymond Abrial and Dominique Cansell. Click'n Prove: Interactive proofs within set theory. In *Theorem Proving in Higher Order Logics*, volume 2758 of *LNCS*, pages 1–24. Springer, 2003.

[3] Patrick Behm, Paul Benoit, Alain Faivre, and Jean-Marc Meynadier. Météor: A successful application of B in a large project. In *FM'99 – Formal Methods*, volume 1708 of *LNCS*, pages 369–387. Springer, 1999.

[4] Karim Berkani, Catherine Dubois, Alain Faivre, and Jérôme Falampin. Validation des règles de base de l'Atelier B. *Technique et Science Informatiques*, 23:855–878, July 2004.

[5] Michael Butler, Philipp Körner, Sebastian Krings, Thierry Lecomte, Michael Leuschel, Luis-Fernando Mejia, and Laurent Voisin. The first twenty-five years of industrial use of the B-Method. In Maurice H. ter Beek and Dejan Ničković, editors, *Formal Methods for Industrial Critical Systems*, volume 12327 of *LNCS*, pages 189–209. Springer, 2020.

[5] He developed a number of other proof tools like an inductive prover or was at the origin of the Click'n'Prove interactive proof interface [2]

[6] Horatiu Cirstea and Claude Kirchner. Using rewriting and strategies for describing the B predicate prover. In *Proceedings of the Workshop on Strategies in Automated Deduction – CADE-15*, pages 23–34, Lindau, Germany, 1998.
**https://inria.hal.science/inria-00098715v1/document**.

[7] Horatiu Cirstea and Claude Kirchner. Theorem proving using computational systems: The case of the B predicate prover. ResearchGate, April 2000.
**https://www.researchgate.net/publication/2628953**.

[8] CLEARSY. *Proof Obligations Reference Manual 3.7*.

[9] Thierry Lecomte. Proving B with Atelier B. In Simon Foster and Augusto Sampaio, editors, *The Application of Formal Methods: Essays Dedicated to Jim Woodcock on the Occasion of His Retirement*, volume 14900 of *LNCS*, pages 329–345. Springer, 2024.

# Memories of Jean-Raymond Abrial in Oxford, the Alps, and Paris

Bernard Sufrin
Emeritus Fellow: Worcester College & Department of Computer Science
Oxford University

"Formalization is not an isolated activity, nor one that is finished once and for all. Implicitly, at least, it is a continuing aspect of every (good) designer's creative work, whether it is performed by a single individual, or what is often more fruitful, by a larger group, or even an entire community. In such situations, however, the (intermediate) results of this process should be given some form of concrete expression, whereupon they may be repeatedly examined, criticized, compared, and recast. It is in this context that the need for formal specification really arises. The notation employed must therefore be construed, above all, as a medium of communication." [1]

## Introduction

I collaborated very closely with Jean-Raymond while he was visiting the PRG[1] in Oxford from Autumn 1979 to mid 1981; and we remained good friends for many years afterwards. In this note I have interleaved a few personal and technical reminiscences of our friendship and collaboration.

My research brief from Tony Hoare when I arrived at the PRG in the autumn of 1978 was to explore ways of publishing the source texts of high-quality software. Academics in the then-tiny group had a lot to do that year: developing and teaching graduate courses, brokering collaborative projects with other parts of the University, *etc.* [2]. I spent some of my remaining time writing and re-writing a document compiler and a text editor[2] in a narrative style. In retrospect these attempts at "code as communication" would be called literate programs, but Knuth had yet to popularise the practice or the tools to support it.

I ended this year feeling uneasy about publishing these programs: they worked perfectly well, but were they of sufficiently high quality to publish? The design principles behind the editor's user

[1] The Programming Research Group of the Computing Laboratory. Until the 1990s this was the *de-facto* Computer Science Department of Oxford University.

[2] Why bother? Please remember that this all went on long before the days of `emacs`, and `word`; TeX was still over the western horizon, and computing facilities in the PRG were still somewhat primitive.

interface (modelessness and undoability) were lost in the details of its implementation. Something was certainly missing … *but I didn't know what.* Then Jean-Raymond arrived, and I began to understand what was wrong.

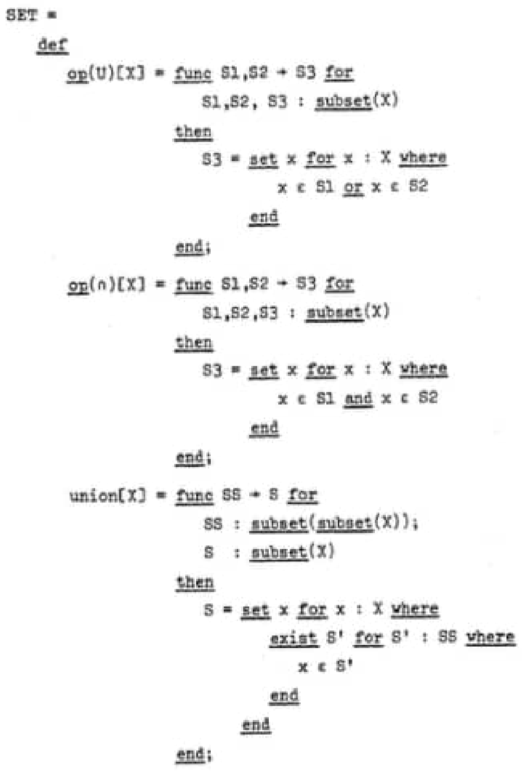
*Finding reasonably priced accommodation for rent in Oxford was then – as it is now – notoriously difficult, but Jean-Raymond and his partner Hélène Villers had found a splendid house on Osney Island in Oxford a yard or so from the river Thames at the corner of South and East Streets. It had a studio-cum-drawing-office for Hélène to work in as she made the transition from academic sociologist to fine artist. I still treasure a few of her early works. The house had a garage that could fit two narrow saloon cars or two dozen bicycles, but J-R – a keen Alpinist – had brought his very large white "expedition van" and it had to be parked in the street outside. Fine until the floods hit Osney.*

## Specifications

Jean-Raymond's first course at Oxford was called "Program Specification". It was about using the language of sets, relations, and functions, to model software systems. It was different from any mathematics course I'd taken as an undergraduate: impressive because every new idea and notation he introduced was immediately illustrated by a couple of serious examples of its application to the description of a situation or system that meant something to the programmers present[a].

The flavour of his "Bourbakiesque" notation and approach, and some of his examples, can be seen at right and in Figures 1–3 – all extracts from the paper that introduced his specification language, then only occasionally called Z [3].

[a]All technical material in this note is either exactly transcribed from papers that sat in my personal archive for many years before being scanned, or a screenshot of part of such a scan. The scans are available on my Oxford website.



```
SET =
  def
    op(∪)[X] = func S1,S2 → S3 for
                   S1,S2, S3 : subset(X)
               then
                   S3 = set x for x : X where
                          x ∈ S1 or x ∈ S2
                        end
               end;
    op(∩)[X] = func S1,S2 → S3 for
                   S1,S2,S3 : subset(X)
               then
                   S3 = set x for x : X where
                          x ∈ S1 and x ∈ S2
                        end
               end;
    union[X] = func SS → S for
                   SS : subset(subset(X));
                   S  : subset(X)
               then
                   S = set x for x : X where
                         exist S' for S' : SS where
                           x ∈ S'
                         end
                       end
               end;
```

Image of an extract from the `SET` chapter of "Specification Language"

There was much emphasis on the modular structuring of specifications: smaller theories, whether intended to be of general use or application-specific were organised as *chapters*, with inter-chapter dependencies made explicit.

The definitions of Figures 1 and 2 are typical of Jean-Raymond's adoption of the extensional approach to relations and functions in his constructions.

What might be surprising from a contemporary point of view, and in view of Jean-Raymond's later work, was the absence of any axioms or rules of reasoning in the "Specification Language" paper. That is not to say that there were no proofs in the examples: just that the commonsense (!) laws of set theory and its derived theories were taken for granted.

The *function* projection of a functional relation in Figure 1 is evidently a pedagogical detail made necessary by the commonsense approach. It precisely defines the domain of the "derived function" of a functional relation, and the meaning of its application to an element of its domain: namely, the uniquely related element in its codomain. Also missing is an explicit definition of the image set $R(x)$ of a singleton $x$ through a relation $R$.

```
REL =
use SET def
   inv[X,X']     = func r → r' for r : X ↔ X'; r': X' ↔ X then
                     r' = rel x' ↔ x for x' : X'; x : X where
                            r(x ↔ x')
                          end
                   end

   op(∘)[X,Y,Z] = func r2,r1 → r3 for r1 : X ↔ Y; r2 : Y ↔ Z; r3 : X ↔ Z then
                     r3 = rel x ↔ z for x : X; z : Z where
                            exist y for y : Y where
                               r1(x ↔ y); r2(y ↔ z)
                            end
                          end
                   end
  ...

   functional[X,Y] = set r for r : X ↔ Y where
                         r'(Y) = X; (r ∘ r') ⊂ ident[Y]
                     given
                         r' = inv(r)
                     end;

   function[X,Y]   = func r → f for r : X ↔ Y; f : X → Y where
                        r ∈ functional[X,Y]
                     then
                        f = func x → y for x : X; y: Y then
                               y = any(R(x))
                            end
                     end
```

Figure 1: An extract from the REL chapter

```
SEQ =
 use SET, NAT, REL def
   segment                   = func n → S for n : NAT; S: subset(NAT) then
                                  S = set i for i : NAT where i<n end
                               end;
   ...
   seq[X]                    = set s for s : NAT ⇸ X where
                                  dom(s) ∈ segment(NAT)
                               end;

   length[X]                 = ...

   op(⋆)[X]                  = func s1,s2 → s3 for s1,s2,s3 : seq[X] then
                                  s3 = s1 ∪ (s2 ∘ f)
                                given
                                  f = iter(inv(succ))(length(s1))
                                end;
   ...
   associated_rel[X]         = func s → r for
                                  s : seq[X];
                                  r : X ↔ X
                                then
                                  r = s ∘ succ ∘ inv(S)
                                end
```

The *associated relation* of a sequence is what we would now call its adjacency relation: it relates elements that are directly adjacent in the sequence.

The description of the result of sequence catenation ($*$) as the union (qua relation) of $s1$ and "$s2$ right shifted by the length of $s1$" is also illustrative of the extensional approach. The proof that the resulting relation is a function whose domain is a `segment` is straightforward.

Figure 2: An extract from the `SEQ` chapter

For his first "realistic" example[3], Jean-Raymond shows the specification and meticulous incremental development of an "algorithmically-structured" function, then its transliteration into Pascal. It replaces multiple adjacent blanks in its input by a single blank.

The use of classes and inheritance in the specification is interesting. Here `spec1` is the class of states whose output has no two consecutive blanks; and `spec2` is the subclass of that class in which the output and input are equivalent modulo (nonempty) sequences of space.

The details of the subsequent development from specification to program are of interest, but constrained space precludes longer discussion here, save to observe that there is no "end to end" specification that captures the *requirement* that the final `out` is the "shortest `equivalent_string` to the (initial) `in`." To remedy this readably would almost certainly give rise to a refactoring of the material of this chapter.

[3]His scare-quotes!

```
EDITING_PROBLEM =
  use RELATIONS, MINI_FIXED_POINT_THEORY def
     state[C]        = class
                         b : C;
                         in, out : seq[C]
                       end;

     spec1[C]        = subclass state[C] where
                         out ∈ no_two_consecutive_blanks
                       given
                         no_two_consecutive_blanks =
                            set s for s : seq[C] where
                              not(associated_rel(s)(b ↔ b))
                            end
                       end;

     spec2[C]        = subclass spec1[C] where
                         equivalent_string(in ↔ out)
                       given
                         equivalent_string = closure(r);
                         r = rel s1↔ s2 for s1, s2: seq[C]
                             where
                               exist x,y,b1,b2 for
                                 x,y : seq[C]
                                 b1,b2 : seq[{b}] - null
                               where
                                 s1 = x*b1*y
                                 s2 = x*b2*y
                               end
                             end
                      end
```

Figure 3: An extract from the EDITING_PROBLEM chapter

## Towards “Oxford Z”

Early the following year Jean-Raymond and I, with our new postdoc (Tim Clement), and Ib Sørensen (a DPhil student who had responded very enthusiastically to the *Program Specification* course), began work on the CAVIAR[4] project. It had been generously funded by **S**tandard **T**elecommunications **L**aboratories.[5] Its goal was to evaluate the effectiveness of formal specification techniques in communicating ideas about “realistic systems” by specifying a system to be used by the company to manage room scheduling and resource provision for the (very many) visitors to its buildings.

The four of us spent the next few months refining the existing specification notation and thinking about CAVIAR with it. We’d meet every afternoon in the smoke-filled office occupied by Jean-Raymond, Tim, and Ib. It was a heady time: when we weren’t changing the specification [of CAVIAR] we’d change the language. Jean-Raymond had the big language ideas and wrote them

[4]Computer Assisted Visitor Information And Resources

[5]Enthusiastically advocated within STL by Bernard Cohen and Tim Denvir.

up frighteningly fast, but he wasn't dogmatic, and he listened to everybody's ideas. Ib Sørensen took responsibility for writing up the CAVIAR specification itself [5], and Jean-Raymond and I experimented with using the notation for other applications, and started to prepare a graduate course on specification that we would give jointly at the start of the next academic year.[6]

What emerged from our deliberations was the "second generation Z" [1, 4]: still rooted in a generically-typed set theory and with "basic library" still presented in the "layered" style of Bourbaki.

By way of illustration, I have shown the whole of the `SET` chapter in Figures 4 and 5. In Figures 7 and 6, I have also shown a handwritten note by Jean-Raymond experimenting with one incremental style of presentation and the opening chapter of my own attempt at an editor specification.

What has changed is that chapters themselves are generic, functions and relations are now given explicit (type-) signatures separated from their definitions, and intended or proven properties of the declared objects are given.[7] One could also present an application-oriented theory as a signature together with intended properties whilst deferring the constructive definitions.

We sent our working papers to Dana Scott, an old friend of the group, and a few other prominent computer scientists. Dana's kindly reply deflated us somewhat. He liked our goals, but not the long windedness of the notation: "… and when it takes you nearly half a page to write down an existential over a simple structure… you should perhaps think of adopting more conventional notations in some places." Fair enough – we resolved to deal with the prolixity!

Another of his remarks "why don't you make all your specifications implementable by restricting your language to … [continuous functions over domains]" also had to be taken seriously. We responded to this by saying that we *wanted* our users to potentially be able to specify the unimplementable: in such cases the process of refinement would uncover the unimplementability and cause the specifiers (and their clients) to think again.

Other critical remarks came from some adherents of the school of algebraic specification – they eventually turned out to be based on worries about the use of the language of set theory.

## "Oxford Z" begins to emerge

Our rethinking of the notation first necessitated that we detach ourselves from the idea that we could somehow make mathematical ideas more palatable to working programmers by packaging them in a notation that resembled a programming language.[8]

It was Hélène who provided the inspiration behind the "box and lines" style of presentation that became the (superficial) trademark of a Z specification. More importantly, though, we reverted to a recognizably more orthodox presentation of mathematics: and spent significant effort on systematising it. Most notably, we unified the notations for introducing and constraining variables

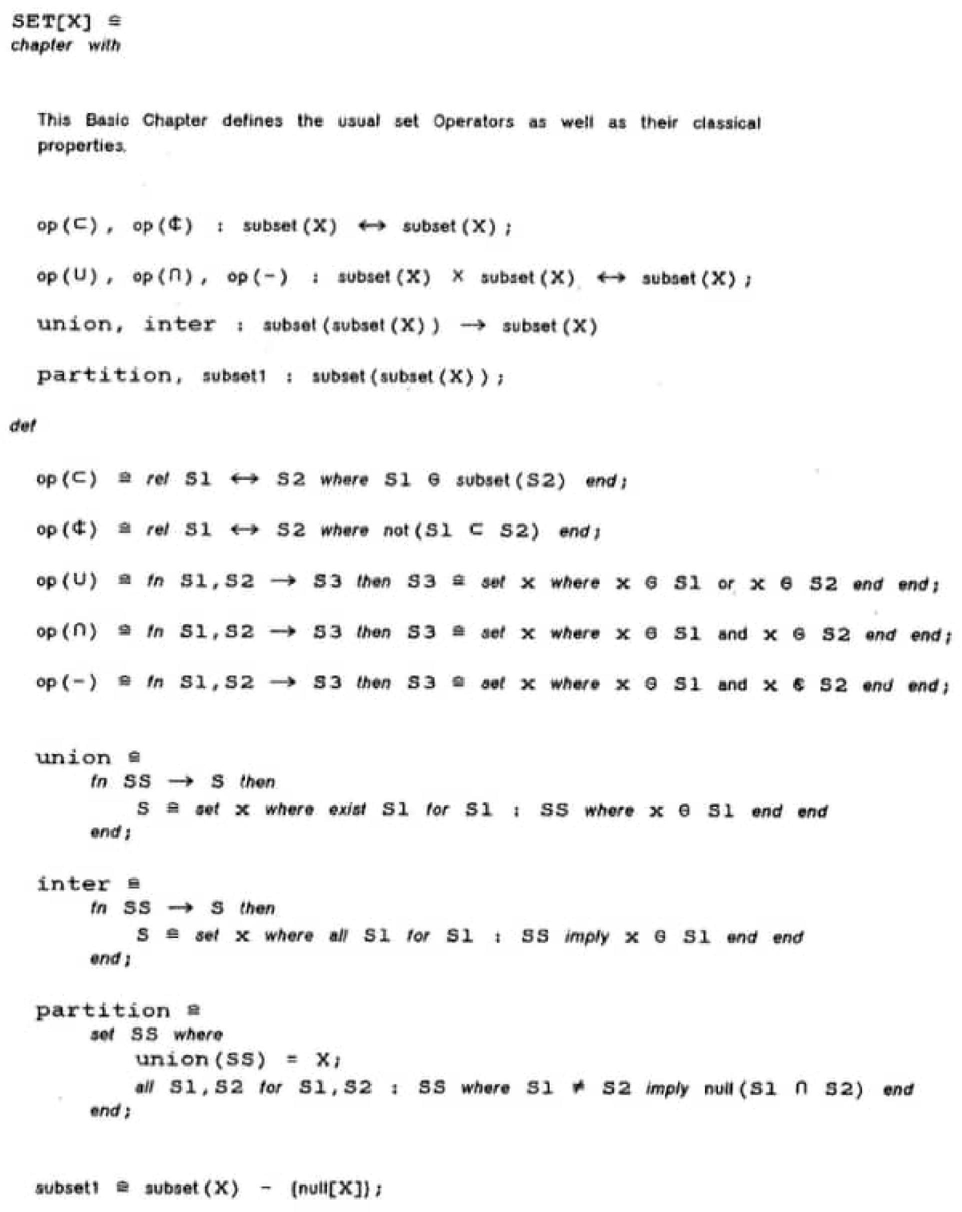
[6]The MSc cohort to which it was delivered is pictured in [2](p30)

[7]For example, the chapter `REL` on relations has fourteen collections of properties of the relational operators defined in it.

[8]I remember a couple of painful sessions of *autocritique* over dinner in Jean-Raymond and Héleǹe's kitchen. The three of us had dinner together very frequently during this period for reasons that I allude to later.

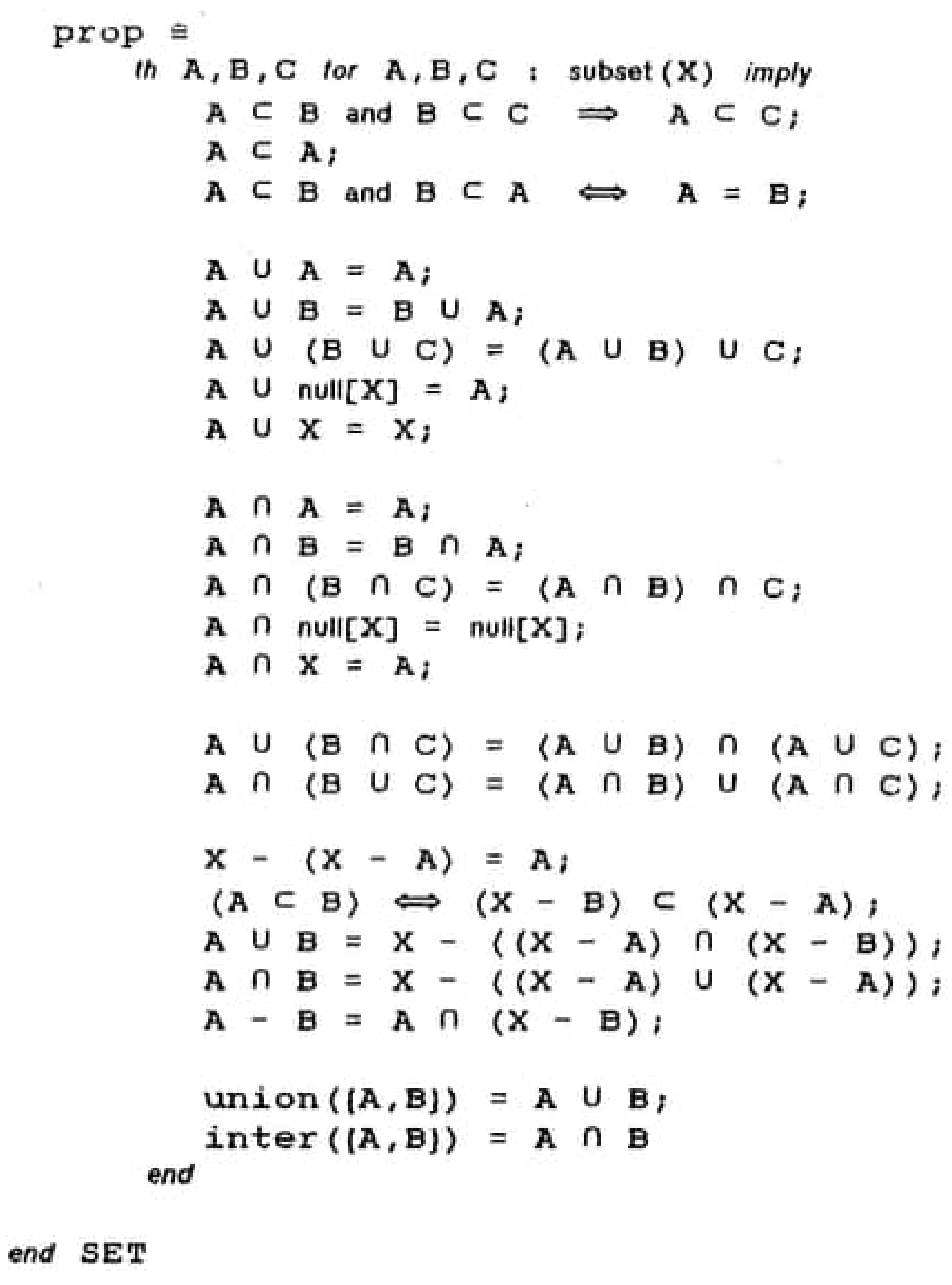
– whether in the signatures of theories, the bodies of structures, or the bound variables of quantified constructs. The ability to name a signature/predicate structure that this liberated gave rise to the notion of a schema.

Comparing Figure 8 with Figure 7 may give an inkling of the simplifications that we had made in this phase, and the firming up of our view that a “state” was really just a collection of related observations, some of which might never appear in the “reified”(Jones) data of an implementation. The simplified notation was promulgated quite widely: especially in the PRG and at INRIA [7]. Discussion of its later evolution and the incorporation of a *calculus* of schemas is beyond the scope of this note.



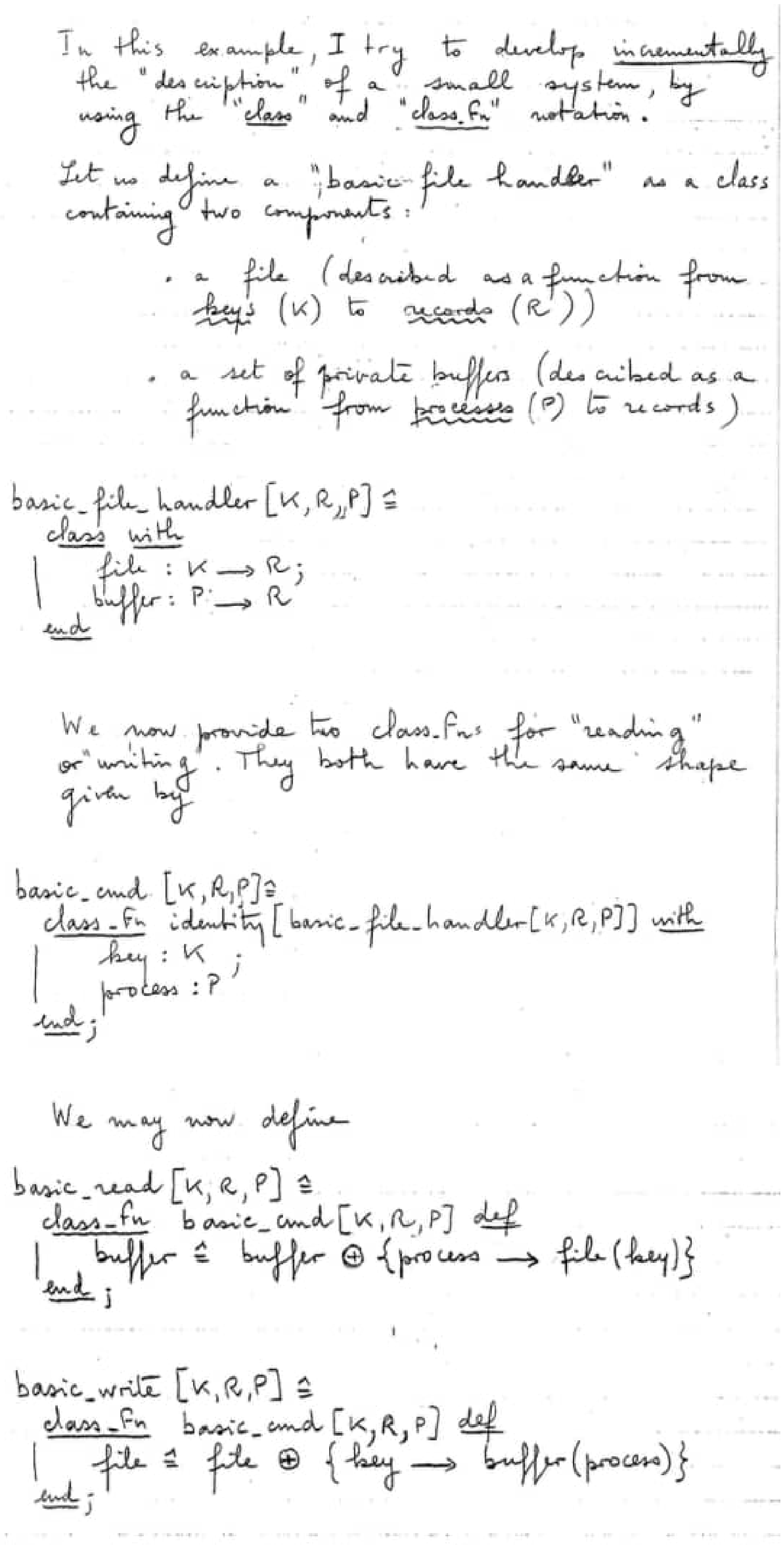
```
SET[X] ≙
chapter with

   This Basic Chapter defines the usual set Operators as well as their classical
   properties.

   op(⊂), op(⊄) : subset(X) ↔ subset(X);

   op(∪), op(∩), op(−) : subset(X) × subset(X) ↔ subset(X);

   union, inter : subset(subset(X)) → subset(X)

   partition, subset1 : subset(subset(X));

def

   op(⊂) ≙ rel S1 ↔ S2 where S1 ∈ subset(S2) end;

   op(⊄) ≙ rel S1 ↔ S2 where not(S1 ⊂ S2) end;

   op(∪) ≙ fn S1,S2 → S3 then S3 ≙ set x where x ∈ S1 or x ∈ S2 end end;

   op(∩) ≙ fn S1,S2 → S3 then S3 ≙ set x where x ∈ S1 and x ∈ S2 end end;

   op(−) ≙ fn S1,S2 → S3 then S3 ≙ set x where x ∈ S1 and x ∉ S2 end end;

   union ≙
       fn SS → S then
           S ≙ set x where exist S1 for S1 : SS where x ∈ S1 end end
       end;

   inter ≙
       fn SS → S then
           S ≙ set x where all S1 for S1 : SS imply x ∈ S1 end end
       end;

   partition ≙
       set SS where
           union(SS) = X;
           all S1,S2 for S1,S2 : SS where S1 ≠ S2 imply null(S1 ∩ S2) end
       end;

   subset1 ≙ subset(X) − {null[X]};
```

Figure 4: First part of the SET chapter of the Basic Library



```
prop ≙
    th A,B,C for A,B,C : subset(X) imply
        A ⊂ B and B ⊂ C  ⟹  A ⊂ C;
        A ⊂ A;
        A ⊂ B and B ⊂ A  ⟺  A = B;

        A ∪ A = A;
        A ∪ B = B ∪ A;
        A ∪ (B ∪ C) = (A ∪ B) ∪ C;
        A ∪ null[X] = A;
        A ∪ X = X;

        A ∩ A = A;
        A ∩ B = B ∩ A;
        A ∩ (B ∩ C) = (A ∩ B) ∩ C;
        A ∩ null[X] = null[X];
        A ∩ X = A;

        A ∪ (B ∩ C) = (A ∪ B) ∩ (A ∪ C);
        A ∩ (B ∪ C) = (A ∩ B) ∪ (A ∩ C);

        X - (X - A) = A;
        (A ⊂ B) ⟺ (X - B) ⊂ (X - A);
        A ∪ B = X - ((X - A) ∩ (X - B));
        A ∩ B = X - ((X - A) ∪ (X - A));
        A - B = A ∩ (X - B);

        union({A,B}) = A ∪ B;
        inter({A,B}) = A ∩ B
    end

end SET
```

Figure 5: Second part of the SET chapter of the Basic Library



In this example, I try to develop incrementally the "description" of a small system, by using the "class" and "class_fn" notation.

Let us define a "basic file handler" as a class containing two components:

. a file (described as a function from keys (K) to records (R))

. a set of private buffers (described as a function from processes (P) to records)

```
basic_file_handler [K,R,P] ≙
  class with
  |   file : K → R;
  |   buffer : P → R
  end
```

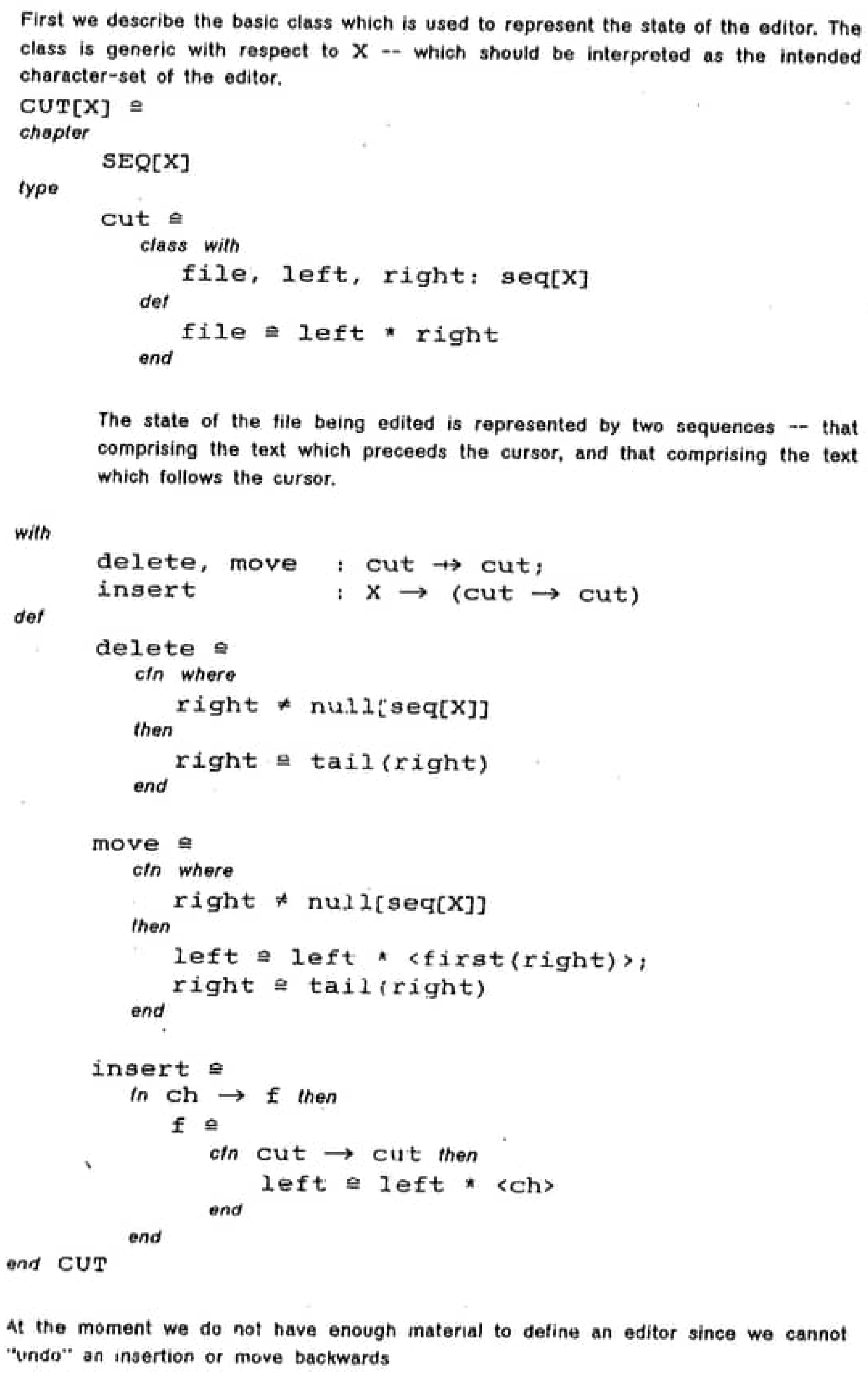
We now provide two class_fns for "reading" or "writing". They both have the same shape given by

```
basic_cmd [K,R,P] ≙
  class_fn identity[basic_file_handler[K,R,P]] with
  |   key : K ;
  |   process : P
  end;
```

We may now define

```
basic_read [K,R,P] ≙
  class_fn basic_cmd [K,R,P] def
  |   buffer ≙ buffer ⊕ {process → file(key)}
  end;

basic_write [K,R,P] ≙
  class_fn basic_cmd [K,R,P] def
  |   file ≙ file ⊕ {key → buffer(process)}
  end;
```

Figure 6: Extract from an experiment in defining a file handler



First we describe the basic class which is used to represent the state of the editor. The class is generic with respect to X -- which should be interpreted as the intended character-set of the editor.

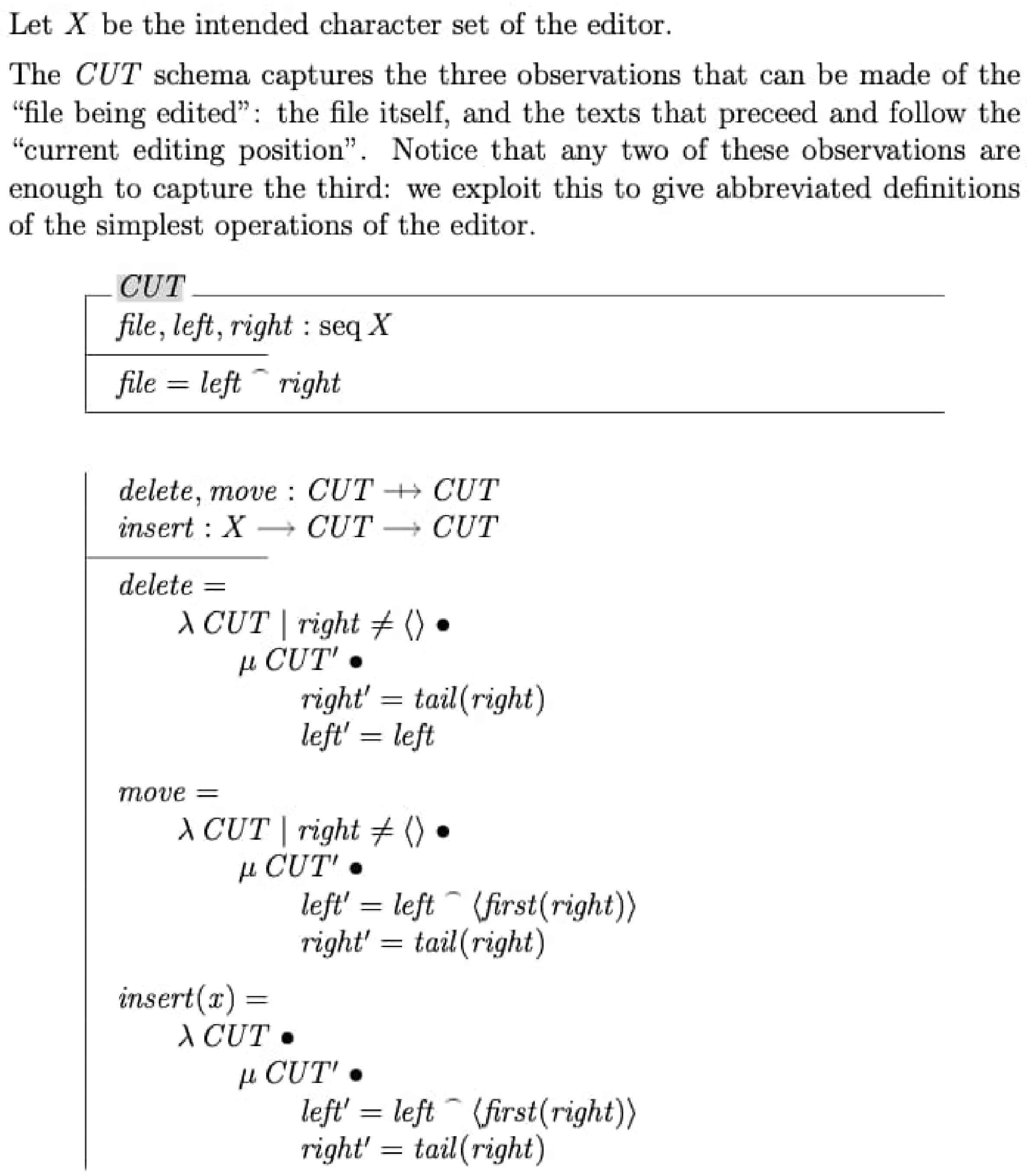
```
CUT[X] ≙
chapter
        SEQ[X]
type
        cut ≙
            class with
                file, left, right: seq[X]
            def
                file ≙ left * right
            end
```

The state of the file being edited is represented by two sequences -- that comprising the text which preceeds the cursor, and that comprising the text which follows the cursor.

```
with
        delete, move    : cut ⇸ cut;
        insert          : X → (cut → cut)
def
        delete ≙
            cfn where
                right ≠ null[seq[X]]
            then
                right ≙ tail(right)
            end

        move ≙
            cfn where
                right ≠ null[seq[X]]
            then
                left ≙ left * <first(right)>;
                right ≙ tail(right)
            end

        insert ≙
            fn ch → f then
                f ≙
                    cfn cut → cut then
                        left ≙ left * <ch>
                    end
            end
end CUT
```

At the moment we do not have enough material to define an editor since we cannot "undo" an insertion or move backwards

Figure 7: Part of the “Display Editor” in “Z”



Let $X$ be the intended character set of the editor.

The $CUT$ schema captures the three observations that can be made of the "file being edited": the file itself, and the texts that preceed and follow the "current editing position". Notice that any two of these observations are enough to capture the third: we exploit this to give abbreviated definitions of the simplest operations of the editor.

```
CUT
  $file, left, right : \operatorname{seq} X$
  ----
  $file = left \frown right$
```

```
  $delete, move : CUT \rightarrowtail\!\!\!\!\!\!\rightarrow CUT$
  $insert : X \longrightarrow CUT \longrightarrow CUT$
  ----
  $delete =$
      $\lambda\, CUT \mid right \neq \langle\rangle \bullet$
          $\mu\, CUT' \bullet$
              $right' = tail(right)$
              $left' = left$

  $move =$
      $\lambda\, CUT \mid right \neq \langle\rangle \bullet$
          $\mu\, CUT' \bullet$
              $left' = left \frown \langle first(right) \rangle$
              $right' = tail(right)$

  $insert(x) =$
      $\lambda\, CUT \bullet$
          $\mu\, CUT' \bullet$
              $left' = left \frown \langle first(right) \rangle$
              $right' = tail(right)$
```

Figure 8: Part of the "CUT" chapter of the "Display Editor" in early "Oxford Z"

## Alpine Adventures

*During the summer of 1980, recovering after the decisive end of a personal relationship, I stayed with Jean-Raymond and Hélène for a few weeks in a borrowed chalet up by the old Olympic downhill course in Grenoble. We did a lot of scrambling together – usually after long bumpy drives along the D1091. Happily, there was room for three of us in the front of the white van, for the mountain scenery was never less than spectacular; but when a fourth joined us, somebody had to rattle about with the equipment in the back.*

*Once we ascended the foreslopes of La Meije from the direction of Deux Alpes. After about four hours scrambling, and long before we had planned to descend, I was bitten horribly behind the knee by a huge horsefly. My whole leg swelled up enormously in about 15 minutes! J-R and our other companion had to help me hobble the long route back down with a nonfunctioning leg. We had to endure a difficult few hours descent before there would be a chance of getting it seen to; and my boot had to be cut off before we reached the van.*

*Eventually, after the long drive back to Grenoble, we reached a hospital. But it was 15th August, "the Assumption" – a day when the contradictions between the formally secular nature of the French state, and the Catholic nature of a significant proportion of the French have been historically resolved by making it a day when* everything *remotely public must close. To my dismay, the toxicologists were on holiday with "Adieu toxico', via Mexico!" on their whiteboard, but somebody gave me some antihistamine to stem the awful swelling, and by the next day it had been replaced by . . . a pervasive itch.*

*Another time, on Petit Taillefer, we had to rescue an inexperienced youth who had fallen on a rocky pitch that was much too hard for him. His companions, rich, elderly*[9] *gentlemen: were scarcely climbers at all. The rescue required us to carry the youth down the mountain for about 6km. Jean-Raymond did most of the carrying on his back, whilst I grunted obligingly in the rear, carrying the youth's equipment.*

## Au revoir

I wrote earlier of Jean-Raymond's frighteningly fast work rate. And for the first few months of 1981 he worked even more quickly – generating large numbers of drafts of new foundations for the language and its basic library, as well as building the prototype of an extensible proof checker, and writing essays inspired by Cliff Jones's book[6]. This work would (eventually) become the basis for **B** and the **B tool**. The rest of the "Z group" found it impossible to keep up with him, and wanted the notation to stay stable for a while. We had materials to prepare for courses and further collaborations with industrial partners to pursue,[10] and a large (5 year) Software Engineering project to prepare for.

Then one Friday lunchtime, Hélène Villers appeared without Jean-Raymond at Brown's restaurant in Oxford to join Cliff and Jill Jones, Linda Forrest and me, and Jill Hoare for lunch. She brought the upsetting news that Jean-Raymond had unilaterally decided that they would both leave Oxford and return to Paris at the end of the following month.[11] She didn't know why, and despite us all remaining good friends with them both for years afterwards none of us ever found out.

> "Those who have the privilege of friendship with Jean-Raymond Abrial have long been aware of the great work in which he has been engaged. It is no less than a complete understanding of the nature of software engineering: ..."
> (from C.A.R. Hoare's prefatory tribute to Abrial in "The B-Book")

## References

[1] Jean-Raymond Abrial. *The Specification Language Z: Syntax and "Semantics"*. Programming Research Group Working Paper, University of Oxford, April 1980.

[2] Bernard Sufrin. Early Days at the PRG ... anecdotes from the Programming Research Group for Tony Hoare's 90th birthday. *FACS FACTS*, 2024-2:25–35, July 2024.

[3] Jean-Raymond Abrial; Stephen A. Schuman; Bertrand Meyer. A Specification Language. In *On the Construction of Programs*, in MacNaghten, A. M.; McKeag, R. M. (eds.). Cambridge University Press, 1980. ISBN 0-521-23090-X.

[4] Jean-Raymond Abrial. *The Specification Language Z: Basic Library*. Programming Research Group Working Paper, University of Oxford, April 1980.

[5] Jean-Raymond Abrial; Bernard Sufrin; Tim Clement; Ib Sørensen. *Specification of CAVIAR: A Computer Aided Visitor Information and Reception System*. Programming Research Group Working Paper, University of Oxford, (undated).

[6] Cliff B. Jones. *Software Development: A Rigorous Approach*. Prentice-Hall International, 1980. ISBN 978-0138218843.

[7] Bernard Sufrin. Formal System Specifications (Notation and Examples). In *Tools and Notions for Program Construction: An advanced course*, D. Néel (ed.). Cambridge University Press, 1982. ISBN 0-52124801-9.

[10] Ib Sørensen had started working with the CICS team at IBM on a project to develop a rational reconstruction of the ideas behind this large, long-lived, and difficult-to-maintain transaction processing system.

[11] At least they had somewhere to live. They had retained Hélène's student-era bathroomless "Chambre-de-bonne" apartment 6 floors' walk upstairs in a town-house near Place Denfert-Rochereau.

# Chronological bibliography of Jean-Raymond Abrial

Henri Habrias[1]

December 2025

*"The very first paper on Z was published in 1980 (at the time, the name Z was not 'invented'), then the book on the B method was published in 1996, and, finally, the book on Event-B was published in 2010. So, 30 years separate Z from Event-B. It is thus clear that I spent a significant time of my scientific professional life working with the same kind of subject in mind, roughly speaking specification languages. I do not know whether this kind of addiction is good or bad, but what I know is that I enjoyed it a lot."* — J.-R. Abrial[2]

## A Langage LTR-2

### 1966

La programmation modulaire appliquée à un système militaire en temps réel, J.-R. Abrial, J. Bourgne, P. Yvon. Centre de Programmation de la Marine, Paris.

## B Grenoble, Socrate and Data Semantics

### 1970

Projet SOCRATE (1) Spécifications générales, J.-R. Abrial, J. Bas, G. Beaume, G. Henneron, R. Morin, G. Vigliano. Communication Institut de Mathématiques Appliquées, University of Grenoble, August.

Système de définition et interrogation de données: application au dossier médical, J.-R. Abrial, G. Beaume, G. Henneron, R. Morin, G. Vigliano, J. Valois, S. Cohen. Symposium de Toulouse.

### 1972

Structure de données et de programmes, cours C4 1ère partie, point de vue existentiel, J.-R. Abrial. Faculty of SCiences, University of Grenoble. **https://www.patstec.fr/MEDIAS/005-005-000015609/8255.pdf**

[1] Aided by Dominique Cansell; edited by Jonathan P. Bowen.

[2] From Z to B and then Event-B: Assigning Proofs to Meaningful Programs, J.-R. Abrial In: E.B. Johnsen, L. Petre (eds), Integrated Formal Methods (IFM 2013). LNCS, vol. 7940, pp. 1–15. Springer. ISBN: 978-3-642-38612-1. doi:10.1007/978-3-642-38613-8-1

### 1973

Data semantics, J.-R. Abrial. Workshop, Alpe d'Huez.

Description sémantique des bases de données, Notes pour le cours donné au 7th workshop, J.-R. Abrial. In: IAG Data Base Series, Brussels, August.

### 1974

Data Semantics, J.-R. Abrial. In: J.W. Klimbie, K.L. Koffeman (eds), Proceedings of the IFIP Working Conference Data Base Management, Cargèse, Corsica, France, April, pp. 1–66. North-Holland. ISBN: 0-7204-2809-2.
https://hal.univ-grenoble-alpes.fr/hal-05150953v1/document

## C Z

### 1977

Manuel du langage Z (Z/13), J.-R. Abrial. IMAG, Grenoble.

Mécanismes de transformations du langage Z (Z/14), Notes internes Z/1 à Z/15, J.-R. Abrial. Service IMA (Informatique et Mathématiques Appliquées) DER (Direction des Etudes et Recherches), Clamart, June. Unpublished.

Utilisation du langage Z pour l'analyse d'une petite application de gestion (Z/15), J.-R. Abrial. EDF, DER IMA, November.

### 1978

Notes de cours sur les spécifications formelles; étude de cas (un système de répertoire automatisé), J.-R. Abrial. École d'été Méthodologie de la programmation, théorie et pratique, INRIA-EDF-CEA, Bréau-sans-Nappe, July.

### 1979

Non Deterministic System Specification, Semantics of Concurrent Computation, J.-R. Abrial, S.A. Schuman. In: Kahn, G. (ed), Semantics of Concurrent Computation. LNCS, vol. 70, pp. 34–35. Springer. ISBN: 978-3-540-09511-8.

### 1980

(1) The Specification Language Z: Basic Library, 30 pages; (2) The Specification Language Z: Syntax and Semantics, 29 pp; (3) An Attempt to Use Z for Defining the Semantics of an Elementary Programming Language, 3 pages (4) A Low Level File Handler Design, 18 pages; (5) Specification of Some Aspects of a Simple Batch Operating System, 37 pages, J.-R. Abrial. Internal Report, Programming Research Group, Oxford University.

The Specification Language Z: Syntax and "Semantics", J.-R. Abrial. Oxford University Computing Laboratory, PRG Technical Report, April.

Specification Language, J.-R. Abrial, S.A. Schuman, B. Meyer. In: R.M. McKeag, A.M. Macnaghten (eds), On the Construction of Programs: An Advanced Course, pp. 343–410. Cambridge University Press.

### 1981

Example 2: KWIC – index generation, J.-R. Abrial, I.H. Sørensen. In: J. Staunstrup (ed), Program Specification (ProgSpec 1981). LNCS, vol. 134, pp. 88–95. Springer. ISBN: 978-3-540-11490-1.

## D ADA

### 1983

Reference manual for the Ada programming language, J.D. Ichbiah, R. Firth, P.N. Hilfinger, O. Roubine, M. Woodger, J.G.P. Barnes, J.-R. Abrial, J.-L. Gailly, J.-C. Heliard, H.F. Ledgard, B.A. Wichmann, B. Krieg-Brueckner. Ada Joint Program Office, ANSI/MIL STD 1815A.

## E Towards B

### 1984

Spécifier ou comment maîtriser l'abstrait, J.-R. Abrial. T.S.I. (Technique Et Science Informatique), vol. 3, no. 3, pp. 201–219. ISSN: 0752-4072.

### 1986

A small case study in program design, J.-R. Abrial. November, unpublished.

### 1987

The Case of Proportional Representation Elections in LCP (A. Coulon, M. Koutchouk, Bull), Deductive Programming (Cl. Pair), J.-R. Abrial's Method (J.-R. Abrial), H. Habrias (ed), IUT of Nantes, Département informatique.

### 1989

A Formal Approach to large Software Construction, J.R. Abrial. In: J.L.A. van de Snepscheut (ed), Mathematics of Program Construction (MPC 1989). LNCS, vol. 375, pp. 1–20. Springer. ISBN: 978-3-540-51305-6. doi:10.1007/3-540-51305-1_1

### 1990

Une approche formelle de la construction des logiciels, J.-R. Abrial. In: H. Habrias (ed), Actes des 3mes journées MOACSI (Méthodes et Outils d'Aide à la Conception de Systèmes d'Information), September.

### 1991

A Refinement Case Study (using the Abstract Machine Notation), J.-R. Abrial. In: J.M. Morris, R.C. Shaw (eds), 4th Refinement Workshop, pp. 51–96. Workshops in Computing, Springer. ISBN: 978-3-540-19657-0.

The B-method, J.-R. Abrial, M.K.O. Lee, D.S. Neilson, I.H. Sørensen, VDM'91 Formal Software Development Methods, January. LNCS, vol. 552, pp. 398–405. Springer. ISBN: 978-3-540-54868-3. doi:10.1007/BFb0020001

## F B

### 1996

Extending B without changing it (for developing distributed systems), J.-R. Abrial. In: 1st B International Conference, H. Habrias (ed), Nantes. ISBN: 2906082252.

Steam boiler control specification problem, J.-R. Abrial. Dagstuhl Seminar, Formal Methods for Industrial Applications 1995, LNCS, vol. 1167, pp. 1–20. Springer. ISBN: 978-3-540-61929-1.

The B-Book: Assigning Programs to Meanings, J.-R. Abrial. Cambridge University Press, ISBN: 978-0511624162. doi:10.1017/CBO9780511624162

## 1997

Introduction à la méthode B, VHS cassettes, Cours et études de cas, J.-R. Abrial. IUT de Nantes.

Construction d'automatismes industriels avec B, J.-R. Abrial. Invited conference talk, AFADL'97, CERT-ONERA, Toulouse, 28–29 May.

## 1998

Introducing Dynamic Constraints in B, J.-R. Abrial, L. Mussat. In: B'98: Recent Advances in the Development and Use of the B Method, Second International B Conference, D. Bert (ed), Montpellier, April 1998, LNCS vol. 1393, pp. 83–128. Springer. ISBN: 978-3-540-64405-7.

Le Cahier des Charges: Contenu, Forme et Analyse (en vue de la Formalisation), J.-R Abrial. June, unpublished.

## 2000

B: 2000 et plus, version 2.2, J.-R Abrial. January, unpublished.

Guidelines to Formal System Studies, J.-R. Abrial. November, unpublished.

Formalism for complete correct system development, J.-R Abrial. January, unpublished

## 2001

Specification and Design of the Leader Election Protocol of IEEE 1394, J.-R. Abrial, D. Cansell, D. Méry. In: C. Shankland, J. Romijn, S. Maharaj (eds), IEEE 1394 (Firewire) Workshop: International Workshop on Applications of Formal Methods to IEEE 1394 Standard. IEEE 1394, University of Stirling, Department of Computing Science & Maths. ISBN: 978-1857691535.

## 2002

Higher-Order Mathematics in B, J.-R. Abrial, D. Cansell, G. Laffitte. In: D. Bert, J.P. Bowen, M.C. Henson, iK. Robinson (eds), ZB 2002:Formal Specification and Development in Z and B (ZB 2002), LNCS, vol. 2272, pp. 170–393. Springer. ISBN: 978-3-540-43166-4.

On Using Conditional Definitions in Formal Theories, J.-R. Abrial, L. Mussat. In: D. Bert, J.P. Bowen, M.C. Henson, iK. Robinson (eds), ZB 2002:Formal Specification and Development in Z and B (ZB 2002), LNCS, vol. 2272, pp. 242–269. Springer. ISBN: 978-3-540-43166-4.

## 2003

B# Toward a synthesis between Z and B, J.-R. Abrial. In: D. Bert, J.P. Bowen, S. King, M. Waldén (eds), ZB 2003: Formal Specification and Development in Z and B. LNCS, vol. 2651, pp. 168–177. Springer. ISBN: 978-3-540-40253-4.

B: passé, présent, futur, J.-R. Abrial, D. Bert, H. Habrias, V. Donzeau-Gouge. Technique et Science Informatiques, pp. 88–118.

A Mechanically Proved and Incremental Development of IEEE 1394 Tree Identify Protocol, J.-R. Abrial, D. Cansell, D. Méry. Formal Aspects of Computing, vol. 14, no. 3, pp. 215–227. doi:10.1007/s001650300002.

Formal Derivation of Spanning Trees Algorithms, J.-R. Abrial, D. Cansell, D. Méry. In: D. Bert, J.P. Bowen, S. King, M. Waldén (eds), ZB 2003: Formal Specification and Development in Z and B. LNCS, vol. 2651, pp. 457–476. Springer. ISBN: 978-3-540-40253-4.

Click'n Prove: Interactive Proofs within Set Theory, J.-R. Abrial, D. Cansell. In: D. Basin, B. Wolff (eds), Theorem Proving in Higher Order Logics (TPHOLs 2003). LNCS, vol. 2758, pp. 1–24. Springer. ISBN: 978-3-540-40664-8.

Event Based Sequential Program Development: Application to Constructing a Pointer Program, J.-R. Abrial. In: K. Araki, S. Gnesi, D. Mandrioli (eds), FME 2003. LNCS, vol. 2805, pp. 51–74. Springer. ISBN: 978-3-540-40828-4.

## 2005

The B-method, J.-R. Abrial, M.K.O. Lee, D.S. Neilson, I.H. Sørensen. In: VDM '91 Formal Software Development Methods, January. LNCS, vol. 552, pp. 398–405. Springer. ISBN: 978-3-540-54868-3. doi:10.1007/BFb0020001

Refinement and Reachability in Event-B, J.-R. Abrial, D. Cansell, D. Méry. In: H. Treharne, S. King, M.C. Henson, S. Schneider (eds), ZB 2005: Formal Specification and Development in Z and B. LNCS, vol. 3455, pp. 222–241. Springer. ISBN: 978-3-540-25559-8.

Rodin deliverable 3.2. Event-B language, C. Métayer, J.-R. Abrial, L. Voisin. Technical Report, University of Newcastle upon Tyne, UK.

The Challenge of Probabilistic Event B – Extended Abstract, C. Morgan, T.S. Hoang, J.-R. Abrial. In: H. Treharne, M.C. Henson, S. Schneider (eds), ZB 2005: Formal Specification and Development in Z and B. LNCS, vol. 3455, pp. 162–171. Springer. ISBN: 978-3-540-25559-8.

Formal Construction of a Non-blocking Concurrent Queue Algorithm, J.-R. Abrial, D. Cansell. Journal of Universal Computer Science (JUCS), vol. , no. 5, pp. 744–770. doi:10.3217/jucs-011-05-074

## G Event-B and Rodin

### 2006

Have we learned from the Vasa Disaster?, J.-R. Abrial. In: Charles Simonyi Symposium: Grand Challenges of Informatics, Academia Europaea Symposium, Budapest, 19–20 September. https://kifu.videotorium.hu/en/recordings/1674/have-we-learned-from-the-wasa-disaster

Roadmap for enhanced languages and methods to aid verification, G.T. Leavens, J.-R. Abrial, D. Batory, M. Butler, et al. In: GPCE '06: Proceedings of the 5th international conference on Generative programming and component engineering, Portland, Oregon, USA, pp. 221–236. ACM. ISBN: 978-1-59593-237-2.

Tools for Developing Large Systems (A Proposal), J.-R. Abrial. In: M. Butler, C.B. Jones, A. Romanovsky, E. Troubitsyna (eds), Rigorous Development of Complex Fault-Tolerant Systems. LNCS, vol. 4157, pp. 387–390. Springer. ISBN: 978-3-540-48265-9.

An Open Extensible Tool Environment for Event-B, J.-R. Abrial, M. Butler, S. Hallerstede, L. Voisin. In: Z. Liu, J. He (eds), Formal Methods and Software Engineering (ICFEM 2006). LNCS, vol. 4260, pp. 588–605. Springer. ISBN: 978-3-540-47460-9,

### 2007

Formal Methods: Theory Becoming Practice, J.-R. Abrial. Journal of Universal Computer Science, vol. 13, no. 5, pp. 619–628. doi:10.3217/jucs-013-05-0619

A System Development Process with Event-B and the Rodin Platform, J.-R. Abrial. In: M. Butler, M.G. Hinchey, M.M. Larrondo-Petrie (eds), Formal Methods and Software Engineering (ICFEM 2007). LNCS, vol. 4789, pp. 1–3. Springer. ISBN: 978-3-540-76648-3.

Deliverable D8 D10.1 "Teaching Materials", J.-R. Abrial, T.S. Hoang, M. Schmalz. Rodin Platform Archives, University of Southampton. https://web-archive.southampton.ac.uk/deploy-eprints.ecs.soton.ac.uk/56/

### 2008

Link State Routing Development, T.S. Hoang, D. Basin, H. Kuruma, J.-R. Abrial. Rodin Platform Archive, University of Southampton. https://web-archive.southampton.ac.uk/deploy-eprints.ecs.soton.ac.uk/31/

Modelling and Proof of a Tree-Structured File System in Event-B and Rodin, K. Damchoom, M. Butler, J.-R. Abrial. In: ICFEM '08: Proceedings of the 10th International Conference on Formal Methods and Software Engineering. LNCS, vol. 5256, pp. 25–44. Springer. ISBN: 978-3-540-88193-3.

## 2009

Doing Mathematics with the Rodin Platform, J.-R. Abrial. Rodin Platform Archive, University of Southampton.
https://web-archive.southampton.ac.uk/deploy-eprints.ecs.soton.ac.uk/138/

Development of a Network Topology Discovery Algorithm, T.S. Hoang, D. Basin, H. Kuruma, J.-R. Abrial. Rodin Platform Archive, University of Southampton.
https://web-archive.southampton.ac.uk/deploy-eprints.ecs.soton.ac.uk/82/

Faultless Systems: Yes We Can! J.-R. Abrial. Computer, vol. 42, no. 9, pp. 30–36. doi:10.1109/MC.2009.283

Event-B patterns and their tool support, T.S. Hoang, A. Fürst, J.-R. Abrial. In: D.V. Hung, P. Krishnan (eds), SEFM 2009, pp. 210–219. IEEE Computer Society.

## 2010

Modeling in Event-B, System and Software Engineering, J.-R. Abrial. Cambridge University Press. ISBN: 978-0-521-89556-9.

Event-B Decomposition for Parallel Programs, T.S. Hoang, J.-R. Abrial. In: M. Frappier, U. Glässer, S. Khurshid, R. Laleau, S. Reeves (eds), Abstract State Machines, Alloy, B and Z (ABZ 2010). LNCS, vol. 5977, pp. 319–333. Springer. ISBN: 978-3-642-11810-4.

Rodin: an open toolset for modelling and reasoning in Event-B, J.-R. Abrial, M. Butler, S. Hallerstede, et al. International Journal of Software Tools Technoly Transfer, vol. 12, pp. 447–466.

Extended Abstracts Collection – Refinement Based Methods for the Construction of Dependable Systems, J.-R. Abrial, M. Butler, R. Joshi, E. Troubitsyna, J.C.P. Woodcock. In: Refinement Based Methods for the Construction of Dependable Systems. Dagstuhl Seminar Proceedings, vol. 9381. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. https://www.dagstuhl.de/09381.

## 2011

Series of courses on Event-B and Rodin, J.-R. Abrial. Microsoft Research.
https://www.youtube.com/watch?v=2GP1pJINVT4

Reasoning about Liveness Properties, T.S. Hoang, J.-R. Abrial. In: S. Qin, Z. Qiu (eds), Formal Methods and Software Engineering (ICFEM 2011). LNCS, vol. 6991, pp. 456–471. Springer. ISBN: 978-3-642-24558-9.

From Requirements to Development: Methodology and Example, W. Su, J.-R. Abrial, R. Huang, H. Zhu. In: S. Qin, Z. Qiu (eds), Formal Methods and Software Engineering (ICFEM 2011). LNCS, vol. 6991, pp. 437–455. Springer. ISBN: 978-3-642-24558-9.

## 2012

Formalizing Hybrid Systems With Event-B, J.-R. Abrial, W. Su, H. Zhu. In: Abstract State Machines, Alloy, B, VDM and Z (ABZ 2012), Pisa. LNCS, vol. 7316, pp. 178–193. Springer. ISBN: 978-3-642-30884-0. doi:10.1007/978-3-642-30885-7_13

Complementary Methodologies for Developing Hybrid Systems with Event-B, W. Su, J.-R. Abrial, H. Zhu. In: T. Aoki, K. Taguchi (eds), Formal Methods and Software Engineering (ICFEM 2012). LNCS, vol. 7635, pp. 230–248. Springer. ISBN: 978-3-642-34280-6.

## 2013

From Z to B and then Event-B: Assigning Proofs to Meaningful Programs, J.-R. Abrial In: E.B. Johnsen, L. Petre (eds), Integrated Formal Methods (IFM 2013). LNCS, vol. 7940, pp. 1–15. Springer. ISBN: 978-3-642-38612-1. doi:10.1007/978-3-642-38613-8-1

Set-Theoretic Models of Computations, J.-R. Abrial. In: Z. Liu, J.C.P. Woodcock, H. Zhu (eds), Theories of Programming and Formal Methods. LNCS, vol. 8051, pp. 1–22. Springer. ISBN: 978-3-642-39697-7.

Event-B patterns and their tool support, T.S. Hoang, A. Fürst, J.-R. Abrial. Software System Modeling, vol. 12, pp. 229–244. doi:10.1007/s10270-010-0183-7

## 2014

The Rodin Platform has turned ten, L. Voisin, J.-R. Abrial. In: Y. Ait Ameur, K.D. Schewe (eds), Abstract State Machines, Alloy, B, TLA, VDM, and Z (ABZ 2014). LNCS, vol. 8477, pp. 1–8. Springer. ISBN: 978-3-662-43651-6.

Formalizing hybrid systems with Event-B and the Rodin Platform, W. Su, J.-R. Abrial, H. Zhu. Science of Computer Programming, vol. 94, part. 2, pp. 164–202. doi:10.1016/j.scico.2014.04.015

Mathematical Case Studies with the Rodin Platform (version 2), J.-R. Abrial. Unpublished.

Transforming Guarded Events into Pre-conditioned Operations, J.-R. Abrial, W. Su. Rodin User and Developer Workshop, slides, June.

## 2015

Spécification, construction et vérification de programmes: le parcours d'une pensée scientifique sur une quarantaine d'années, J.-R. Abrial. Séminaire: Prouver les programmes: pourquoi, quand, comment? de Gérard Berry, Collège de France. **https://www.college-de-france.fr/fr/agenda/seminaire/prouver-les-programmes-pourquoi-quand-comment/specification-construction-et-verification-de-programmes-le-parcours-une-pensee-scientifique-sur-une**

Formal Development of a Real-Time Operating System Memory Manager, W. Su, J.-R. Abrial, G. Pu, B. Fang. In: 20th International Conference on Engineering of Complex Computer Systems (ICECCS), Gold Coast, Queensland, Australia, pp. 130–139.

An Exercise in Mathematical Engineering: Stating and Proving Kuratowski Theorem, J.-R. Abrial. In: M. Leucker, C. Rueda, F. Valencia (eds), Theoretical Aspects of Computing (ICTAC 2015). LNCS, vol. 9399, pp. 3–27. Springer. ISBN: 978-3-319-25149-3.

## 2016

Formal Proof of the Weak Goodstein Theorem, J.-R. Abrial. Event-B Day, Tokyo, November. ArXiv. `https://arxiv.org/abs/1701.01673`

Modelling and Refining Hybrid Systems in Event-B and Rodin, M. Butler, J.-R. Abrial, R. Banach. In: L. Petre, E. Sekerinski (eds), From Action Systems to Distributed Systems, The Refinement Approach, pp. 29–42. Chapman and Hall/CRC. doi:10.1201/b20053

## 2017

Proving Weak and Strong Goodstein Theorems, D. Cansell, J.-R. Abrial. Unpublished.

Aircraft landing gear system: approaches with Event-B to the modeling of an industrial system, W. Su, J.-R. Abrial. International Journal of Software Tools Technology Transfer, vol. 19, pp. 141–166. doi:10.1007/s10009-015-0400-3

## 2018

On B and Event-B: Principles, Success and Challenges, J.-R. Abrial In: M. Butler, A. Raschke, T.S. Hoang, K. Reichl (eds), Abstract State Machines, Alloy, B, TLA, VDM, and Z (ABZ 2018). LNCS, vol. 10817, pp. 31–35. Springer. ISBN: 978-3-319-91270-7. doi:10.1007/978-3-319-91271-4_3

Formal modelling of list based dynamic memory allocators, B. Fang, M. Sighireanu, G. Pu, W. Su, J.-R. Abrial, M. Yang, L. Qiao. Science China: Information Science, vol. 61, 122103. doi:10.1007/s11432-017-9280-9

## 2020

The ABZ-2018 Case Study with Event-B, J.-R. Abrial. International Journal of Software Tools Technology Transfer, vol. 22, pp. 257–264. doi:10.1007/s10009-019-00525-3

## 2021

Examples of using the Instantiation Plug-in, D. Cansell, J.-R. Abrial. The 9th Rodin User and Developer Workshop, 8 June, Ulm, Germany (Virtual).

## 2022

Ordinals less than $\varepsilon_0$, D. Cansell, J.-R. Abrial. Unpublished.

## 2023

Constructing the Real Numbers using RODIN and EBRP's plugin, J.-R. Abrial, D. Cansell. In: The 10th Rodin User and Developer Workshop, 30 May, Nancy, France.

Génération télescopique de sous-ensemble d'ordinaux à l'aide de la plateforme Rodin, D. Cansell, J.-R. Abrial. Project EBRP. Unpublished.

## 2025

The Proved Construction of a Protocol with an Example Inspired by the Paxos Protocol. D. Cansell, J.-R. Abrial In: M. Leuschel, F. Ishikawa (eds), Rigorous State-Based Methods (ABZ 2025). LNCS, vol. 15728, pp. 143–160. Springer. ISBN: 978-3-031-94532-8. doi:10.1007/978-3-031-94533-5_9

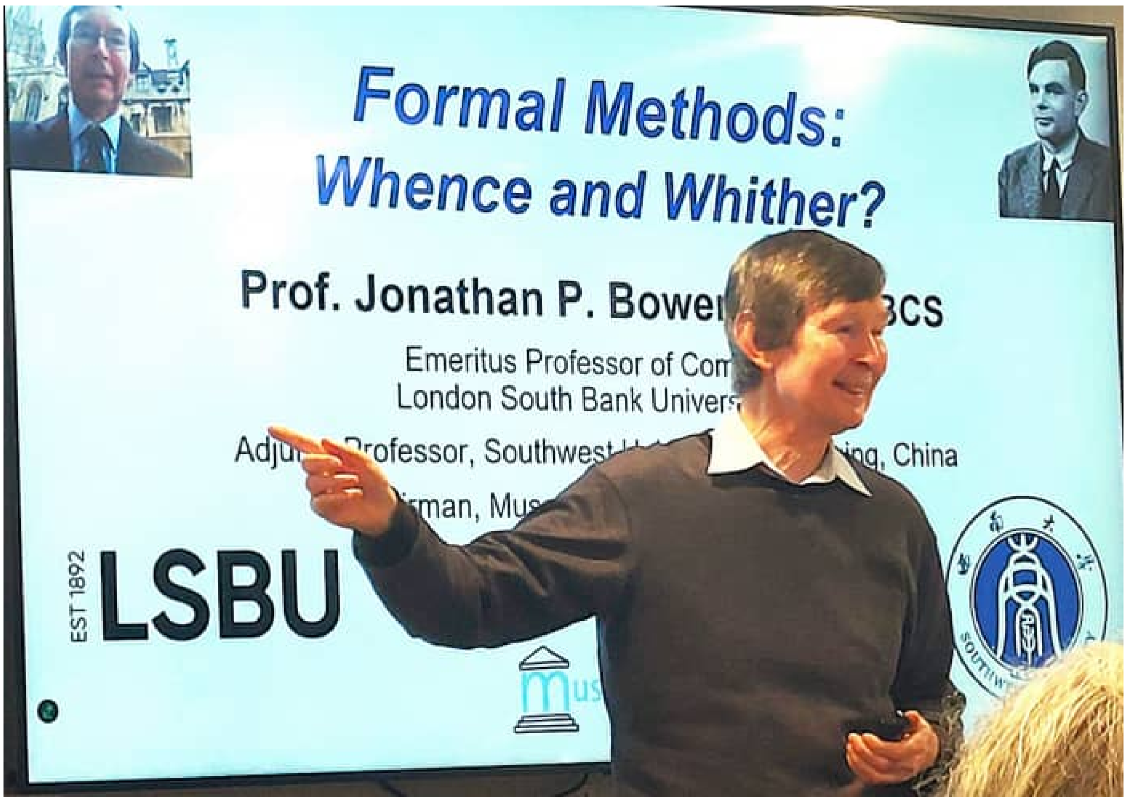
# BCS FACS 2025 Landin Lecture
# **Formal Methods: Whence and Whither?**

Jonathan P. Bowen

4th December 2025

BCS London office

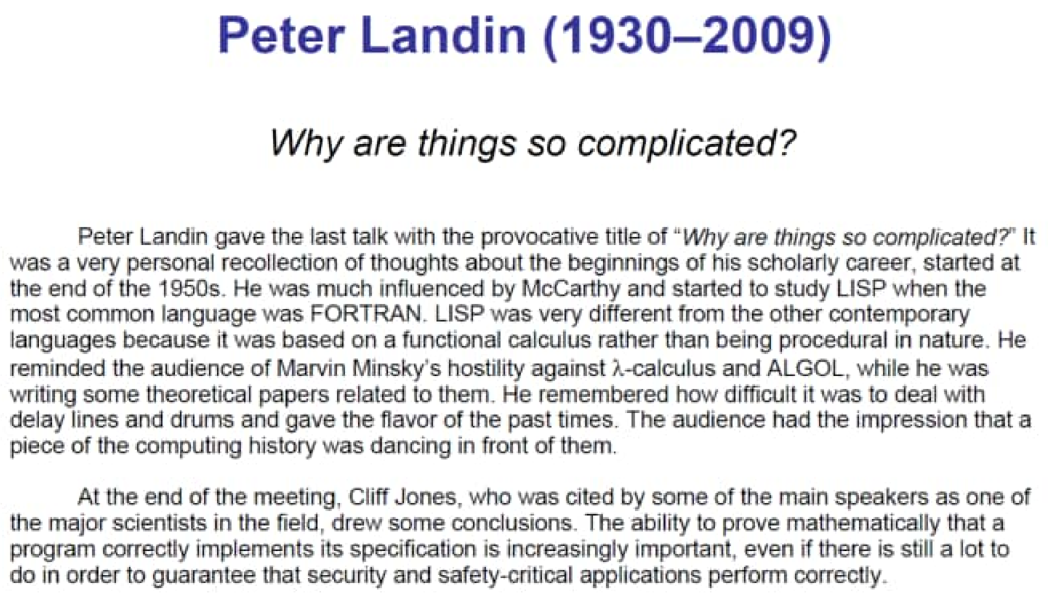
(**Reported by:** Brian Monahan and Keith Lines)



*Jonathan P. Bowen, FBCS FRSA, is Chairman of Museophile Limited (founded in 2002), an Emeritus Professor at London South Bank University, where he was Professor of Computing, establishing and heading the Centre for Applied Formal Methods from 2000.*

At this year's Landin seminar, Jonathan Bowen marked his standing down as FACS Chair by giving a presentation that drew on his considerable experience in formal methods, as both an academic and as a software engineer. This important talk covered a lot of historical ground concerning formal methods since its inception and underlined the continuing importance of this topic, and potentially foreshadowing future developments involving AI and so on.

After a brief summary of his own extensive career, Jonathan spoke briefly about one of (the only?) occasions that he met Peter Landin himself, at a 2001 seminar titled "*Program Verification and Semantics: The early work*" for which Peter was one of the organisers. Peter Landin himself spoke at this conference, giving the last presentation titled "*Why are things so complicated?*" (see below). Jonathan mostly recalls how Peter was truly on terrific form that day and how much laughter ensued!



## Peter Landin (1930–2009)

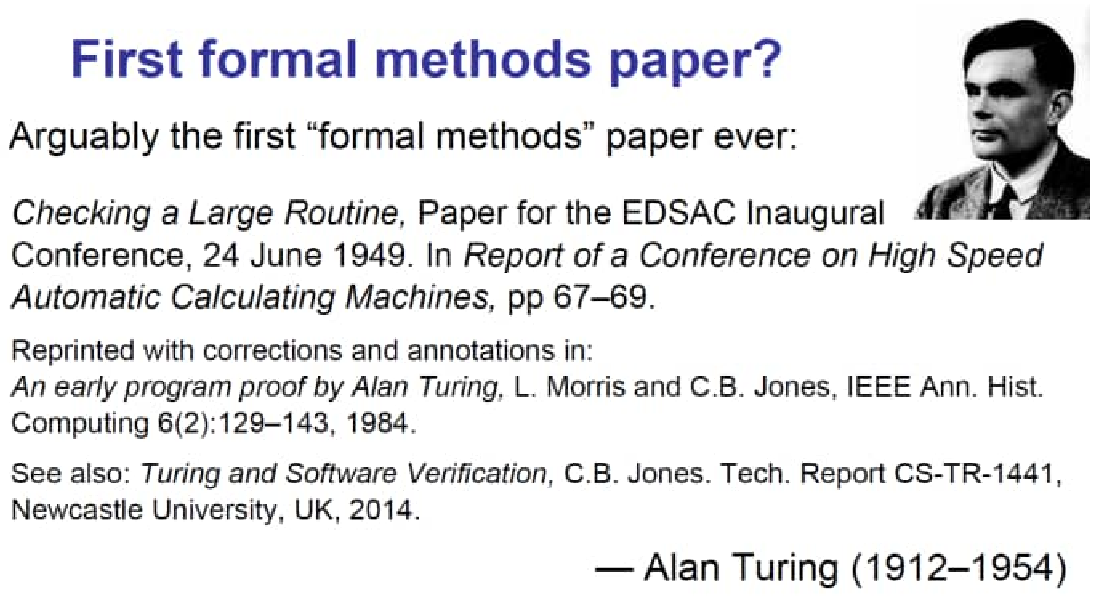
*Why are things so complicated?*

Peter Landin gave the last talk with the provocative title of "*Why are things so complicated?*" It was a very personal recollection of thoughts about the beginnings of his scholarly career, started at the end of the 1950s. He was much influenced by McCarthy and started to study LISP when the most common language was FORTRAN. LISP was very different from the other contemporary languages because it was based on a functional calculus rather than being procedural in nature. He reminded the audience of Marvin Minsky's hostility against λ-calculus and ALGOL, while he was writing some theoretical papers related to them. He remembered how difficult it was to deal with delay lines and drums and gave the flavor of the past times. The audience had the impression that a piece of the computing history was dancing in front of them.

At the end of the meeting, Cliff Jones, who was cited by some of the main speakers as one of the major scientists in the field, drew some conclusions. The ability to prove mathematically that a program correctly implements its specification is increasingly important, even if there is still a lot to do in order to guarantee that security and safety-critical applications perform correctly.

As suggested above, Jonathan spoke of many things relating to "formal methods" and its applications – so many in fact that it presents your reporters with something of a difficulty – what aspects do we report on here? Suffice to say that we mention here an eclectic sample of topics from Jonathan's talk – and we encourage the reader to look through Jonathan's slides and watch the YouTube video for themselves (see below for links).

Jonathan's first half of his talk gave an account of the history of formal methods, albeit with a distinctly UK flavour. Arguably, it was Alan Turing himself who initiated the entire subject with a brilliant *two-page* paper presented at the EDSAC Inaugural Conference in 1949, containing references to many of the ideas of "assertions", "dashed states" and of course, verifying that the computation came to an end (termination). Unfortunately, it was neglected for well over a couple of decades.



## First formal methods paper?

Arguably the first "formal methods" paper ever:

*Checking a Large Routine,* Paper for the EDSAC Inaugural Conference, 24 June 1949. In *Report of a Conference on High Speed Automatic Calculating Machines,* pp 67–69.

Reprinted with corrections and annotations in:
*An early program proof by Alan Turing,* L. Morris and C.B. Jones, IEEE Ann. Hist. Computing 6(2):129–143, 1984.

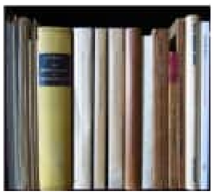
See also: *Turing and Software Verification,* C.B. Jones. Tech. Report CS-TR-1441, Newcastle University, UK, 2014.

— Alan Turing (1912–1954)

Jonathan also dealt with the important matter of where did the term "Formal Methods" come from – and what does it really mean? First of all, it seems that the actual term "Formal Methods" was used in the title of a monograph by Dutch philosopher and logician Evert Willem Beth in 1962:

***Formal Methods:***
***An Introduction to Symbolic Logic and to the Study of Effective Operations in Arithmetic and Logic*** **(1962)**

Evert Willem Beth (1908–1964), Dutch philosopher and logician



Earliest book with "formal methods" in the title?

EVERT W. BETH
FORMAL METHODS
AN INTRODUCTION TO SYMBOLIC LOGIC AND TO THE STUDY OF EFFECTIVE OPERATIONS IN ARITHMETIC AND LOGIC

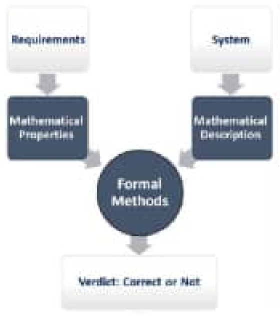
As to the etymological origins of the term "Formal Methods", Jonathan presented the following dictionary entries:

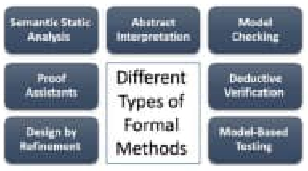
> **Formal** [L formalis, f. forma: See FORM n., -AL.] a. . . . d. <u>*Logic. Concerned with the form, not the matter of reasoning.*</u>
>
> **Methods** [ L *methodus* f. Gk *methodos* pursuit of knowledge, mode of investigation f. meta (see META-) + *hodos* way.] I Procedure for attaining an object. 1. . . . 2. A mode of procedure; a (defined or systematic) way of doing a thing. *esp*. (w. specifying wd or wds) in accordance with a particular theory or as associated with a particular person. . . . II Systematic arrangement, order. 1. . . . 3. <u>*The branch of logic that deals with the description and arrangement of arguments or propositions for the investigation or exposition of a truth*</u>,

In point of fact, Jonathan's view of formal methods is perhaps best summed up here:

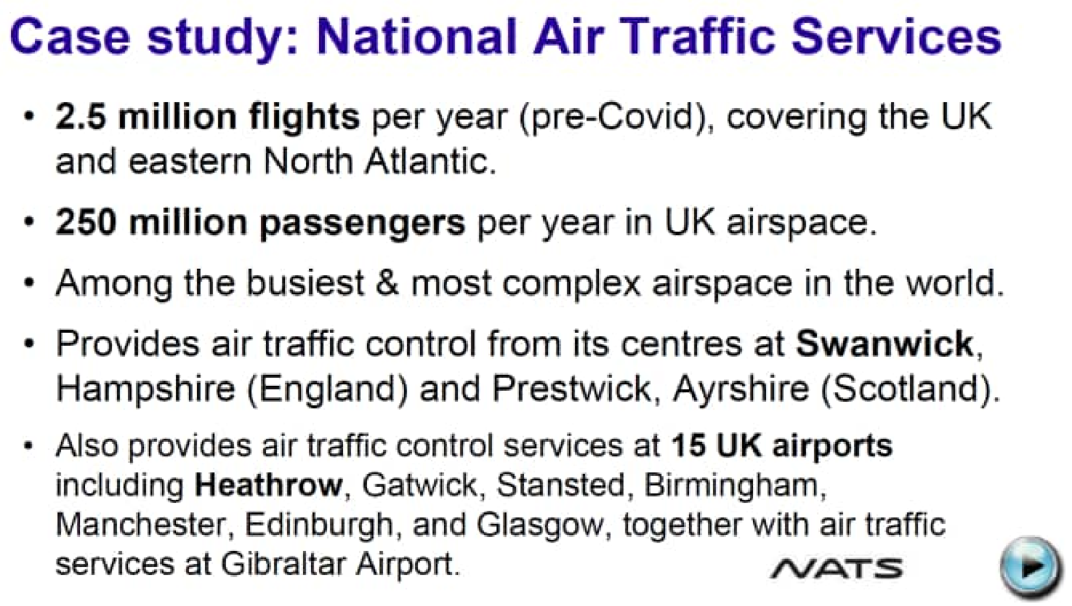
## Formal methods

- Term established by late 1970s
  - Next stage from structured design
  - Mathematical basis
- Formal specification and (optionally) proof:
  - Validation (correct specification)
  - Verification (correct implementation wrt spec.)
- But engineers *calculate* rather than prove





The core of Jonathan's talk centred upon the formal methods work done in relation to the National Air Traffic Services (NATS) for the UK, an area clearly relevant to safety-critical systems.

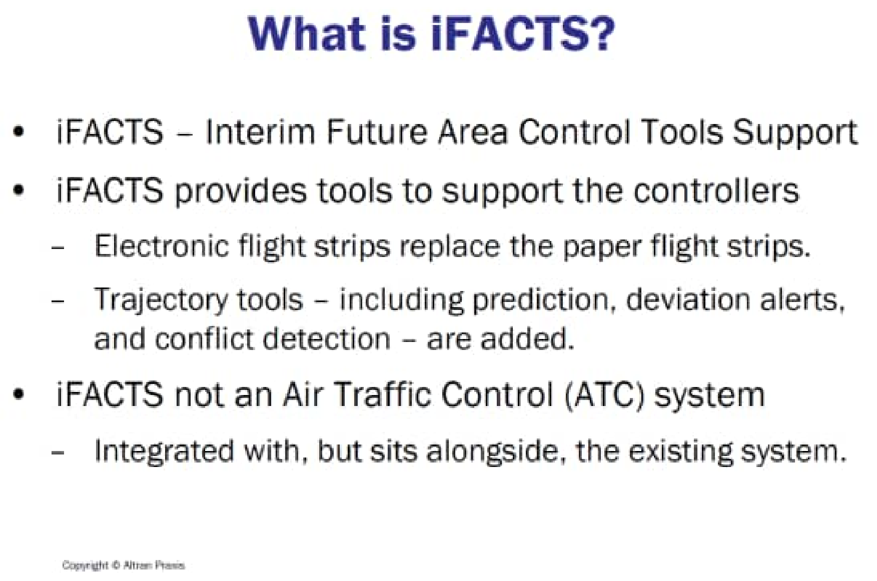


## Case study: National Air Traffic Services

- **2.5 million flights** per year (pre-Covid), covering the UK and eastern North Atlantic.
- **250 million passengers** per year in UK airspace.
- Among the busiest & most complex airspace in the world.
- Provides air traffic control from its centres at **Swanwick**, Hampshire (England) and Prestwick, Ayrshire (Scotland).
- Also provides air traffic control services at **15 UK airports** including **Heathrow**, Gatwick, Stansted, Birmingham, Manchester, Edinburgh, and Glasgow, together with air traffic services at Gibraltar Airport.

NATS

The formal methods work involved a case study for the Interim Future Area Control Tools Support (iFACTS) project \cite{iFACTS} within NATS. Although not an air traffic control system itself, iFACTS provides vital services and support to NATS and presented a scenario in which the value of formal specification surely could not be doubted.

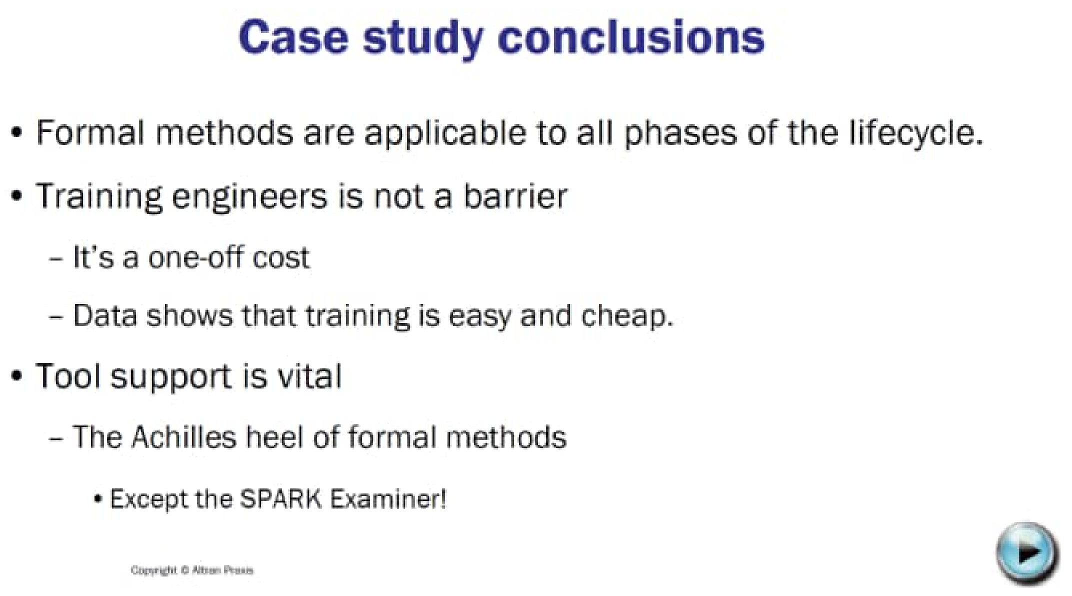


## What is iFACTS?

- iFACTS – Interim Future Area Control Tools Support
- iFACTS provides tools to support the controllers
  - Electronic flight strips replace the paper flight strips.
  - Trajectory tools – including prediction, deviation alerts, and conflict detection – are added.
- iFACTS not an Air Traffic Control (ATC) system
  - Integrated with, but sits alongside, the existing system.

Copyright © Altran Praxis

The iFACTS specification consisted of a combination of Z notation, Mathematica, state tables (a perfectly respectable formal method, that apparently required "no training") enhanced by informal, but structured, English language descriptions. The formal methods work was a major success and led to clearer specifications using a variety of formal methods, improved implementations, and enhanced test case development linking the two. Such work helped give a considerable level of assurance to the quality and effectiveness of the developed software.



## Case study conclusions

- Formal methods are applicable to all phases of the lifecycle.
- Training engineers is not a barrier
  - It's a one-off cost
  - Data shows that training is easy and cheap.
- Tool support is vital
  - The Achilles heel of formal methods
    - Except the SPARK Examiner!

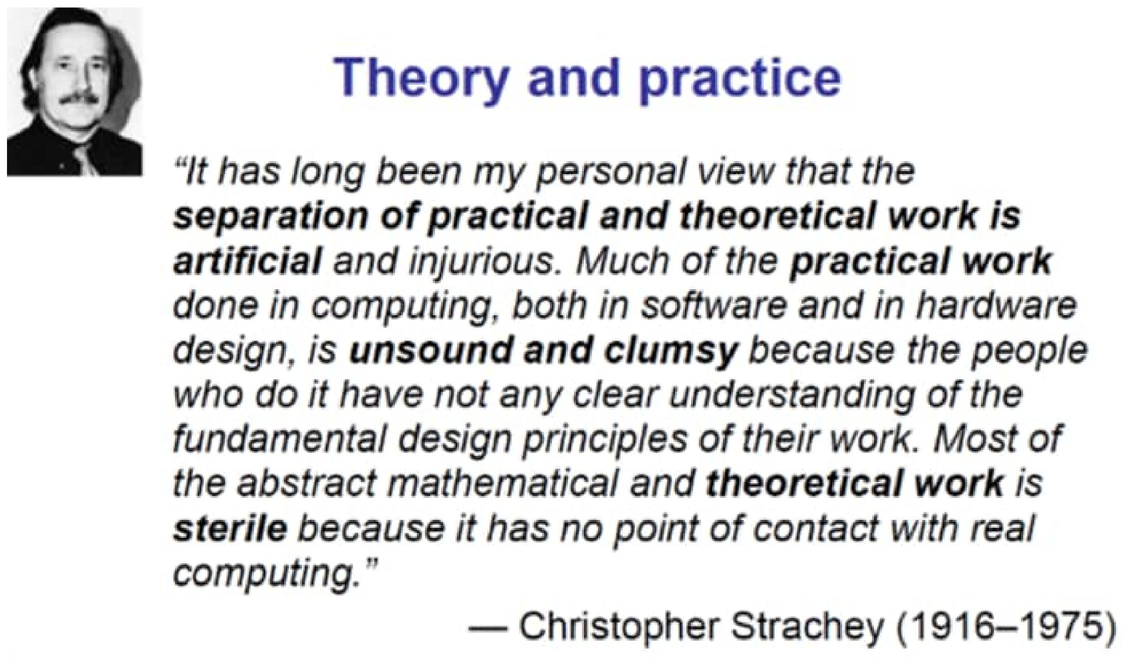
Copyright © Altran Praxis

At this point, the "Achilles heel" of applying formal methods reveals itself – lack of sufficient tool support for practical application by software engineers and developers. Although specifications and implementations can each be written, it is with the validation of their correspondence which remains the biggest barrier. As Jonathan in effect remarked, engineers prefer to calculate things, rather than prove theorems.

Moving now to the final part of Jonathan's talk where the focus turned to Jonathan's own thoughts concerning the future of formal methods. In addition to hoping that further tools support will emerge, Jonathan's view here fully embraced the idea that with software development moving into the age of AI coding and such like, formal methods needs to do so as well. The risk of not doing so is starkly obvious – to avoid being seen as irrelevant and outmoded.

In clearly accepting Jonathan's insightful and necessary observation here, it seems that combining AI with formal methods will require the development of significant additional tools support, something that demands considerable insight into how formal methods can be applied and help software designers and practitioners produce high quality software systems that meet well-understood, meaningful specifications. AI could potentially help in all sorts of positive ways with that, perhaps using the emerging class of Large Reasoning Models that use neurosymbolic techniques to helpfully go well beyond what the era of so-called "expert systems" ever achieved.

The following quote from Christopher Strachey neatly summarises the ongoing agenda of FACS:



## Theory and practice

*"It has long been my personal view that the* ***separation of practical and theoretical work is artificial*** *and injurious. Much of the* ***practical work*** *done in computing, both in software and in hardware design, is* ***unsound and clumsy*** *because the people who do it have not any clear understanding of the fundamental design principles of their work. Most of the abstract mathematical and* ***theoretical work*** *is* ***sterile*** *because it has no point of contact with real computing."*

— Christopher Strachey (1916–1975)

Strachey's remark here echoes a similar remark, reputedly said by another famous computer scientist, Donald Knuth: "The best theory is inspired by practice. The best practice is inspired by theory."

Jonathan's presentation was an excellent and inspirational "Au revoir", rather than goodbye, to FACS as Jonathan (much to his successor's relief) will remain on the FACS committee. Despite ongoing challenges with acceptance, adoption and understanding of formal methods, the work of FACS continues. . .

The seminar ended with a Q&A session. One point concerned the title of Peter Landin's seminal 1966 paper "The next **700** programming languages". The choice of 700 seems a little random, why not 70 or 7000 etc? One theory goes that the title was originally intended to be "The next **100** programming languages". But in those days of typewritten documents, the "1" got confused at the printer for a "7"!

Links to the slides and YouTube video can be found at the BCS meeting page here:

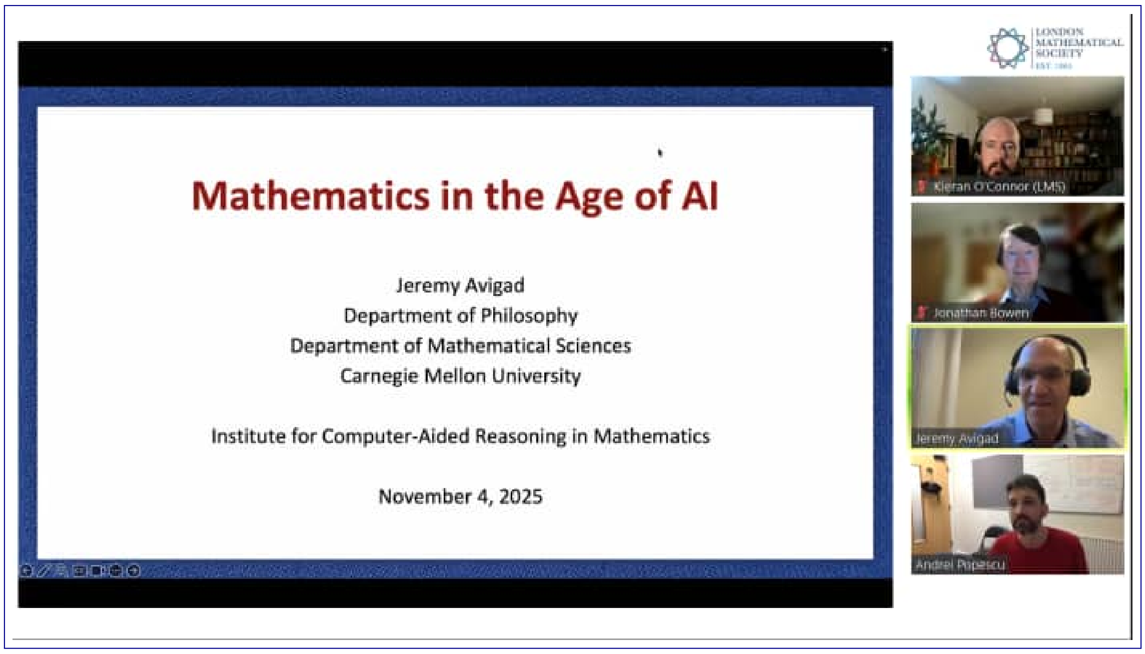
**https://www.bcs.org/events-calendar/2025/december/hybrid-facs-agm-and-peter-landin-semantics-seminar/**

# BCS FACS/LMS Evening Seminar
# **Mathematics in the Age of AI**

Jeremy Avigad
Department of Mathematical Sciences
Carnegie Mellon University

6 November 2025

(**Reported by:** Andrei Popescu)


Mathematics in the Age of AI

Jeremy Avigad
Department of Philosophy
Department of Mathematical Sciences
Carnegie Mellon University

Institute for Computer-Aided Reasoning in Mathematics

November 4, 2025

In his talk, Jeremy Avigad described how a family of technologies often grouped under the umbrella term *AI for mathematics* are beginning to reshape mathematical practice. These technologies include interactive theorem proving, automated reasoning and machine learning – recently combined in the emerging field of neurosymbolic AI.

Jeremy centred his presentation on recent success stories that he views as turning points, demonstrating that mathematics has much to gain from these developments. Examples include the mobilisation of the Lean proof assistant community to certify, in a short amount of time, a theorem from the cutting edge of contemporary mathematical research (a foundational result in the condensed mathematics programme of Scholze and Clausen); the use of a SAT solver to decide a remaining case of Keller's conjecture; the application of reinforcement learning to disprove conjectures in graph theory; and the deployment of neurosymbolic systems such as AlphaProof to win silver and then gold medals at the International Mathematical Olympiad. These efforts have also driven advances in collaboration technologies (for example through leanblueprint) and revealed the value of combining disparate tools to achieve results more efficiently and reliably – for instance, using a SAT solver for raw search power and a proof assistant for certification.

Along the way, Jeremy addressed common points of scepticism from mathematicians, particularly the view that AI tools operate mainly in finite settings whereas mathematicians care predominantly about the infinite. He argued that finiteness plays a much larger role in mathematical reasoning than is often acknowledged. In his words: "When we go looking for mathematical objects, it's not like we're shining a flashlight around the Platonic realm looking for these objects. We're always interacting with them through these finite expressions and constructions that we can use to describe them."

He also discussed the newly launched Institute for Computer-Aided Reasoning in Mathematics (ICARM) at Carnegie Mellon – one of six such institutes in the US recently funded by the NSF – whose mission is to help mathematicians use these technologies effectively, to build interdisciplinary bridges, and to support broad, equitable access.

Jeremy concluded by expressing both enthusiasm and optimism about the evolution of AI for mathematics, while emphasising the need to place people at the centre: the key question should not be "how can mathematicians use the technology" but rather "how can technology serve mathematicians".

The talk lasted one hour, followed by a 45-minute Q&A session in which Jeremy answered questions ranging from technical issues to broader societal topics, including natural-language interfaces, education, mathematical discovery and the interoperability of proof assistants.

The talk took place online via Zoom on 6 November 2025. It had 350 registrants and 175 live participants.

**YouTube video:** https://www.youtube.com/watch?v=2fxXPq8SSkQ&t=235s

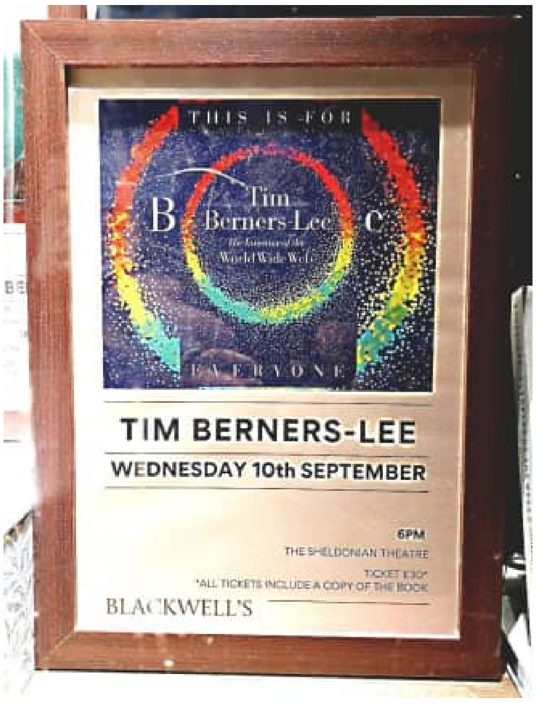
**Slides:** https://www.bcs.org/media/125dru4n
/lms_facs_seminar_with_jeremy_avigad.pdf

# Book review: *This is for Everyone* by Tim-Berners Lee

*A formal Oxonian view*

November 2025

(**Reviewed by:** Jonathan P. Bowen)



**Left:** Notice publicising a Tim Berners-Lee event in Oxford.
**Right:** Tim Berners-Lee in conversation at the Sheldonian Theatre.

Among the delights of living in Oxford are the serendipitous opportunities to attend interesting events. In early September, I was walking past Blackwell's bookshop in Broad Street and noticed a framed poster in the window (see left above) advertising a forthcoming event featuring Tim Berners-Lee, the inventor of the World Wide Web, to take place in the historic Wren-designed Sheldonian Theatre, immediately opposite. I booked online to attend the event, featuring Tim Berners-Lee in conversation with a journalist (see right above). This format worked well for the general audience.

Included in the ticket price was Tim Berners-Lee's latest book, *This is for Everyone* (published in 2025 by Macmillan [1, 4]), emphasising the altruistic nature of the Web, the technology of which he convinced CERN, the European Council for Nuclear Research in Switzerland, where he invented the Web, to give away to the world for free. This is a significant reason why the Web was so successful. At the time, the University of Minnesota tried to introduce licensing fees for competing software, the text-based Gopher system, which killed that technology very rapidly. Even Bill Gates of Microsoft realised that there was no point in competing with the Web and instead embraced it with its own Web browser.

The European Union was no wiser than many others. I remember attending a large meeting on the Web in Brussels during the early 1990s with Joe Stoy of Oxford, where an EU official intimated that the Web as it stood was just a prototype for what was to come, no doubt with the idea that the EU would be leading the way. I thought at the time how wrong he was. The Web was already expanding fast and this was the future. Joe Stoy was hopeful that Oxford could become the centre of Web developments in Europe, with Berners-Lee's Oxford connection. However that was not to be. Tim Berners-Lee had already left for MIT in the United States, where much subsequent development of the Web was centred through the World Wide Wide Consortium (W3C). He did not even attend the Brussels meeting. INRIA in France became the official European Centre for W3C, but as is often the way, most of the action was in the US in practice, especially with Berners-Lee's move there.

Before the Web and Gopher, the prevalent approach for online file access was using FTP, the File Transfer Protocol, which was more heavyweight in its approach to access. I myself maintained at "Archive Service" at the Programming Research Group (PRG) in Oxford in the late 1980s, accessible via FTP and even via automated email requests. This included formal methods information, especially for the Z notation. Once the Web became generally available in the early 1990s, with our own Web server at the PRG in the Oxford University Computing Laboratory, I moved most of the online access over to Web pages, including hyperlinks to the older FTP resources. These Web-based formal methods resources included hyperlinks to other related resources on Web servers around the world. The site [2, 3] became part of the Virtual Library, originated by Arthur Secret and Tim Berners-Lee at CERN, as a means to categorise and find information online, before search engines became widespread and reliable. With subsequent improvements in search engine technology and collaborative and altruistic encyclopedia resources like Wikipedia (which is greatly admired by Berners-Lee), the Virtual Library, and the formal methods resource that was part of it, are now just of historical interest.

The book *This is for Everyone* is a personal account of Tim Berners-Lee's invention of the Web and subsequent developments. The first chapter, *Early Days*, covers his pre-CERN experiences, including his time at Queen's College in Oxford, studying Physics as an undergraduate. I was one year behind him, studying Engineering Science at University College, immediately opposite Queen's College on the High Street. We would have attended the same sets of mathematics lectures as first-year students, which were combined for undergraduate physicists and engineers in the large lecture room at the Natural History Museum in Oxford. Our paths never crossed as students in Oxford, but I did meet him later in a conference setting, and when he returned to Oxford as a Professor. He would have just missed the comedian Rowan Atkinson, who studied for a Master's degree in Engineering Science at Queen's College, just after Berners-Lee left Oxford – and taught me all I used to know about Nyquist plots (graphical representations of frequency response in the complex plane for a control system) during practical sessions!

Tim Berners-Lee obviously enjoyed the hardware aspects of computing and well as software, relating his use of the 7400 series of binary logic integrated circuits, and designing a computer terminal with a television and an improvised keyboard from an adding machine, eventually connected to a PDP-8 minicomputer. He recalls his delight in having his first "Turing machine", a Motorola 6800 8-bit microprocessor on a single chip, in his last year at Oxford during the mid-1970s. His mathematical interest is indicated by his decoration of the Queen's College clock tower with Euler's famous identity, $e^{i\pi}+1=0$, together with fellow physics students after his final exams, in the hope of deflecting suspicion towards mathematics students.

He notes that at the time of finishing as an undergraduate with a first-class degree, he probably had not heard of Silicon Valley in California, where he might have been drawn had he done so. Instead, he worked for the electronics company Plessey in Poole, on the south coast of England, in Dorset. However, in 1980, he was offered a job at CERN, changing the course of his career and the history of the world forever. He describes the networking war of the European OSI (Open Systems Interconnection) versus TCP/IP, the American protocol adopted by the US Department of Defence, which made CERN wary of it. Of course, TCP/IP won as the universal underlying protocol of the internet internationally to this day.

Most of the book relates to Tim Berners-Lee's experience of first inventing the Web at CERN and then following its exponential adoption, expansion, and development. This covers Berners-Lee's concept of the "Semantic Web", underpinned with XML (eXtensible Markup Language), an extension of the Web's HTML HyperText Markup Language, to generalise text to data with underlying meaning. This included the Resource Description Format (RDF) language, but the approach we not adopted by companies like Microsoft, who were suspicious of the concept. Unlike the original Web, Berners-Lee's idea of the Semantic Web never took off in quite the same way, perhaps unfortunately for the formal methods community, whose ideas are, of course, based on formal semantics. The term "Semantic Web" is certainly not in general use nowadays.

Moving to the present, the Web presents significant ethical issues with the development of social media, and now the recent rapid advances in artificial intelligence (AI) technology. In 2016, Tim Berners-Lee returned to Oxford University, based at Christ Church, the largest and one of the most traditional colleges at Oxford, where Albert Einstein previously resided during his time at Oxford in the early 1930s. Berners-Lee admits that at this time, he believed that powerful AI was still a long way away. His office at Christ Church has views of both Christ Church Meadows, leading down to the River Thames, and to the rear, the Dean of Christ Church's private walled garden. In Victorian times, this garden was frequented by the Dean's daughter Alice Liddell, of "Alice in Wonderland" fame, and the mathematician Charles Dodgson, also known as the author Lewis Carroll. Tim Berners-Lee and I both have daughters called Alice, perhaps (or in my case certainly) due to this Oxford connection.

Also based at the Oxford University Department of Computer Science (formerly the Computing Laboratory), Tim Berners-Lee has been able to consider the possibility of individuals owning their own data, and providing access under their control, rather than the current situation of large hi-tech corporations owning such data for their own financial ends. When just back in Oxford, Tim Berners-Lee won the ACM Turing Award, computer science's nearest equivalent of the Nobel Prize. He informed his parents, both of whom had known Alan Turing personally as early programmers at Manchester University. Vint Cerf, co-inventor of the TCP/IP internet protocols and also a previous Turing Award winner, was present at the award ceremony in San Francisco. Cerf had to whisper to Berners-Lee that he should wind up his speech so they could all have dinner!

In his recent thoughts on AI and machine learning, Berners-Lee bemoans the role of *Cambridge Analytica* in influencing the UK vote on Brexit, and the US vote for Trump, winning the electoral college majority, but not the overall popular vote, in 2016. As a fellow Oxonian, I can only agree!

The final chapters of the book reflect on the development of the Web with hindsight, including the issues of social media and AI. The book ends on an optimistic note, but it remains to be seen if this is warranted. Certainly, for formal methods and mathematics in general, the opportunity for AI to aid in proofs, most likely in an interactive manner, promises to help reduce the drudgery of large proofs, especially as the availability of large language models (LLMs) covering existing proofs increases. Prover software systems like the Lean proof assistant are already using this to effect, with improvement likely in the coming years. In 2025, approaches based on the Chinese DeepSeek generative AI (GenAI) software and Lean to aid theorem proving are being investigated [5, 6]. More similar developments can be expected in the future. So the advances in AI could well be a synergistic opportunity for the field of formal methods in the years ahead.

## References

[1] Berners-Lee, T., with Witt, S. (2025). *This is for Everyone*. Macmillan.

[2] Bowen, J.P. (1994). The World Wide Web Virtual Library: Formal Methods. *Virtual Library*. **https://formalmethods.fandom.com**

[3] Bowen, J.P. (2011). Formal Methods Online Publication Resources. *FACS FACTS*, **2011**(1):16, December. BCS. **https://www.bcs.org/media/3082/facs-dec11.pdf**

[4] Pan Macmillan (2025). Stephen Fry and Sir Tim Berners-Lee | This is For Everyone Audiobook. *YouTube*. **https://www.youtube.com/watch?v=Hs_KqUQ6LGI**

[5] Huang, S., Song, P., George, R.J., and Anandkumar, A. (2025). LeanProgress: Guiding Search for Neural Theorem Proving via Proof Progress Prediction. *arXiv*. doi:10.48550/arXiv.2502.17925

[6] Lu, J., Emond, K., Yang, K., Chaudhuri, S., Sun, W., and Chen, W. (2025). Lean Finder: Semantic Search for Mathlib That Understands User Intents. *arXiv*. doi:10.48550/arXiv.2510.15940

## Forthcoming Events

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 Miyazawa 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

**Keith Lines**
*FACS Chair*

**Roger Carsley**
*Treasurer*

**Alvaro Miyazawa**
*Secretary and Seminar Organiser*

**Jonathan Bowen**
*Newsletter Co-editor*
*(Emeritus FACS Chair and BCS Liaison)*

**John Cooke**
*Publications*
*(Emeritus Treasurer)*

**Margaret West**
*Inclusion Officer*

**Tim Denvir**
*Emeritus Newsletter Co-editor*

**Brian Monahan**
*Newsletter Co-editor*

**Andrei Popescu**
*LMS Liaison*

**Ana Cavalcanti**
*FME Liaison*

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

BCS-FACS
c/o Keith Lines (Chair)
National Physical Laboratory, Teddington, TW11 0LW
Email: **keith.lines@npl.co.uk**

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 subscriber. An archive of messages is accessible under:

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

including facilities for subscribing and unsubscribing.

The additional list: **facs-event@jiscmail.ac.uk** 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.