Experiences from the European ProCoS Projects: Provably Correct Systems

JONATHAN BOWEN, School of Computer Science and Digital Technologies, London South Bank University, London, United Kingdom of Great Britain and Northern Ireland, Museophile Limited, Oxford, United Kingdom of Great Britain and Northern Ireland, and RISE, Southwest University, Chonqing, China
MARTIN FRÄNZLE, Department für Informatik, Carl von Ossietzky Universität Oldenburg, Oldenburg, Germany
ERNST-RÜDIGER OLDEROG, Department für Informatik, Carl von Ossietzky Universität Oldenburg, Oldenburg, Germany
DINES BJORNER, Technical University of Denmark, Lyngby, Denmark
MICHAEL HANSEN, Department of Applied Mathematics and Computer, Technical University of Denmark, Lyngby, Denmark
HANS LANGMAACK, Christian-Albrechts-Universität zu Kiel, Kiel, Germany
ZHIMING LIU, RISE, Southwest University, Chongqing, China
URSULA MARTIN, Mathematical Institute, University of Oxford, Oxford, United Kingdom of Great Britain and Northern Ireland

This article presents the collaborative European ESPRIT ProCoS projects on "Provably Correct Systems", and associated initiatives of the 1990s. The influence of the projects in the field of formal methods is also discussed. A general overview of the projects is provided, together with a number of reminiscences by those involved with the projects, including the influence on the subsequent careers of participants. The projects addressed the issues of connecting formal approaches at different connected levels of formality, including requirements, specification, and compilation down to machine code and even directly into hardware. The investigations were based on a representative subset of the programming language Occam, which was subsequently extended by elements indicative of particular problems of compilation, yet not found in Occam,

J. Bowen, M. Fränzle, and E.-R. Olderog authors edited this paper; all authors provided reminiscences.
Authors' Contact Information: Jonathan Bowen (corresponding author), School of Computer Science and Digital Technologies, London South Bank University, London, Southwark, United Kingdom of Great Britain and Northern Ireland, Museophile Limited, Oxford, United Kingdom of Great Britain and Northern Ireland, RISE, Southwest University, Chonqing, China; e-mail: jpbowen@gmail.com; Martin Fränzle, Department für Informatik, Carl von Ossietzky Universität Oldenburg, Oldenburg, Niedersachsen, Germany; e-mail: martin.fraenzle@uni-oldenburg.de; Ernst-Rüdiger Olderog, Department für Informatik, Carl von Ossietzky Universität Oldenburg, Oldenburg, Niedersachsen, Germany; e-mail: Ernst.Ruediger.Olderog@informatik.uni-oldenburg.de; Dines Bjorner, Technical University of Denmark, Lyngby, Capital Region of Denmark, Denmark; e-mail: bjorner@gmail.com; Michael Hansen, Department of Applied Mathematics and Computer, Technical University of Denmark, Lyngby, Capital Region of Denmark, Denmark; e-mail: mire@dtu.dk; Hans Langmaack, Christian-Albrechts-Universität zu Kiel, Kiel, Schleswig-Holstein, Germany; e-mail: hl@informatik.uni-kiel.de; Zhiming Liu, RISE, Southwest University, Chongqing, Chongqing, China; e-mail: zhimingliu88@swu.edu.cn; Ursula Martin, Mathematical Institute, University of Oxford, Oxford, England, United Kingdom of Great Britain and Northern Ireland; e-mail: Ursula.Martin@maths.ox.ac.uk.

ACM 1433-299X/2026/3-ART00
https://doi.org/10.1145/3803555

and the related Transputer microprocessor. In practice, two of the most important and enduring results after the projects were Duration Calculus and Unifying Theories of Programming, both subfields of formal methods, with associated communities of researchers and practitioners.

CCS Concepts: • **Software and its engineering → Formal software verification**; *Semantics*; • **Social and professional topics**;

Additional Key Words and Phrases: Formal methods, software development, history of computing

ACM Reference Format:
Jonathan Bowen, Martin Fränzle, Ernst-Rüdiger Olderog, Dines Bjorner, Michael Hansen, Hans Langmaack, Zhiming Liu, and Ursula Martin. 2026. Experiences from the European ProCoS Projects: Provably Correct Systems. *Form. Asp. Comput.* 00, JA, Article 00 (March 2026), 29 pages. https://doi.org/10.1145/3803555

1 Introduction

A major event for fundamental computer science research in 1987 was the idea "from Brussels", i.e., from the EU, to launch its strategic programme called "ESPRIT" (European Strategic Programme for Research and Development in Information Technology) and within that framework offer funding opportunities for European **Basic Research Actions** (**BRA**), which aimed at promoting scientific cooperation and exchange across Europe. Given that funding framework, it of course required an additional impetus for forming a European consortium and submitting a funding proposal that finally led to the "Provably Correct Systems" project (ProCoS for short): During an international symposium in Brussels, Tony Hoare gave a lecture about his stay as a guest researcher with J Moore at **Computational Logic Inc.** (**CLI**) in Austin, Texas, and about Moore's so-called "stack" of translation correctness proofs from the Pascal-like higher programming language Micro-Gypsy to assembly language, machine language, and finally hardware, the latter with a gate-level interpretation of Boolean operations [4]. All proofs were carried out using the Boyer-Moore Theorem Prover. The correctness of the prover was assumed, based on its extensive testing, yet no formal certificates were established for the proof engines themselves and the various, quite complex and extensive, much less thoroughly tested theories, embedded in them. Due to these dependencies on an unverified theorem-proving environment, the stack, though constituting an impressive advance in end-to-end verification of a software engineering infrastructure at its time, was still not completely verified.

For Tony Hoare and Dines Bjørner (see Section 2.1), this gave rise to the idea of rigorously formalizing all relevant correctness properties and to prove them comprehensively and mathematically. As such, proof might well adopt convenient tools like, e.g., algebraic rewriting rules concerning, e.g., statement sequences in programs akin to the rewriting rules employed in theorem proving, albeit only if the rewriting rules had previously been verified against lower-level formalizations of program semantics in axiomatic, denotational, or structural operational style. This was meant to extend the "stack" into a grounded "tower" featuring a deeper and more solid foundation.

Overarching software and hardware system development and securing both, as well as all the interfaces involved, by formal proof, this "tower" was aimed at achieving maximum safety in building IT systems. This is how the name "**Provably Correct Systems**" (**ProCoS**) came about for two consecutive ESPRIT BRA projects and an associated international working group. The projects and the working group were major collaborative European initiatives aimed at investigating the comprehensive use of formal methods in a coordinated manner.

The original partner sites on the first ProCoS project (ESPRIT BRA 3104, from May 1989 to February 1992), coordinated by the Technical University of Denmark (DTU, Danmarks Tekniske Universitet, formerly Danmarks Tekniske Højskole, or DTH), were [6, 7, 119]:

— The Technical University of Denmark – DTH (Denmark).
— Christian-Albrechts-Universität zu Kiel (Germany).
— University of Oxford (United Kingdom).
— Royal Holloway and Bedford New College (United Kingdom).
— Århus Universitet (Denmark).
— University of Manchester (United Kingdom).

The ProCoS II project (ESPRIT BRA 7071, duration 1992–1995) was coordinated by Oxford University and the sites consisted of [17, 43, 106]:

— University of Oxford (coordinator, United Kingdom).
— Carl von Ossietzky Universität Oldenburg (Germany).
— Christian-Albrechts-Universität zu Kiel (Germany).
— The Technical University of Denmark – DTU (Denmark).

These were to become the core sites for the legacy of ProCoS overall. The subsequent ProCoS-WG Working Group (1994–1997) included 25 partners from both academia and industry [18, 44].

ProCoS supported several conferences, including ZUM'94 [33], FTRTFT'94 [94, 118], ZUM'95 [39], and ZUM'97 [40]. Many ProCoS personnel attended the Dagstuhl Seminar on the Steam Boiler Control Specification Problem, 5–9 June 1995 [1]. ProCoS tutorials were presented at FME'93 [30], RTS'93 [31], FTRTFT'94 [42], and FME'96 [95]. The associated ProCoS Working Group also organized a number of meetings, culminating with one at the University of Reading (United Kingdom) in 1997, co-located with the ZUM'97 conference [40].

These conferences were complemented by a series of regular workshops being organized by ProCoS initiative partners and assembling the members of the ProCoS projects and the ProCoS Working Group:

— DTU: 27–30 October 1989, Svaneke, Bornholm, Denmark.
— Kiel: 23–27 April 1990, Malente, Germany.
— Oxford: 23–24 July 1990, St Edmund Hall, Oxford, UK (review).
— DTU: 3–7 December 1990, Golf Hotel, Viborg, Denmark.
— Oxford: 25–27 June 1991, St Benet's Hall, Oxford, UK.
— DTU: 14–18 October 1991, Gl. Avernæs Hotel, near Odense, Fyn, Denmark (Symposium).
— DTU: 19–22 October 1992, Lyngby, Denmark.
— Oxford: 5–8 January 1993, St John's College, Oxford, UK.
— Oldenburg: 20–23 September 1993, Wiefelstede, near Oldenburg, Germany.
— DTU: 17–21 January 1994, Lyngby, Denmark.
— Oxford: 10–12 January 1995, Computing Laboratory, Oxford, UK.
— DTU: 21–23 August 1995, Hotel Marina, Vedbæk, Denmark.
— Oldenburg: 11–13 March 1996, Oldenburg, Germany
— Reading: 7–9 April 1997, University of Reading, UK (with ZUM'97).

ProCoS started in parallel with the development of the World Wide Web. Project-related web resources were established at several ProCoS sites, with the lead resources at Oxford University. The LaTeX document preparation system was used for project documents, with a ProCoS "style" file used for project documentation, including a standardized front cover with a project logo (see Figure 1). A ProCoS bibliography using BibTeX was maintained online, and an updated version of this was used for some of the references in this current article. A "ProCoS" e-mail list was established on the United Kingdom JISCmail facility that continues to this day with posts related to formal methods.

This section has provided a summary of ProCoS-related activities during the 1990s, as background and reference in the rest of the article. The next section provides some more detailed reports on activities during and after the ProCoS projects, from a variety of individual perspectives.

2 Reminiscences

In this section, we present some individual reminiscences by participants in the ProCoS projects. This includes some details of personnel at each of the main project partners. As well as experiences during the ProCoS projects themselves, some of the effects on subsequent careers are also covered.

Fig. 1. ProCoS project logo.

2.1 Technical

University of Denmark, by Dines Bjørner

The Start: An IFIP WG2.3 Meeting. On 13 November 1987, I attended an IFIP (International Federation for Information Processing) WG2.3 (Working Group 2.3: Programming Methodology) meeting at the Château Du Point d'Oye in the Vallon part of Belgium. I had presented a topic for discussion. At IFIP WG2.3 meetings, one did not present articles: one suggested a topic for discussion, say 5–10 minutes, and, under the skillful guidance of the chairman, the members then decided whether or not it was worth listening to that topic! It was about the programming methodology used in the Dansk Datamatik Center development of the CHILL and Ada compilers. My topic centered around a software development graph (see Figure 2).

Each of the boxes, from a programming methodology point of view, represents a formal specification according to some theory: Strachey and Scott, McCarthy and Painter, Reynolds, and many others. And each of the arrows represents one or another form of reification. From a theoretical computer science point of view, boxes "stood" for some kind of algebra, and arrows for algebraic injections.

After the talk, there was a coffee break, but Tony Hoare held me back, asked for my notebook, and then wrote in it (see Figure 3). In his subtle style, Tony then wondered: "could this, perhaps, be a basis for an ESPRIT project?" Yes, was my answer. I then worked out both a group of colleagues and the basis for a proposal. Jonathan Bowen (see Section 2.3) provided the wonderful logo (see Figure 1). And so it went!

At the March 2015 ProCoS gathering in London, in my talk on 9 March [5], 27½ years later, I displayed the same image from my notebook – and, again, afterward, Tony Hoare, asked for the notebook and wrote on the right-hand side page, the bottom two blue lines (see Figure 3):

I'm still trying to understand this.
Tony Hoare 9 March 2015

Chinese Connections. Independent of all this, I had Zhou Chaochen visit my DTU institute some winters, for three months duration at a time, since 1982. He was also invited, with his family, for half a year. They were to come on 1 July 1989. [1] At the same time, we were about to obtain ESPRIT funding for the ProCoS proposal. So I sat down, spent perhaps a whole day on 5 June formulating a 2–3-page fax, which I then sent to two of my "connections" in Beijing: one to Prof. Wang Shengwei, director of the **People's Liberation Army** (**PLA**) Research Center and one to Prof. Xu Kongshi, Director of the Chinese Academy of Science's Institute for Computing Technology, a forerunner of today's Institute of Software. Prof. Wang Shengwei was a close confidante of the then Minister for Science, Technology and Industry of the PLA: Ding Henggao – with whom together with his wife, Madame Nie Li, daughter of the military leader of the Long March, Marshal Nie Rhongzhen (1899–1992), I have had dinner, in Beijing, as well as with him and some of his staff, in my home

[1] Some of original text has been removed here with consent by the author.

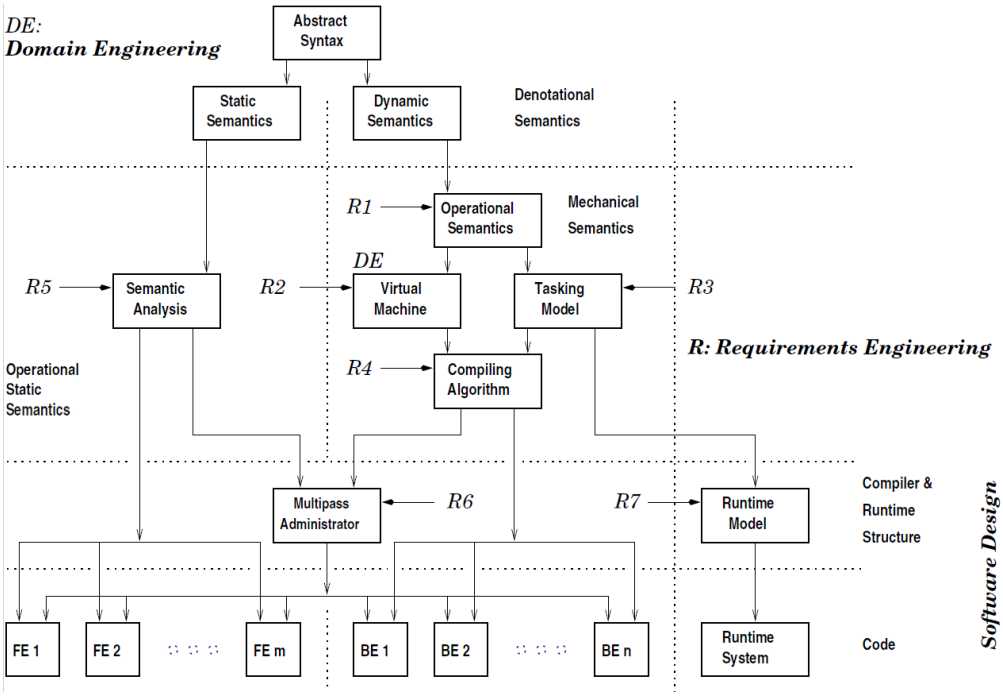
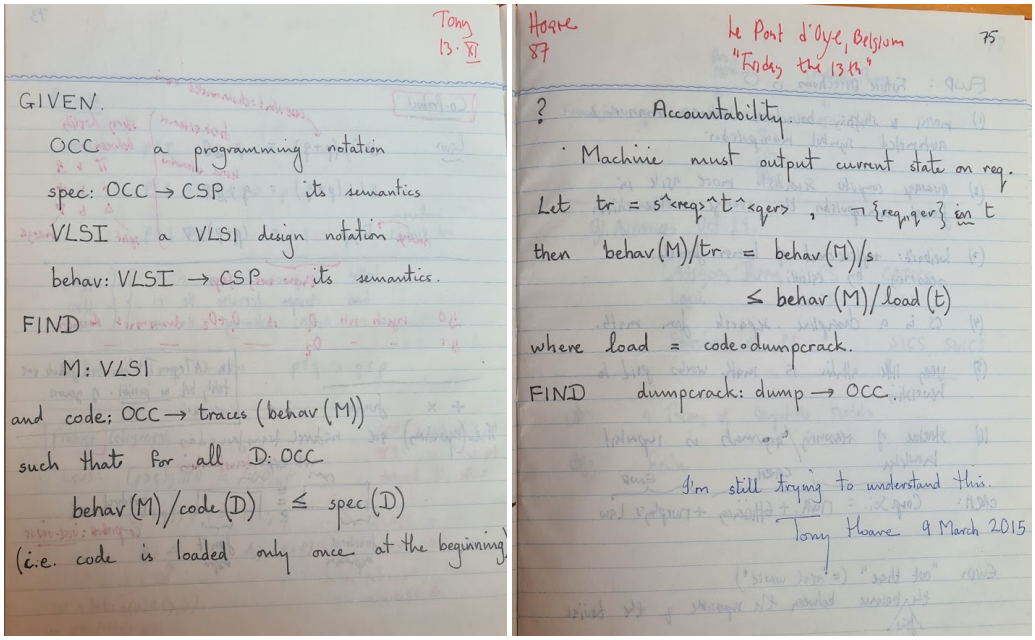


Fig. 2. Software development graph.

in Denmark! In the fax, I expressed our happiness at soon receiving the Zhou family in Denmark, that the ProCoS project had been approved, and that a main focus of its research was that of programming methodologies for real-time, safety-critical systems. I received two reply faxes within 24 hours: "Oh, yes, the whole family would come." And they came. Instead of six months, it became four years!

The Viborg Meeting. One of the ProCoS project meetings took place at the Golf Hotel in Viborg, Denmark. My colleague Prof. **Erling Vagn Sørensen** (**EVS**) gave a talk on how electric switching circuits exhibited "spikes" and how that affected their logic. It was a fine, clear talk – which obviously inspired Zhou Chaochen, Tony Hoare, and Anders Peter Ravn. After EVS's talk, there was a break before lunch. During that break, the ***Duration Calculus*** (**DC**) was conceived. In a neighboring conference room, the three spent an hour in front of a whiteboard! Soon after, *A Calculus of Durations* [185] appeared.

UNU/IIST. In May 1991, I was headhunted to become the first and founding UN Director of the upcoming UN University's International Institute for Software Technology, located in Macau. I served for five years: 1 July 1982 – 30 June 1997. I was able to have Søren Prehn with UNU/IIST from 1 August 1992 and Zhou Chaochen from 1 September 1992. During the years 1993–1997, UNU/IIST hosted more than 60 young scholars from China, Vietnam, the Philippines, India, Mongolia, both Koreas (!), Nepal, Argentina, Cameroon, Nigeria, and so on. UNU/IIST's charter was to enable developing countries to become developers of sophisticated software. I chose to do so by propagating state-of-the-art software development methods and their underlying science. We identified the UNU Fellows, as they were, through giving two-week courses at universities in these countries and selected those participants who asked probing questions.



GIVEN

OCC a programming notation
spec: OCC → CSP its semantics
VLSI a VLSI design notation
behav: VLSI → CSP its semantics

FIND

M: VLSI

and code: OCC → Traces (behav(M))
such that for all D: OCC

behav(M)/code(D) ≤ spec(D)

(i.e., code is loaded only once at the beginning)

? Accountability

Machine must output current state on *req*.

Let $tr = s\hat{}\langle req\rangle\hat{}t\hat{}\langle qer\rangle$, $\neg\{req, qer\}$ in t
then behav(M)/tr = behav(M)/s
≤ behav(M)/load(t)
where load = code ∘ dumpcrack.

FIND dumpcrack: dump → OCC.

Fig. 3. Two pages from Dines Bjørner's notebook.

My staff, besides Zhou Chaochen and Søren Prehn, were such computing scientists as Chris W. George, Dang Van Hung, Richard Moore, Tomasz Janowski, and Kees Middelburg. I, and later Chris George, would lead two to three of these, as a Programming Methodology Group, and Zhou Chaochen, two to three as a more Theoretical Computer Science Group. Usually, the UNU Fellows would arrive by September, during these years, and be exposed to a month of lectures by all of us. During that month, they were then assigned to one of the groups, and a study topic was tentatively identified.

Zhou Chaochen's group primarily worked around Duration Calculus topics. My and Chris George's group was involved with domain-specific topics: *Run-time Scheduling of Trains* (China), *Ministry of Finance Software* (Vietnam), *Multi-script Editing* (Mongolia), *Radio Telephony* (The Philippines), and so on. Over these and the following years, where first Zhou Chaochen, and, finally, Mike Reed, succeeded me as Directors, several guest lecturers spent typically six months

at UNU/IIST. Also at UNU/IIST, we eventually hosted both an IFIP WG2.3 meeting and a ProCoS Working Group meeting! Today, we can look back and conclude that UNU/IIST, with its programme of formal methods, domain-specific software development, and Duration Calculus has been the origin of many studies and course directions at several Chinese and some Indian universities: notably at **East China Normal University** (**ECNU**) in Shanghai, with He Jifeng, who also spent 1998–2005 at UNU/IIST, after working at Oxford University with Tony Hoare.

Looking Back and Acknowledgments. There is no longer an identifiable core of colleagues at my former institute (DTU) who work as we did in the 1990s, including on ProCoS. My "stint" at UNU/IIST appears to have brought an end to that. I, anyway, thank my former colleagues, Hans Henrik Løvengreen, Michael Reichhardt Hansen (see Section 2.6), and Hans Rischel for loyal and kind collaboration. I am also grateful to Chris W. George, Tony Hoare, Dang Van Hung, Tomasz Janowski, Hans Langmaack (see Section 2.2), Kees Middelburg, and Richard Moore, for inspiring science and UN-dedicated collaboration.

2.2 Christian-Albrechts-Universität zu Kiel, by Martin Fränzle and by Hans Langmaack

Being interested in translator correctness from the very onset of compilation of high-level programming languages to machine code [80], Hans Langmaack and his research group at Christian-Albrechts Universität Kiel became in the late 1980s intrigued by Computational Logic's "stack" of translation correctness proofs from Micro-Gypsy, a Pascal-like programming language, via assembly language and machine language to hardware. As this major achievement was enabled by J Moore's consistent use of novel interactive theorem-proving technology, namely the Boyer-Moore Theorem Prover [55], work in the research group was rapidly expanded to construction [56, 58] and application [176] of interactive theorem proving technology.

At the same time, it became clear that such an approach could significantly enhance, yet not fully assert translator correctness due to its dependency on complex and ultimately unverified proof engines, on extensive axiomatizations of the highly non-trivial logical theories incorporated in such provers, and on recourse to unverified compiler components necessary, e.g., during initial compiler bootstraps. The idea of establishing an ESPRIT Basic Research Action addressing system verification by rigorously formalizing all relevant correctness properties and proving them comprehensively and mathematically therefore matched the interests of the group ideally. Because Dines Bjørner had held the so-called Danish Chair at the University of Kiel in 1981 and Ernst-Rüdiger Olderog had visited Oxford, there gaining important inspiration for his habilitation thesis "Nets, Terms and Formulas" in 1988 (published as a book in 1991 [149]), Hans Langmaack's group at Kiel and Ernst-Rüdiger Olderog's at Oldenburg, both located in northern Germany, were invited to take part in this ESPRIT Basic Research Action 3,104 ProCoS, with Kiel focusing on end-to-end compiler correctness.

It was then, in 1989, with the start of the ProCoS projects, that three young students, well before attaining their diplomas (at that time, the German equivalent of an MSc) were invited into Hans Langmaack's office for an appointment featuring an undefined, apparently confidential topic. The three, namely Martin Fränzle, Yassine Lakhnech, and Markus Müller-Olm, were puzzled and surprised by both the invitation and the fact that only a few hours later, they were the youngest members of the ProCoS team to be formed. As research students, they joined a team of established researchers at Christian-Albrechts-Universität Kiel, namely Debora Weber-Wulff, Bettina Buth, and Karl-Heinz Buth, all three full-time research assistants working toward their PhD titles at the time, Dr. Burghard von Karger, who was working on his habilitation, and of course Prof. Dr. Hans Langmaack. The profound impact of ProCoS on all their careers cannot be denied, with five of the young researchers later also becoming professors in formal methods or related areas of computer science.

But ProCoS has not only shaped their careers, but also their perspective on how to do science, as all their meandering journeys through the beautiful territories of computer science started with a crash course in demanding and rewarding scientific travel: for the kick-off meeting of ProCoS, we went by a series of trains from Kiel to Copenhagen, then entered an overnight ferry to Rønne on the island of Bornholm, got to know our new project colleagues right on that ferry (and simulated a bit of scientific work ibidem), and after a bus transfer, ended up in a beautiful hotel in Svaneke on Bornholm, which had the most splendid buffets we had ever eaten. No wonder that most of us stayed in academia and formal methods ever since!

The Years 1989–1995 of ProCoS I and II. In ProCoS I, the focus of the team at Kiel was on contributing a fully verified compiler infrastructure to ProCoS' interconnected and mutually dependent network of correctness arguments, which was inspired by – and extended – the "stack" pioneered by Computational Logic Inc., see [4]. The resulting "tower" constituted a considerable broadening of the then prevalent perspective on formal methods beyond program verification, emphasizing that program verification against a reference semantics of the particular programming language is crucial, yet by no means sufficient: compilers, run-time environments, device drivers, and operating system services, or even the underlying hardware, may all introduce deviations from the reference semantics, and thus deserve verification with the same scrutiny as the application programs [117].

While a compiler will generally be (re-)used significantly more frequently than an individual application program, rendering "proven in use" arguments somewhat more reliable for compilers than for application programs, multiple long-standing errors in compilers and compiler optimizations had surfaced by this time. Comprehensive compiler verification thus seemed a must, and it soon became clear that this endeavor goes much deeper than code generator verification confirming identity between the reference semantics of the particular programming language and the machine semantics of the code generated from it. Compiler front-ends, in their strive to analyse the syntactic structure of the source program and to deploy it as an abstract data structure for the code generator, may modify the semantics of the compiled code, just like code optimizations pursued by the compiler could do. More subtly, even the compilation of the compiler itself, which is usually written in a high-level programming language rather than directly in machine code, may modify all the aforementioned compiler components, thus potentially invalidating even the most rigorous proofs of front-ends, code generators, and code optimizers [57].

These problems were consequently addressed in a series of interrelated scientific approaches pursued at Kiel, most of them leading to profound PhD and habilitation theses: Bettina Buth and Karl-Heinz Buth, in their PhD theses [58] and [56], established interactive verification technology based on term rewriting for mechanically proving the semantic alignment between operational and denotational semantic descriptions. Debora Weber-Wulff [176] exploited interactive theorem proving for the verification of compiler front-ends, while Martin Fränzle and Markus Müller-Olm addressed code-generator correctness, culminating in Markus Müller-Olm's PhD thesis [146] applying a refinement-algebraic approach to an advanced imperative programming language incorporating not only concurrency and synchronous communication, but also commands for directly controlling hard real-time behavior of code segments. The challenge of tackling the recursive digression of the compiler implementation itself relying on host languages and the correctness of their respective implementation was solved by adopting a bootstrapping approach to compiler implementation and elucidating the interdependency arguments implied by such a bootstrap [57].

These results secured the whole code generation process from a considerable extension of the Occam programming language [143] all the way down to actual machine code for the Transputer [177]. Our most prominent extension of Occam was the addition of expressive real-time control

mechanisms able to strictly confine execution times of whole code segments, both from below and from above. The ability of the code generator to translate these hard real-time constraints expressed at the source-code level and invariably enforce them at the machine level, as well as our ability to rigorously verify this timing-preserving code generator, hinges on theoretical foundations concerning the semantics of embedded real-time systems, which were deepened during the second phase of ProCoS.

Burghard von Karger, in his habilitation thesis that emerged from the ProCoS projects in general, and an extended research stay in Tony Hoare's group at Oxford in particular, linked the algebraic approach of program design and verification advocated by Tony Hoare [103] with the deductive style based on temporal logics that was suggested by Amir Pnueli [159]. Therefore, he had to embed predicate algebras (more commonly known as Boolean algebras), such as relation algebra and the algebra of predicate transformers, together with temporal logics, into a common semantic framework called Temporal Algebra [174, 175]. Martin Fränzle took the very expressive metric temporal logic DC [185] that had been developed by Zhou Chaochen, Tony Hoare, and Anders P. Ravn within the ProCoS project, and investigated its suitability as a source language for automatic synthesis of embedded real-time systems from abstract specifications. The satisfiability problem of DC was known to be undecidable over dense time [184], but through a series of physics-inspired semantic models more exactly capturing the behavior of digital hardware over dense time and novel decidability and approximability results derived from these, automatic synthesis procedures from DC directly to digital real-time hardware could be devised [69, 72] and expressive decidable subsets of DC with tractable complexity identified [71]. Subsequently, these principles were transferred to the field of hybrid discrete-continuous systems [70], which ProCoS also helped to establish [82, 187], where they sparked research on robust automatic verification procedures providing verdicts almost always, despite general undecidability, namely whenever a system property is not invalidated by infinitesimally small variations in the parameters of the hybrid system.

With its "tower" of correctness arguments spanning from physical hardware to high-level programming, ProCoS rapidly became a landmark project of comprehensive formal methods, leading to intense scientific exchange inside and outside the ProCoS consortium, the latter witnessed by Kiel's and ProCoS's leading role in organizing the FTRTFT'94 conference on Formal Techniques in Real-Time and Fault-Tolerant Systems at Lübeck, Germany [118].

The Years After. The stimulating atmosphere of the ProCoS projects led to long-standing scientific cooperations and personal friendships. Building on the research performed in ProCoS, the group in Kiel collaborated with universities of Karlsruhe (Gerhard Goos) and Ulm (Friedrich W. von Henke) in the project Verifix (1995–2002) funded by the German Research Foundation (DFG) [78]. Its aim was the verification of compilers for realistic programming languages. Arguing about compiler correctness started from a compiling specification describing the correspondence of source and target language in formal terms. The project used abstract state machines to formalize this correspondence.

Martin Fränzle later on moved to the Technical University of Denmark for his first professorship, where he collaborated extensively with Michael R. Hansen [73–76, 148] (see Section 2.6) and also with Hans-Henrik Løvengren during the years 2002–2004. These collaborations endured Martin's move to Oldenburg University in 2004 through regular mutual visits and in particular a Velux Visiting Professorship in 2006–2008 as well as a sabbatical in 2023–2024 at DTU, all accompanied by Martin's wife Barbara. Amiable lunches at the homes of Ulrike and Michael R. Hansen and of Annemette Lind and Anders P. Ravn, as well as meeting Kari and Dines Bjørner (author of Section 2.1) at home or in as cosy refined restaurants, were always an integral part of these visits. At Oldenburg, collaboration with the former ProCoS peers Ernst-Rüdiger Olderog (see Section 2.5)

and Henning Dierks started immediately as if ProCoS had never ended. Martin Fränzle also visited Oxford and the Chinese Academy of Sciences, to which Zhou Chaochen had returned, repeatedly after the end of the ProCoS projects.

Martin Fränzle is still collaborating intensely with Naijun Zhan from the ProCoS-WG Working Group and joined the SETSS School on Engineering Trustworthy Software Systems in 2024 [32], which was co-organized by the ProCoS alumni Zhiming Liu (see Section 2.7) and Jonathan Bowen (cf. Section 2.3). These visits were always as scientifically stimulating as enjoyable due to long-lasting friendships, and were, of course, balanced by visits from Kgs. Lyngby, Denmark, and Beijing, PR China, to Oldenburg, as well as by PhD student exchange.

Particularly memorable events decades later, highlighting the long-standing effects of the ProCoS projects, were the 25 years of ProCoS reunion at the office of the **British Computer Society** (**BCS**) in London [101], where almost all former members of Kiel's ProCoS team reconvened and met their international peers and colleagues from the ProCoS projects and the ProCoS-WG Working Group, as well as the 2025 Festkolloquium on the occasion of Martin Fränzle's 60th anniversary, after some 35 years, still featuring the former ProCoSians Hans Langmaack with his wife Annemarie, Jonathan Bowen, Michael R. Hansen, Naijun Zhan, and Ernst-Rüdiger Olderog [29].

2.3 University of Oxford, by Jonathan P. Bowen

The first introduction to ProCoS that I had was when Tony Hoare invited me to dinner at St John's College, Oxford, at relatively short notice [26, 65], with a hand-written Post-it Note during the winter of 1988–9, in the days before Tony started using e-mail:

> Jonathan,
> would you be free some
> time this week end to
> join a meeting of the Esprit
> Procos partners in St John's.
> Starting for supper tomorrow (Fri)
> 7.00pm St John's lodge Tony

On my arrival at the porters' lodge of St John's College, there were several academics, whose faces were largely new to me. Following a convivial dinner, a meeting was convened by Tony with Dines Bjørner (Denmark), Cliff Jones (UK), Hans Langmaack (Germany), and various others in attendance, including me, discussing proposed ProCoS collaborations. This led to a collaborative ESPRIT BRA, the ProCoS Project on "Provably Correct Systems" [7, 11].

Although I became employed by the UK SAFEMOS project on "Totally Verified Systems" with Cambridge (University and SRI) and the company Inmos, the work was closely connected with that of the ProCoS project, at least at Oxford. Both projects used the Transputer microprocessor instruction set of Inmos [12, 114] and the associated high-level programming language Occam [113], based on Tony Hoare's **Communicating Sequential Processes** (**CSP**) [102], with parallelism as an integral part of the language for convenient implementation on multiple Transputers.

The first ProCoS project was based around an Occam-like programming language and a subset of the Transputer instruction set. Early in the project, I specified the latter using the Z notation [12], and an operational semantics was also produced by Paritosh Pandya, a visitor to Oxford from India, with the help of this Z specification [157]. Tony Hoare, with the help of He Jifeng, who was employed on the ProCoS project itself, developed an approach to formal compiling specification in a form very close to a logic program (e.g., in Prolog) together with a proof of correctness. We applied this approach to the **ProCoS Level 0** (**PL$_0$**) programming language, essentially a simple

subset of Occam [37, 110]. I also investigated the use of the Lisp functional programming language for compiling this language as an alternative approach.

I mainly studied the relational compiling specification using the Prolog logic programming language, which resulted in the possibility of both a prototype compiler [14], and even a decompiler [16] since Prolog is a declarative language. The ProCoS programming language was later extended by He Jifeng with recursion (PLR_0) [91]. The original ProCoS project was continued with a reduced number of partners, led by Tony Hoare at Oxford University, as the ProCoS II project [17, 107, 108], and I continued by involvement with funding from a UK EPSRC project.

Formal methods are often used in safety-critical systems because of the desirability of the best possible integrity of the software in such systems. Victoria Stavridou, afterward Coleman, from Royal Holloway, and a visiting academic at Oxford for a while, and I investigated these issues in survey articles at conferences [46, 47]. Standards are an important aspect of the encouragement or even enforcement of the use of formal methods in critical systems [15, 48, 50, 52]. One survey article in this area [48] received a best article award for the year (the Charles Babbage Premium [51]) and was translated into French [49], despite being rejected by the first journal to which it was submitted. It then became my most highly cited article to this day. This taught me that article submission can be something of a lottery in practice!

The ProCoS II project investigated compilation directly into programmable hardware (e.g., on a **Field Programmable Gate Array** or (**FPGA**)) in the form of a netlist of basic electronic components, as an alternative to standard compilation for execution on a processor. This followed ideas originally developed by Ian Page at Oxford and pursued more formally by He Jifeng, with some modest help by me [99]. This approach was presented within a ProCoS tutorial edited by Anders Ravn [163]. Real-time issues were investigated by He Jifeng [92] and we also considered compiler optimization issues [19, 90, 93]. Additionally, I investigated hardware compilation of the ProCoS gas burner case study using logic programming.

As well as research, the ProCoS projects gave a sense of comradery across the various European partners, including moments of levity, such as a photomontage with captions for the FME'93 conference. Tony Hoare was an especially influential and inspirational leader of the projects [21], but two Chinese members, Zhou Chaochen and He Jifeng, also contributed important ideas developing *Duration Calculus* [185] and the seeds of ***Unifying Theories of Programming*** (**UTP**) [109].

I helped in producing collaborative overview articles and tutorials of the ProCoS approach [31, 42] and in organizing a final report in the *EATCS Bulletin* [43]. I also maintained an online ProCoS II bibliography in BibTEX format, an online FTP and then web-based archive of ProCoS material, a LATEX style file with a logo and standard front page for ProCoS reports, and a ProCoS mailing list on the UK JISCmail facility that has continued in existence to this day. All these facilities proved helpful in a collaborative international project like ProCoS.

A three-volume interim deliverable was produced for the initial ProCoS project in 1990. A final deliverable was also produced at the end of the ProCoS project [8]. Four volumes in two folders were produced. These are available in the History of Computing Collection, run by John Tucker and others at the University of Swansea in Wales since 2007 [173], which also includes archival articles by Dines Bjørner and me. The collection covers general material associated with the ProCoS projects, project-related articles, documents specifically associated with the ProCoS I and ProCoS II projects, material connected with the FTRTFT'94 Provably Correct Systems tutorial, and reading material suggested for participants at the start of the ProCoS project [26].

After ProCoS. The initial two ProCoS projects continued with the ProCoS-WG Working Group of 25 European industrial and academic partners [18, 44]. I moved from Oxford to the University of Reading as a lecturer in 1995 and continued to organize ProCoS-WG from there, with periodic

meetings culminating in a meeting at the University of Reading in association with the ZUM'97 conference [40].

He Jifeng continued at Oxford, working with Tony Hoare, culminating in their 1998 magnum opus UTP [109]. Tony Hoare has noted He Jifeng's significant contribution to the mathematical aspects of the UTP approach in connecting different formal semantics [178]. UTP resulted in a subsequent international community of researchers and an associated International Symposium on Unifying Theories of Programming, started in 2006.

After Oxford, He Jifeng then moved to **United Nations University, International Institute for Software Technology** (**UNI/IIST**) in Macau, working under the leadership of Zhou Chaochen [22], who hired him. I worked there on using He Jifeng's research to animate the semantics of the hardware description language Verilog using the logic programming language Prolog during the summer of 1999 [23, 38]. I continued collaborating with He Jifeng on hardware compilation [34, 35], and attending his Shanghai Festschrift celebrations for his 70th and 80th birthdays in 2013 [24] and 2023 [45, 54].

I became Professor of Computing at **London South Bank University** (**LSBU**) in 2000. Huibiao Zhu, who had been a visitor at UNU/IIST, studied for a PhD based on UTP at LSBU, under my supervision. Subsequently, he became a very successful and productive academic at **East China Normal University** (**ECNU**) in Shanghai, where I was a visitor, giving lectures on the Z notation in 2013. We have continued collaboration to this day on UTP-related research [169, 188–191] and organization [53]. I also collaborated on research with Zhiming Liu (see Section 2.7) [154] and became an adjunct professor at Southwest University in Chongqing, China, under his direction. We have co-organized the SETSS School on Engineering Trustworthy Software Systems, started in 2014 and held annually, with a break during the COVID pandemic, and subsequently restarted again [32]. This provides tutorials by international speakers for postgraduate students, mainly from around China. In parallel, I maintained periodic contact with Tony Hoare, for example, interviewing him in 2006 at Microsoft Research in Cambridge for the Computer History Museum in California [41, 170].

In 2015, I co-organized a two-day ProCoS reunion event with Ernst-Rüdiger Olderog (see Section 2.5) at the BCS London office, with presentations by both original ProCoS participants and younger researchers influenced by ProCoS [101]. I presented on the connections and community created by the ProCoS initiatives, in the context of the **Community of Practice** (**CoP**) model [25]. Without the initial stimulus of ProCoS, this international collaboration for decades after the original project would certainly never have happened in the way that it did. The ESPRIT funding opportunities, and for me, the ProCoS initiatives in particular, have enabled research collaborations that would have been impossible otherwise, around Europe initially, and subsequently around the world. In 2018, we celebrated the 20th anniversary of the UTP book [109], also at the BCS London office, where both He Jifeng, visiting from China, and Tony Hoare gave presentations [27]. Sadly, this would be the last time I saw Tony, before his passing in 2026, during the final editing of this article.

2.4 Royal Holloway and Bedford New College, by Ursula Martin

In the mid-1980s, my career shifted from abstract mathematics to computer science, and I joined the Formal Methods research group of Cliff Jones at the University of Manchester. Cliff was a wonderful mentor and inclusive leader, getting me involved in a research culture of collaboration, conferences, and major projects that were totally new to me. It was through Cliff that I became involved in the planning for what became ProCoS, receiving an eye-opening education from a master in how these things were done by leaders of the field. The ProCoS tower of levels for the development of embedded real-time computer systems took shape before our eyes as Dines Bjørner drew multi-colored boxes and arrows on old-school acetates.

In 1988, I took up an academic position in the new Computer Science Department at Royal Holloway University of London, as did Victoria Coleman, who had recently completed a PhD in hardware verification with Manchester's Doug Edwards. The ProCoS bid had been successful – delivered to Brussels by hand, by someone flying down from Denmark with the required large number of article copies – and we at Royal Holloway were one of the seven nodes of Phase 1, alongside Oxford University, the Technical University of Denmark at Lyngby, Christian-Albrechts-Universität Kiel, Universität Oldenburg, Århus University, and the University of Manchester.

I particularly remember our intense meetings, organized with some style by our Danish hosts. For one, we were instructed to arrive at a ferry terminal in Copenhagen, where we caught the night ferry to the magical island of Bornholm, taking over a charming small hotel for a few days, and walking on the beach between work sessions. During another meeting in Denmark, I managed to mislay my passport: somehow, Dines' team ensured that I had a trouble-free journey home, and I was welcomed at London Heathrow airport by an official politely enquiring "Are you Dr Martin, we are expecting you".

My own research during this period remained close to my mathematical roots, developing aspects of the theory of term rewriting, subsequently deployed in various practical systems [142]. Victoria collaborated with ProCoS members Jonathan Bowen (see Section 2.3) and Anders P. Ravn (Denmark) to publish a slew of significant and highly cited articles on formal methods and system safety [46–48, 83]. These laid the foundation for her influential later work in the UK and USA, in particular on DefStan 00-56, the procurement standard that the UK Ministry of Defence uses to regulate the procurement of programmable electronic systems.

After ProCoS. In my later career, I worked at the University of St Andrews and Queen Mary University of London, before moving to Oxford in 2015, and was elected a Fellow of the Royal Academy of Engineering in 2017. Verification work has included applying Hoare Logic to control engineering [2], and investigating practical ways of embedding real-number theorem proving in larger systems [79]. More recently, my research broadened to consider the social and cultural context of computer science and mathematics, and I led the first scholarly investigation of the mathematics of Ada Lovelace [112]. I was honored to be made a Dame in the 2025 King's Birthday Honors list.

Victoria Coleman has had an outstanding career in high-tech leadership in the USA, and is currently the CEO of Acubed, the Silicon Valley Innovation Center of Airbus, and Head of Research and Technology, North America for Airbus. She is a Visiting Professor in EECS at UC Berkeley, and previously served as the 37th Chief Scientist of the United States Air Force, Arlington, Virginia, and the 22nd Director of DARPA. She was elected a Fellow of the US National Academy of Engineering in 2025: a most distinguished and influential ProCoS alumnus.

2.5 Carl von Ossietzky Universität Oldenburg, by Ernst-Rüdiger Olderog

I became involved in the preparations of the ProCoS project as a scientific assistant in the research group of Hans Langmaack (see Section 2.2) at the University of Kiel. In this role, I accompanied Langmaack to a meeting in Oxford, where the proposal was discussed. When the project was approved by "Brussels", Hans Langmaack generously transferred one of the positions for research assistants to my upcoming research group at the University of Oldenburg.

The beginning of the ProCoS project coincided with the start of my professorship at the University of Oldenburg. I then became the site leader of Oldenburg in ProCoS. For me, it was also the start of using e-mail as a means of communication and LaTeX as a means of writing scientific articles. Both were indispensable for the collaboration with scientists of several international sites in the project. Jonathan Bowen (see Section 2.3) skillfully designed the layout for all project documents to

be written in these years. Also notable is that the payment of ESPRIT projects like ProCoS was calculated in ECU (European Currency Unit, with fixed conversion rates to the various national currencies), a precursor of the Euro, was introduced later in 1999.

Stephan Rössig was my first research assistant at Oldenburg, who was funded by ProCoS, aiming for a PhD. Soon afterward, Dr. Michael Schenke joined my group, also working on the topic of ProCoS and aiming at a habilitation. In Oldenburg, our task inside the ProCoS "tower of layers" [117] (comprising requirements, specifications, programs, machine code, and hardware) was the correct transformational design of real-time systems from the level of program specifications to the level of Occam-like programs. Results were published in [153, 166, 167].

Besides regular project-wide workshops, an exchange of scientists from the different sites took place. He Jifeng from Oxford and Michael R. Hansen (see Section 2.6) from DTU in Lyngby visited Oldenburg for longer periods. In turn, Michael Schenke stayed in Oxford as a Visiting Research Fellow. With Anders P. Ravn, a lifelong cooperation [151, 152] and friendship started.

The ProCoS project inspired my group to participate in several other research projects: **Correct Communication Networks** (**CoNs**) funded by Philips Research Laboratories in Aachen, UniForM Workbench with Bernd Krieg-Brückner and Jan Peleska from the University of Bremen and the company Elpro LET in Berlin, working on signaling systems for trams, and KORSO (Correct Software) funded by the German Federal Ministry for Education and Research (BMBF). The latter project enabled Jürgen Bohn [9] to provide mechanical support and validation of the transformational design rules developed for ProCoS. To this end, he employed the proof assistant LAMBDA [59].

The Legacy of ProCoS. A major achievement in the ProCoS project was the development of the DC for formalizing real-time requirements [185]. At Oldenburg, we studied DC formulas mostly at the implementable level, called *DC implementables* and introduced by Anders P. Ravn [162]. Within the UniForM project, Henning Dierks developed PLC-Automata as an abstract automata model for **Programmable Logic Controllers** (**PLCs**), which are a widespread industrial platform used in control applications. He defined two consistent semantics for PLC-Automata, one based on DC and one based on timed automata. He showed that an automatic synthesis procedure can generate a PLC-Automata from a consistent specification in terms of DC implementables. The semantics based on timed automata enabled an automatic verification of real-time properties of PLC-Automata, using the UPPAAL model checker [120].

In her PhD thesis, Cheryl Kleuker developed *Constraint Diagrams* for the visual specification of real-time requirements in an assumption/commitment style, with its semantics in the Duration Calculus. DC implementables can be expressed by a subclass of Constraint Diagrams. Jochen Hoenicke generalized DC implementables to the uniform pattern of *counterexample formulas* [111]. The idea is that each formula describes a single behavior pattern that must *not* occur. Such formulas have proven useful as a semantic backbone for real-time requirements occurring in automotive applications [160].

Many of these results are presented in the book [150] on formal specification and automatic verification of real-time systems based on Duration Calculus, timed automata, and PLC-Automata.

The Duration Calculus inspired two further developments in my research group. Andreas Schäfer extended the DC to a multi-dimensional Shape Calculus, in which it is possible to reason about movements in real-time and space, for instance, of robots [165]. Further, in collaboration with Anders P. Ravn, my group developed a dedicated **Multi-Lane Specification Language** (**MLSL**) for reasoning about the safety of car traffic on multi-lane roads of different types (motorways, country roads, urban traffic) [100].

Within my group in Oldenburg, and on topics related to the ProCoS project, Stephan Rössig, Stephan Kleuker, Cheryl Kleuker, Jürgen Bohn, Henning Dierks, Jochen Hoenicke, and Andreas

Schäfer completed their PhD theses, and Michael Schenke completed his habilitation. Of these, four have been appointed professors at universities of applied sciences.

During the ProCoS II project, in 1994, I also received my greatest distinction: the Leibniz Prize of the German Research Foundation (DFG). It resulted in sizable extra funding for five years during which the research initiated by ProCoS could be pursued further. At our department in Oldenburg, the topic of correct systems was pursued further in two major DFG-funded research initiatives: the Collaborative Research Center **Automatic Verification and Analysis of Complex Systems**, 2004–2015 (**AVACS**) of the universities of Oldenburg, Freiburg, and Saarbrücken, and the Research Training Group SCARE (System Correctness under Adverse Conditions, 2012–2021) for PhD students in Oldenburg.

2.6 Duration Calculus, by Michael Reichhardt Hansen

The Beginning. At the ProCoS workshop in Viborg, Denmark, in December 1990, E. V. Sørensen presented a case study addressing requirements for gas burners. Sitting next to Zhou Chaochen, I remember that he whispered two observations:

— Requirements are properties about intervals.
— The notion: state duration, is important.

At that time, Zhou Chaochen and Anders P. Ravn were visiting Tony Hoare at Oxford University, and a beautiful logic, namely *Duration Calculus*, was developed and published in 1991 [185].

The ProCoS Years of 1989–1995 at DTU. The development of DC led to a fantastic stimulating period with research activities spanning from case studies, methodology, fundamentals, and tool support, all with DC as a common denominator. These activities involved, among others, Dines Bjørner (see Section 2.1), Kirsten Mark Hansen, Michael R. Hansen, Zhiming Liu (see Section 2.7), Hans Henrik Løvengreen, Jens Nordahl, Anders P. Ravn, Hans Rischel, Peter Sestoft, Jens Ulrik Skakkebæk, and E. V. Sørensen, where Zhou Chaochen, also at DTU, was the central person.

The "Gas Burner" case study presented by E. V. Sørensen materialized into a very influential publication [164] by Ravn, Rischel, and K. M. Hansen, where a method, within a Duration Calculus framework, was developed for specification and analysis involving safety and functionality requirements, high-level design, and trace-based specifications of control programs. This article contains the first formal correctness proof for the gas burner case study. The first investigation of Duration Calculus as a basis for specification and analysis of hybrid systems was *Extended Duration Calculus for Hybrid Systems*, by Zhou, Ravn, and Hansen [187]. Furthermore, a language for describing networks of communicating discrete and continuous processes was presented by Zhou, Wang, and Ravn, in [186], with a semantics expressed in Extended Duration Calculus and a form inspired by [183].

In the context of dependable computing, a probabilistic duration calculus was introduced by Zhiming Liu, A. P. Ravn, E. V. Sørensen and Zhou Chaochen in [138]. Furthermore, in [172], E. V. Sørensen, J. Nordahl, and N. H. Hansen described how a probabilistic dependability model can be derived from a CSP model. Fundamental results, by M. R. Hansen and Zhou Chaochen, relating Duration Calculus to **Interval Temporal Logic** (**ITL**), by Ben Moszkowski, appeared in [86] and results by Zhou Chaochen, M. R. Hansen and P. Sestoft, on decidable and undecidable subsets of Duration Calculus appeared in [184]. An interactive verification tool was developed by J. U. Skakkebæk [171] and a first automated verification tool exploiting a decidable subset of Duration Calculus is DCVALID [156] by P. K. Pandya.

In his thesis *Design of Embedded Real-Time Computing Systems* for Doctor Technices [162], Anders P. Ravn presented a unified framework for the design of embedded real-time systems with an emphasis on step-wise refinements ranging from top-level requirements to implementable

controller programs. Control programs are expressed in a subset of Duration Calculus named *DC implementables*. It is shown, in work by H. Dierks [66], how a consistent collection of such implementables can be implemented by a PLC automaton.

Mobility between ProCoS sites had a major rôle. Zhou Chaochen had longer stays at Oxford University and the Technical University of Denmark, and from 1992, had the United Nations University in Macau as his primary affiliation. During 1992–93, Michael R. Hansen visited the group of Ernst-Rüdiger Olderog (see Section 2.5) in Oldenburg and [184] was partially written during this visit. A course on Duration Calculus was a part of this visit, resulting in lecture notes aiming at developing the logic so that it could be presented, with applications, in a coherent manner. This led to [87] by M. R. Hansen and Zhou Chaochen. In addition to Anders P. Ravn's stay in Oxford, he also visited the group of Hans Langmaack at Kiel during 1994–95, and lectured on his doctoral dissertation.

The Years after ProCoS. The ProCoS years were characterized by a project-wise stimulating atmosphere with collaboration and mutual inspiration. The beauty is that the spirit and collaboration were long-lasting. It was a huge pleasure to arrange the first international workshop on Duration Calculus at the 10th European Summer School in Logic, Language and Information, in August 1998, where Zhou Chaochen and Ernst-Rüdiger Olderog were invited speakers, and the program comprised presentations of 16 accepted articles. A special issue of *Formal Aspects of Computing* [115] was dedicated to selected articles from this event.

In the area of component-driven design and object-orientation, Anders P. Ravn, based at Aalborg University from 1999, had an ongoing collaboration with Zhiming Liu and colleagues from UNI/IIST (in Macau) and the Institute of Software, Chinese Academy of Sciences (in Beijing), e.g., [62, 64]. Furthermore, his collaboration with Ernst-Rüdiger Olderog and colleagues from Oldenburg University continued. For example, in [168] by M. Schenke and A. P. Ravn, refinements from control problems to programs are considered, and in [100], by M. Hilscher, S. Linker, E.-R. Olderog, and A. P. Ravn the **Multi-Lane Spacial Logic** (**MLSP**) is introduced.

My collaboration and regular visits to the groups of Ernst-Rüdiger Olderog and Martin Fränzle at Oldenburg University were always enjoyable, stimulating experiences, in a unique research environment. It was most refreshing to collaborate on model checking and aspects like robustness and discounting (sooner is better than later) in a Duration Calculus context and to have a closer investigation of MLSL, together with Martin and Heinrich Ody, e.g., [73–76, 148]

The close collaboration with Zhou Chaochen continued, including several visits to UNU/IIST, from which I have the dearest memories. Working with Zhou was fantastic, and experiencing UNU/IIST was unforgettable. Selected joint work with Zhou, Dang Van Hung, and P. K. Pandya are [84, 85, 181].

The culmination of the collaboration was the completion of the 2004 monograph *Duration Calculus: A Formal Approach to Real-Time Systems* [182]. The joint efforts in the process, with high-level considerations on telling a coherent story, as well as technical developments aiming at a consistent and complete exposition, will always be in my mind.

2.7 A View from China, by Zhiming Liu

Experiences of the ProCoS Project. Although I spent only four months with ProCoS, the project introduced me to researchers who became long-term collaborators and provided a decisive boost to my academic career.

My first ProCoS meeting, held on the island of Fyn in October 1991, brought together many who would later shape my research path, including Jonathan Bowen and He Jifeng. The meeting exemplified the collaborative spirit of ProCoS, linking young researchers with established leaders in real-time and safety-critical systems.

At the same time, Dines Bjørner and Zhou Chaochen were preparing the launch of the UNU/IIST in Macao, scheduled to open in 1992. Dines, at the workshop dinner, was therefore passing the ProCoS coordinator's baton to Tony Hoare – an agreement sealed, naturally, over good food and even better wine. I was surprised by Tony's speech: warm, emotional, and laced with humor – so unlike the reserved impression he gave when Zhou's four former students, including myself, were invited to have Christmas lunch with his family at his home in 1989.

Probabilistic Duration Calculus. Zhou Chaochen suggested that I collaborate with Anders P. Ravn and Ealing V. Sørensen (EV) on developing a probabilistic extension of DC to analyse the reliability of fault-prone real-time systems. EV, an expert in safety analysis, proposed a gas burner example (see Section 2.6), whose unreliable flame detector became the starting point for this line of work.

This proposal resonated with my doctoral research on formal techniques for fault tolerance [128, 134, 135], where I had developed an approach to *fault-tolerant programming by transformations*. To the best of my knowledge, this was the first systematic formal method to address fault tolerance, providing a formal semantics for *faults*, *errors*, *failures*, and *fault-tolerant behavior*. My draft thesis also included an early chapter on *probabilistic Hoare logic*, which aimed at characterizing fault tolerance in probabilistic terms. Although this chapter was removed on my examiner's advice, it anticipated many of the later concerns with probabilistic verification, and I now regard it as an important step toward quantifying the reliability of fault-tolerance and security of systems, *cyber-physical systems with AI components* in particular.

Building on this background, EV and I, with support from Zhou and Anders, developed the ***Probabilistic Duration Calculus*** (**PDC**). The calculus combined *probabilistic automata* for modeling with DC formulas for specifying requirements, and introduced axioms and an induction rule for computing the *satisfaction probability* $\mu(D)$ of a formula D. Later extensions employed probability matrices for more efficient calculation. To demonstrate applicability, we analysed a communication protocol over an unreliable channel [88, 89], with results published in the inaugural issue of *High Integrity Systems* [139].

The first version of PDC was presented at the International Workshop on Responsive Systems in Saitama, Japan (1992) [138], and subsequently appeared in an edited volume. This work marked the beginning of a research direction that connected formal semantics of fault tolerance with probabilistic verification, influencing later developments in the field.

My First Visit to UNU/IIST. On the way to the International Workshop on Responsive Systems in Japan, Zhou invited EV and me to visit UNU/IIST, and each give a seminar. When I arrived, the institute was barely three months old, but was already positioning itself as an international centre for research in formal methods.

The seminars we delivered were not only the first academic events at UNU/IIST, but also the first public lectures on formal methods in Macao. To emphasize their significance, Zhou personally prepared an announcement in traditional Chinese, which was published in the *Macau Daily*. At that time, both formal methods and theoretical computer science were little known in the region, and special care was taken to make the talks accessible to the local audience. These inaugural events reflected the institute's mission to introduce rigorous methods into new academic environments and to build a foundation for future research and training in software technology.

The seminar room, though modest, was full. Following opening remarks by Dines Bjørner and an introduction by Zhou, EV gave the first talk, and I followed. The audience listened attentively and, in local fashion, remained polite during the presentations, with lively discussions developing during the breaks and afterward.

Formal Methods Research and Education at UNU/IIST. UNU/IIST was a lean institute, with rarely more than ten academics – all committed to formal methods – supported by about half a dozen

administrative staff, called "general" staff. In its two-storey building, research was informally divided into two groups: the "theory" group upstairs and the "practice" group downstairs, a distinction often mentioned humorously, but reflecting the institute's dual commitment to foundational research and practical application. The downstairs group, led for most of the time by Chris George, concentrated on the RAISE method, focusing on tools, semantics, and applications.

As Jonathan Bowen noted in the *Times Higher Education Supplement* [22], UNU/IIST was "a pearl in software technology" – small in size but of remarkable value. UNU/IIST pursued an ambitious mission: *to help developing countries* [2] *strengthen their software development capacity through training and collaboration.* Fellows from nearly all regions of the Global South came for eight-month research visits, returning with training in formal methods and experience in collaborative projects. Staff also delivered courses in Macao and abroad, on five continents, ensuring that formal methods first took root in many countries through UNU/IIST initiatives.

Alumni now lead research groups worldwide, and in China, the institute became known as the "Huangpu Military Academy" [3] of software engineering. Visitors included leading figures such as Tony Hoare, Bill Roscoe, Ernst-Rüdiger Olderog, Anders Ravn, Jonathan Bowen, and Paritosh Pandya. An army of about ten UNU/IIST fellows, led by Zhou, attended the FTRTFT'94 conference in Lübeck, Germany, partly sponsored by ProCoS [118]. A later highlight was the 2007 Festschrift in Macao celebrating the joint 70th birthdays of Dines and Zhou – evidence that rigorous logic and vibrant community could flourish together [116].

Continued Research on DC and UTP at UNU/IIST. The upstairs group provided continuity in theoretical innovation. Initially led by Zhou Chaochen, its research and training centred on DC. Under his guidance, UNU/IIST became a hub for DC research, producing completeness proofs, decidability results, algorithms for linear duration invariants, and semantic advances such as *finite divergence* and *superdense time*, later consolidated in the DC monograph [182]. Notable contributions also came from fellows, including the PhD theses of Xiaoshan Li [124] and Naijun Zhan [180]. Applications extended to tool semantics, specification languages, and real-time semantics for CSP, Verilog, and RAISE.

In 1997, Zhou became Director of UNU/IIST and, in 1998, He Jifeng joined as Senior Research Fellow, assuming leadership of the upstairs group. He broadened the scope to the UTP, providing a semantic foundation for linking diverse programming paradigms within a unified logical framework. This shift strongly influenced both research directions and advanced training at the institute.

In 2001, I spent an eight-month sabbatical at UNU/IIST, collaborating with He Jifeng and Xiaoshan Li at the University of Macao. Our work focused on the formal use of UML within the UTP framework [126], at a time when UML was widely criticized by the formal methods community. I joined UNU/IIST as a Research Fellow in 2002, and with the fellows, this line of research was further developed [127, 133, 136]. It led to the creation of *rCOS: A Refinement Calculus of Object Systems* [97, 98], which extended UTP to provide a semantic foundation for object-oriented programming. rCOS was later broadened to define semantics of component-based architectures and to extend the concept of design by contracts [144] into a framework for *component-based design by interface contracts* [96, 131, 132].

Further Development of rCOS. After He Jifeng's departure in 2005, the upstairs group became the rCOS group, continuing until the institute's reorganization by the United Nations University in 2013. During this period, UNU/IIST established a joint PhD programme with the University of

[2] Now commonly referred to as the "Global South".

[3] The Huangpu Military Academy, founded by Sun Yat-sen in 1924, trained revolutionary military leaders who shaped modern China.

Pisa, and a postdoctoral programme, supported by research grants from the Macau Science and Technology Development Fund (FDCT).

The main focus of research and training was the further development of rCOS, including application case studies and tool support [60, 63, 64, 81, 122, 123, 125, 137, 140]. rCOS extended refinement calculus to component- and object-based systems, providing techniques for specification, verification, and stepwise refinement aligned with industrial software engineering practices. It became a bridge between theory and practice, offering fellows methods that combined rigorous formal foundations with applicability in large-scale system development.

The rCOS framework also served as the basis for UNU/IIST training in software engineering, particularly model-driven software development, and was disseminated through teaching activities in multiple countries across the Global South. I taught in Brazil, China, Iran, Kazakhstan, Nigeria, Peru, Tunisia, South Africa, and Vietnam. There were over 30 UNU/IIST fellows, PhD students, and postdoctoral fellows who worked on rCOS.

Subsequent research has advanced automated tool support and extended rCOS in the superimposition style, a key idea inherited from UTP, to model-driven architectures of ***cyber-physical systems*** (**CPS**) [61, 154, 155] and ***human-cyber-physical systems*** (**HCPS**) [141]. A comprehensive review of the thematic ideas of rCOS is presented in the Festschrift article [129] dedicated to Professor He Jifeng's 80th birthday.

Final Remarks. With the memories presented here, I would like to acknowledge the impact of the ProCoS project and two of its major achievements – DC and the UTP – on the development of my academic career. I am also deeply grateful to Professor Mathai Joseph, my PhD supervisor; Professor Zhou Chaochen, my master's supervisor; and Professor He Jifeng, my colleague and mentor in research, for their guidance and influence, both as exemplary human beings and as scholars of the highest calibre.

No account of these memories would be complete without a tribute to Anders Ravn – one of the finest friends and mentors I have had in both life and career. A more detailed version of this memoir, including a full section dedicated to Anders, has been published in the July 2025 issue of the *FACS FACTS* newsletter of the BCS-FACS Specialist Group [130].

3 Related Projects

Overall, the following ProCoS-related initiatives took place during the 1990s, funded by the EU within the ESPRIT strategic programme unless otherwise stated:

- ProCoS project (no. 3104, 1989–1992).
- ProCoS II project (no. 7071, 1992–1995).
- ProCoS-WG Working Group (no. 8694, 1994–1997).
- UK EPSRC Provably Correct Hardware/Software Co-design project at Oxford University (1993–1996).
- ESPRIT/NSF ProCoS-US (EC-US027, 1993–1997) European/US travel grant on Provably Correct Hardware Compilation with Geoffrey Brown, Cornell University, USA.
- KIT (Keep in Touch) travel grant (1993–1998) with Zhou Chaochen, UNU/IIST, Macau.
- PROCORSYS KIT (Keep in Touch) travel grant (KIT 142, 1994–1997) with Augusto Sampaio, Departamento de Informatica, Universidade Federal de Pernambuco, Brazil.
- EPSRC Visiting Fellowship (1996–1997) to study Provably Correct Real Time Systems at Oxford University for Michael Schenke, University of Oldenburg, Germany.

The related collaborative United Kingdom **Information Engineering Directorate** (**IED**) DTI/SERC SAFEMOS project ran for 3½ years from 1989 to 1993 [13, 36], culminating in an edited book, *Toward Verified Systems* [20]. The project partners were:

— University of Cambridge Computer Laboratory.
— Cambridge SRI.
— Inmos (later SGS-Thomson Microelectronics Ltd).
— Oxford University Computing Laboratory.

In parallel with ProCoS, a number of ESPRIT "BRA" in Computer Science started [67] (later to be called "projects" [68]). The plans of 12 of them were announced in the *Bulletin of the* ***European Association for Theoretical Computer Science*** **(EATCS)**, number 39 in October 1989, and in numbers 40 and 41 in February and June 1990. Most notably, one of them was ESPRIT BRA 3006 CONCUR (Theories of Concurrency: Unification and Extension), coordinated by J. C. M. Baeten. As with some other ESPRIT projects, CONCUR organized its own conference, also called CONCUR. The first one took place in Amsterdam, Netherlands, in 1990 [3]. CONCUR remains an annual conference, with the last one occurring in Aarhus, Denmark, in 2025 [10].

The ESPRIT BRA 3104 ProCoS project was motivated by the "stack" of CLI, studying pervasive systems verification with the goal of building a system from verified, hierarchically stacked components (see also Section 1) [179]. The CLI stack also motivated the projects Verisoft and Verisoft XT, funded by the German Federal Ministry of Education and Research (BMBF) from 2003 to 2010, and coordinated by Wolfgang J. Paul in Saabrücken, and involving a number of academic and industrial partners. The main difference was that the components were significantly more complex than at CLI. The mission of the project was [158]:

(i) to develop the technology which permits the pervasive formal verification of entire computer systems consisting of hardware, system software, communication systems, and applications.
(ii) to demonstrate in collaboration with industry this technology with several prototypes.

Unlike ProCoS, this project insisted on tool-supported verification, using state-of-the-art methods. The verification was performed with the theorem prover Isabelle/HOL.

In 2002 and 2003, Tony Hoare advocated the verifying compiler as a long-term goal, as a "grand challenge" for computer science research [104, 105]. Such a compiler proves automatically that a program is correct before running it. Correctness of a program is defined by placing assertions at strategic points in the program text, particularly at the interfaces between components. To fully realize this goal, a major international research initiative is called for. As a start, a conference on ***Verified Software: Theories, Tools, Experiments*** **(VSTTE)** was arranged at the ETH Zürich in 2005 [145]. This conference developed into a series as the 16th conference VSTTE 2024 demonstrates [161].

4 Conclusion

This article has provided an overview of the European ProCoS projects and related research initiatives during the 1990s. More detailed views by individuals involved with these projects and initiatives are also presented, covering both experiences during the projects and the effects on careers after the projects.

It is interesting to note that one of the most far-reaching and enduring results from the European ProCoS projects was due in no small part through the efforts led by a Chinese scientist. Zhou Chaochen (now in Beijing), with the help of Anders P. Ravn and Tony Hoare, formulated the *(*DC during the first ProCoS project [185]. DC was not foreseen in the original project proposal, but it resulted in a subsequent international community of researchers. Some of the reminiscences in this article capture the birth and development of DC from various viewpoints.

In summary, it is intended that these firsthand accounts may be useful for future researchers investigating the nature of collaborative research and how communities can be fostered well into the future after a period of explicit funding. Indeed, many of the benefits of such collaboration (like DC) are typically realized years and often decades after the original funded initiatives.

Acknowledgments

The authors acknowledge the legacy of other participants on the ProCoS projects who were unable to contribute directly to this article, most notably Tony Hoare (1934–2026), Anders P. Ravn (1947–2019), and Zhou Chaochen, all key project members, and progenitors of the Duration Calculus. Victoria Coleman checked part of this article. Dines Bjørner dedicates his section to the memory of Søren Prehn. Zhiming Liu dedicates his section to Erling V. Sørensen. We dedicate the entire article to Tony Hoare [65], who passed away as this article was finalized, and Anders P. Ravn [28].

References

[1] J.-R. Abrial, E. Börger, and H. Langmaack (Eds.). 1996. *Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control.* LNCS, Vol. 1165, Springer. DOI: https://doi.org/10.1007/BFb0027227

[2] R. Arthan, U. Martin, E. A. Mathiesen, and P. Oliva. 2009. A general framework for sound and complete Floyd-Hoare logics. *ACM Transactions on Computational Logic* 11, 1 (2009), 1–31. DOI: https://doi.org/10.1145/1614431.1614438

[3] J. C. M. Baeten and J. W. Klop (Eds.). 1990. *CONCUR'90 – Theories of Concurrency: Unification and Extension, Amsterdam, The Netherlands, Proc.* LNCS, Vol. 458, Springer. DOI: https://doi.org/10.1007/BFb0039045

[4] W. R. Bevier, W. A. Hunt Jr., J. S. Moore, and W. D. Young. 1989. An approach to systems verification. *Journal of Automated Reasoning* 5, 4 (1989), 411–428. DOI: https://doi.org/10.1007/BF00243131

[5] D. Bjørner. 2017. How it all begin – as seen from Denmark, See [101], 1–6. DOI: https://doi.org/10.1007/978-3-319-48628-4_1

[6] D. Bjørner et al. 1990. Action 3104: Provably correct systems – PROCOS. See [67], 112–115.

[7] D. Bjørner, C. A. R. Hoare, J. P. Bowen, J. He, H. Langmaack, E.-R. Olderog, U. Martin, V. Stavridou, F. Nielson, H. Riis Nielson, et al. 1989. A ProCoS project description: ESPRIT BRA 3104. *Bulletin of the European Association for Theoretical Computer Science* 39 (1989), 60–73. Retrieved from http://researchgate.net/publication/256643262

[8] D. Bjørner, H. Langmaack, C. A. R. Hoare, et al. 1993. Provably Correct Systems: ProCoS I Final Deliverable. (1993). ESPRIT ProCoS BRA 3104.

[9] J. Bohn. 1998. *Mechanical Support and Validation of a Design Calculus for Communicating Systems by a Logic Based Proof System.* Ph. D. Dissertation. University of Oldenburg, Germany. Retrieved from https://d-nb.info/953866963 Report Nr. 2/1998.

[10] P. Bouyer and J. van de Pol (Eds.). 2025. In *Proceedings of the 36th International Conference on Concurrency Theory.* LIPIcs, Vol. 348, Schloss Dagstuhl — Leibniz-Zentrum für Informatik. DOI: https://doi.org/10.4230/LIPIcs.CONCUR.2025

[11] J. P. Bowen. 1990. ESPRIT BRA ProCoS project, See [147], 248. Back Matter: Posters.

[12] J. P. Bowen. 1990. Formal specification of the ProCoS/SAFEM OS instruction set. *Microprocessors and Microsystems* 14, 10 (1990), 631–643. DOI: https://doi.org/10.1016/0141-9331(90)90038-W

[13] J. P. Bowen. 1990. SAFEMOS: Demonstration of the possibility of totally verified systems, See [147], 249. Back Matter: Posters.

[14] J. P. Bowen. 1992. From programs to object code using logic and logic programming. In *Proceedings of the Code Generation – Concepts, Tools, Techniques (Workshops in Computing).* R. Giegerich and S. L. Graham (Eds.), Springer, 173–192. DOI: https://doi.org/10.1007/978-1-4471-3501-2_10

[15] J. P. Bowen. 1993. Formal methods in safety-critical standards. In *Proceedings of the Software Engineering Standards Symposium.* IEEE, 168–177. DOI: https://doi.org/10.1109/SESS.1993.263953

[16] J. P. Bowen. 1993. From programs to object code and back again using logic programming: Compilation and decompilation. *Journal of Software Maintenance: Research and Practice* 5, 4 (1993), 205–234. DOI: https://doi.org/10.1002/smr.4360050403

[17] J. P. Bowen. 1993. A ProCoS II project description: ESPRIT basic research project 7071. *Bulletin of the European Association for Theoretical Computer Science* 50 (1993), 128–137. Retrieved from http://researchgate.net/publication/2521581

[18] J. P. Bowen. 1994. A ProCoS-WG working group description: ESPRIT basic research 8694. *Bulletin of the European Association for Theoretical Computer Science* 53 (1994), 136–145.

[19] J. P. Bowen. 1994. Rapid compiler implementation. See [90], Chapter 10, 141–169.

[20] J. P. Bowen (Ed.). 1994. *Towards Verified Systems.* Real-Time Safety Critical Systems, Vol. 2. Elsevier.

[21] J. P. Bowen. 1995. A brief history of algebra and computing: An eclectic Oxonian view. *IMA Bulletin* 31, 1 (1995), 6–9.

[22] J. P. Bowen. 1999. Beijing takes the software pearl in Macau's crown. *The Times Higher Education Supplement* 1409 (1999), 13.

[23] J. P. Bowen. 2000. Combining operational semantics, logic programming and literate programming in the specification and animation of the Verilog Hardware Description Language. In *Proceedings of the Integrated Formal Methods*. W. Grieskamp, T. Santen, and B Stoddart (Eds.), LNCS, Vol. 1945, Springer, 277–296. DOI: https://doi.org/10.1007/3-540-40911-4_16

[24] J. P. Bowen. 2013. A relational approach to an algebraic community: From Paul Erdős to He Jifeng. In *Proceedings of the Theories of Programming and Formal Methods*. Z. Liu, J. C. P. Woodcock, and H. Zhu (Eds.), LNCS, Vol. 8051, Springer, 54–66. DOI: https://doi.org/10.1007/978-3-642-39698-4_4

[25] J. P. Bowen. 2017. Provably correct systems: Community, connections, and citations, See [101], 313–328. DOI: https://doi.org/10.1007/978-3-319-48628-4_13

[26] J. P. Bowen. 2019. *A Personal Formal Methods Archive*. ResearchGate. DOI: https://doi.org/10.13140/RG.2.2.31943.65447

[27] J. P. Bowen. 2019. Unifying Theories of Refinement. *FACS FACTS* 2019, 1 (2019), 7–9. Retrieved from https://www.bcs.org/media/5204/facs-dec19.pdf#page=7.

[28] J. P. Bowen. 2020. In memoriam: A tribute to five formal methods colleagues. *FACS FACTS* 2020, 1 (2020), 13–29. Retrieved from https://www.bcs.org/media/5842/facs-jun20.pdf#page=13.

[29] J. P. Bowen. 2025. Report on Prof. Dr. Martin Fränzle and his Festschrift Symposium. *FACS FACTS* 2025, 2 (2025), 28–34. Retrieved from https://www.bcs.org/media/yd4ocehl/facs-jul25.pdf#page=28.

[30] J. P. Bowen, B. Buth, E.-R. Olderog, and A. P. Ravn. 1993. *Provably Correct Systems – Tutorial Material for Formal Methods Europe 93*. ProCoS II document [ID/DTH APR 20/1]. Technical University of Denmark. In [121].

[31] J. P. Bowen, M. Fränzle, E.-R. Olderog, and A. P. Ravn. 1993. Developing correct systems. In *Proceedings of the 5th Euromicro Workshop on Real-Time Systems*. IEEE, 176–187. DOI: https://doi.org/10.1109/EMWRT.1993.639088

[32] J. P. Bowen, C. Gomes, and Z. Liu (Eds.). 2025. *Engineering Trustworthy Software Systems, 6th International School, SETSS 2024*. LNCS, Vol. 15584, Springer. DOI: https://doi.org/10.1007/978-981-96-4656-2

[33] J. P. Bowen and J. A. Hall (Eds.). 1994. *Z User Workshop, Cambridge 1994: Proceedings of the 8th Z User Meeting, Cambridge, 29–30 June 1994*. Springer. DOI: https://doi.org/10.1007/978-1-4471-3452-7

[34] J. P. Bowen and J. He. 2001. An approach to the specification and verification of a hardware compilation scheme. *The Journal of Supercomputing* 19, 1 (2001), 23–39. DOI: https://doi.org/10.1023/A:1011184310224

[35] J. P. Bowen and J. He. 2006. An algebraic approach to hardware compilation. In *Proceedings of the Modern Formal Methods and Applications*. H. A. Gabbar (Ed.), Springer, 151–176. DOI: https://doi.org/10.1007/1-4020-4223-X_7

[36] J. P. Bowen, J. He, R. W. S. Hale, and J. M. J. Herbert. 1995. Towards verified systems: The SAFEMOS project. In *Proceedings of the Mathematics of Dependable Systems*. C. Mitchell and V. Stavridou (Eds.), Institute of Mathematics and Its Applications Conference Series, Vol. 55, Oxford University Press, 23–48. Retrieved from http://researchgate.net/publication/2525857

[37] J. P. Bowen, J. He, and P. K. Pandya. 1990. An approach to verifiable compiling specification and prototyping. In *Proceedings of the Programming Language Implementation and Logic Programming*. P. Deransart and J. Maluszynski (Eds.), LNCS, Vol. 456, Springer, 45–59. DOI: https://doi.org/10.1007/BFb0024175

[38] J. P. Bowen, J. He, and Q. Xu. 2000. An animatable operational semantics of the Verilog Hardware Description Language. In *Proceedings of the 3rd IEEE International Conference on Formal Engineering Methods*. IEEE, 199–207. DOI: https://doi.org/10.1109/ICFEM.2000.873820

[39] J. P. Bowen and M. G. Hinchey (Eds.). 1995. *ZUM'95: The Z Formal Specification Notation: 9th International Conference of Z Users, Limerick, Ireland, September 7–9, 1995*. LNCS, Vol. 967, Springer. DOI: https://doi.org/10.1007/3-540-60271-2

[40] J. P. Bowen, M. G. Hinchey, and D. Till (Eds.). 1997. *ZUM'97: The Z Formal Specification Notation: 10th International Conference of Z Users, Reading, UK, April 3–4, 1997*. LNCS, Vol. 1212, Springer. DOI: https://doi.org/10.1007/BFb0027279

[41] J. P. Bowen and C. A. R. Hoare. 2006. *Oral History of Sir Antony Hoare*. Technical Report X3698.2007. Computer History Museum, California, USA. Retrieved from http://www.computerhistory.org/collections/catalog/102658017 Catalog Number 102658017.

[42] J. P. Bowen, C. A. R. Hoare, M. R. Hansen, A. P. Ravn, H. Rischel, E.-R. Olderog, M. Schenke, M. Fränzle, M. Müller-Olm, J. He, and et al. 1994. Provably correct systems – FTRTFT'94 tutorial. In *Proceedings of the 3rd International School and Symposium, Formal Techniques in Real Time and Fault Tolerant Systems*. Christian-Albrechts-Universität, Lübeck, Germany. Retrieved from http://researchgate.net/publication/2420842 School Material.

[43] J. P. Bowen, C. A. R. Hoare, H. Langmaack, E.-R. Olderog, and A. P. Ravn. 1996. A ProCoS II project final report: ESPRIT basic research project 7071. *Bulletin of the European Association for Theoretical Computer Science* 59 (1996), 76–99. http://researchgate.net/publication/2255515

[44] J. P. Bowen, C. A. R. Hoare, H. Langmaack, E.-R. Olderog, and A. P. Ravn. 1998. A ProCoS-WG working group final report: ESPRIT working group 8694. *Bulletin of the European Association for Theoretical Computer Science* 64 (1998), 63–72. Retrieved from http://researchgate.net/publication/2527052

[45] J. P. Bowen, Q. Li, and Q. Xu (Eds.). 2023. *Theories of Programming and Formal Methods, Essays Dedicated to Jifeng He on the Occasion of His 80th Birthday*. LNCS, Vol. 14080, Springer. DOI: https://doi.org/10.1007/978-3-031-40436-8

[46] J. P. Bowen and V. Stavridou. 1992. Formal methods and software safety. In *Proceedings of the IFAC Symposium on Safety of Computer Control Systems 1992*. H. Frey (Ed.), Pergamon Press, 93–98. DOI: https://doi.org/10.1016/B978-0-08-041893-3.50020-3

[47] J. P. Bowen and V. Stavridou. 1993. The industrial take-up of formal methods in safety-critical and other areas: A perspective. In *Proceedings of the Industrial-Strength Formal Methods*. J. C. P. Woodcock and P. G. Larsen (Eds.), LNCS, Vol. 670, Springer, 183–195. DOI: https://doi.org/10.1007/BFb0024646

[48] J. P. Bowen and V. Stavridou. 1993. Safety-critical systems, formal methods and standards. *Software Engineering Journal* 8, 4 (1993), 189–209. DOI: https://doi.org/10.1049/sej.1993.0025

[49] J. P. Bowen and V. Stavridou. 1993. Systèmes à Süreté de Fonctionnement Critique, Méthodes Formelles et Normes. *Génie Logiciel and Systèmes Experts* 30 (1993), 55–82.

[50] J. P. Bowen and V. Stavridou. 1994. Safety-critical systems and formal methods. *Real-Time Safety Critical Systems* 2. (1994), 3–33. DOI: https://doi.org/10.1016/B978-0-444-89901-9.50010-0

[51] J. P. Bowen and V. Stavridou. 1995. Charles babbage premium award winners. *Software Engineering Journal* 10, 1 (1995), 2. DOI: https://doi.org/10.1049/sej.1995.0001

[52] J. P. Bowen and V. Stavridou. 1999. Safety-critical systems and formal methods. In *Proceedings of the High-Integrity System Specification and Design*. J. P. Bowen and M. G. Hinchey (Eds.), Springer, 485–528. DOI: https://doi.org/10.1007/978-1-4471-3431-2

[53] J. P. Bowen and H. Zhu (Eds.). 2017. *Unifying Theories of Programming: 6th International Symposium*. LNCS, Vol. 10134, Springer. DOI: https://doi.org/10.1007/978-3-319-52228-9

[54] J. P. Bowen and H. Zhu. 2023. Jifeng He at Oxford and beyond: An appreciation, See [45], 1–6. DOI: https://doi.org/10.1007/978-981-96-4656-2_18

[55] R. S. Boyer and J. S. Moore. 1986. Overview of a theorem-prover for a computational logic. In *Proceedings of the 8th International Conference on Automated Deduction*. J. H. Siekmann (Ed.), Lecture Notes in Computer Science, Vol. 230, Springer, 675–678. DOI: https://doi.org/10.1007/3-540-16780-3_133

[56] B. Buth. 1995. *Operation Refinement Proofs for VDM-like Specifications*. Ph. D. Dissertation. University of Kiel, Germany. Retrieved from https://d-nb.info/944607624

[57] B. Buth, K.-H. Buth, M. Fränzle, B. von Karger, Y. Lakhnech, H. Langmaack, and M. Müller-Olm. 1992. Provably correct compiler development and implementation. In *Proceedings of the 4th International Conference on Compiler Construction*. U. Kastens and P. Pfahler (Eds.), LNCS, Vol. 641, Springer, 141–155. DOI: https://doi.org/10.1007/3-540-55984-1_14

[58] K.-H. Buth. 1994. *Techniques for Modelling Structured Operational and Denotational Semantics Definitions with Term Rewriting Systems*. Ph. D. Dissertation. University of Kiel, Germany. Retrieved from https://d-nb.info/943313988

[59] N. Chapman, S. Finn, and M. P. Fourman. 1994. Datatypes in L2. In *Proceedings of the Higher Order Logic Theorem Proving and Its Applications: 7th International Workshop*. T. F. Melham and J. Camilleri (Eds.), LNCS, Vol. 859, Springer, 128–143. DOI: https://doi.org/10.1007/3-540-58450-1_39

[60] X. Chen, J. He, Z. Liu, and N. Zhan. 2007. A model of component-based programming. In *Proceedings of the International Symposium on Fundamentals of Software Engineering, International Symposium*. F. Arbab and M. Sirjani (Eds.), LNCS, Vol. 4767, Springer, 191–206. DOI: https://doi.org/10.1007/978-3-540-75698-9_13

[61] X. Chen and Z. Liu. 2017. Towards interface-driven design of evolving component-based architectures, See [101], 121–148. DOI: https://doi.org/10.1007/978-3-319-48628-4_6

[62] Z. Chen, A. Hannousse, D. V. Hung, I. Knoll, X. Li, Z. Liu, Y. Liu, Q. Nan, J. C. Okika, A. P. Ravn, et al. 2008. Modelling with relational calculus of object and component systems – rCOS. In *Proceedings of the Common Component Modeling Example: Comparing Software Component Models*. A. Rausch, R. Reussner, R. Mirandola, and F. Plášil (Eds.), LNCS, Vol. 5153, Springer, 116–145. DOI: https://doi.org/10.1007/978-3-540-85289-6_6

[63] Z. Chen, X. Li, Z. Liu, V. Stolz, and L. Yang. 2007. Harnessing rCOS for tool support – The CoCoME experience. In *Proceedings of the Formal Methods and Hybrid Real-Time Systems, Essays in Honor of Dines Bjørner and Chaochen Zhou on the Occasion of Their 70th Birthdays, Papers presented at a Symposium held in Macao*. C. B. Jones, Z. Liu, and J. Woodcock (Eds.), LNCS, Vol. 4700, Springer, 83–114. DOI: https://doi.org/10.1007/978-3-540-75221-9_5

[64] Z. Chen, Z. Liu, A. P. Ravn, V. Stolz, and N. Zhan. 2009. Refinement and verification in component-based model-driven design. *Science of Computer Programming* 74, 4 (2009), 168–196. DOI: https://doi.org/10.1016/j.scico.2008.08.003 Special Issue on the Grand Challenge.

[65] T. Denvir, J. He, C. B. Jones, A. W. Roscoe, J. Stoy, B. Sufrin, and J. P. Bowen. 2024. Tony hoare @ 90. *FACS FACTS* 2024, 2 (2024), 5–42. Retrieved from https://www.bcs.org/media/1wrosrpv/facs-jul24.pdf#page=5.

[66] H. Dierks. 1999. Synthesizing controllers from real-time specifications. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 18, 1 (1999), 33–43. DOI: https://doi.org/10.1109/43.739057

[67] Directorate General XIII (Ed.). 1990. *Synopses of Basic Research Actions and Working Groups.* ESPRIT Synopses, Vol. 8. Commission of the Europeam Communities. Telecommuncations, Information Industries and Innovation.

[68] Directorate General XIII (Ed.). 1993. *IT R&D Programme: Basic Research – Summaries of Esprit Projects, Working Groups and Networks of Excellence.* Summaries of Esprit project, working groups and networks of excellence, Vol. EUR 15375 EN/1. European Commission. Telecommuncations, Information Market and Exploitation of Research.

[69] M. Fränzle. 1997. *Controller Design from Temporal Logic: Undecidability need not matter.* Ph. D. Dissertation. University of Kiel, Germany. Retrieved from https://d-nb.info/951730746

[70] M. Fränzle. 1999. Analysis of hybrid systems: An ounce of realism can save an infinity of states. In *Proceedings of the International Workshop on Computer Science Logic.*, J. Flum and M. Rodríguez-Artalejo (Eds.), LNCS, Vol. 1683, Springer, 126–140. DOI: https://doi.org/10.1007/3-540-48168-0_10

[71] M. Fränzle. 2002. Take it NP-easy: Bounded model construction for Duration Calculus. In *Proceedings of the International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems.* W. Damm and E.-R. Olderog (Eds.), LNCS, Vol. 2469, Springer, 245–264. DOI: https://doi.org/10.1007/3-540-45739-9_16

[72] M. Fränzle. 2004. Model-checking dense-time Duration Calculus. *Formal Aspects of Computing* 16, 2 (2004), 121–139. DOI: https://doi.org/10.1007/S00165-004-0032-Y

[73] M. Fränzle and M. R. Hansen. 2005. A robust interpretation of Duration Calculus. In *Proceedings of the 2nd International Colloquium on Theoretical Aspects of Computing.* D. V. Hung and M. Wirsing (Eds.), LNCS, Vol. 3722, Springer, 257–271. DOI: https://doi.org/10.1007/11560647_17

[74] M. Fränzle and M. R. Hansen. 2007. Deciding an interval logic with accumulated durations. In *Proceedings of the 13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems.* O. Grumberg and M. Huth (Eds.), LNCS, Vol. 4424, Springer, 201–215. DOI: https://doi.org/10.1007/978-3-540-71209-1_17

[75] M. Fränzle and M. R. Hansen. 2009. Efficient model checking for Duration Calculus. *International Journal of Software and Informatics* 3, 2–3 (2009), 171–196.

[76] M. Fränzle, M. R. Hansen, and H. Ody. 2015. No need knowing numerous neighbours. In *Proceedings of the Correct System Design: Symposium in Honor of Ernst-Rüdiger Olderog on the Occasion of His 60th Birthday.* R. Meyer, A. Platzer, and H. Wehrheim (Eds.), Springer, 152–171. DOI: https://doi.org/10.1007/978-3-319-23506-6_11

[77] M.-C. Gaudel and J. Woodcock (Eds.). 1996. *FME'96: Industrial Benefit and Advances in Formal Methods.* LNCS, Vol. 1051, Springer. DOI: https://doi.org/10.1007/3-540-60973-3

[78] G. Goos and W. Zimmermann. 1999. Verification of compilers. In *Proceedings of the Correct System Design, Recent Insight and Advances (to Hans Langmaack on the Occasion of his Retirement from his Professorship at the University of Kiel).* E.-R. Olderog and B. Steffen (Eds.), LNCS, Vol. *1710,* Springer, 201–230. DOI: https://doi.org/10.1007/3-540-48092-7_10

[79] H. Gottliebsen, R. Hardy, O. Lightfoot, and U. Martin. 2013. Applications of real number theorem proving in PVS. *Formal Aspects of Computing* 25 (2013), 993–1016. DOI: https://doi.org/10.1007/s00165-012-0232-9

[80] A. A. Grau, U. Hill, and H. Langmaack. 1967. *Handbook of Automatic Computation I b: Translation of Algol 60.* Grundlehren der mathematischen Wissenschaften, Vol. 137, Springer, Berlin/New York.

[81] A. Griesmayer, Z. Liu, C. Morisset, and S. Wang. 2013. A framework for automated and certified refinement steps. *Innovations in Systems and Software Engineering* 9, 1 (2013), 3–16. DOI: https://doi.org/10.1007/S11334-012-0183-6

[82] R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel (Eds.). 1993. *Hybrid Systems.* LNCS, Vol. 736, Springer. DOI: https://doi.org/10.1007/3-540-57318-6

[83] K. M. Hansen, A. P. Ravn, and V. Stavridou. 1998. From safety analysis to software requirements. *IEEE Transactions on Software Engineering* 24, 7 (1998), 573–584. DOI: https://doi.org/10.1109/32.708570

[84] M. R. Hansen and D. V. Hung. 2007. A theory of Duration Calculus with application. In *Domain Modeling and the Duration Calculus, International Training School, Shanghai, China, September 17–21. 2007, Advanced Lectures.* C. George, Z. Liu, and J. Woodcock (Eds.), LNCS, Vol. *4710,* Springer, 119–176. DOI: https://doi.org/10.1007/978-3-540-74964-6_3

[85] M. R. Hansen, P. K. Pandya, and C. Zhou. 1995. Finite divergence. *Theoretical Computer Science* 138, 1 (1995), 113–139. DOI: https://doi.org/10.1016/0304-3975(94)00145-9

[86] M. R. Hansen and C. Zhou. 1991. Semantics and completeness of Duration Calculus. In *Proceedings of the Workshop/School/Symposium of the REX Project (Research and Education in Concurrent Systems).* J. W. de Bakker, C. Huizing, W.-P. de Roever, and G. Rozenberg (Eds.), LNCS, Vol. *600,* Springer, 209–225. DOI: https://doi.org/10.1007/BFB0031994

[87] M. R. Hansen and C. Zhou. 1997. Duration Calculus: Logical foundations. *Formal Aspects of Computing* 9, 3 (1997), 283–330. DOI: https://doi.org/10.1007/BF01211086

[88] H. Hansson and B. Jonsson. 1989. A framework for reasoning about time and reliability. In *Proceedings of the 10th IEEE Real-Time System Symposium.* IEEE, 102–111. DOI: https://doi.org/10.1109/REAL.1989.63561

[89] H. Hansson and B. Jonsson. 1994. A logic for reasoning about time and reliability. *Formal Aspects of Computing* 6, 5 (1994), 512–535. DOI: https://doi.org/10.1007/BF01211866

[90] J. He (Ed.). 1994. *Provably Correct Systems: Modelling of Communication Languages and Design of Optimized Compilers.* McGraw-Hill.

[91] J. He and J. P. Bowen. 1991. *Compiling Specification for ProCoS Language PLR_0*. ProCoS document [OU HJF 6]. Oxford University Computing Laboratory. Retrieved from http://researchgate.net/publication/319069362
[92] J. He and J. P. Bowen. 1992. Time interval semantics and implementation of a real-time programming language. In *Proceedings of the 4th Euromicro Workshop on Real-Time Systems*. IEEE, 110–115. DOI: https://doi.org/10.1109/EMWRT.1992.637480
[93] J. He and J. P. Bowen. 1994. Specification, verification and prototyping of an optimized compiler. *Formal Aspects of Computing* 6, 6 (1994), 643–658. DOI: https://doi.org/10.1007/BF03259390
[94] J. He, C. A. R. Hoare, M. Fränzle, M. Müller-Olm, E.-R. Olderog, M. Schenke, M. R. Hansen, A. P. Ravn, and H. Rischel. 1994. Provably correct systems, See [118], 288–335. DOI: https://doi.org/10.1007/3-540-58468-4
[95] J. He, C. A. R. Hoare, M. Müller-Olm, E.-R. Olderog, M. Schenke, M. R. Hansen, A. P. Ravn, and H. Rischel. 1996. The ProCoS Approach to the Design of Real-Time Systems: Linking Different Formalisms. (March 1996). Presented as a tutorial at the FME'96 symposium [77].
[96] J. He, X. Li, and Z. Liu. 2005. Component-based software engineering. In *Proceedings of the 2nd International Colloquium on Theoretical Aspects of Computing*. , D. V. Hung and M. Wirsing (Eds.), LNCS, Vol. 3722, Springer, 70–95. DOI: https://doi.org/10.1007/11560647_5
[97] J. He, X. Li, and Z. Liu. 2006. rCOS: A refinement calculus of object systems. *Theoretical Computer Science* 365, 1-2 (2006), 109–142. DOI: https://doi.org/10.1016/J.TCS.2006.07.034
[98] J. He, Z. Liu, X. Li, and S. Qin. 2004. A relational model for object-oriented designs. In *Proceedings of the 2nd Asian Symposium on Programming Languages and Systems*. W.-N. Chin (Ed.), LNCS, Vol. *3302,* Springer, 415–436. DOI: https://doi.org/10.1007/978-3-540-30477-7_28
[99] J. He, I. Page, and J. P. Bowen. 1993. Towards a provably correct hardware implementation of Occam. In *Proceedings of the Correct Hardware Design and Verification Methods*. G. J. Milne and L. Pierre (Eds.), LNCS, Vol. *683,* Springer, 214–225. DOI: https://doi.org/10.1007/BFb0021726
[100] M. Hilscher, S. Linker, E.-R. Olderog, and A. P. Ravn. 2011. An abstract model for proving safety of multi-lane traffic manoeuvres. In *Proceedings of the International Conference on Formal Engineering Methods*. S. Qin and Z. Qiu (Eds.), LNCS, Vol. *6991,* Springer, 404–419. DOI: https://doi.org/10.1007/978-3-642-24559-6_28
[101] M. G. Hinchey, J. P. Bowen, and E.-R. Olderog (Eds.). 2017. *Provably Correct Systems*. Springer. DOI: https://doi.org/10.1007/978-3-319-48628-4
[102] C. A. R. Hoare. 1985. *Communicating Sequential Processes*. Prentice Hall International.
[103] C. A. R. Hoare. 1985. The mathematics of programming. In *Proceedings of the 5th International Conference on Foundations of Software Technology and Theoretical Computer Science*. S. N. Maheshwari (Ed.), LNCS, Vol. *206,* Springer, 1–18. DOI: https://doi.org/10.1007/3-540-16042-6_1
[104] C. A. R. Hoare. 2002. Towards the verifying compiler. In *Formal Methods at the Crossroads. From Panacea to Foundational Support, 10th Anniversary Colloquium of UNU/IIST, the International Institute for Software Technology of The United Nations University, 2002, Revised Papers*. B. K. Aichernig and T. S. E. Maibaum (Eds.), LNCS, Vol. *2757,* Springer, 151–160. DOI: https://doi.org/10.1007/978-3-540-40007-3_10
[105] C. A. R. Hoare. 2003. The verifying compiler: A grand challenge for computing research. In *Compiler Construction, 12th International Conference, CC 2003, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2003, Warsaw, Poland, April 7–11, 2003, Proceedings*., G. Hedin (Ed.), LNCS, Vol. *2622,* Springer, 262–272. DOI: https://doi.org/10.1007/3-540-36579-6_19
[106] C. A. R. Hoare et al. 1993. Project 7071: Provably correct systems – PROCOS II. See [68], 280–283.
[107] C. A. R. Hoare and J. P. Bowen. 1991. ProCoS II research project – parts A and B. (1991). Oxford University Computing Laboratory, UK.
[108] C. A. R. Hoare and J. P. Bowen. 1992. Basic Research Project 7071 ProCoS II Technical Annex. (1992). Oxford University Computing Laboratory, UK.
[109] C. A. R. Hoare and J. He. 1998. *Unifying Theories of Programming*. Prentice Hall Europe.
[110] C. A. R. Hoare, J. He, J. P. Bowen, and P.K. Pandya. 1990. An algebraic approach to verifiable compiling specification and prototyping of the ProCoS level 0 programming language. In *Proceedings of the Annual ESPRIT Conference Brussels*. Directorate-General XIII of the Commission of the European Communities (Ed.), Kluwer Academic Publishers, 804–818. DOI: https://doi.org/10.1007/978-94-009-0705-8_65
[111] J. Hoenicke. 2006. *Combination of Processes, Data, and Time*. Ph. D. Dissertation. University of Oldenburg, Germany. Retrieved from https://d-nb.info/981576087 Report Nr. 9/2006.
[112] C. Hollings, U. Martin, and A. Rice. 2018. *Ada Lovelace: The Making of a Computer Scientist*. Bodleian Library, Oxford.
[113] INMOS Limited. 1988. *Occam 2 Reference Manual*. Prentice Hall International.
[114] INMOS Limited. 1988. *Transputer Instruction Set – A Compiler Writer's Guide*. Prentice Hall International.
[115] C. B. Jones and M. R. Hansen. 2004. Editorial. *Formal Aspects of Computing* 16, 2 (2004), 95. DOI: https://doi.org/10.1007/s00165-004-0047-4 Duration Calculus: A Logical Approach to Real-Time Systems.

[116] C. B. Jones, Z. Liu, and J. Woodcock (Eds.). 2007. *Formal Methods and Hybrid Real-Time Systems, Essays in Honor of Dines Bjørner and Chaochen Zhou on the Occasion of Their 70th Birthdays, Papers presented at a Symposium held in Macao, China, September 24–25, 2007.* LNCS, Vol. 4700, Springer. DOI: https://doi.org/10.1007/978-3-540-75221-9

[117] H. Langmaack. 1997. The ProCoS approach to correct systems. *Real Time Systems* 13, 3 (1997), 253–275. DOI: https://doi.org/10.1023/A:1007963427189

[118] H. Langmaack, W.-P. de Roever, and J. Vytopil (Eds.). 1994. *Formal Techniques in Real-Time and Fault-Tolerant Systems, Third International Symposium Organized Jointly with the Working Group Provably Correct Systems – ProCoS, Lübeck, Germany, September 19–23, Proceedings.* LNCS, Vol. 863, Springer. DOI: https://doi.org/10.1007/3-540-58468-4

[119] H. Langmaack and A. P. Ravn. 1994. The ProCoS project: Provably correct systems. See [20], 249–265. Appendix B.

[120] K. G. Larsen, P. Petterson, and Y. Wang. 1997. Uppaal in a nutshell. *International Journal on Software Tools for Technology Transfer* 1 (1997), 134–152. DOI: https://doi.org/10.1007/S100090050010

[121] P. G. Larsen (Ed.). 1993. *Tutorial Material for FME'93: Industrial-Strength Formal Methods. Proc. 1st International Symposium of Formal Methods Europe, Odense, Denmark.*

[122] D. Li, X. Li, Z. Liu, and V. Stolz. 2011. Interactive transformations from object-oriented models to component-based models. In *Proceedings of the International Workshop on Formal Aspects of Component Software.* F. Arbab and P. Csaba Ölveczky (Eds.), LNCS, Vol. *7253,* Springer, 97–114. DOI: https://doi.org/10.1007/978-3-642-35743-5_7

[123] D. Li, X. Li, Z. Liu, and V. Stolz. 2013. Support formal component-based development with UML profile. In *Proceedings of the 22nd Australian Conference on Software Engineering.* IEEE Computer Society, 191–200. DOI: https://doi.org/10.1109/ASWEC.2013.31

[124] X. Li. 1993. *A Mean Value Calculus.* Ph. D. Dissertation. Software Institute, Chinese Academy of Sciences, Beijing, China.

[125] X. Li and Z. Liu. 2007. Prototyping system requirements model. In *Proceedings of the 1st International Workshop on Harnessing Theories for Tool Support in Software.* G. Pu and V. Stolz (Eds.), Electronic Notes in Theoretical Computer Science, Vol. *207,* Elsevier, 17–32. DOI: https://doi.org/10.1016/J.ENTCS.2008.03.083

[126] X. Li, Z. Liu, and J. He. 2001. Formal and use-case driven requirement analysis in UML. In *Proceedings of the 25th International Computer Software and Applications Conference.* IEEE Computer Society, 215–224. DOI: https://doi.org/10.1109/CMPSAC.2001.960619

[127] X. Li, Z. Liu, and J. He. 2004. A formal semantics of UML sequence diagram. In *Proceedings of the 15th Australian Software Engineering Conference.* IEEE Computer Society, 168–177. DOI: https://doi.org/10.1109/ASWEC.2004.1290469

[128] Z. Liu. 1991. *Fault-tolerant Programming By Transformations.* Ph. D. Dissertation. University of Warwick, United Kingdom.

[129] Z. Liu. 2023. Linking formal methods in software development – A reflection on the development of rCOS, See [45], 52–84. DOI: https://doi.org/10.1007/978-3-031-40436-8_3

[130] Z. Liu. 2025. The ProCoS project and Duration Calculus: A personal memoir. *FACS FACTS* 2025, 2 (2025), 13–27. Retrieved from https://www.bcs.org/media/yd4ocehl/facs-jul25.pdf

[131] Z. Liu, J. He, and X. Li. 2004. Contract oriented development of component software. In *Proceedings of the Exploring New Frontiers of Theoretical Informatics, IFIP 18th World Computer Congress, TC1 3rd International Conference on Theoretical Computer Science..* J.-J. Lévy, E. W. Mayr, and J. C. Mitchell (Eds.), IFIP, Vol. 155, Kluwer/Springer, 349–366. DOI: https://doi.org/10.1007/1-4020-8141-3_28

[132] Z. Liu, J. He, and X. Li. 2004. rCOS: Refinement of component and object systems. In *Proceedings of the 3rd International Symposium on Formal Methods for Components and Objects.* F. S. de Boer, M. M. Bonsangue, S. Graf, and W.-P. de Roever (Eds.), LNCS, Vol. *3657,* Springer, 183–221. DOI: https://doi.org/10.1007/11561163_9

[133] Z. Liu, J. He, X. Li, and Y. Chen. 2003. A relational model for formal object-oriented requirement analysis in UML. In *Proceedings of the 5th International Conference on Formal Engineering Methods.* J. S. Dong and J. Woodcock (Eds.), LNCS, Vol. *2885,* Springer, 641–664. DOI: https://doi.org/10.1007/978-3-540-39893-6_36

[134] Z. Liu and M. Joseph. 1992. Transformation of programs for fault-tolerance. *Formal Aspects of Computing* 4, 5 (1992), 442–469. DOI: https://doi.org/10.1007/BF01211393

[135] Z. Liu and M. Joseph. 1999. Specification and verification of fault-tolerance, timing, and scheduling. *ACM Transactions on Programming Language Systems* 21, 1 (1999), 46–89. DOI: https://doi.org/10.1145/314602.314605

[136] Z. Liu, X. Li, and J. He. 2002. Using transition systems to unify UML models. In *Proceedings of the International Conference on Formal Engineering Methods.* C. George and H. Miao (Eds.), LNCS, Vol. *2495,* Springer, 535–547. DOI: https://doi.org/10.1007/3-540-36103-0_54

[137] Z. Liu, V. Mencl, A. P. Ravn, and L. Yang. 2006. Harnessing theories for tool support. In *Proceedings of the 2nd International Symposium on Leveraging Applications of Formal Methods, Verification and Validation.* IEEE Computer Society, 371–382. DOI: https://doi.org/10.1109/ISOLA.2006.49

[138] Z. Liu, A. P. Ravn, E. V. Sørensen, and C. Zhou. 1992. A probabilistic Duration Calculus. In *Proceedings of the Responsive Computer Systems.* H. Kopetz and Y. Kakuda (Eds.), Vol. 7, Dependable Computing and Fault-Tolerant Systems, Springer, 29–52. DOI: https://doi.org/10.1007/978-3-7091-9288-7_3

[139] Z. Liu, A. P. Ravn, E. V. Sørensen, and C. Zhou. 1994. Towards a calculus of system dependability. *High Integrity Systems* 1, 1 (1994), 49–75.
[140] Z. Liu and R. Venkatesh. 2005. Methods and tools for formal software engineering. In *Proceedings of the 1st Working Conference on Verified Software: Theories, Tools, and Experiments.* B. Meyer and J. Woodcock (Eds.), LNCS, Vol. 4171, Springer, 31–41. DOI: https://doi.org/10.1007/978-3-540-69149-5_4
[141] Z. Liu and J. Wang. 2020. Human-cyber-physical systems: Concepts, challenges, and research opportunities. *Frontiers of Information Technology and Electronic Engineering* 21, 11 (2020), 1535–1553. DOI: https://doi.org/10.1631/FITEE.2000537
[142] U. Martin and T. Nipkow. 1989. Boolean unification – The story so far. *Journal of Symbolic Computation* 7, 3–4 (1989), 275–293. DOI: https://doi.org/10.1016/S0747-7171(89)80013-6
[143] D. May. 1983. OCCAM. *ACM SIGPLAN Notices* 18, 4 (1983), 69–79. DOI: https://doi.org/10.1145/948176.948183
[144] B. Meyer. 1991. Design by contract. In *Proceedings of the Advances in Object-Oriented Software Engineering.* Prentice Hall, 1–50.
[145] B. Meyer and J. Woodcock (Eds.). 2008. *Verified Software: Theories, Tools, Experiments, 1st IFIP TC 2/WG 2.3 Conference, VSTTE 2005, Zurich, Switzerland, 2005, Revised Selected Papers and Discussions.* LNCS, Vol. 4171, Springer. DOI: https://doi.org/10.1007/978-3-540-69149-5
[146] M. Müller-Olm. 1997. *Modular Compiler Verification – A Refinement-Algebraic Approach Advocating Stepwise Abstraction.* LNCS, Vol. 1283, Springer. DOI: https://doi.org/10.1007/BFB0027453
[147] J. E. Nicholls (Ed.). 1990. *Z User Workshop, Oxford 1989: Proceedings of the 4th Annual Z User Meeting Oxford, 15 December 1989.* Springer. DOI: https://doi.org/10.1007/978-1-4471-3877-8
[148] H. Ody, M. Fränzle, and M. R. Hansen. 2016. Discounted Duration Calculus. In *Proceedings of the 27th Nordic Workshop on Programming Theory.* J. S. Fitzgerald, C. L. Heitmeyer, S. Gnesi, and A. Philippou (Eds.), LNCS, Vol. *9995,* 577–592. DOI: https://doi.org/10.1007/978-3-319-48989-6_35
[149] E.-R. Olderog. 1991. *Nets, Terms and Formulas: Three Views of Concurrent Processes and Their Relationship.* Cambridge University Press. DOI: https://doi.org/10.1017/CBO9780511526589
[150] E.-R. Olderog and H. Dierks. 2008. *Real-Time Systems: Formal Specification and Automatic Verification.* Cambridge University Press. DOI: https://doi.org/10.1017/CBO9780511619953
[151] E.-R. Olderog, A. P. Ravn, and J. U. Skakkebæk. 1996. Refining system requirements to program specifications. In *Proceedings of the Formal Methods for Real-Time Computing.* C. Heitmeyer and D. Mandrioli (Eds.), Trends in Software, Vol. 5, John Wiley & Sons, 107–134.
[152] E.-R. Olderog, A. P. Ravn, and R. Wisniewski. 2017. Linking discrete and continuous models, applied to traffic manoeuvrers. See [101], 95–120. DOI: https://doi.org/10.1007/978-3-319-48628-4_5
[153] E.-R. Olderog and S. Rössig. 1993. A case study in transformational design of concurrent systems. In *Proceedings of the Colloquium on Trees in Algebra and Programming.* M.-C. Gaudel and J.-P. Jouannaud (Eds.), LNCS, Vol. 668, Springer, 90–104. DOI: https://doi.org/10.1007/3-540-56610-4_58
[154] E. Palomar, X. Chen, Z. Liu, S. Maharjan, and J. P. Bowen. 2016. Component-based modelling for scalable smart city systems interoperability: A case study on integrating energy demand response systems. *Sensors* 16, 11 (2016), 1810. DOI: https://doi.org/10.3390/s16111810
[155] E. Palomar, Z. Liu, J. P. Bowen, Y. Zhang, and S. Maharjan. 2014. Component-based modelling for sustainable and scalable smart meter networks. In *Proceeding of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks.* IEEE Computer Society, 1–6. DOI: https://doi.org/10.1109/WOWMOM.2014.6918927
[156] P. K. Pandya. 2001. Specifying and deciding quantified discrete-time Duration Calculus formulae using DCVALID. In *Proceedings of the Workshop of Real-Time Tools.* P. Pettersson (Ed.), Retrieved from https://www.researchgate.net/publication/2422376 Satellite event of CONCUR 2001: 12th Interational Conference on Concurrency Theory.
[157] P. K. Pandya and J. P. Bowen. 1989. *An Operational Semantics for the ProCoS Level 0 Assembly Language.* ProCoS document [OU PKP 1]. Oxford University Computing Laboratory.
[158] W. J. Paul. 2008. Towards a worldwide verification technology. In *Proceedings of the Working Conference on Verified Software: Theories, Tools, and Experiments.* B. Meyer and J. Woodcock (Eds.), LNCS, Vol. *4171,* Springer, 19–25. DOI: https://doi.org/10.1007/978-3-540-69149-5_2
[159] A. Pnueli. 1977. The temporal logic of programs. In *Proceedings of the 18th Annual Symposium on Foundations of Computer Science.* IEEE Computer Society, 46–57. DOI: https://doi.org/10.1109/SFCS.1977.32
[160] A. Post, J. Hoenicke, and A. Podelski. 2011. rt-inconsistency: A new property for real-time requirements. In *Proceedings of the International Conference on Fundamental Approaches to Software Engineering.* D. Giannakopoulou and F. Orejas (Eds.), LNCS, Vol. *6603,* Springer, 34–49. DOI: https://doi.org/10.1007/978-3-642-19811-3_4
[161] J. Protzenko and A. Raad (Eds.). 2025. *Verified Software. Theories, Tools and Experiments – 16th International Conference, VSTTE 2024, Prague, Czech Republic, 2024, Revised Selected Papers.* LNCS, Vol. 15525, Springer. DOI: https://doi.org/10.1007/978-3-031-86695-1

[162] A. P. Ravn. 1995. *Design of Embedded Real-Time Computing Systems.* Technical Report ID-TR 1995-170. Technical University of Denmark. Thesis of the degree of Doctor Technices.

[163] A. P. Ravn et al. 1993. Tutorial material, formal methods europe '93: Industrial-strength formal methods. In *Proceedings of the Provably Correct Systems.* P. G. Larsen (Ed.), Formal Methods Europe, Odense, Denmark, 437–450.

[164] A. P. Ravn, H. Rischel, and K. M. Hansen. 1993. Specifying and verifying requirements of real-time systems. *IEEE Transactions on Software Engineering* 19, 1 (1993), 41–55. DOI: https://doi.org/10.1109/32.210306

[165] A. Schäfer. 2007. Axiomatisation and decidability of multi-dimensional Duration Calculus. *Information and Computation* 205, 1 (2007), 25–64. DOI: https://doi.org/10.1016/J.IC.2006.08.005

[166] M. Schenke. 1999. Transformational design of real-time systems. Part II: From program specifications to programs. *Acta Informatica* 36, 1 (1999), 67–96. DOI: https://doi.org/10.1007/S002360050154

[167] M. Schenke and E.-R. Olderog. 1999. Transformational design of real-time systems. Part I: From requirements to program specifications. *Acta Informatica* 36, 1 (1999), 1–65. DOI: https://doi.org/10.1007/S002360050153

[168] M. Schenke and A. P. Ravn. 1996. Refinement from a control problem to programs. In *Proceedings of the Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control.* J.-R. Abrial, E. Börger, and H. Langmaack (Eds.), Springer, 403–427. DOI: https://doi.org/10.1007/BFb0027247

[169] F. Sheng, H. Zhu, J. He, Z. Yang, and J. P. Bowen. 2019. Theoretical and practical aspects of linking operational and algebraic semantics for MDESL. *ACM Transactions on Software Engineering and Methodology* 28, 3 (2019), 14. DOI: https://doi.org/10.1145/3295699

[170] L. Shustek, C. A. R. Hoare, and J. P. Bowen. 2009. An interview with C.A.R. Hoare. *Communications of the ACM* 52, 3 (2009), 38–41. DOI: https://doi.org/10.1145/1467247.1467261

[171] J. U. Skakkebæk. 1994. *A Verification Assistant for a Real-Time Logic.* Doctoral dissertation. Technical University of Denmark.

[172] E. V. Sørensen, J. Nordahl, and N. H. Hansen. 1993. From CSP models to Markov models. *IEEE Transactions on Software Engineering* 19, 6 (1993), 554–570. DOI: https://doi.org/10.1109/32.232021

[173] J. Tucker et al. 2007. History of Computing Collection. Swansea University. Retrieved from https://collections.swansea.ac.uk/s/history-of-computing-collection/

[174] B. von Karger. 1998. Temporal algebra. *Mathematical Structures in Computer Science* 8, 3 (1998), 277–320. DOI: https://doi.org/10.1017/S0960129598002540

[175] B. von Karger. 2002. Temporal algebra. In *Proceedings of the Algebraic and Coalgebraic Methods in the Mathematics of Program Construction, International Summer School and Workshop.* R. C. Backhouse, R. L. Crole, and J. Gibbons (Eds.), LNCS, Vol. *2297,* Springer, 309–385. DOI: https://doi.org/10.1007/3-540-47797-7_9

[176] D. Weber-Wulff. 1997. *Contributions to Mechanical Proofs of Correctness for Compiler Front Ends.* Ph. D. Dissertation. University of Kiel, Germany. Retrieved from https://d-nb.info/951730800

[177] C. Whitby-Strevens. 1985. The transputer. In *Proceedings of the 12th Annual Symposium on Computer Architecture .* T. F. Gannon, T. Agerwala, and C. V. Freiman (Eds.), IEEE Computer Society, 292–300. DOI: https://doi.org/10.1145/327070.327269

[178] J. Woodcock. 2021. Hoare and He's unifying theories of programming. In *Proceedings of the Theories of Programming: The Life and Works of Tony Hoare.* C. B. Jones and J. Misra (Eds.), Association for Computing Machinery, 285–316. DOI: https://doi.org/10.1145/3477355.3477369

[179] W. D. Young. 1994. System verification and the CLI stack. See [20], 225–248. Appendix A.

[180] N. Zhan. 2000. *Higher-order Duration Calculus and its Applications.* Ph. D. Dissertation. Software Institute, Chinese Academy of Sciences, Beijing, China. In Chinese.

[181] C. Zhou and M. R. Hansen. 1997. An adequate first order interval logic. In *Proceedings of the International Symposium on Compositionality.* W.-P. de Roever, H. Langmaack, and A. Pnueli (Eds.), LNCS, Vol. *1536,* Springer, 584–608. DOI: https://doi.org/10.1007/3-540-49213-5_23

[182] C. Zhou and M. R. Hansen. 2004. *Duration Calculus – A Formal Approach to Real-Time Systems.* Springer. DOI: https://doi.org/10.1007/978-3-662-06784-0

[183] C. Zhou, M. R. Hansen, A. P. Ravn, and H. Rischel. 1991. Duration specifications for shared processors. In *Proceedings of the Formal Techniques in Real-Time and Fault-Tolerant Systems.* J. Vytopil (Ed.), LNCS, Vol. 571, Springer, 21–32. DOI: https://doi.org/10.1007/3-540-55092-5_2

[184] C. Zhou, M. R. Hansen, and P. Sestoft. 1993. Decidability and uNdecidability results for Duration Calculus. In *Proceedings of the 10th Annual Symposium on Theoretical Aspects of Computer Science.* P. Enjalbert, A. Finkel, and K. W. Wagner (Eds.), LNCS, Vol. *665,* Springer, 58–68. DOI: https://doi.org/10.1007/3-540-56503-5_8

[185] C. Zhou, C. A. R. Hoare, and A. P. Ravn. 1991. A calculus of durations. *Information Processing Letters* 40, 5 (1991), 269–276. DOI: https://doi.org/10.1016/0020-0190(91)90122-X

[186] C. Zhou, W. Ji, and A. P. Ravn. 1996. A formal description of hybrid systems. In *Proceedings of the Hybrid Systems III: Verification and Control.* R. Alur, T. A. Henzinger, and E. D. Sontag (Eds.), LNCS, Vol. *1066,* Springer, 511–530. DOI: https://doi.org/10.1007/BFb0020972

[187] C. Zhou, A. P. Ravn, and M. R. Hansen. 1993. An extended Duration Calculus for hybrid real-time systems, See [82], 36–59. DOI: https://doi.org/10.1007/3-540-57318-6

[188] H. Zhu, J. He, and J. P. Bowen. 2008. From algebraic semantics to denotational semantics for Verilog. *Innovations in Systems and Software Engineering: A NASA Journal* 4, 4 (2008), 341–360. DOI: https://doi.org/10.1007/s11334-008-0069-9

[189] H. Zhu, J. He, J. Li, and J. P. Bowen. 2011. Algebraic approach to linking the semantics of web services. *Innovations in Systems and Software Engineering: A NASA Journal* 7, 3 (2011), 209–224. DOI: https://doi.org/10.1007/s11334-011-0172-1

[190] H. Zhu, S. Qin, J. He, and J. P. Bowen. 2009. PTSC: Probability, time and shared-variable concurrency. *Innovations in Systems and Software Engineering: A NASA Journal* 5, 4 (2009), 271–284. DOI: https://doi.org/10.1007/s11334-009-0100-9

[191] H. Zhu, F. Yang, J. He, J. P. Bowen, J. W. Sanders, and S. Qin. 2012. Linking operational semantics and algebraic semantics for a probabilistic timed shared-variable language. *The Journal of Logic and Algebraic Programming* 81, 1 (2012), 2–25. DOI: https://doi.org/10.1016/j.jlap.2011.06.003

Received 3 July 2025; revised 13 January 2026; accepted 17 March 2026

Author Queries

Q1: AU: Please verify that the title, footnotes, author names, and affiliations are correct as set. If any changes are needed to the author list, please provide them, explain the reasons for the changes, and per journal policy, please provide e-mails from all authors noting that all changes have their approval.

Q2: AU: Please verify that the mailing address and e-mail address for correspondence are correct as set.

Q3: AU: Please confirm that all of the abbreviations and expansions are correct as they appear and used consistently throughout.

Q4: AU: If your proof includes supplementary material, please confirm that all supplementary material is cited in the text.

Q5: AU: If your proof includes links to Web sites, please verify that the links are valid and will direct readers to the correct Web page.

Q6: AU: Please review the alt-text you provided for figures in your article and confirm that it is correct. If alt-text is missing for any of the figures, please add the alt-text [via vendor-defined mechanism] in the proof, as alt-text is required for all figures. Alt-text is a brief description of the image that is read by screen readers or other assistive technology. It provides information about an image's purpose for the reader and provides context on how the image relates to the page content.

Q7: AU: If your proof includes figures, please check the figures in your proof carefully. If any changes are needed, please provide a revised figure file.

Q8: AU: Please provide the names of all authors in references [6], [8], [106], [163], and [173].

Q9: AU: Please provide the issue number in references [7], [17], [18], [22], [43], [44], [49], [50], [79], and [120].

Q10: AU: Please provide the access date in reference [173].