Formal Methods Communities of Practice: A Survey of Personal Experience

Jonathan P. Bowen[1,2]
[0000−0002−8748−6140]

and

Peter T. Breuer[3]
[0000−0003−2296−6175]

[1] London South Bank University
School of Engineering
Borough Road, London SE1 1AA, UK

[2] Southwest University
Centre for Research and Innovation in Software Engineering (RISE)
Faculty of Computer and Information Science
Chongqing 400715, China

[3] Hecusys LLC
Atlanta, Georgia, USA

jonathan.bowen@lsbu.ac.uk
http://www.jpbowen.com

Abstract. In this paper, we discuss certain Communities of Practice (CoP) in the field of formal methods, used for software engineering, especially with respect to state-based notations. The multiple communities involved with formal methods are examined here as related CoPs. In this context, the CoPs involved are open communities encouraging participation by al those interested, both in research and application. The authors have both been involved with formal methods over several decades, for most of their careers, and it is hoped that their observations in this paper may help future community building to further the development of formal methods, and software engineering in general. A bibliography is included at the end of the paper.

1 Background

> *. . . a job is about a lot more than a paycheck. It's about your dignity. It's about respect. It's about your place in your community.*
>
> – Joe Biden

The first author of this paper has been involved in building and investigating communities [32], both in the area of formal methods [45], especially the Z notation [51], and also in museum-related [7,60] and arts-related [37,52] contexts.

This has been facilitated by the increasing possibility of worldwide virtual communities without geographic bounds [12]. This paper considers aspects of community building for formal methods, especially in the context of the *Community of Practice* (CoP) approach to understanding the evolution of communities that are based around an area of developing knowledge [90,91]. Note that it is also possible to visualize and formalize communities, especially if online [32,52]. In addition, patterns in citations can be investigated formally [54].

Bowen first became involved with community building at the Oxford University Computing Laboratory's Programming Research Group (PRG) in the late 1980s. He was a Research Officer working on formal methods [69] and specifically the Z notation [26,64] at the time. He also became involved with the European ESPRIT **ProCoS I** and **II** projects on *Provably Correct Systems*, led by Tony Hoare at Oxford, Dines Bjørner at DTH in Denmark, and others in the early 1990s [8,23,48].

The subsequent **ProCoS-WG** Working Group of 25 partners around Europe existed to organize meetings and workshops in the late 1990s [24]. The **ProCoS-WG** final report in 1998 [49] presented comments by members of the group, including those who joined after the start of its formation. For example, Prof. Egon Börger [39] of the University of Pisa in Italy participated at many **ProCoS-WG** meetings. He was an invited speaker at the ZUM'97 conference [47] and, with Jean-Raymond Abrial and Prof. Hans Langmaack of the University of Kiel, he organized an important set of case studies formalizing a Steam Boiler problem in a variety of formal notations [4], including a number of contributions by **ProCoS-WG** members. He used **ProCoS-WG** to present his work on the correctness theorem for a general compilation scheme for compiling Occam programs to Transputer code [17]. The influence of the **ProCoS** initiative has continued for decades after the original projects and Working Group [35,68].

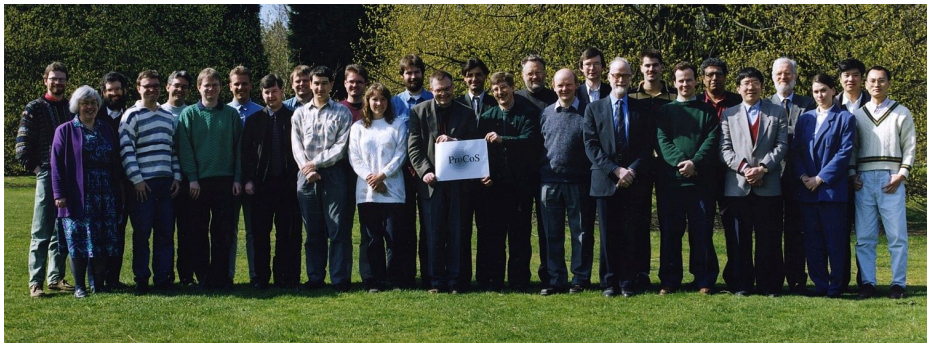

Fig. 1. Participants at the **ProCoS-WG** meeting, held in 1997 at the University of Reading, and co-located with the ZUM'97 conference.

See Figure 1 for a group photograph of participants at the final **ProCoS-WG** workshop held at the University of Reading in the UK during 1997, co-located with the ZUM'97 conference on the Z notation [47]. At this conference, participants in the Z community were introduced to Abstract State Machines (ASM) [19].

In 2006, a number of formal methods researchers contributed to a book on *Software Specification Methods* [61], based around a common case study, including use of the Z notation [31] and ASM [18]. In the same year, a *Festkolloquium* at Schloss Dagstuhl in Germany brought together many of these contributors [6], resulting in an article in a subsequent *Festschift* volume [44].

The BCS-FACS (Formal Aspects of Computing Science) Specialist Group forms a community of those interested in formal methods in the United Kingdom. Such a community depends on a core committee to keep it active. Although it has been in existence since 1978, there as a period when activitied declined in the 1990s. From the early 2000s, a new committee as formed with Bowen as Chair, leading to a renewal of activities. For example, in December 2003, the BCS-FACS Workshop *Teaching Formal Methods: Practice and Experience* was held at Oxford Brookes University [10]. The group also holds regular evening seminars mainly at the BCS London office and selected talks have appeared as chapters in an edited book [11].

In 2008, the newly formed *Abstract State Machines, B and Z: First International Conference, ABZ 2008* started in London, UK, edited by Egon Börger (ASM), Michael Butler (B-Method), myself (Z notation), and Paul Boca as a local organizer [15,16]. This was an extension of the previous ZB conferences, that were a combination of previously separate B and Z conferences. In 2011, a special issue of selected and extended papers from the ABZ 2008 conference was produced for the *Formal Aspects of Computing* journal, edited by Egon Börger, myself, Michael Butler, and Mike Poppleton [14]. More recently, a 2018 book on *Modeling Companion for Software Practitioners* using the ASM approach, co-authored by Egon Börger and Alexander Raschke, has appeared [20,36].

2 State-based Formal Methods

> *The model should not dictate but reflect the problem.*
>
> – Egon Börger [20]

Some key state-based formal methods are summarized in Figure 2. There have been a number of comparative studies of formal methods approaches. For example, a 1996 Steam Boiler Control case study competition book demonstrated different formal methods [4,5]. A 2001 book [59] (second edition in 2006 [61]) presented the questions that should be answered in developing an example invoicing case study using a variety of formal methods.

Formal methods overlap with communities of researchers in other areas that require a rigorous basis to their approaches. For example, formal methods are beneficial for compilers [53,62,63], Hardware Description Languages (HDL) [57,95],

Alloy: Daniel Jackson [71].
ASM: (Abstract State Machines) Egon Börger & Yuri Gurevich [13,20].
B-Method & Event-B: Jean-Raymond Abrial [2,3].
TLA: (Temporal Logic of Actions) Leslie Lamport [75].
VDM: (Vienna Development Method) Dines Bjørner & Cliff Jones [72].
Z: Jean-Raymond Abrial & Mike Spivey [1,70,85].

Fig. 2. Some key state-based formal methods with their major progenators/promulgators.

Human-Cyber-Physical Systems (HCPS) [76], logic programming [22,25,28], safety-critical systems [29,88], security [55,56], software maintenance [40,41,86], software testing [74,87], etc. That said, there can be some resistance to accepting formal methods in some communities within software engineering as a whole [9,46,67]. Education and training are important aspects with respect to the acceptance and promulgation of formal methods [30,34,58].

3 Communities of Practice

> *Everything we do is practice for something greater than where we currently are. Practice only makes for improvement.* – Les Brown

A *Community of Practice* (CoP) [90] is a social science concept for modelling the collaborative activities of professional communities [12] with a common goal over time [91,92]. It can be relevant in a variety of contexts, for example agile methods [73,93], student teaching in higher education [77,78,79,80], and developing large organizations [84]. Information on the successful creation of a CoP is available [89]. CoPs are typically open communities and it is in this context that they are discussed in this paper.

A CoP modelling approach can be used in various scenarios, for example in the context of this paper, formal methods communities [33,35,51]. A CoP consists of:

1. A **domain** of knowledge and interest. In the case of formal methods, this is the application of mathematical approaches to computer-based specification, modelling and development.
2. A **community** based around this domain. For formal methods, like other academically-based disciplines, this includes conference organizers and programme committee members that are interested in formal methods as core facilitators, conference presenters and delegates as participants, as well as other researchers and practitioners involved with developing and using formal methods.
3. The **practice** undertaken by the community in this domain, developing its knowledge, sometimes formalized as a *Body of Knowledge* (BoK) [51]. The

formal methods community encourages the transfer of research ideas into practical use [65,66]. Some formal methods approaches have been used in industrial-scale software-based projects, although information on these can be diffcult to promulgate due to commercial sensitivities and Non-Disclosure Agreements.

There are various stages in the development of a CoP [90]:

1. **Potential:** There needs to be an existing network of people to initiate a CoP. In the case of formal methods as a whole, researchers interested in theoretical computer science, especially discrete mathematics and logic, were the starting point. For example, the inital Z meetings were held in Oxford with an informal proceedings [81,21], due to the location of the Programming Research Group there.
2. **Coalescing:** The community needs to establish a rhythm to ensure its continuation. In the case of many successful formal methods, a regular specialist workshop is typically established initially. For the Z notation, a more formal Z User Meeting (ZUM) was established, together with a Z User Group (ZUG) established in 1992 [50]. Initially meetings were in the United Kingdom, but it then became an international conference in 1995 [43]. Online information was maintained, initially as a FTP service with an associated Z FORUM electronic mailing list [27].
3. **Maturing:** The community must become more enduring. An initial workshop series may become a more formal conference series and establish itself internationally. With maturity, there may be a combining with other formal methods. For example, the International Conference of Z Users became the ZB conference in 2000 [42], combined with the B-Method, and then the ABZ conference, combining ASM, the B-Method, and the Z notation in a single conference in 2008 [15]. This conference has continued through to the present [82]. Another sign of maturity is the production of a standard, e.g., the international ISO/IEC standard for the Z notation [70]. The FTP service became a website and the Z FORUM mailing list was linked with the `comp.specification.z` newsgroup [27] (now part of Google Groups).
4. **Stewardship:** The community needs to respond to its environment and develop appropriately. A particular formal methods community should interact with related organizations, e.g., those associated with similar formal methods. Overall, Formal Methods Europe (FME, `https://www.fmeurope.org`) has acted as a stewarding organization internationally, developing beyond the bounds of Europe, and organizing the regular FME conference from 1993 [94], becoming the FM conference more recently, and following on from the original VDM conferences.
5. **Legacy:** All CoPs eventually end; if successful they morph into further communities. State-based formal methods communities such as those around ASM, B, VDM, Z, etc., have coalesced around the ABZ conference, which continues to this day, as noted above [82]. The various CoPs around these approaches are at different levels of development with respect to their CoP

evoluation. Currently, as of 2021, there is much activity around the B-Method and the related Event-B. Research around ASM is also still active. However, research on VDM and Z is now somewhat dormant. Exactly how all these related communities will continue is something that is worth considering and planning for at the appropriate time.

It remains to be seen precisely what legacy the various state-based formal methods, especially those associated with the ABZ conference, leave in the future. For the moment, the various communities continue to come together through the ABZ conference, as well as other more informal and individual interactions.

It is interesting to reflect on the occurrence of various formal methods and tools in the titles of papers in the two most recent ABZ conference proceedings for 2020 [83] and 2021 [82], as reported by Bowen [38]. Event-B is the most popular formal methods, with 14 papers. 11 papers mention the Rodin tool, providing Event-B tool support. There are nine papers with ASM in the title (including two mentioning the associated ASMETA toolset). Alloy, a Z-like language with tool support, is mentioned in three paper titles, as is the ProB tool providing tool support for B. The Atelier B, UML-B, and UPPAAL tools are each mentioned in one title. TLA, VDM, and Z are not mentioned in any paper titles. So, the "A" (ASM and Alloy) and "B" (mainly Event-B with the associated Rodin and ProB tools) in conference title "ABZ" are still active with respect to research, especially strongly in the cases Event-B/Rodin and ASM. However, the "Z" part of he conference has essentially disappeared. That said, Z is still an inspiration for some formal methods research and is still used in industrial projects, even if not widely publicized. Tools are increasingly important for industrial use of formal methods at scale.

4 Conclusion

Education is for improving the lives of others and for leaving your community and world better than you found it.

– Marian Wright Edelman

As discussed in this paper, there are a number of competing state-based formal methods for modelling computer-based systems. The communities associated with formal methods have developed since the 1980s, such as the Z notation, the B-Method, Event-B, and ASM. Each has their own advantages and disadvantages, which are beyond the scope of this paper to provide in detail. Each have their own community of adherents, that have now somewhat merged with the establishment of the ABZ conference in 2008 [15]. By their nature, formal methods communities tend to be open communities, with participation by both researchers and practitioners actively encouraged.

The various interrelated formal methods communities may be seen as examples of Communities in Practice (CoP) in action. CoPs can potentially merge and create new CoPs. For example, the B-Method and then Event-B were developed after the Z notation largely by the same progenitor, Jean-Raymond Abrial, with some in the Z community subsequently becoming part of the B community.

A Community of Practice depends on people with different skills for success, be it for ideas, vision, organization, etc. Typically, a small number of key personnel are needed for the successful launch of any new formal methods community. Figure 2 provided some key initial personnel for a selection of state-based formal methods. A successful CoP then needs to reach a critical mass in size, following some of the developments covered in this paper.

The earliest formal methods communities were formed around VDM and Z, which grew up in parallel, although Z concentrated on formal specification at a high level with little tool support, whereas VDM also considered refinement towards program code more explicitly. These are now in the late stages of a CoP. ASM developed separately somewhat later and its flexibility has proved useful on modelling systems at a high level. Due to the lack of refinement and tool support in Z, the B-Method and then Event-B were developed to more explicitly handle these aspects with some compromise on the high-level nature of the language. Alloy also provides tool support using a Z-like language that is useful as a prototyping tool, perhaps for parts of a largerly Z specification that could benefit from closer investigation for example. Judging by the activities reported in recent ABZ conferences, Event-B is the most active CoP at the moment.

Predicting the future is always difficult, but formal methods communities have been successful enough to leave their mark on the computer science community as a whole. Certainly the most active formal methods CoPs have shifted from consideration of fundamental ideas to tool support, enabling better potential for industrial usage. The experience of the authors is mainly with state-based formal methods, but the examples in this paper may also be applicable to model-checking communities, for example. In any case, the authors hope that the experiences presented here will help future researchers in developing a Community of Practice based on their own research.

Acknowledgments: The authors are grateful to many formal methods colleagues for inspiration and collaboration over the years. Jonathan Bowen thanks Museophile Limited for financial support.

Bibliography

1. Abrial, J.R.: Data semantics. In: Klimbie, J.W., Koffeman, K.L. (eds.) IFIP TC2 Working Conference on Data Base Management. pp. 1–59. Elsevier Science Publishers (North-Holland) (Apr 1974)
2. Abrial, J.R.: The B-Book: Assigning Programs to Meanings. Cambridge University Press (1996)
3. Abrial, J.R.: Modeling in Event-B: System and Software Engineering. Cambridge University Press (2010)
4. Abrial, J.R., Börger, E., Langmaack, H. (eds.): Formal Methods for Industrial Applications: Specifying and Programming the Steam Boiler Control, Lecture Notes in Computer Science, vol. 1165. Springer (1996). https://doi.org/10.1007/BFb0027227
5. Abrial, J.R., Börger, E., Langmaack, H.: The steam boiler case study: Competition of formal program specification methods. In: Formal Methods for Industrial Applications [4], pp. 1–12. https://doi.org/10.1007/BFb0027227
6. Abrial, J.R., Glässer, U. (eds.): 06191 Summary – Rigorous Methods for Software Construction and Analysis. No. 06191 in Dagstuhl Seminar Proceedings, Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany (May 2006), `http://drops.dagstuhl.de/opus/volltexte/2006/665`
7. Beler, A., Borda, A., Bowen, J.P., Filippini-Fantoni, S.: The building of online communities: An approach for learning organizations, with a particular focus on the museum sector. In: Hemsley, J., Cappellini, V., Stanke, G. (eds.) EVA 2004 London Conference Proceedings. pp. 2.1–2.15. EVA Conferences International, University College London, UK (2004), `https://arxiv.org/abs/cs/0409055`
8. Bjørner, D., Hoare, C.A.R., Bowen, J.P., He, J., Langmaack, H., Olderog, E.R., Martin, U., Stavridou, V., Nielson, F., Nielson, H.R., Barringer, H., Edwards, D., Løvengreen, H.H., Ravn, A.P., Rischel, H.: A ProCoS project description: ESPRIT BRA 3104. Bulletin of the European Association for Theoretical Computer Science **39**, 60–73 (1989), `http://researchgate.net/publication/256643262`
9. Black, S.E., Boca, P.P., Bowen, J.P., Gorman, J., Hinchey, M.G.: Formal versus agile: Survival of the fittest. Computer **42**(9), 37–45 (2009). https://doi.org/10.1109/MC.2009.284
10. Boca, P.P., Bowen, J.P., Duce, D.A. (eds.): Teaching Formal Methods: Practice and Experience. Electronic Workshops in Computing (eWiC), FACS, BCS (15 Dec 2006), `http://www.bcs.org/ewic/tfm2006`
11. Boca, P.P., Bowen, J.P., Siddiqi, J.I. (eds.): Formal Methods: State of the Art and New Directions. Springer (2010). https://doi.org/10.1007/978-1-84882-736-3
12. Borda, A., Bowen, J.P.: Virtual collaboration and community. In: Information Resources Management Association (ed.) Virtual Communities: Concepts, Methodologies, Tools and Applications, chap. 8.9, pp. 2600–2611. IGI Global (2011)
13. Börger, E.: The origins and development of the ASM method for high-level system design and analysis. Journal of Universal Computer Science **8**(1), 2–74 (2002). https://doi.org/10.3217/jucs-008-01-0002
14. Börger, E., Bowen, J.P., Butler, M.J., Poppleton, M.: Editorial. Formal Aspects of Computing **23**(1), 1–2 (2011). https://doi.org/10.1007/s00165-010-0168-x
15. Börger, E., Butler, M.J., Bowen, J.P., Boca, P.P. (eds.): Abstract State Machines, B and Z: First International Conference, ABZ 2008, London, UK, September 16–18, 2008, LNCS, vol. 5238. Springer (2008). https://doi.org/10.1007/978-3-540-87603-8

16. Börger, E., Butler, M.J., Bowen, J.P., Boca, P.P. (eds.): ABZ 2008 Conference: Short papers. BCS, London, UK (2008)
17. Börger, E., Durdanovic, I.: Correctness of compiling Occam to Transputer code. The Computer Journal **39**(1), 52–92 (1996)
18. Börger, E., Gargantini, A., Riccobene, E.: ASM. In: Habrias and Frappier [61], chap. 6, pp. 103–119. https://doi.org/10.1002/9780470612514
19. Börger, E., Mazzanti, S.: A practical method for rigorously controllable hardware design. In: Bowen et al. [47], pp. 151–187. https://doi.org/10.1007/BFb0027289
20. Börger, E., Raschke, R.: Modeling Companion for Software Practitioners. Springer (2018). https://doi.org/10.1007/978-3-662-56641-1
21. Bowen, J.P. (ed.): Proceedings of the Third Annual Z Users Meeting. Oxford University Computing Laboratory, Rewley House, Wellington Square, Oxford, UK (16 Dec 1988), `http://researchgate.net/publication/2526997`
22. Bowen, J.P.: From programs to object code and back again using logic programming: Compilation and decompilation. Journal of Software Maintenance: Research and Practice **5**(4), 205–234 (1993). https://doi.org/10.1002/smr.4360050403
23. Bowen, J.P.: A ProCoS II project description: ESPRIT Basic Research project 7071. Bulletin of the European Association for Theoretical Computer Science **50**, 128–137 (1993), `http://researchgate.net/publication/2521581`
24. Bowen, J.P.: A ProCoS-WG Working Group description: ESPRIT Basic Research 8694. Bulletin of the European Association for Theoretical Computer Science **53**, 136–145 (Jun 1994)
25. Bowen, J.P.: Rapid compiler implementation. In: He [62], chap. 10, pp. 141–169
26. Bowen, J.P.: Formal Specification and Documentation using Z: A case study approach. International Thomson Computer Press (1996), `http://researchgate.net/publication/2480325`
27. Bowen, J.P.: Comp.specification.z and Z FORUM frequently asked questions. LNCS, vol. 1493, pp. 407–415. Springer (1998). https://doi.org/10.1007/978-3-540-49676-2_25
28. Bowen, J.P.: Combining operational semantics, logic programming and literate programming in the specification and animation of the Verilog hardware description language. In: Grieskamp, W., Santen, T., Stoddart, B. (eds.) Integrated Formal Methods. LNCS, vol. 1945, pp. 277–296. Springer (2000). https://doi.org/10.1007/3-540-40911-4_16
29. Bowen, J.P.: The ethics of safety-critical systems. Communications of the ACM **43**(4), 91–97 (2000). https://doi.org/10.1145/332051.332078
30. Bowen, J.P.: Experience teaching Z with tool and web support. ACM SIGSOFT Software Engineering Notes **26**(2), 69–75 (2001). https://doi.org/10.1145/505776.505794
31. Bowen, J.P.: Z. In: Habrias and Frappier [61], chap. 1, pp. 3–20. https://doi.org/10.1002/9780470612514, `http://researchgate.net/publication/319019328`
32. Bowen, J.P.: Online communities: Visualization and formalization. In: Blackwell, C. (ed.) Cyberpatterns 2013: Second International Workshop on Cyberpatterns – Unifying Design Patterns with Security, Attack and Forensic Patterns. pp. 53–61. Oxford Brookes University, Abingdon, UK (Jul 2013), `http://arxiv.org/abs/1307.6145`
33. Bowen, J.P.: A relational approach to an algebraic community: From Paul Erdős to He Jifeng. In: Liu, Z., Woodcock, J.C.P., Zhu, H. (eds.) Theories of Programming and Formal Methods. LNCS, vol. 8051, pp. 54–66. Springer (2013). https://doi.org/10.1007/978-3-642-39698-4_4

34. Bowen, J.P.: The Z notation: Whence the cause and whither the course? In: Liu, Z., Zhang, Z. (eds.) Engineering Trustworthy Software Systems. LNCS, vol. 9506, pp. 104–151. Springer (2016). https://doi.org/10.1007/978-3-319-29628-9_3
35. Bowen, J.P.: Provably Correct Systems: Community, connections, and citations. In: Hinchey et al. [68], pp. 313–328. https://doi.org/10.1007/978-3-319-48628-4_13
36. Bowen, J.P.: Egon Börger and Alexander Raschke: Modeling companion for software practitioners. Formal Aspects of Computing **30**(6), 761–762 (2018). https://doi.org/10.1007/s00165-018-0472-4
37. Bowen, J.P.: A personal view of EVA London: Past, present, future. In: Weinel, J., Bowen, J.P., Diprose, G., Lambert, N. (eds.) EVA London 2020: Electronic Visualisation and the Arts. pp. 8–15. Electronic Workshops in Computing (eWiC), BCS, London, UK (2020). https://doi.org/10.14236/ewic/EVA2020.2
38. Bowen, J.P.: ABZ 2021 conference report. FACS FACTS **2021**(2), 65–70 (Jul 2021), `https://www.bcs.org/media/7577/facs-jul21.pdf`
39. Bowen, J.P.: Communities and ancestors associated with Egon Börger and ASM. In: Raschke, A., Riccobene, E., Schewe, K.D. (eds.) Logic, Computation and Rigorous Methods. Lecture Notes in Computer Science, vol. 12750, pp. 96–120. Springer (2021). https://doi.org/10.1007/978-3-030-76020-5_6
40. Bowen, J.P., Breuer, P.T., Lano, K.C.: A compendium of formal techniques for software maintenance. Software Engineering Journal **8**(5), 253–262 (1993). https://doi.org/10.1049/sej.1993.0031
41. Bowen, J.P., Breuer, P.T., Lano, K.C.: Formal specifications in software maintenance: From code to Z++ and back again. Information and Software Technology **35**(11–12), 679–690 (1993). https://doi.org/10.1016/0950-5849(93)90083-F
42. Bowen, J.P., Dunne, S., Galloway, A., King, S. (eds.): ZB 2000: Formal Specification and Development in Z and B: First International Conference of B and Z Users, York, UK, August 29 – September 2, 2000, LNCS, vol. 1878. Springer (2000). https://doi.org/10.1007/3-540-44525-0
43. Bowen, J.P., Hinchey, M.G. (eds.): ZUM'95: The Z Formal Specification Notation: 9th International Conference of Z Users, Limerick, Ireland, September 7–9, 1995, LNCS, vol. 967. Springer (1995). https://doi.org/10.1007/3-540-60271-2
44. Bowen, J.P., Hinchey, M.G.: Ten commandments ten years on: Lessons for ASM, B, Z and VSR-net. In: Abrial, J.R., Glaesser, U. (eds.) Rigorous Methods for Software Construction and Analysis. LNCS, vol. 5115, pp. 219–233. Springer (2009). https://doi.org/10.1007/978-3-642-11447-2_14
45. Bowen, J.P., Hinchey, M.G.: Formal methods. In: Gonzalez, T., Díaz-Herrera, J., Tucker, A.B. (eds.) Computing Handbook, vol. Computer Science and Software Engineering, pp. 1–25. Chapman and Hall / CRC Press, 3rd edn. (2014). https://doi.org/10.1201/b16812-80
46. Bowen, J.P., Hinchey, M.G., Janicke, H., Ward, M., Zedan, H.S.M.: Formality, agility, security, and evolution in software engineering. In: Hinchey, M.G. (ed.) Software Technology: 10 Years of Innovation in IEEE Computer, chap. 16, p. 384. Wiley-IEEE Press (2018). https://doi.org/10.1002/9781119174240.ch16
47. Bowen, J.P., Hinchey, M.G., Till, D. (eds.): ZUM'97: The Z Formal Specification Notation: 10th International Conference of Z Users, Reading, UK, April 3–4, 1997, LNCS, vol. 1212. Springer (1997). https://doi.org/10.1007/BFb0027279
48. Bowen, J.P., Hoare, C.A.R., Langmaack, H., Olderog, E.R., Ravn, A.P.: A ProCoS II project final report: ESPRIT Basic Research project 7071. Bulletin of the European Association for Theoretical Computer Science **59**, 76–99 (Jun 1996), `http://researchgate.net/publication/2255515`

49. Bowen, J.P., Hoare, C.A.R., Langmaack, H., Olderog, E.R., Ravn, A.P.: A ProCoS-WG Working Group final report: ESPRIT Working Group 8694. Bulletin of the European Association for Theoretical Computer Science **64**, 63–72 (1998), http://researchgate.net/publication/2527052
50. Bowen, J.P., Nicholls, J.E. (eds.): Z User Workshop, London 1992: Proceedings of the Seventh Annual Z User Meeting, London, 14–15 December 1992. Workshops in Computing, Springer (1993). https://doi.org/10.1007/978-1-4471-3556-2
51. Bowen, J.P., Reeves, S.: From a Community of Practice to a Body of Knowledge: A case study of the formal methods community. In: Butler, M., Schulte, W. (eds.) FM 2011: Formal Methods. LNCS, vol. 6664, pp. 308–322. Springer (2011). https://doi.org/10.1007/978-3-642-21437-0_24
52. Bowen, J.P., Wilson, R.J.: Visualising virtual communities: From Erdős to the arts. In: Dunn, S., Bowen, J.P., Ng, K.C. (eds.) EVA London 2012: Electronic Visualisation and the Arts. pp. 238–244. Electronic Workshops in Computing (eWiC), BCS (2012), http://arxiv.org/abs/1207.3420
53. Breuer, P.T., Bowen, J.P.: A PREttier Compiler-Compiler: Generating higher-order parsers in C. Software: Practice and Experience **25**(11), 1263–1297 (1995). https://doi.org/10.1002/spe.4380251106
54. Breuer, P.T., Bowen, J.P.: Empirical patterns in Google Scholar citation counts. In: SOSE 2014: IEEE 8th International Symposium on Service Oriented System Engineering. pp. 398–403. IEEE (2014). https://doi.org/10.1109/SOSE.2014.55
55. Breuer, P.T., Bowen, J.P.: Fully encrypted high-speed microprocessor architecture: the secret computer in simulation. International Journal of Critical Computer-Based Systems **9**(1–2), 26–55 (2019). https://doi.org/10.1504/IJCCBS.2019.098797
56. Breuer, P.T., Bowen, J.P., Palomar, E., Liu, Z.: On security in encrypted computing. In: Naccache, D., Xu, S., Qing, S., Samarati, P., Blanc, G., Lu, R., Zhang, Z., Meddahi, A. (eds.) Information and Communications Security. LNCS, vol. 11149, pp. 192–211. Springer (2018). https://doi.org/10.1007/978-3-030-01950-1_12
57. Breuer, P.T., Madrid, N.M., Bowen, J.P., France, R., Petrie, M.L., Kloos, C.D.: Reasoning about VHDL and VHDL-AMS using denotational semantics. In: Borrione, D. (ed.) DATE'99: Conference on Design, Automation and Test in Europe. pp. 346–352. ACM (1999). https://doi.org/10.1109/DATE.1999.761144
58. Cerone, A., Roggenbach, M., Schlingloff, B.H., Schneider, G., Shaikh, S.A.: Teaching formal methods for software engineering – ten principles. Informatica Didactica **9** (2015), https://www.informaticadidactica.de/uploads/Artikel/Schlinghoff2015/Schlinghoff2015.pdf
59. Frappier, M., Habrias, H. (eds.): Software Specification Methods: An overview using a case study. FACIT, Springer (2001). https://doi.org/10.1007/978-1-4471-0701-9
60. Giannini, T., Bowen, J.P. (eds.): Museums and Digital Culture: New perspectives and research. Series on Cultural Computing, Springer (2019). https://doi.org/10.1007/978-3-319-97457-6
61. Habrias, H., Frappier, M. (eds.): Software Specification Methods. ISTE (2006). https://doi.org/10.1002/9780470612514
62. He, J. (ed.): Provably Correct Systems: Modelling of communication languages and design of optimized compilers. International Series in Software Engineering, McGraw-Hill (1994)
63. He, J., Bowen, J.P.: Specification, verification and prototyping of an optimized compiler. Formal Aspects of Computing **6**(6), 643–658 (1994). https://doi.org/10.1007/BF03259390

64. Henson, M.C., Reeves, S., Bowen, J.P.: Z logic and its consequences. Computing and Informatics **22**(3–4), 381–415 (2003). https://doi.org/10289/1571
65. Hinchey, M.G., Bowen, J.P. (eds.): Applications of Formal Methods. Series in Computer Science, Prentice Hall International (1995)
66. Hinchey, M.G., Bowen, J.P. (eds.): Industrial-Strength Formal Methods in Practice. FACIT, Springer (1999). https://doi.org/10.1007/978-1-4471-0523-7
67. Hinchey, M.G., Bowen, J.P., Glass, R.L.: Formal methods: Point-counterpoint. Computer **29**(4), 18–19 (1996). https://doi.org/10.1109/MC.1996.10044
68. Hinchey, M.G., Bowen, J.P., Olderog, E.R. (eds.): Provably Correct Systems. NASA Monographs in Systems and Software Engineering, Springer (2017). https://doi.org/10.1007/978-3-319-48628-4
69. Hinchey, M.G., Bowen, J.P., Rouff, C.: Introduction to formal methods. In: Hinchey, M.G., Rash, J., Truszkowski, W., Gordon-Spears, D.F. (eds.) Agent Technology from a Formal Perspective, pp. 25–64. Springer (2006). https://doi.org/10.1007/1-84628-271-3_2
70. ISO: Information technology – Z formal specification notation – syntax, type system and semantics. International Standard 13568, ISO/IEC (Jul 2002)
71. Jackson, D.: Software Abstractions: Logic, Language, and Analysis. The MIT Press (2011)
72. Jones, C.B.: Systematic Software Development Using VDM. Series in Computer Science, Prentice Hall International (1986)
73. Kahkonen, T.: Agile methods for large organizations-building Communities of Practice. In: Agile Development Conference. pp. 2–10. IEEE (2004). https://doi.org/10.1109/ADEVC.2004.4
74. Kapoor, K., Bowen, J.P.: Test conditions for fault classes in Boolean specifications. ACM Transactions on Software Engineering and Methodology (TOSEM) **16**(3), 1–12 (2007). https://doi.org/10.1145/1243987.1243988
75. Lamport, L.: The temporal logic of actions. ACM Transactions on Programming Languages and Systems **16**(3), 872–923 (May 1994). https://doi.org/10.1145/177492.177726
76. Liu, Z., Bowen, J.P., Liu, B., Tyszberowicz, S., Zhang, T.: Software abstractions and Human-Cyber-Physical Systems architecture modelling. In: Bowen, J.P., Liu, Z., Zhang, Z. (eds.) Engineering Trustworthy Software Systems. Lecture Notes in Computer Science, vol. 12154, pp. 159–219. Springer (2020). https://doi.org/10.1007/978-3-030-55089-9_5
77. Mavri, A., Ioannou, A., Loizides, F.: Cross-organisational Communities of Practice: Enhancing creativity and epistemic cognition in higher education. The Internet and Higher Education **49** (2021). https://doi.org/10.1016/j.iheduc.2021.100792
78. McDonald, J., Cater-Steel, A. (eds.): Communities of Practice: Facilitating Social Learning in Higher Education. Springer (2017). https://doi.org/10.1007/978-981-10-2879-3
79. McDonald, J., Cater-Steel, A. (eds.): Implementing Communities of Practice in Higher Education: Dreamers and Schemers. Springer (2017). https://doi.org/10.1007/978-981-10-2866-3
80. Nunes, R., Pedrosa, D., Fonseca, B., Paredes, H., Cravino, J., Morgado, L., Martins, P.: Enhancing students' motivation to learn software engineering programming techniques: A collaborative and social interaction approach. In: Antona, M., Stephanidis, C. (eds.) Universal Access in Human-Computer Interaction: Access to Learning, Health and Well-Being. Lecture Notes in Computer Science, vol. 9177, pp. 189–201. Springer (2015). https://doi.org/10.1007/978-3-319-20684-4_19

81. Oxford University Computing Laboratory: Proceedings of Z Users Meeting (8 Dec 1987). https://doi.org/10.13140/RG.2.2.20103.34724
82. Raschke, A., Méry, D. (eds.): Rigorous State-Based Methods: 8th International Conference, ABZ 2021, Ulm, Germany, June 9–11, 2021, Proceedings, Lecture Notes in Computer Science, vol. 12709. Springer (2021). https://doi.org/10.1007/10.1007/978-3-030-77543-8
83. Raschke, A., Méry, D., Houdek, F. (eds.): Rigorous State-Based Methods: 7th International Conference, ABZ 2020, Ulm, Germany, May 27–29, 2020, Proceedings, Lecture Notes in Computer Science, vol. 12071. Springer (2020). https://doi.org/10.1007/978-3-030-48077-6
84. Schulte, B.: The Organizational Embeddedness of Communities of Practice Exploring the Cultural and Leadership Dynamics of Self-organized Practice. Springer (2020)
85. Spivey, J.M.: The Z Notation: A Reference Manual. Series in Computer Science, Prentice Hall International, 2nd edn. (1992)
86. van Zuylen, H.J. (ed.): The REDO Compendium: Reverse Engineering for Software Maintenance. John Wiley & Sons (1993)
87. Vilkomir, S.A., Bowen, J.P.: From MC/DC to RC/DC: Formalization and analysis of control-flow testing criteria. Formal Aspects of Computing **18**(1), 42–62 (2006). https://doi.org/10.1007/s00165-005-0084-7
88. Vilkomir, S.A., Bowen, J.P., Ghose, A.K.: Formalization and assessment of regulatory requirements for safety-critical software. Innovations in Systems and Software Engineering: A NASA Journal **2**, 165–178 (2006). https://doi.org/10.1007/s11334-006-0006-8
89. Webber, E.: Building Successful Communities of Practice. Blurb (2016)
90. Wenger, E.: Communities of Practice: Learning, Meaning, and Identity. Cambridge University Press (1998)
91. Wenger, E., McDermott, R.A., Snyder, W.: Cultivating Communities of Practice: A Guide to Managing Knowledge. Harvard Business School Press (2002)
92. Wenger-Trayner, E., Wenger-Trayner, B.: Introduction to communities of practice: A brief overview of the concept and its uses (2015), `https://wenger-trayner.com/introduction-to-communities-of-practice/`
93. Wohllebe, A., Götz, M.: Communities of Practice for functional learning in agile contexts: Definition approach and call for research. International Journal of Advanced Corporate Learning **41**, 62–69 (2021). https://doi.org/10.3991/ijac.v14i1.21939
94. Woodcock, J.C.P., Larsen, P.G. (eds.): FME'93: Industrial-Strength Formal Methods, First International Symposium of Formal Methods Europe, Odense, Denmark, April 19–23, 1993, Proceedings, Lecture Notes in Computer Science, vol. 670. Springer (1993). https://doi.org/10.1007/BFb0024633
95. Zhu, H., Bowen, J.P., He, J.: From operational semantics to denotational semantics for Verilog. In: Margaria, T., Melham, T. (eds.) Correct Hardware Design and Verification Methods. LNCS, vol. 2144, pp. 449–464. Springer (2001). https://doi.org/10.1007/3-540-44798-9_34