Howard Barringer
The Man who Invented the Past

Klaus Havelund*

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, USA
klaus.havelund@jpl.nasa.gov

Abstract

This article is an introduction to Professor Howard Barringer, in honor of his 60th birthday on December 20, 2011, which was celebrated by the HOWARD-60 workshop (Higher-Order Workshop on Automated Runtime verification and Debugging), held on the same day at University of Manchester.

1 A 60 Second Overview

Howard Barringer was born on December 20, 1951, is married to Margaret, and has three children. This forms the concrete part of Howard's life. Beyond this, Howard has had an abstract life centred around mathematics, physics, and computer science. At secondary school (1964-1969) Howard moved into the science stream and finished with 'A' levels in mathematics (pure and applied) and physics. He went on to University of Manchester where he first received a B.Sc in Physics (1972), then an M.Sc in Computer Science (1973), and finally a Ph.D in Computer Science (1978).

His adult academic life has been centred at University of Manchester. He became a Research Associate in Computer Science at Manchester in 1975, a Lecturer in 1977, a Senior Lecturer in 1986 and was then rapidly promoted to Professor in Computer Science in 1987. For the majority of his career, his research and teaching has been focussed around the development and application of logics, in particular temporal and modal logics, in the specification, design, and analysis of software and hardware systems. Howard insisted on the importance of past time logic in temporal logic, and hence got named *"the man who invented the past"*[1]. He has taught classes in the theory of computation, compiling techniques, specification and verification, concurrency, modal and temporal logic, algorithms, and programming in Java. Howard also spent a significant portion of his career in senior and highly influential administrative positions at University of Manchester.

He has been invited to present over 100 seminars and research lectures in Austria, Belgium, Canada, China (Beijing, Shanghai, Wuhan), Denmark, England, France, Germany, Greece, Israel, Italy, the Netherlands, Norway, Scotland, South Africa, Spain, Sweden, USA (Arizona, California, New York, Pennsylvania, Texas) and Wales. He was visiting professor at Kings College (2001 and 2006), and visited Silicon Valley numerous times including NASA Ames Research Center, Moffett Field, CA in 2002 and 2003, and the Computer Science Laboratory at SRI International, Menlo Park, CA in 2002. It has been observed that, temporally speaking, a series of Mars Rovers were launched after Howard visited NASA. Howard was one of the

*The writing of this article was carried out at Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

[1]This title was assigned to Howard in connection with his 60th birthday by Andrei Voronkov.

founding editors of the *Journal of Logic and Computation* in 1989 and is now Co-Chief Editor with Dov Gabbay. He is also Editor for *Journal of Applied Logic*.

2 Initial Research Phase

Howard started his research activities in systems programming (1973–1979). The research for the M.Sc and Ph.D degrees was concerned with compiling techniques for the ALGOL 68 programming language. As a Research Assistant after his doctorate, his work involved the design and implementation of portable compiling systems, and resulted in a low-level target machine language (TML) (Barringer, Capon and Phillips, 1979) [20] and a simpler high-level systems implementation language (MUPL). This line of work continued during the first year in which he was employed as a lecturer within the Department of Computer Science and resulted in the implementation of an improved, higher level, intermediate target language (TL) that enabled straightforward implementation of an improved systems implementation languages (MUSL).

During the end of the 1970s, Howard developed an interest in formal methods, causing a change of direction. This change coincided with Cliff Jones joining Manchester University as professor. Together with Howard's Ph.D student Jen Cheng they worked on a logic for partial functions, see (Barringer, Cheng and Jones, 1984) [22]. During the first half of the 1980ties, his interest otherwise centred around compositional temporal logic, specifically techniques for specification and verification of parallel and distributed systems. Initially, results were obtained for the axiomatic verification of Ada tasks (Barringer and Mearns, 1982) [21] and (Barringer and Mearns, 1986) [23]. He became a member of the CEC Ada-Europe Working Group on *Formal methods for Specification and Development* (1983–1985), was chairman for this group from 1985–88, and was additionally member of the Ada UK Working Group on Formal Methods from 1984–87.

In 1981-1982 he went on two three-week trips to 12 university and industrial research laboratories in the United States to have discussions with researchers about their verification methods. The research undertaken was published as a volume in the Springer-Verlag Lecture Notes in Computer Science series (Barringer, 1985) [1]. The study led to ideas on specifying systems in a modular and compositional fashion based on the use of temporal logics – including reasoning about the past – which were presented in (Barringer and Kuiper, 1983) [44]. As a result of this work, Howard spent a couple of weeks at the Weizmann Institute of Science, Israel, at the invitation of Professor Amir Pnueli in order to collaborate on developing compositional temporal proof systems for parallel languages (the goal of compositionality in temporal proof systems had been standing for approximately eight years).

That visit marked the beginning of a strong collaboration with Pnueli, leading to general techniques for constructing compositional temporal proof systems for both shared variable and message based communication mechanisms in parallel programming languages (Barringer, Kuiper and Pnueli, 1984) [47], (Barringer, Kuiper and Pnueli, 1985) [49] and for fully abstract concurrency models (Barringer, Kuiper and Pnueli, 1986) [50]. Other researchers joined the work and visited Manchester, including Professors Willem-Paul de Roever and Zhou Chaochen. In 1986, at the invitation of Professor Zhou Chaochen, he presented a research lecture series on temporal logic and its applications in concurrency (a total of 32 hours of lectures was presented at the Academy of Sciences, Beijing, Wuhan University and Fudan University, Shanghai, over a four week period).

Further work led, for example, to a temporal fixed point calculus (Banieqbal and Barringer, 1986) [104], as well as what was probably the first practical implementation of a decision procedure for checking validity of a linear temporal logic covering infinite past, present and infinite future, undertaken by one of Howard's M.Sc students, Graham Gough, in 1984. Elements of that implementation were used in prototyping a system enabling model checking, both linear and branching time, of finite state programs presented in arbitrary *finite-state* programming languages (Barringer and Gough, 1988) [51] and (Barringer, Fisher and Gough, 1989) [56].

The work on compositional temporal logic, formal specification and decision procedures led to a collaboration with Professor Dov Gabbay (King's College London) from 1986 to 1994 on a novel approach to programming with temporal logic. The usual view of temporal logic (as in model checking) is to evaluate a formula over a structure. In this new work, this was turned around, and they proposed an imperative view that dynamically created a model for a formula. Thus for a temporal formula representing a specification of some desired system, the imperative view provided a mechanism for animating, or directly simulating, the specification. This led to the development of the *MetateM* system, described in the book: *The Imperative Future: Principles of Executable Temporal Logic* (Barringer, Fisher, Gabbay, Owens and Reynolds, 1996) [3].

During the period 1986–1994, Howard was also involved in the development of techniques for the specification and verification of synchronous hardware systems, specifically resulting in a CAD environment for the hardware design language ELLA (at the time viewed as a potential European standard for high-level digital design). This was a result of an industrial project. Of particular interest on the verification side was the novel symbolic state-based equivalence checking, combining state-based exploration with theorem proving to enable the equivalence checking of infinite state systems in a largely automated fashion (Barringer, Gough, Monahan and Williams, 1995) [63]. His interest in hardware design and verification continued during the years 1993–2000 through his leadership of the Rainbow project, a system for comprehensive design and formally-based analysis tools, including a CTL* model checker developed by his Ph.D student Willem Visser, for asynchronous chip designs (Barringer, Fellows, Gough, Jinks, Marsden and Williams, 1996)[67], (Barringer, Fellows, Gough, Jinks and Williams, 1997) [71], (Barringer, Fellows, Gough and Williams, 1997) [69], (Visser, Barringer, Fellows, Gough and Williams, 1997) [72] and (Visser and Barringer, 2000) [31].

3 University Administration

Howard has committed a significant effort to the administration of University of Manchester. As well as undertaking nearly every academic-led administrative role in the department, he has held senior posts in the university itself. (He even claims to have had a total of ten days of management training.) As professor he has been head of the formal methods group since 1996. He was head of the Computer Science Department in the period 1991–96. His success in this role led to an invitation to become Pro-Vice-Chancellor for Research in the period 1996–2001.

As Pro-Vice-Chancellor for Research Howard performed many extraordinary tasks in order to help shape and guide University policy and strategy in research and graduate education matters. He was for example asked to conduct an academic review of the Chemistry department, and bring forward academic and financial plans that would support, or otherwise, a case for major refurbishment of their laboratories, caused by unsafe working conditions. He was also asked to head a research review of Medicine, working together with a very high-powered group consisting of medical, biological and dental specialists, causing a turnaround and upward trend within the faculty.

He played a major leading role in preparing the University for the 2001 Research Assessment Exercise (RAE), where British universities are compared and ranked relative to each other. His personal goal was to lift the University's research ranking to be in the top ten (from the mid twenties), possibly the top large civic full-breadth UK University. The University achieved an uplift in quality assessment and was considered to be truly back within the top ten UK research universities. He has furthermore represented the University of Manchester on many major international missions, e.g. to China, Japan, South Africa, Syria, USA, Canada, Singapore, and Indonesia.

He warned many years ago that each year away from full-time personal research requires at least two years for recovery. Since his administrative career was during the years 1991–2001, he should now be well beyond half way to full recovery. Retiring to become Emeritus Professor should hopefully speed up this process.

4 Return to Research

From 2002 when Howard visited NASA Ames Research Center, Moffett Field, California, USA, for the first time, a set of new research directions caught his attention. He and colleagues Dimitra Giannakopoulou and Corina Pasareanu at NASA Ames co-developed a novel solution to generate weakest environment assumptions necessary for a given component to achieve a desired behavioral property. This technique was applied to models of a part of the plan execution engine of K9, an experimental autonomous Mars rover, and to the plan execution engine of the Remote Agent, part of an AI solution for autonomous control of the experimental Deep-Space 1 spacecraft (Giannakopoulou, Pasareanu and Barringer, 2002) [76] and (Giannakopoulou, Pasareanu and Barringer, 2005) [33].

An interesting unpublished piece of work during his visit to NASA Ames is a formalization of a new planning and scheduling system, that the robotics group there was working on. The complexity of the system was such, that it was difficult to get the design defined precisely during the design phase. Howard formalized the system as a VDM model, during the visit, in order to facilitate the communication.

Another line of research ongoing at NASA Ames at the time was the study of logics and systems for monitoring systems executions. This field is also sometimes referred to as *runtime verification*. The fundamental idea is to instrument a program (or more generally: a system) to emit events during its execution, and check the resulting execution generated event stream against a formal specification. Howard has contributed in several substantial ways to this field. A wide variety of different languages and logics had been proposed for specifying such trace properties, initially of course inspired by already existing logics from model checking, including temporal logics and regular expressions. Researchers at NASA Ames wanted to go beyond the propositional case and be able to specify properties over traces of data carrying events. In addition, Howard was dissatisfied that there was no unifying base logic from which these different temporal logics could be built. Researchers had in the past searched for efficient algorithms for the evaluation of restricted sub-logics, e.g. pure past time LTL, pure future LTL, extended regular expressions, metric temporal logic, and so forth. However, for monitoring program executions one could consider relaxing the constraints somewhat. Based on his experience with executable temporal logics, e.g. MetateM, this led to a collaboration with Allen Goldberg, Klaus Havelund (both at NASA Ames) and Kousik Sen (summer intern at NASA Ames from University of Illinois), to the development of the Eagle fixed point logic and corresponding Java evaluation engine (Barringer, Goldberg, Havelund and Sen, 2004) [78], restricted to finite traces. The logic is based on recursive rules defined over primitives for next (one step forwards in a trace), previous (one step backwards in a trace) and concatenation and/or fusion (of two traces). Rules can be parameterized by other rules, and by other typed data parameters. The data parameterization provides some limited form of data quantification in the logic; in particular, it facilitates definition of metric and real-time temporal logics, and in general permits the embedding of a whole range of temporal logics.

The Eagle implementation was, however, somewhat complex. From this experience, and again his previous experience with MetateM, Howard initiated the design and implementation of a simpler rule-based system, RuleR, which can be viewed as an even lower-level abstract machine into which various monitoring specifications (from different languages) can be compiled. The resulting logic was designed in collaboration with David Rydeheard (University of Manchester) and Klaus Havelund, and was published as (Barringer, Rydeheard and Havelund, 2007) [83] and (Barringer, Rydeheard and Havelund, 2008) [37]. Independently and at the same time, Dov Gabbay developed the idea of reactive Kripke structures, in which traversal from one world to another can influence other connections between worlds. This has led to a collaboration between Howard and Dov on what they call *Reactive Grammars*. A subsequent collaboration with Ylies Falcone (University of Grenoble, France), Howard's Ph.D student Giles Reger (University of Manchester), David Rydeheard, and Klaus Havelund led to a theory named Quantified Event Automata (QEA), an automaton-based approach to monitoring, focusing on efficient monitoring as well as expressiveness of the logic, see (Barringer, Falcone, Havelund, Reger and Rydeheard, 2012) [98].

The work on runtime monitoring logics naturally leads to the question: what to do when a monitored system fails to conform with the specification, that the execution is verified against. Howard and colleagues David Rydeheard, Brian Warboys (University of Manchester), Dov Gabbay, and John Woods (University of British Columbia, Canada) have since 2005 studied this problem as what they refer to as *evolvable systems*. The idea is to consider a system as consisting of a supervisee (the actual system), and a supervisor (a monitor) which monitors the supervisee, and triggers corrective action when the supervisee violates the specification installed in the supervisor. See (Barringer and Rydeheard, 2005) [15], (Barringer, Rydeheard, Warboys and Gabbay, 2007) [82], (Barringer, Rydeheard and Gabbay, 2007) [84], and (Barringer, Gabbay and Woods, 2012, parts 1 and 2) [41, 42]. Another interesting problem is how to infer a specification from a program's execution traces. This problem was studied by Howard's Ph.D student Giles Reger in a collaboration with Howard and David Rydeheard, see (Reger, Barringer and Rydeheard, 2013a) [99] and (Reger, Barringer and Rydeheard, 2013b) [100].

5 Conclusion

Howard Barringer has, as can be seen from the above, been interested in a wide spectrum of work, and has in addition taken on leadership roles when asked to do so, helping other people on their way, always "stepping up to the plate". It has been pointed out by colleagues, that one of his most important personality traits is, that he cares about other people. He is always ready to discuss just about any subject thrown at him, with a positive and open mind. This positive spirit combined with a very sharp mind, and being always willing to open an Eclipse editor and start writing Java, as well as a Latex document and write formulas, is what makes Howard admired and liked by so many. We are told that he went mountain climbing with John Rushby in the past. Clearly, it helped form a solid guy.

Acknowledgements

I would like to thank Andrei Voronkov and Margarita Korovina for having organized the HOWARD-60 workshop, which gave all participants an opportunity to celebrate Howard Barringer. I would also like to thank Cliff B. Jones and Andrei Voronkov for their comments and

edits to this article.

References

Authored Books

[1] Barringer H. (1985): A Survey of Verification Techniques for Parallel Programs, Lecture Notes in Computer Science, Vol. 191, 120 pages, Springer-Verlag.

Edited Books

[2] Proceedings of Colloquium on Temporal Logic and Specification. Edited by Barringer H., Banieqbal B. and Pnueli A., LNCS 398, Springer-Verlag, October 1989.

[3] The Imperative Future: Principles of Exectuable Temporal Logic. Edited by Barringer H., Fisher M.D., Gabbay D.M., Owens R.P. and Reynolds M. Research Studies Press, May 1996.

[4] Advances in Temporal Logic. Edited by Barringer H., Fisher M.D., Gabbay D.M. and Gough G.D. Volume 16, Applied Logic Series, Kluwer Academic Publishers, Dordrecht, December 1999.

[5] We Will Show Them! Essays in Honour of Dov Gabbay: Volume 1. Edited by Artemov S., Barringer H., DAvila Garcez A., Lamb L. and Woods J. Kings College Press, October 2005.

[6] We Will Show Them! Essays in Honour of Dov Gabbay: Volume 2. Edited by Artemov S., Barringer H., DAvila Garcez A., Lamb L. and Woods J. Kings College Press, October 2005.

[7] Proceedings of the Fifth Workshop on Runtime Verification (RV 2005). Edited by Barringer H., Finkbeiner B., Gurevich Y. and Sipma H.B. ENTCS, Volume 144, Number 4, May 2006.

[8] Proceedings of the 1st International Conference on Runtime Verification (RV 2010). Edited by Barringer H., Falcone Y., Finkbeiner B., Havelund K., Lee I., Pace G., Rosu G., Sokolsky O. and Tillman N. LNCS 6418, Springer, November 2010.

Invited Chapters in Books

[9] Barringer H. (1987): The Use of Temporal Logic in the Compositional Specification of Concurrent Systems. In Temporal Logic and its Applications, ed. A.P.Galton, Academic Press Inc. (London) Limited, pp. 53 90.

[10] Barringer H. and Gabbay D.M. (1991): The Imperative Future: Past Successes $\Rightarrow$ Future Actions. In Logic from Computer Science, (ed. Y.N. Moschovakis), Proc. of a Workshop held at MSRI Berkeley, Springer Verlag, pp. 1 16.

[11] Barringer H., Gabbay D.M. and Reynolds M.D. (1996): Introduction to Temporal Logic. In The Imperative Future: Principles of Executable Temporal Logic, Research Studies Press, pp. 3 32.

[12] Barringer H., Fisher M.D., Gabbay D.M., Gough G.D. and Owens R.P. (1996): MetateM: A Framework for Executing Temporal Logic. Revised and updated version of workshop paper in The Imperative Future: Principles of Executable Temporal Logic, Research Studies Press, pp. 68 114.

[13] Barringer H. and Gabbay D.M. (2005): Modal Varieties of Temporal Logic. In Handbook of Temporal Reasoning in Artificial Intelligence (eds Fisher, Gabbay and Vila), pp 119-165, Elsevier.

[14] Barringer H., Gabbay D.M. and Woods J (2005): Temporal Dynamics of Support and Attack Networks: From Argumentation to Zoology. In Mechanizing Mathematical Reasoning, Essays in Honor of Jorg B. Siekmann on the Occasion of His 60th Birthday, LNCS 2605, pp 59-98, Springer, 2005.

[15] Barringer H. and Rydeheard D.E. (2005): Modelling Evolvable Systems: A Temporal Logic View. In We Will Show Them! Essays in Honour of Dov Gabbay, Kings College Press, 2005.

[16] Barringer H., Gabbay D.M. and Woods J (2008): Network Modalities: an exploration paper. In G. Gross and K.U. Schulz, editors, Linguistics, Computer Science and Language Processing: Festschrisfft for the 60th Birthday of Franz Guentner, College Publications, 2008.

[17] Barringer H., Rydeheard D.E. and Gabbay D.M (2010): Reactive Grammars: an exploration paper. In: Volume in honour of Yaacov Choueka, LNCS, Springer, 2010.

[18] Barringer H. and Gabbay D.M. (2010): Modal and Temporal Argumentation Networks. In: Time for Verification, Essays in Memory of Amir Pnueli, LNCS 6200, pp 1-25, Springer 2010.

[19] Barringer H. and Havelund K. (2011): Internal versus External DSLs for Trace Analysis (Extended Abstract). In Proceedings of 2nd International Conference on Runtime Verification, San Francisco, LNCS 7186, pp 1-3, Springer, 2012

Journal Articles

[20] Barringer H., Capon P.C. and Phillips R. (1979): The Portable Compiling Systems of MUSS. Software-Practice and Experience, Vol. 9, pp 645 655.

[21] Barringer H. and Mearns I. (1982): Axioms and Proof Rules for Ada Tasks. IEE Proceedings, Vol. 129, Pt. E, No. 2.

[22] Barringer H., Cheng J.H. and. Jones C.B. (1984): A Logic Covering Undefinedness in Program Proofs. Acta Informatica, Vol. 21, No. 3, pp. 251 269, 1984.

[23] Barringer H. and Mearns I. (1986): A Proof System for Ada Tasks. The Computer Journal, Vol. 29, No. 5, October 1986, pp. 404 415.

[24] Barringer H. (1987) Up and Down the Temporal Way. The Computer Journal, Vol.30, No.2, pp.134 148.

[25] Bjoerner D., Hoare C.A.R., Bowen J., He JiFeng, Langmaack H., Olderog E.-R., Martin U., Stavridou V., Nielson F. and H.R., Barringer H., Edwards D., Loevengreen H.H., Ravn A.P., Rischel H. (1989): A ProCoS Project Description: Esprit BRA 3104. Bulletin of EATCS, Number 39, pp. 60 73.

[26] de Roever W.P., Barringer H., Courcoubetis C., Gabbay D., Gerth R., Jonsson B., Pnueli A., Reed M., Vytopil J. and Wolper P. (1990): Formal Methods and Tools for the Development of Distributed and Real Time Systems: SPEC – Esprit Project 3096 (Extended Abstract). Bulletin of EATCS, Number 40, pp. 117 133.

[27] Barringer H., Fisher M.D., Gabbay D.M., Gough G.D. and Owens R.P. (1995): MetateM: An Introduction. Formal Aspects of Computing, Vol. 7, No. 5, Springer International, pp. 533 549.

[28] Barringer H., Gough G.D., Monahan B.Q. and Williams A.J. (1995): Symbolic Equivalence Checking for the ELLA Hardware Description Language. Journal of Brazilian Computer Society, Special Issue on EDA, pp. 49 62.

[29] Barringer H., Brough D., Fisher M., Hunter A., Owens R., Gabbay D., Gough G., Hodkinson I., McBrien P. and Reynolds M. (1996): Languages, Meta-Languages and MetateM, A discussion paper. Journal of the IGPL, Vol. 4, No. 2, pp. 255 272.

[30] Barringer H., Gough G., Monahan B., and Williams A. (1996): A Process Algebra Foundation for Reasoning about Core ELLA. The Computer Journal, Volume 39(4), pp. 303 324.

[31] Visser W. and Barringer H. (2000): Practical CTL* Model Checking - Should SPIN be Extended? International Journal of Software Tools for Technology Transfer, volume 2, issue 4, pp. 350 365.

[32] Barringer H., Fellows D., Gough G.D., and Williams A.J. (2002): Rainbow: Development, Simulation and Analysis Tools for the Asynchronous Micro-pipeline Hardware Design. The Computer Journal, volume 45(1), pp. 2 11, Special Focus: Formal Methods in Computation.

[33] Giannakopoulou D., Pasareanu C.S., and Barringer H. (2005): Component Verification with Automatically Generated Assumptions. Automated Software Engineering, Volume 12, number 3, pp 297-320, Springer Science, July 2005.

[34] Artho C., Barringer H., Goldberg A., Havelund K., Khurshid S., Lowry M., Pasareanu C., Rosu G., Sen K., Visser W. and Washington R. (2005): Combining Test Case Generation and Runtime Verification. Theoretical Computer Science 336, pp. 209-234, Elsevier.

[35] Inggs C.P. and Barringer H. (2006): CTL* Model Checking on a Shared-Memory Architecture. Formal Methods of System Design, Vol. 29, No. 2, pp. 135-155, Springer.

[36] Pasareanu C.S., Giannakopoulou D, Bobaru M.G, Cobleigh J.M. and Barringer H. (2008): Learning to Divide and Conquer: Applying the L* Algorithm to Automate Assume-Guarantee Reasoning. Formal Methods in System Design, Vol. 32, pp175-205, Springer.

[37] Barringer H., Rydeheard D.E. and Havelund K. (2008): Rule systems for run-time monitoring: from Eagle to RuleR (extended version). Journal of Logic and Computation, (JLC Advanced Access published Nov 21, 2008), Oxford University Press., June 2010, Vol. 20, No. 3, pp 675-706

[38] Yang N., Barringer H. and Zhang N. (2008): A Purpose-Based Access Control Model. Journal of Information Assurance and Security, Vol.1, pp 51-58.

[39] Barringer H., Gabbay D. and Rydeheard D.E. (2009): Modelling of Evolvable Systems: Part 1: A logical framework. Logic Journal of the IGPL, December 2009, Vol. 17, pp 631 - 696. 2010.

[40] Barringer H., Groce A., Havelund K. and Smith M.: Formal Analysis of Log Files. AIAA Journal of Aerospace Computing, Information and Communications, Vo. 7, No 11, pp365-390, 2010.

[41] H. Barringer H,, Gabbay D.M. and Woods J. (2012): Temporal, numerical and metalevel dynamics in argumentation networks. Argument & Computation, 3:2-3, 143-202.

[42] Barringer H. , Gabbay D.M. and Woods J. (2012): Modal and Temporal Argumentation Networks. Argument & Computation, 3:2-3, 203-227.

Refereed Conference Articles

[43] Barringer H. and Lindsey C.H. (1977): The Manchester ALGOL 68 Compiler - Part 1. In Proceedings of the 5th Annual III Conference, Guidel, France, pp 145-165, INRIA.

[44] Barringer H. and Kuiper R. (1983): Towards the Hierarchical, Temporal Logic, Specification of Concurrent Systems. In Proceedings of the STL/SERC Workshop on the Analysis of Concurrent Systems Cambridge, September 1983, LNCS Vol. 207 pp.157 183.

[45] Barringer H. (1984): Formal Specification Techniques for Parallel and Distributed Systems: A Survey. In Proceedings of the Third Joint Ada Europe/AdaTEC Conference, Brussels, 26 28 June 1984, The Ada Companion Series, (ed. J. Teller), Cambridge University Press.

[46] Barringer H. and Kuiper R. (1984): Hierarchical Development of Concurrent Systems in a Temporal Logic Framework. Proceedings of the NSF/SERC Seminar on Concurrency, CMU, Pittsburgh, July 1984, LNCS Vol. 197, pp. 45 75.

[47] Barringer H., Kuiper R. and Pnueli A. (1984): Now You May Compose Temporal Logic Specifications. In Proceedings of the 16th ACM Symposium on the Theory of Computing, Washington, pp. 51 63.

[48] Barringer H. (1985): Specifying Ada Tasks, Proc. of the Ada UK Conference Ada — The Next Ten Years, York. In Ada UK News, Vol. 6, No. 2.

[49] Barringer H., Kuiper R. and Pnueli A. (1985): A Compositional Approach to a CSP-like Language. In Formal Models in Programming, ed. E.J. Neuhold and G. Chroust, North Holland, Proc. of the IFIP TC2 Working Conference The Role of Abstract Models in Information Processing, Vienna, pp. 207 227.

[50] Barringer H., Kuiper R. and Pnueli A. (1986): A Really Abstract Concurrent Model and its Temporal Logic. In Proceedings of the ACM Symposium on Principles of Programming Languages, St. Petersburg, pp. 173 183.

[51] Barringer H., Gough G.D. (1988): A Semantics Driven Temporal Verification System. Proc. of European Symposium on Programming, Nancy, LNCS 300, pp. 21 34.

[52] Barringer H., Edwards D.A. and Stavridou V. (1988): Formal Specification and Verification of Hardware: A Comparative Case Study. Proc. 25th ACM/IEE Design Automation Conference, Los Angeles, pp. 197-204, 1988. Reprinted in Formal Verification of Hardware Design (ed. Michael Yoeli), IEEE Computer Society Press (1991).

[53] Barringer H. and Gabbay D.M. (1988): Executing Temporal Logic: Review and Prospects. In Proc. Concurrency 88, Hamburg, LNCS 335, pp. 104 105.

[54] Barringer H., Fisher M.D., Gabbay D.M., Gough G.D. and Owens R.P. (1990): MetateM: A Framework for Executing Temporal Logic. Proc. of REX Workshop, Stepwise Refinement of Distributed Systems: Models, Formalisms and Correctness, Mook, May/June 1989, LNCS 430, pp. 94 129.

[55] Banieqbal B. and Barringer H. (1989): Temporal Logic with Fixpoints. In Proc. Colloquium on Temporal Logic and Specification, Altrincham, 1987, LNCS 398, pp. 62 74.

[56] Barringer H., Fisher M.D. and Gough G.D. (1989): Fair SMG and Linear Time Model Checking. In Proc. of Workshop on Automatic Verification Methods for Finite State Systems, Grenoble, June 12th-14th, 1989, LNCS 407, pp. 133 150.

[57] Barringer H., Fisher M.D., Gabbay D.M. and Hunter A. (1991): MetaProgramming in MetateM. In Proc. of Knowledge Representation '91, Morgan Kaufmann.

[58] Barringer H. and Fisher M.D. (1991): Concurrent MetateM Processes –A Language for Distributed AI. In Proc. European Simulation Multiconference '91.

[59] Barringer H., Gough G.D. and Monahan B.Q. (1992): Operational Semantics for Hardware Design Languages. In Proc. of the Advanced Research Workshop on Correct Hardware Design Methodologies, edited by P.Prinetto and P.Camurati, pp. 313 334, Elsevier.

[60] Barringer H., Gough G.D., Longshaw T.B., Monahan B.Q., Peim M. and Williams A.J. (1992): Semantics and verification for Boolean Kernel ELLA using IO automata. In Proc. of the Advanced Research Workshop on Correct Hardware Design Methodologies, edited by P.Prinetto and P.Camurati, pp. 65 90, Elsevier.

[61] Barringer H.,.Gough G.D, Longshaw T.B., Monahan B.Q., Peim M. and Williams A.J. (1993): Formal Verification Support for ELLA. In Proceedings of the 1993 DTI/SERC JFIT Conference, Keele.

[62] Barringer H., Dixon C. and Fisher M.D. (1994): A Graph-Based Approach to Resolution in Temporal Logic. In Proceedings of the First International Conference, ICTL '94, LNAI 827, pp. 415 429.

[63] Barringer H., Gough G.D., Monahan B.Q. and Williams A.J. (1995a): Symbolic Verification of hardware systems. In Proceedings of ASP-DAC/CHDL/VLSI'95, Chiba, Japan, pp. 631 636.

[64] Barringer H., Gough G.D., Monahan B.Q. and Williams A.J. (1995b): A Design and Verification Environment for ELLA. In Proceedings of ASP-DAC/CHDL/VLSI'95, Chiba, Japan, pp 685-691.

[65] Barringer H., Gough G.D., Monahan B.Q. and Williams A.J. (1995): Formal Support for the ELLA Hardware Description Language. In Proceedings of CHARME'95, Frankfurt, LNCS 987, pp. 225 246.

[66] de Melo A.C.V. and Barringer H. (1995): A Foundation for Formal Reuse of Hardware. In Proceedings of CHARME'95, Frankfurt, LNCS 987, pp. 124 145.

[67] Barringer H., Fellows D., Gough G.D., Jinks P.J., Marsden B.W. and Williams A.J. (1996): Design and Simulation in Rainbow: A framework for Asynchronous Micropipeline Circuits. In A.G. Bruzzone and U.J.H. Kerckhoffs, editors, Proceedings of the European Simulation Symposium (ESS'96), pp. 567 571, Vol. 2. Genoa, Italy, October 1996, Society for Computer Simulation International.

[68] Visser W. and Barringer H. (1997): Memory Efficient State Storage in Spin. In J.-C. Gregoire, G.J. Holzmann, D.A. Peled, editors, Proceedings of the Second Workshop on the Spin Verification System (August 1996), DIMACS Series in Discrete Mathematics and Theoretical Science, pp. 185 203, Vol. 32, American Mathematical Society.

[69] Barringer H., Fellows D., Gough G.D. and Williams A.J. (1997): Abstract Modelling of Asynchronous Micropipeline Systems using Rainbow. In C.D. Kloos, E. Cerny, editors, Hardware Description Languages and their Applications, Proceedings of IFIP TC10 WG10.5 International Conference on Computer Hardware Description Languages and their Applications (CHDL'97), Toledo, pp. 285 304, Chapman & Hall.

[70] de Melo A.C.V. and Barringer H. (1997): Minimization of Synchronous Processes Preserving Bisimulation. In S. Bampi, M. Lubaszzewski, editors, Proceedings of the X Brazilian Symposium on Integrated Circuit Design (SBCCI '97), pp. 293 302.

[71] Barringer H., Fellows D., Gough G.D., Jinks P.J. and Williams A.J. (1997): Multi-view design of Asynchronous Micropipeline Systems using Rainbow. In R. Reis, L. Claesen, editors, IX IFIP International Conference on Very Large Scale Integration (VLSI'97), Gramado, Chapman & Hall.

[72] Visser W., Barringer H., Fellows D., Gough G.D. and Williams A.J.(1997): Efficient CTL* model checking for analysis of Rainbow designs. In H.F. Li, D.K. Probst, editors, Advances in Hardware Design and Verification, Proceedings of IFIP TC10 WG10.5 International Conference on Correct Hardware and Verification Methods, Montreal, pp. 128 145, Chapman & Hall.

[73] Visser W. and Barringer H. (1998): CTL* Model Checking for SPIN. In Proceedings of 4th Workshop on Automata Theoretic Verification with the SPIN Model Checker (SPIN98), Paris, Ecole Nationale Superieure des Telecommunications - ENST 98 S 002.

[74] Inggs, C.P. and Barringer H. (2002): On the Parallelisation of Model Checking. In Proceedings of the Second Workshop on Automated Verification of Critical Systems (AVoCS 2002), Technical Report CSR-02-6, School of Computer Science, the University of Birmingham, England, April 2002.

[75] Inggs, C.P. and Barringer H. (2002): Effective State Exploration for Model Checking on a Shared Memory Architecture. In Proceedings of the Workshop on Parallel and Distributed Model Checking (PDMC 2002), held in association with Concur 2002, Brno, August 2002.

[76] Giannakopolou, D., Pasareanu, C. and Barringer H. (2002): Assumption Generation for Software Component Verification. In Proceedings of 17th IEEE/ACM Automated Software Engineering Conference, Edinburgh, September 2002. (this paper received an ACM SigSoft Distinguished Paper Award and was the Best in Conference)

[77] Barringer H., Giannakopolou, D. and Pasareanu, C. (2003): Proof Rules for Automated Compositional Verification through Learning. In Proceedings of the International Workshop on Specification and Verification of Component Based Systems, Helsinki, September 2003.

[78] Barringer H., Goldberg A., Havelund K. and Sen K. (2004): Rule-based Runtime Verification. In Proceedings of VMCAI04, Venice, LNCS Vol. 2937, Springer, pp 44–57, January 2004.

[79] Barringer H., Goldberg A., Havelund K. and Sen K. (2004): Run-time Monitoring with LTL in Eagle. Proceedings of PADTAD 04, Santa Fe, New Mexico, April 2004, IEEE Computer Society, IDPDS04, Volume 17, Number 17.

[80] Inggs, C.P. and Barringer H. (2005): CTL* Model Checking on a Shared-Memory Architecture. In Proceedings of the Workshop on Parallel and Distributed Model Checking (PDMC 2004), in association with Concur 2004, ENTCS, Vol. 128, No. 3, pp 107–123, September 2005.

[81] Inggs, C.P., Barringer H. , Nenadic, A. and Zhang N. (2004): Model Checking a Security Protocol. In Proceedings of SATNAC 2004, Spier Wine Estate, Western Cape, RSA, September 2004.

[82] Barringer H., Rydeheard D.E., Warboys B.C. and Gabbay D. (2007): A Revision-based Logical Framework for Evolvable Software. In Proceedings of the 25th IASTED International Multi-Conference, Software Engineering, February, Innsbruck, Austria, pp78-83.

[83] Barringer H., Rydeheard D.E. and Havelund K. (2007): Rule systems for run-time monitoring: from Eagle to RuleR. In Proceedings of the 7th International Workshop on Runtime Verification, Vancouver, March 2007, LNCS Vol. 4839, pp 111-125, Springer.

[84] Barringer H., Rydeheard D.E. and Gabbay D. (2007): From Runtime Verification to Evolvable Systems. In Proceedings of the 7th International Workshop on Run-time Verification, Vancouver,

March 2007, LNCS Vol. 4839, pp 97–110, Springer.

[85] Barringer H., Rydeheard D.E. and Gabbay D. (2007): A Logical Framework for Monitoring and Evolving Software Components. In 1st IEEE International Conference on Theoretical Aspects of Software Engineering (TASE07), Shanghai, China, IEEE Computer Press, pp 373-383, June 2007.

[86] Baran J. and Barringer H. (2007): A Grammatical Representation of Visibly Pushdown Languages. In 14th Workshop on Logic, Language, Information and Computation (WoLLIC07), LNCS, Vol. 4576, pp 111, July 2007.

[87] Yang N., Barringer H. and Zhang N. (2007): A Purpose-Based Access Control Model. In 3rd International Symposium on Information Assurance and Security (IAS 07), IEEE CS Press, August 2007.

[88] Baran J., Barringer H. (2008): Forays into Sequential Composition and Concatenation in Eagle. Runtime Verification. 8th International Workshop, RV2008, Budapest, Hungary, March 2008. Selected Papers. LNCS Vol. 5289, pp 69-85, Springer.

[89] Bujorianu M.C., Bujorianu M.L and Barringer H. (2009): A Formal Framework for User Centric Control of Probabilistic Multi-Agent Cyber-Physical Systems. Selected papers of CLIMA workshop, LNCS, Vol. 5405, pp 97-116, Springer 2009.

[90] Bujorianu M.C., Bujorianu M.L and Barringer H. (2009): A Unifying Specification Logic for Cyber-Physical Systems. In Proceedings of 17th Mediterranean Conference on Control, Automation, IEEE Computer Society Press, pp 1166-1171, 2009.

[91] Barringer H. Havelund K., Rydeheard D.E. and Groce A.(2009): Rule Systems for Runtime Verification: A Short Tutorial. Runtime Verification, 9th International Workshop, RV 2009, Grenoble, France, June 26-28, 2009. Selected Papers. LNCS, Vol. 5779, pp 1-24, Springer 2009.

[92] Bujorianu M.C. and Barringer H.(2009): An Integrated Specification Logic for Cyber-Physical Systems. ICECCS 2009: pp 291-30.

[93] Barringer H. , Groce A., Havelund K., Smith M.(2010): An Entry Point for Formal Methods: Specification and Analysis of Event Logs. Proceedings of the FMA Workshop, November 2009. EPTCS, 2010.

[94] Afifi, D., Rydeheard, D. and Barringer H. (2010): ESAT: A Tool for Animating Logic-Based Specifications of Evolvable Component Systems. Proceedings of the International Conference on Runtime Verification, RV 2010, Malta, Nov 1-4, LNCS, Vol. 6418, pp 469474, Springer 2010.

[95] Barringer H., Havelund K., Kurklu E. and Morris R.: Checking Flight Rules with TraceContract — Application of a Scala DSL for Trace Analysis. In Scala Days 2011, Stanford, June 2011.

[96] Barringer H., Havelund K.: TraceContract: A Scala DSL for Trace Analysis. In Proceedings of FM11, 17th International Symposium on Formal Methods, Limerick, Ireland, June 2011.

[97] Bujorianu M.L., Bujorianu M.C. and Barringer H. (2011): Systems Theory in an Analytic Setting. In Proceedings of 50th Conference CDC-ECC 2011, IEEE Computer Society, 2011.

[98] Barringer H., Falcone Y., Havelund K, Reger G. and Rydeheard D.: Quantified Event Automata: Towards Expressive and Efficient Runtime Monitors. In Proceedings of FM 2012, 18th International Symposium on Formal Methods, Paris, France, LNCS vol. 7436, pp 6884, August 2012.

[99] Reger, G., Barringer H. and Rydeheard D. (2013): A Pattern-Based Approach to Parametric Specification Mining. In Proceedings of 28th IEEE/ACM Automated Software Engineering Conference, Palo Alto, November 2013.

[100] Reger, G., Barringer H. and Rydeheard D. (2013): Automata-based Pattern Mining from Imperfect Traces. 2nd International Workshop on Software Mining, Palo Alto, 2013.

Other Conference Articles

[101] Barringer H. (1983): On the Development of Parallel/Distributed Programs. Proceedings of the BCS-FACS/SERC Workshop on Specification and Verification, York.

[102] Babb E., Barringer H. and Gabbay D.M. (1991): HF-PLL — A Meta-level Rewrite Language for Non-linear Planning. European Workshop on Planning (EWSP-91), GMD, Sankt Augustin.

[103] Barringer H., Gough G.D., Monahan B.Q. and Williams A.J. (1995): The State Evolution Method for Verifying Hardware Systems. Poster presentation at CHARME'95.

Technical Reports

[104] Banieqbal B. and Barringer H. (1986): A Study of an Extended Temporal Language and a Temporal Fixed Point Calculus. Technical Report UMCS-86-10-2, Department of Computer Science, University of Manchester.

[105] Fisher M.D. and Barringer H. (1986): Program Logics: A short survey. Technical Report UMCS-86-11-1, Department of Computer Science, University of Manchester.

[106] Barringer H., Gough G.D., Longshaw T.B., Monahan B.Q., Peim M. and Williams A.J. (1992): Towards the semantics and verification for ELLA. Technical Report UMCS-92-4-6, Department of Computer Science, 50 pages, May 1992.

[107] Barringer H., Gough G.D., Monahan B.Q. and Williams A.J. (1993): Formal Semantic Model for ELLA. Technical report UMCS-93-2-1, Department of Computer Science, 160 pages, February 1993.

[108] Ying Zhang and Barringer H. (1994): A Reified Temporal Logic for Nonlinear Planning. Technical Report UMCS-94-7-1, Department of Computer Science, 24 pages, July 1994.

[109] Ying Zhang, Carlisle D. and Barringer H. (1994): Domain Constraint Maintenance. Technical Report UMCS-94-8-1, Department of Computer Science, 28 pages, August 1994.

[110] Ying Zhang and Barringer H. (1994): Constraint Logic Planning. Technical Report UMCS-94-9-2, Department of Computer Science, 33 pages, September 1994.

[111] Ying Zhang and Barringer H. (1994): Action Constraint Maintenance. Technical Report UMCS-94-9-1, Department of Computer Science, 24 pages, August 1994.