Formal Modeling and Analysis of Timed Systems: Technology Push or Market Pull?

Boudewijn R. Haverkort[1,2]

[1] Embedded Systems Institute, Eindhoven
boudewijn.haverkort@esi.nl
http://www.esi.nl/
[2] Design and Analysis of Communication Systems, University of Twente
http://www.utwente.nl/ewi/dacs/

Abstract. In this short paper I will address the question whether the methods and techniques we develop are applied well in industrial practice. To address this question, I will make a few observations from the academic field, as well as from industrial practice. This will be followed by a concise analysis of the cause of the perceived gap between the academic state-of-the-art and industrial practice. I will conclude with some opportunities for improvement.

1 Background

Twenty-five years ago, in september 1986, I received an engineering degree in Computer Science from the University of Twente. Since then, I have been working in the field of performance and dependability evaluation of computer and communication systems, either developing theory, methods and tools, or working on applications. My work on theory includes Markovian models of all sorts, queueing network analysis techniques, rare event simulation techniques, quasi-birth-death processes (QBDs), and, over the last 10 years, much work on stochastic model checking [2].

Although I have published by far the most on theory and tools, cf. [10], I did address quite some application fields as well, including computer networking (e.g., ATM and B-ISDN networks, WLAN and token ring access networks, TCP/IP flow and congestion control), fault-tolerant distributed systems (e.g., fault-tolerant multiprocessor systems, system survivability with application to the Google file system), and embedded systems (e.g., embedded control systems, or wearable power-constrained communication devices). Lately, I have become more involved in SCADA systems, as well as with intrusion detection in network systems as part of my activities at the University of Twente, and with a wide variety of embedded computer-control systems as part of my activities at the Embedded Systems Institute (ESI), the latter in close cooperation with leading high-tech industries such as ASML, NXP, Océ, Philips and Thales.

U. Fahrenberg and S. Tripakis (Eds.): FORMATS 2011, LNCS 6919, pp. 18–24, 2011.

2 Key Question Addressed

Being able to look back at 25 years of scientific work in this field, it is a good time to ask what has been achieved in the generic field of modeling and analysis of timed systems, and how valuable that has been, and for whom?

In the past three years, I already had the opportunity to reflect on the achievements of the field of performance evaluation at large, through invited presentations at three European conferences [5,8,9]. In this short paper, I would like to go a step further, and take my experience with industry while being employed at the ESI since 2009, much more into account. The key question that I address then is: ***Are the methods and techniques we develop being used in industrial practice?*** With ***we*** in this question, I do mean the academic community, that is, the typical participants of conferences like FORMATS, QEST, CAV, E-PEW, MASCOTS, etc.

To address this question, I will start with a few observations from the academic field, as well as from industrial practice, followed by a concise analysis, and some directions towards the future.

3 Observations from the Academic Field

Let me first address five observations that I think I can honestly make about the academic field that addresses models and techniques for the evaluation of quantitative system properties, such as timeliness, performance and reliability. I restrict to five only, for conciseness, the list of course is longer and more detailed. The list is not intended to blame anyone or any sub-community; I am just critically assessing the work being done by the community I have been working in happily myself for 25 years!

A1. The key theoretical results in most operations research related fields, such as queueing networks, stochastic Petri nets, optimization and discrete-event simulation, have been obtained 25 years or longer ago. The number of truly new results in this field in the last 25 years is rather small; one can think about models for self-similar traffic, improved EM-fitting of phase-type distributions, or the logarithmic reduction algorithm for QBDs.

A2. Important new results have been achieved in the field of explicit state-space-based methods, including storage structures like BDD's and MTBDD's, as well as efficient model checking and numerical algorithms, in the last 25 years.

A3. Key results have been attained regarding the structured (sometimes compositional) description of systems and system behavior, using model- or analysis-specific languages, having a formal basis, thus forming a good starting point for further analysis and synthesis. Think of timed-automata of various sorts, advances in process algebra, etc.

A4. Much more developer-friendly (this is not necessarily *user*-friendly) and powerful software tools have become available, that allow for larger case

studies to be addressed with the techniques referred to in A1–3. Note that this trend is largely a result the availability of commodity computing power since the mid 1980's, hence, of Moore's law, and far less so of academic improvements.

A5. A large share of the recent work in academia is addressing (minor) extensions of methods and techniques (cf. A1–2), of description techniques (cf. A3) as well of as supporting tools (cf. A4), simply because theory allows us to do so ("technology push"), rather than that there is well-pronounced need for this from industry ("market pull").[1]

4 Observations and Trends in Industry

Similar as done for academia, let me also address five key observations from industry. Again, the list is longer, and there will be nuances to all of the items. I compiled this list as by-product of the process towards a new research agenda for ESI [4]. As such, it primarily addresses the embedded systems field, however, I do think that these observation apply more generally.

I1. There is an ever-increasing growth of complexity of systems and system software, for a variety of reasons, including the inclusion of legacy code or third-party components, increased openness, higher user expectations, the use of multicore hardware, mixed criticality of the software, and the advent of systems-of-systems.

I2. Given this increasing complexity, both system design and system test and integration, in terms of the classical V-model for system development, are severely challenged, technically, but also regarding time and cost constraints.

I3. There is an increased interest in model-driven design, however, in most industries with a focus on modeling as a kind of high-level programming, and code generation taking over part of the intensive coding work, thus saving time and costs.

I4. A sense of urgency regarding modeling and quantitative system evaluation is felt, but it is undirected as of yet. Modeling and analysis, as we advocate it, is not a natural component in the design process of software-intensive systems. This is very much unlike the practice in many other system design areas, like hardware, mechanical or optical system design.

I5. When modeling is used to guide the design process, that is, to explore the design space and to make trade-offs, this is either done on the basis of back-of-the-envelope models (or spread-sheets at best), or on the basis of extremely detailed simulation models that are, in fact, system prototypes implemented in a simulator environment. In most cases, the modeling techniques we have been developing, are *not* being used.

[1] A colleague of mine, from a different field in computer science, referred to this as "Harley Davidson extension work", liked by Harley Davidson hobbyists, irrelevant for anyone else, but annually displayed at length at nic(h)e workshops and conferences around the world.

5 Analysis

As should be clear from the two sets of observations, there is a world to gain here. Industry is in need, and academia wants to deliver. However, what is delivered, does not appear to be the right answer. Hence, the answer to the initially posed question, in my opinion, clearly is not positive. The academic community is doing great things, however, the uptake is slow. In other words, we have been doing things right, that is, correct, but the question more is, did we do the right things? It is too easy to say that industry is to blame, because they are not accepting what we deliver and say is good for them.

Why is it that "our methods" are not being used? Why do industrial developers not use what we deliver? Again, there is no definitive answer to this, however, I again would like to make a few observations, that might help in bringing these worlds closer together. As before, the list can be made longer, the current items might not be all orthogonal, still they do provide some insight.

1. The "modeling and analysis" community, that is, we, have become disconnected from the computer systems community. In the 1960's and 1970's, the researchers working on multi-programmed time-sharing computer systems and computer networks, respectively, also did make the models needed to dimension them, see, e.g., [6]. Also the early work on what are now called formal methods, was done by researchers developing protocol systems, in the 1980's, see, e.g., [1,7]. The models and techniques being developed, were clearly developed with a "market pull". By now, it appears we have moved much more towards a "technology push" model, with almost completely disjunct communities and conferences, pushing methods and techniques to the market that are not being bought, for various reasons, be it complexity of use, or simply because they do not have the right functionality. Should we be surprised that our methods are not used?
2. Academic key performance indicators (KPI's), especially those from the sciences, do force researchers to publish many papers, at highly ranked conferences and in top-quality journals. Being part of a community in which one fits well and in which the work is well received, is part of academic survival. It is unfair to solely blame researchers for this. However, there is a side effect that cannot be ignored either.
3. A large part of computer science research has developed itself strongly along the science-axis. The *analysis* of (existing) systems is prevalent. However, the engineering and constructive side is crucial to industry, as this is the line along which value is created. In the end, ICT systems are man-made systems that need to be built, preferably using solid engineering principles. A revitalization of computer engineering, or probably the science of computer engineering, is needed.
4. The study of quantitative properties of computer systems and to develop appropriate modeling and analysis techniques for that, does require an active attitude towards measurements and experimentation, for at least three reasons: (i) to calibrate the system models (parametrization), (ii) to validate

the models themselves, and (iii) to validate modeling approaches. We see too little of this [12]; if we do not validate our models and modeling techniques ourselves, can we expect others to simply believe us and buy them?

5. Going out there, that is, working on real industrial cases is very challenging. Many researchers do feel safer at home, working on the models and techniques they have been working on for a long time, thus avoiding to enter unknown territory. Of course, for an individual Ph.D.-student—the work horse of modern research—it is very risky to embark on a project that is largely industry-driven. At the same time, industry is not always as open as needed to embark on such joint projects. Both professors and senior researchers and developers in industry should take their responsibility in progressing the field.
6. Industrial practice asks for extreme scalability of methods and techniques. The community made great improvements here, especially in the field of symbolic functional verification. We have to go a long way for scalable quantitative analysis, let alone for synthesis while preserving quantitative constraints. Let these needs be leading! This might mean, that paradigm shifts are needed, e.g., moving to mean-field analysis, instead of exact explicit state-space analysis.
7. True industrial use, that is, in system development and not just in research departments, of the techniques and tools we like so much, does require the embedding in daily work routines. For many industries, this means that a nice such-and-such tool, will not be used if it is not part of, e.g., the Matlab Simulink toolset, or cannot be connected to IBM's Rational Rhapsody family of products. Your tool is not their tool! At best, our tools are seen as "engine" in some larger process chain. We cannot deny this. How many academic researchers know which tools are being used in industrial practice? How different is this from other engineering disciplines?

6 Opportunities!

Having made two sets of observations, as well as an analysis, helps us in determining how to improve things. In the end, we all want to have impact with our work, to have the story that our work is being used extensively in the design of new series of products by major industries.

Here, even more than before, I am very careful. I do not have the final recipe, however, I do have some thoughts, which I like to share with you, before I address a way-of-working we execute and advocate at ESI: the industry-as-lab approach.

Please note that my thoughts below do not exclude at all good fundamental research. However, I do say that the balance between fundamental and applied research should probably be shifted, in the engineering discipline we should like to be. Probably the difference between applied and fundamental research is overemphasized here; maybe there is just good and not-so-good research.

1. Wouldn't it be great to see our tools and techniques have true impact? In my opinion this will much more easily happen if we do incorporate industry in

our research projects more firmly, right from the start. This does go beyond doing some realistic industrial case study at the end of a 4-year project. What is needed is industrial involvement in the problem definition and a continuous dialogue between the problem owner (industry) and the solution provider. This does require serious investments from either side.

2. If I would be a medical doctor working in an academic medical centre, I would teach, I would do research, and I would really treat people, even operate them. A similar mix of activities is often seen in Architecture or in Engineering faculties. How rare is this in computer science! It might be an idea to have sabbaticals more often across borders, that is, an academic really working in a development lab in industry, and a computer system of software engineer spending a semester in academia. National funding agencies could accommodate this. In the Netherlands, the Ministry of Economic Affairs supported the latter scheme during the crisis year 2010. This could even taken further, in that such exchanges form part of the human capital agenda of industries and academia alike.
3. The Dutch Technology Foundation STW does foster programs for joint research, which are being defined by committees with members from industry and academia. Moreover, industry does invest in these programs in that they cater for 50% of the costs. In doing so, the problem owner is actively involved. This leads to challenging projects, in which both industry and academia have to deliver.

Finally, at ESI we have successfully followed a scheme which we have called ***industry-as-lab*** after a model proposed by Potts for the software engineering field [11]. At ESI we tailored it to the high-tech embedded systems field, encouraged by new insights in open innovation [3].

Potts observed that most research projects in software engineering follow the traditional research-then-transfer paradigm. In a way, this is a different wording for "technology push". However, the effective result of such projects is often questionable; Potts even classifies some of the typical project goals simply as naive. According to Potts, the industry-as-lab approach sacrifices revolution, but strongly fosters evolution. Industry does not like revolution most of the time, it does like evolution. The key idea is that industry is involved in problem identification at the outset, and jointly with academic partners forms a project description and consortium; this is a true market pull. Interaction between industry and academic partners is intensive, with Ph.D.-students spending time in industry every week, meeting system and software engineers, joining meetings to better understand what the true problems are, and to show what can be learned or gained from more academic approaches. Note that this does go well beyond the industry-academic interaction in many European projects, where project partners meet a couple of times per year, and only interact superficially. Industry-as-lab allows for a continuous large-scale (experimental) validation of research work. It changes the sequential research-*then*-transfer paradigm into a continuous research-***and***-transfer paradigm, allowing very short feedback cycles.

The industry-as-lab way of working is very challenging, but very rewarding as well, leading to good research results achieved under realistic constraints, hence, leading to more direct applicability. Being able to state that one's newly created technique is really used in industry for developing their new so-and-so product, also helps in gaining academic credits; at ESI, we have seen many examples of that.

7 Epilogue

In this short paper I have addressed the question how well the formal modeling and analysis techniques for quantitative system properties, as we know and like time, are taken up in industry. By carefully observing what happens in academia and industry, I have to conclude that despite great developments in academia, the industrial uptake is disappointing. And understandably so, I am afraid. But this is not to say that this cannot improve. Instead, I proposed a number of ways to improve the required interaction in order to shift from a technology push situation, to a market pull situation. The industry-as-lab paradigm, as practiced by ESI, implements such a shift, thereby truly bridging between academia and industry in an open innovation setting.

References

1. Bolognesi, T., Brinksma, E.: Introduction to the ISO Specification Language LOTOS. Computer Networks 14, 25–59 (1987)
2. Baier, C., Haverkort, B.R., Hermanns, H., Katoen, J.-P.: Performance Evaluation and Model Checking Join Forces. Communications of the ACM 53(9), 76–85 (2010)
3. Chesbrough, H.W.: The Era of Open Innovation. MIT Sloan Management Review 44(3), 35–41 (2003)
4. The Embedded Systems Institute Strategic Research Agenda 2011 Forthcoming fall (2011), `http://www.esi.nl/`
5. Thomas, N., Juiz, C. (eds.): EPEW 2008. LNCS, vol. 5261. Springer, Heidelberg (2008)
6. Frenkel, K.A.: Big Blue's Time-Sharing Pioneer. Communications of the ACM 30(10), 824–828 (1987)
7. Holzmann, G.J.: Design and Validation of Computer Protocols. Prentice Hall, Englewood Cliffs (1990)
8. Proceedings IEEE MASCOTS. IEEE CS Press, London (September 2009)
9. Proceedings of the International Conference on Operations Research. Springer, München (September 2010)
10. My DBLP page, `http://www.informatik.uni-trier.de/ ley/db/indices/a-tree/h/Haverkort:Boudewijn_R=.html`
11. Potts, C.: Software-Engineering Research Revisited. IEEE Software 10(5), 19–28 (1993)
12. Tichy, W.F.: Should Computer Scientists Experiment More? IEEE Computer 31(5), 32–40 (1998)